STATUS OF THESIS

Title of thesis: Security, Trust and Privacy (STP) Model for Federated Identity and Access Management (FIAM) Systems

I, ZUBAIR AHMAD KHATTAK

hereby allow my thesis to be placed at the Information Resource Center (IRC) of Universiti Teknologi PETRONAS (UTP) with the following conditions:

1. The thesis becomes the property of UTP
2. The IRC of UTP may make copies of the thesis for academic purposes only.
3. This thesis is classified as

☑ Confidential

□ Non-confidential

If this thesis is confidential, please state the reason:
Protecting the Copyright of the Source Code

The contents of the thesis will remain confidential for 5 years.

Remarks on disclosure:

	Endorsed by
Signature of Author	Signature of Supervisor
Permanent address:	Name of Supervisor
House #. C-G-14,	Dr. Suziah Sulaiman
Apartment Orkid,	
Jalan Cemara Taman Bukit Serdang 47000	
Seri Kembangan, Kuala Lumpur, Malaysia.	
Date : 01/07/2013	Date :

UNIVERSITI TEKNOLOGI PETRONAS

SECURITY, TRUST AND PRIVACY (STP) MODEL FOR FEDERATED IDENTITY AND ACCESS MANAGEMENT (FIAM) SYSTEMS

by

ZUBAIR AHMAD KHATTAK

The undersigned certify that they have read, and recommend to the Postgraduate Studies Programme for acceptance this thesis for the fulfillment of the requirements for the degree stated.

Signature: ________________________________

Main Supervisor: Dr. Suziah Sulaiman

Signature: ________________________________

Co-Supervisor: Dr. Jamalul-Lail Ab Manan

Signature: ________________________________

Head of Department: Dr. Jafreezal Bin Jaafar

Date: ________________________________

SECURITY, TRUST AND PRIVACY (STP) MODEL FOR FEDERATED IDENTITY AND ACCESS MANAGEMENT (FIAM) SYSTEMS

By

ZUBAIR AHMAD KHATTAK

A Thesis

Submitted to the Postgraduate Studies Programme

as a Requirement for the Degree of

DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER AND INFORMATION SCIENCES

UNIVERSITI TEKNOLOGI PETRONAS

BANDAR SERI ISKANDAR

PERAK

JULY 2013

DECLARATION OF THESIS

Title of thesis	Security, Trust and Privacy (STP) Model for Federated Identity and Access Management (FIAM) Systems

I, ZUBAIR AHMAD KHATTAK

hereby declare that the thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at UTP or other institutions.

Witnessed by

Signature of Author

Permanent address:
House #. C-G-14,
Apartment Orkid,
Jalan Cemara Taman Bukit Serdang 47000
Seri Kembangan, Kuala Lumpur, Malaysia.

Signature of Supervisor

Dr. Suziah Sulaiman

Date :

Date :

DEDICATION

This thesis is dedicated to *my beloved parents, brothers, sister and my lovely wife*.

ACKNOWLEDGEMENT

First of all, I am extremely grateful to Almighty Allah, the most beneficent, forgiving and generous, for His sanctifies and intensifies my valor to polish and complete this work successfully. I am extremely grateful to Dr. Suziah Sulaiman for her incessant encouragement, precious suggestions, inspiration, and unending vigor to guide me in the completion of this research. I am as well vastly appreciative to my field-supervisor Dr. Jamalul-Lail Ab Manan for his eager attentiveness, caring research directions, valuable technical suggestions and ethical support. I am also grateful to the joint effort of Universiti Teknologi PETRONAS, Malaysia and MIMOS Berhad, Malaysia management making it possible for me to conduct the implementation of this research.

I am extremely grateful to Prof. Dr. Kai Rannenberg (Goethe University Frankfurt, Germany), Prof. Dr. Audun Jøsang (UiO, Norway) and Dr. Reiner Sailor (IBM, USA) for their time and precious technical views to enhance this research.

I am enormously grateful to my beloved parents, brothers, sister, wife and all my family members whose hands always rise in prayer for my success and are the source of my inspiration all through this work and my life. I am also grateful to all of my friends who always welcome me whenever I need their support.

ABSTRACT

The federated identity and access management systems facilitate the home domain organization users to access multiple resources (services) in the foreign domain organization by web single sign-on facility. In federated environment the user's authentication is performed in the beginning of an authentication session and allowed to access multiple resources (services) until the current session is active. In current federated identity and access management systems the main security concerns are: (1) In home domain organization machine platforms bidirectional integrity measurement is not exist, (2) Integrated authentication (i.e., username/password and home domain machine platforms mutual attestation) is not present and (3) The resource (service) authorization in the foreign domain organization is not via the home domain machine platforms bidirectional attestation. Furthermore, absence of bidirectional trust in federated organizations machine platforms may cause the threats such as worms, phishing via Trojans, keyloggers and rootkits). The Trusted Computing solutions (e.g., trusted platform module and mutual attestation technique) may assist to overcome the machine platforms security and trust issues in federated identity and access management systems. However, the use of mutual attestation scheme in a federated environment may lead to the machine platforms measurement (security credential) privacy issue. The aforementioned issues motivated this research to construct a practicable and unified security, trust and privacy solution for federated organizations to collaborate in a secured, trustworthy and privacy-enhanced fashion.

In this work Shibboleth was chosen to construct a practicable and unified security, trust and privacy framework. The proposed solution: (i) integrates mutual attestation technique with Shibboleth basic user authentication mechanism, (ii) ensures bi-directional platform trust formation between the home domain identity provider and client machine, (iii) ensures privacy of the home domain machine platforms security credentials conserved at the foreign domain and (iv) resource authorization in a foreign domain linked with the home domain machine platforms bi-directional.

integrity measurement

The research methodology used in this research was divided into four different phases: (1) The framework foundation phase, (2) The design of framework phase, (3) The test-bed prototype implementation phase and (4) The assessment of the framework. The experiment result assessment suggests that the trusted computing bidirectional platform integrity measurement integration with the Shibboleth basic authentication mechanism: (1) Intensify federated organizations machine platforms security and trust and (2) Reduces the home domain machine platforms measurement privacy concern. In addition to that the assessment of the result also shows that sharing of machine platforms security credentials in inter-domain or intra-domain setup depends on the trust association (i.e., strong or weak). The newly created data connector - mutual integrity provider is flexible because in the future it can accommodate any other mutual attestation scheme. The mutual attestation performance measurement and benchmarking of client and server machine platforms shows an increase in the attestation time with the increase of in the number of stored measurement log. The increase in the attestation time because of the mutual attestation scheme (i.e., integrity measurement architecture) used in this work which heavily depends on the stored measurement log. The mutually attested machine platforms privacy is conserved in a foreign domain via trusted attribute - mutual platform integrity.

ABSTRAK

Identiti persekutuan dan sistem pengurusan akses memudahkan pengguna-pengguna organisasi *home domain* untuk mengakses pelbagai sumber (perkhidmatan) dalam organisasi domain asing melalui kemudahan laman sesawang *single sign-on*. Dalam persekitaran bersekutu pengesahan pengguna dilakukan pada awal sesi pengesahan dan dibenarkan untuk mengakses pelbagai sumber (perkhidmatan) sehingga sesi semasa menjadi aktif. Dalam identiti persekutuan semasa dan sistem pengurusan akses, ciri utama keselamatan yang perlu dititik beratkan adalah: (1) ketidak wujudan platform mesin ukuran integriti dwiarah di organisasi *home domain* (2) ketidak hadiran pengesahan bersepadu (iaitu, nama pengguna/ kata laluan dan platform penyaksian bersama mesin *home domain*) (3) sumber (perkhidmatan) kebenaran dalam organisasi domain asing tidak melalui platform mesin pengesahan dwiarah *home domain*. Tambahan pula, ketiadaan amanah dwiarah dalam platform mesin organisasi persekutuan boleh menyebabkan ancaman seperti *worms*, dan *phishing* melalui *Trojan*, *keyloggers* dan *rootkit*). Penyelesaian *Trusted Computing* (misalnya, modul platform yang dipercayai dan teknik pengesahan bersama) boleh membantu untuk mengatasi isu-isu keselamatan dan amanah platform mesin dalam identiti persekutuan dan sistem pengurusan akses. Walau bagaimanapun, penggunaan skim pengesahan bersama dalam persekitaran bersekutu boleh membawa kepada isu privasi ukuran platform mesin (tauliah keselamatan). Isu-isu yang dinyatakan di atas memberi motivasi untuk menjalankan kajian ini bagi membina keselamatan bersatu yang praktis, amanah dan penyelesaian privasi untuk organisasi bersekutu agar dapat bekerjasama dengan cara yang selamat, boleh dipercayai dan mutu privasi yang dipertingkatkan.

Dalam kajian ini *Shibboleth* telah dipilih untuk membina rangka kerja keselamatan, amanah dan privasi yang praktik dan bersatu. Penyelesaian yang dicadangkan (i) mengintegrasi teknik pengesahan bersama dengan mekanisme asas pengesahan pengguna *Shibboleth*, (ii) memastikan pembentukan amanah platform dua-arah antara pembekal identiti *home domain* dan mesin pelanggan, (iii) memastikan privasi platform mesin kelayakan keselamatan *home domain* dipelihara di domain asing dan (iv) memastikan kebenaran sumber di dalam domain asing dikaitkan dengan ukuran integriti dwiarah platform mesin *home domain*.

Kaedah kajian yang digunakan dalam penyelidikan ini telah dibahagikan kepada empat fasa berbeza: (1) asas rangka kerja, (2) reka bentuk rangka kerja, (3) ujian prototaip fasa pelaksanaan dan (4) penilaian rangka kerja. Penilaian hasil eksperimen menunjukkan bahawa integrasi ukuran platform dwiarah *trusted computing* dengan mekanisme asas pengesahan *Shibboleth*: (1) meningkatkan keselamatan dan kepercayaan platform mesin organisasi bersekutu dan (2) mengurangkan kebimbangan terhadap privasi ukuran platform mesin *home domain*. Di samping itu penilaian keputusan juga menunjukkan bahawa perkongsian kelayakan keselamatan platform mesin dalam persekitaran *inter-domain* atau *intra-domain* bergantung kepada *trust association* (iaitu, samada kuat atau lemah). Penyambung data yang baru diwujudkan - pembekal integriti bersama adalah fleksibel kerana pada masa akan datang ia boleh menampung apa-apa skim pengesahan bersama yang lain. Prestasi pengukuran dan penanda aras pengesahan bersama milik platform mesin pelanggan dan pelayan menunjukkan peningkatan dalam masa penyaksian dengan penambahan bilangan log pengukuran yang tersimpan. Peningkatan dalam masa penyaksian disebabkan oleh skim pengesahan bersama (iaitu, seni bina integriti ukuran) yang digunakan dalam kerja-kerja ini yang amat bergantung kepada ukuran log tersimpan. Privasi platform mesin yang saling disahkan adalah dipelihara dalam domain asing melalui sifat kepercayaan - integriti platform bersama.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACAs	Attestation Collector Agents
AIK	Attestation Identity Key
AVAs	Attestation Validation Agents
AuthN	Authentication
ASP	Authentication Service Provider
AuthR	Authorization
BIOS	Basic Inpput Output Service
BS	Blind Signature
BTGS	Blind Token Generating Service
CoT	Circle of Trust
CRTM	Core Root of Trust for Measurement
CS	Corroboration (Validation) Service
DC	Data Connector
DoD	Department of Defense
DoR	Department of Research
DS	Descovery Service
DDL	Device Driver Library
DAA	Direct Anonymous Attestation
EUSTPF	Emergent Unified Security, Trust and Privacy Framework
EK	Endorsement Key
XML	Extensible Markup Language
FAPs	Federated Authentication Providers
FIAM	Federated Identity and Access Management
FIM	Federated Identity Management
FD	Foreign Domain
HD	Home Domain
HTTP	Hyper Text Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Secure

IAM	Identity and Access Management
IMSs	Identity Masking Schemes
IdP	Identity Provider
IAIK	Institute for Applied Information Processing and Communication
IBAC	Integrity Based Access Control
IMA	Integrity Measurement Architecture
IMBs	International Business Machines
LDAP	Lightweight Directory Access Protocol
MA	Mutual Attestation
MIR	Mutual Integrity Resolver
MIPDC	Mutual Integrity Provider Data Connector
MPI	Mutual Platform Integrity
MdFs	Multi Domain Federations
OPs	OpenID Providers
OSAIS	Organization for Advancement of Structured Information Standards
PCRs	Platform Configuration Registers
PUSTPF	Practicable Unified Security, Trust and Privacy Framework
PrivacyCA	Privacy Certificate Authorihty
PBA	Property Based Attestation
PKI	Public Key Infrastrcture
RUP	Rational Unified Process
RA	Remote Attestation
RSA	Rivest, Shamir and Adleman Algorithm
SET	Secure Electronic Transactions
SHA-1	Secure Hash Algorithm -1
SSL	Secure Socket Layer
SAML	Security Assertion Markup Language
STP	Security, Trust and Privacy
SP	Service Provider
Shibd	Shibboleth Daemon
SSO	Single Sign-On

SaS	Software as a Service
SGML	Standard Generalized Markup Langauge
SML	Stored Measurement Log
TLS	Transport Layer Security
TC	Trusted Computing
TCG	Trusted Computing Group
TSS	TCG Software Stack
TCPA	Trusted Computing Platform Alliance
TPM	Trusted Platform Module
TTP	Trusted Third Party
TMAP	Trustworthy Mutual Attestation Protocol
USTPF	Unified Security, Trust and Privacy Framework
VRCERT	Validation of Received Certificate
VRN	Validation of Received Nonce
VRPCR	Validation of Received Platform Configuration Register
VRSML	Validation of Received Stored Meaurement Log
WSSO	Web Single Sign-On
WAYF	Where Are You From

CHAPTER 1

INTRODUCTION

1.1 Introduction

This chapter explains the motivation, research problem statement, objectives, scope, questions to be addressed, the adopted research methodology, and related research activities. This chapter is wrapped-up with research contributions and thesis organization.

1.2 Motivation

The Security, Trust and Privacy (STP) unification in a federated environment is challenging because of the unbalanced relationship among STP. In a federated setting, the home domain and foreign domain organizations are increasingly using the user’s identification information and the attributes in Authentication (AuthN) and Authorization (AuthR) processes. In such federated settings, the home domain users and foreign domain Service Providers (SPs) often put their trust in the home domain Identity Provider (IdP) for critical tasks such as home domain user AuthN, attribute resolution and their correct assertion. Cryptographic protocol techniques can be used to protect sensitive information while they are being transferred on top of communication links between: (1) the Home Domain user and IdP machines and (2) the Home Domain IdP and foreign domain SP machines. However, cryptographic protocols only provide communication link security but not a safeguard for the home domain IdP and the user’s machine platform against the theft of private information.

Malevolent programs, Trojans, viruses and worms, are the threats that cause the aforementioned risks. For example, in malware based attacks, the invader may deceive the home domain user or home domain IdP by installing software over their machines to remotely allow the invader to capture the keyboard inputs. In addition to that, the invaders may also be interested in acquiring the user's login credentials from the user's or the IdP's machines. Therefore, the home domain user's and IdP's machine platforms infection by malevolent threats may lead these machine platforms into a dishonest or un-trusted state.

The Trusted Computing [14], [127] based security and trust solutions will most likely overcome such issues in a federated setting. However, to establish mutual trust between the communicating machine platforms, these machines need to exchange their platform security credentials (i.e., measurements) which will probably lead to the platform privacy issue. Therefore, trusted computing based security demands the sharing of all private information which is related to the proof of ownership of the platform. On the other hand, privacy demands the minimal sharing of private information so that platforms cannot be traceable or linkable to a particular user, entity or data. Hence, among STP, there are some conflicting issues that have to be resolved and harmonized.

To demonstrate the STP unification challenge, consider a federated research collaboration scenario that consists of two organizations: (1) the Department of Defence (DoD) which can be called a home domain organization and (2) the Department of Research (DoR) which is also known as a foreign domain organization. The DoD: (1) is managed autonomously, (2) is responsible for the management of users and their machine platform registration, which involves AuthN and attestation, and (3) possesses privacy conserving related policies. The home domain consists of an entity call the IdP which is responsible for performing user AuthN. The foreign domain is a Resource Provider (RP)/ SP organization. However, there are concerns in such a scenario that include: (1) weak AuthN, (2) missing machine platform mutual attestation and (3) resource AuthR in the foreign domain is not on the basis of successful mutual attestation "Trusted Attribute". In addition to that, releasing of mutually attested machine platform security credentials (i.e., platform measurements) between the attesting machines raises platform privacy concerns. The STP concerns are further exemplified with the help of the following example:

Farida is a senior research officer in organization “A” and physically working in organization “A”. Farida has given consent specifically to her Principal Researcher (PR), to access her new innovative security software design and implementation document which resides at the organization “A”. The PR believes that Farida’s recent new innovation has the potential to have huge success and profit in the near future. The PR would like to “use” her consent to refer her design to an External Principal Researcher (EPR), who is also a notably experienced practitioner in that field, at organization “B” outside her home domain. It has also been assumed and expected that she would also give access consent to some of her previous product designs and test results to the EPR. In such a scenario, when the relevant documents are being released from organization “A” to organization “B”, the process of releasing should be in such a way that only those permitted persons (i.e., the EPR) at the organization “B” can access the documents. In this example the STP concerns are as below:

Security

- Existing solution permits organizations to authenticate within its own organization, i.e. organization “B” can only verify the local PR authenticity. In some situations, a PR may be authentic, but the machine (i.e., desktop/laptop) that he/she is utilizing and the Identity System (i.e., IdP machine) may not be in a trusted state.

- There may be non-existence of platform mutual attestation in its infrastructure, for example, between the IdP and PR in organization “B”.

- There may be non-existence of integrated AuthN and integrity verification schemes, for example, within organization “B”.

Trust

- Mutual Trust does not exist in the federated setting. In the federated setting, the user’s trust is within its own organization only. Similarly, the foreign domain puts its trust into the home domain IdP for AuthN, attribute resolution and the assertion tasks. Therefore, the absence of mutual trust between the home domain organization entities may lead to security risks.

- For example, in organization “A”, resource AuthR is not on the basis of organization “B’s” user’s and IdP machines’ successful mutual attestation.

Privacy

- The machines’ platform security credentials (i.e., integrity measurements) which are being exchanged between communicating machines, while it helps to strengthen trust, may also raise privacy concerns.

1.3 Problem Statement

The existing security measures such as anti-Trojans/malware/spam and the Secure Socket Layer (SSL), based on software based security solutions instead of hardware, [6, p. 134] cannot determine the communicating machine platforms mutual integrity measurement within the Web Single Sign-On (WSSO) schemes (i.e., trusted or otherwise) [1], [2], [3], [153]. In third party based federated identity and resource (service) access systems software based encryption and digital signature are utilized to secure the information in transit, guarantee identity credentials integrity and home domain IdP authentication to the foreign domain SP [11], but what about the communicating machine platforms mutual integrity measurement? Therefore, the absence of mutual trust (i.e., bidirectional platforms mutual integrity measurement, reporting and validation) in WSSO schemes could lead to security and trust concerns, and challenges [67], [122], [137], [4], [5], [7], [8], [9], [10], [11], [132] in the federated identity and resource (service) access management systems. Furthermore, the STP concerns in the federated identity and resource (service) access mode are described through a threat model in section 4.2.2. It is possible that the message integrity, in transit, may be protected [6] but the communicating platform health may not be trusted (i.e., infected by malware).

In third party based proxy WSSO scheme (e.g., Shibboleth) SSL is used to secure the communication channel in online transactions between the client-IdP, IdP-SP and client-SP. However, it is incapable to counter against malevolent code injection attacks such as attacker may inject dishonest portion of code onto the honest portion of code and service requester identity spoofing attack [154], [155]. Online E-commerce and federated identity and resource (service) access management system has one thing common that both make use of untrusted global network (i.e., Internet) in online transaction. In e-commence online transaction SSL is used to protect confidentiality of data (e.g., credit card numbers in online shopping) [6]. Whereas, in online federated identity and resource (service) access environment SSL is used to protect: (i) user basic authentication data (e.g., username/pwd) between the home domain client and IdP machine in untrusted network. However, in both cases communicating machine platforms integrity state always remains in a question (i.e., trusted or untrusted).

Jensen [8] provides a structured survey with many others on security, trust and privacy challenges in federated identity and resource (service) access management systems. In online federated identity and resource (service) access the prime security concern is an identity theft. Bertino et al. [156] commented that identity theft is hard to prevent in online federated resource (service) access scenario because the home domain client machine online presence is mandatory for an authentication to the home domain IdP machine (i.e., 24x7 online) and to access geographically federated resource (service) located in a foreign domain. Therefore, the UN/PWD pairs in the login process as well as stocking of the home domain users UN/PWD pairs at the IdP machine must be protected [4]. Madsen et al. [64] points that an identify theft flaw in a federated access scenario may allow an attacker to access unauthorized resources (services) in an unbroken chain. The misuse of authentication credentials and identity attributes stored at the home domain IdP must be protected [156] in away that only trusted machine perform such operations.

The federated identity and resource (service) access systems are not full filling the identity and attribute provider’s trustworthiness requirements [157]. In third party based true WSSO scheme the home domain client machine trusts the home domain IdP machine to correctly identify the authenticated users through an authentication mechanism. Whereas, the foreign domain SP trusts the home domain IdP machine for an authenticated user correct attribute fetching from attribute authority and assertion. In worst case scenario untrusted home domain IdP machine may behave dishonestly [11].

The basic privacy concern in all federated identity and resource (service) access system is entity (i.e., user identity or hardware identity) identity information [11]: (1) Utilization, (2) Compilation, (3) Stocking them and (4) Intra or Inter domain sharing. In this work privacy threat (i.e., discloser) for home domain IdP and client machine focuses on the protection and use of their platform security credentials by the home domain IdP and foreign domain machine.

In summary, neither the client, IdP and SP machines in the proxy true WSSO nor the client and SP machines in the native true WSSO:

- have the right mechanism to create mutual trust formation in between communicating platforms.
- can fight against any malevolent activity initiated by the invader.

In following the problem of the security and trust in federated identity and resource (service) access are discussed. Whereas later, the attestation scheme limitation and privacy concerns in federated resources (services) access enviornement are discussed.

- *Home Domain Dishonest Client Platform*: This refers to a situation in which a user is authenticated to the home domain in Shibboleth [12], [13], and [146] (or to the foreign domain in the native true WSSO scheme). In both native true WSSO and proxy true WSSO cases the home domain and foreign domain do not have any knowledge about a client’s machine platform integrity status (i.e., whether in a trusted state or not). So, such an un-trusted machine may lead to security threats (e.g., user credential theft).
- *Home Domain Dishonest Identity Provider Platform*: This refers to a situation in which the home domain IdP machine may be tampered by a malevolent activity. In Shibboleth [12] and [13], an open source Federated

Identity and Access Management (FIAM) based on the proxy true WSSO design, the entity IdP must always be online to authenticate the user and deliver AuthN and AuthR information to the SPs. So, such an infected IdP machine, which has become a dishonest machine and may carry out malevolent actions, may lead to a massive user login credential theft and the attacker may misuse them to access sensitive resources.

- *Foreign Domain Dishonest Service Provider Platform*: This refers to a situation in which the foreign domain SP machine may be compromised by an attack. So, a dishonest SP machine may become a stepping stone for an attacker to access sensitive resources or user credentials.

The mutual attestation and privacy limitations in trusted computing remote attestation schemes are:

- *Remote Machine Attestation Technique Limitations*: The two main limitations of the remote attestation scheme are: (1) it cannot mutually evaluate the client and challenger machines' health and (2) the concern on platform measurement privacy.

1.4 Research Objective

The main objective of the research is to develop a Practicable Unified Security, Trust and Privacy Framework (PUSTPF) for federated identity and resource (service) access management. To achieve this obective, a number of specific tasks have been defined:

- To select and specify STP aspects/ charactesistics (given in section 4.2.3), as the necessary building blocks for a PUSTPF.

- To select and extend the Remote Attestation (RA) protocol into a Trustworthy Mutual Attestation Protocol (TMAP) for the proposed framework.

- To solve the privacy issue in the TMAP of the proposed framework.

- To integrate TMAP with a user's basic AuthN mechanism in a federated resource (service) access environment for the proposed framework.

- To benchmark the performance (i.e., attestation time vs number of measurement logs) of the bidirectional integrity measurement technique with the existing solution.

- To construct a PUSTPF test-bed prototype that merges the selected STP aspects/ characteristics given below (for detail see section 4.2.3):

  - Integration of the TMAP with a user basic AuthN (i.e., username and password),

  - Mutual trust formation between communicating machines in a federated resource (service) access environment, and

  - Privacy conservation in the proposed TMAP.

1.5 Research Scope

The overall scope of the work is depicted in Figure 1.1 below.

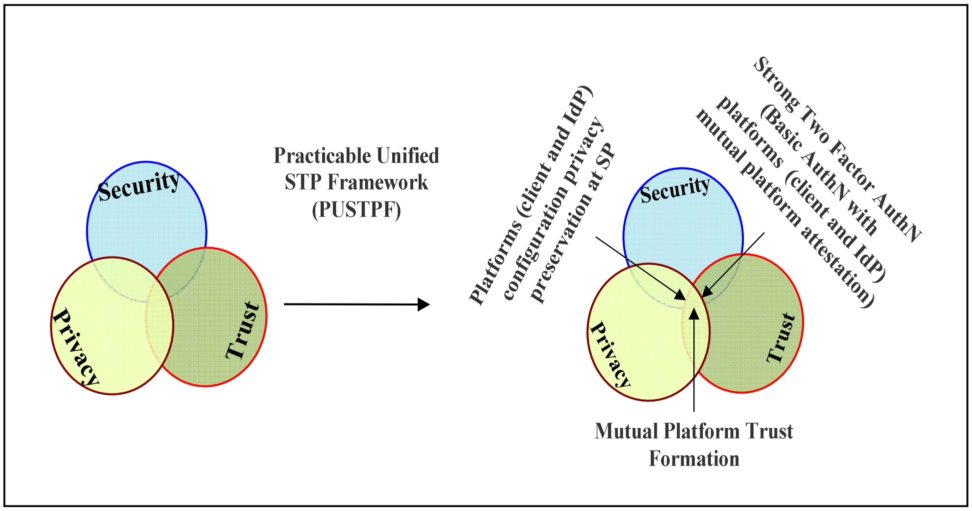

Security
Privacy
Trust
Practicable Unified STP Framework (PUSTPF)
Platforms (client and IdP) configuration privacy preservation at SP
Strong Two Factor AuthN (Basic AuthN with platforms (client and IdP) mutual platform attestation)
Security
Privacy
Trust
Mutual Platform Trust Formation

Figure 1.1: Scope of the research

The scope of the research is restricted to the following:

- The STP aspects (section 4.2.4), the building blocks of a PUSTPF.
- The existing trust aspects (section 4.2.3.2) in remote attestation schemes and to select and extend remote attestation for the TMAP for the proposed framework.
- The privacy aspect (section 4.2.3.3) of mutually attested machine (i.e., client and server) platform measurements in the TMAP for the proposed framework.
- The security aspect (section 4.2.3.1) in integrating the TMAP with a user basic AuthN mechanism (i.e., username and password) in a federated resource (service) access environment of the proposed framework.
- The mutual trust formation between communicating machines in a federated resource (service) access environment of the framework.

1.6 Research Questions

The following research questions are to ensure the relevance of the research focus in the federated resource (service) access environment.

- Why do we choose certain technologies/approaches/methods to build the proposed framework?
  - Why the Trusted Computing based security solution is chosen?
  - Why the hardware rooted secure and trusted component solutions are chosen?
  - Why the machine mutual attestation is used instead of the remote attestation?
  - Why is there a need for a practicable framework?

- What are the choices of architectures from the past research that can be adopted/ adapted works for the federated resource (service) access environment?

- What STP aspects (section 4.2.3) are essential/ desired to be part of a PUSTPF?

- How to integrate the proposed TMAP for it to work harmoniously with Shibboleth in the proxy true WSSO schemes/ FIAM mode?

- How can the AuthN and mutual attestation processes are implemented to ensure that home domain and foreign domain organization processes (such as AuthN, mutual attestation and AuthR) happen in a fully integrated and unified way?

- How can the privacy in a TMAP are mitigated for parties with conflicting conditions between two federated organizations such as: (i) in the proxy true WSSO scheme and (ii) in the native true WSSO scheme.

- How can all of the STP aspects (section 4.2.3) be implemented to realize the proposed PUSTPF?

1.7 Research Activities

The Fig. 1.2 below specifies detailed research activities in each research methodology phase (Fig. 4.1).

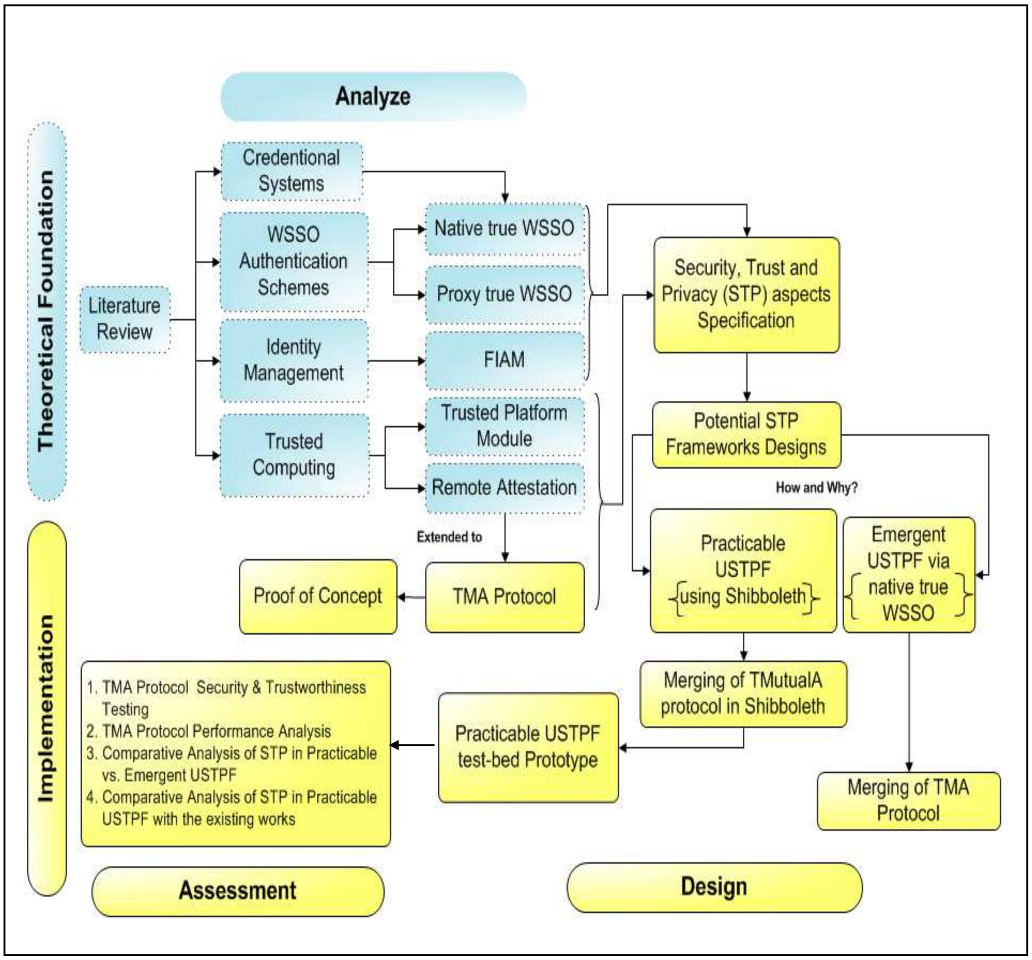

Analyze
Theoretical Foundation
Literature Review
Credentional Systems
WSSO Authentication Schemes
Identity Management
Trusted Computing
Native true WSSO
Proxy true WSSO
FIAM
Trusted Platform Module
Remote Attestation
Extended to
TMA Protocol
Proof of Concept
Security, Trust and Privacy (STP) aspects Specification
Potential STP Frameworks Designs
How and Why?
Practicable USTPF {using Shibboleth}
Emergent USTPF via {native true WSSO}
Merging of TMutualA protocol in Shibboleth
Merging of TMA Protocol
Practicable USTPF test-bed Prototype
1. TMA Protocol Security & Trustworthiness Testing
2. TMA Protocol Performance Analysis
3. Comparative Analysis of STP in Practicable vs. Emergent USTPF
4. Comparative Analysis of STP in Practicable USTPF with the existing works
Implementation
Assessment
Design

Figure 1.2: Research activities

1.8 Research Contributions

The contributions of this research work are:

- *Integration of Basic User AuthN Mechanisim with the TMAP*: Intigration of a TMAP with a Shibboleth user basic AuthN to validate the home domain user authenticity (i.e., by username/password proof of knowledge) as well as the home domain IdP and client machine platforms mutual integrity. To access a protected resource in a foreign domain the results of user basic AuthN and home domain communicating machine platforms integrity check must be true. In case of abortive user basic AuthN or machine platforms mutual imtegrity measurement the user is not authentic to access a protected resource in a foreign domain.

- *Home Domain IdP and Client Machine Platforms Privacy Conservation*: Providing a solution for the HD machine platform's measurement (i.e., platforms security credentials) privacy problem in the federated resource access environment by preventing the exposure of: (1) Home Domain IdP platforms security credentials to the clients, and (2) Home Domain client and IdP machines' platform security credentials to the foreign domains.

- *MutualIntegrityProviderDataConnector (MIPDC)*: Providing the own DataConnector (DC) (i.e., the MutualIntegrityProviderDataConnector (MIPDC) which is in-control of invoking Corroboration (i.e., Validation) Service (CS) located at the IdP machine to carry out platform mutual attestation.

- *Trustworthy Mutual Attestation Protocol (TMAP)*: Proposed TMAP solution for: (i) A non-Trusted Third Party (TTP), Emergent Unified Security, Trust and Privacy Framework (EUSTPF), federated identity and resource (service) access system and (ii) A Trusted Third Party (TTP), Practicable and Unified Security, Trust and Privacy Framework (PUSTPF), federated identity and resource (service) access management system.

- *Unified STP for Federated Identity and Access Management System*: Proposing and constructing a PUSTPF prototype which integrates the TMAP in the FIAM system (such as Shibboleth) which uses the proxy true WSSO scheme for user AuthN. It should be mentioned here that in the proposed solution, to access a resource in a foreign domain, four steps must be followed, namely: (1) Both AuthN verification and mutual platform attestation must be validated, (2) Mutual trust formation between communicating machines for resource (service) access, (3) Privacy conservation of client and IdP machines at the foreign domain and (4) Resource access in the foreign domain based on the "trusted mutual attestation attribute" in a federated environment.

- *Practicability of the Unified Framework*: The practicability of the proposed USTPF is very high. First, the framework makes use of an Open source FIAM system Shibboleth. So, each HD organization needs to set its own Shibboleth IdP and FD, and the Shibboleth SP entities, respectively. Second, only the HD IdP needs major changes to perform the HD clients' and IdP's machine platform mutual attestation which is quite viable and easy because both entities are members of the same network (internal).

- *Enhanced Security and Trust in the Home Domain Machine Platforms*: The security and trust between the HD client's and IdP's machines are enhanced because the TMAP mutually assures that the HD client and IdP machine platforms are secured and trustworthy.

- *Open Source Solution*: Open source and standard Trusted Computing technology based mutual attestation scheme notion proof of concept implementation is carried out for the native true WSSO scheme. In addition to that, the security and trustworthiness of the attested machine platforms is tested against a rootkit attack execution and the performance analysis is carried out of the TMAP in PUSTPF.

- *Test-bed Prototype Construction*: The PUSTPF test-bed prototype is constructed by combining the pros of the Shibboleth, an open source and a standard FIAM solution with the Trusted Computing standards and open source solutions (e.g., TC for the Java platforms, IMA and TPM).

1.9 Thesis Organization

Chapter 1 discusses the motivation, research problem statement, objectives, scope, questions, methodology, activities, and contributions.

Chapter 2 provides the background study of the research work. This chapter presents the concept of identity in a virtual environment. This is followed by an explanation on what Identity and Access Management (IAM) is, its basic processes, and the distinct IAM modes. It then describes different WSSO AuthN schemes, their related types and their differences. This chapter also explains the FIAM mode and its related elements, pros and cons, basic standards and technologies (such as, Security Assertion Markup Language (SAML) [17] and [18], Shibboleth and SSL), and the association between the federated mode and the proxy true WSSO scheme. The identity masking schemes such as the public key digital signature and the blind signature, their respective pros and cons are also explained. In addition, the Trusted Computing notions such as: (1) Trusted Platform Module [16], [125], [126], (2) Trusted boot, (3) Privacy Certificate Authority (PrivacyCA), (4) Attestation, and (5) open source trusted computing service packages for the Java (tm) platforms are included in this chapter. This chapter ends by describing the Integrity Measurement Architecture (IMA) [15], what is integrity, the Integrity objective and the architectural design of the integrity measurement.

Chapter 3 provides a critical review of the previous studies on the techniques, technologies and standards that are pertinent to the current research. It presents a detailed literature review with the sub-sections: (1) Elucidation on the distinct IAM mode failures and advancement, (2) An overview of trust and security in the FIAM mode, (3) A description of the related work allied to the unification of STP,(4) The related works on the identity masking schemes and the practicability of these schemes in a real environment, and (5) An explanation on the different machine platform attestation techniques, related challenges, practicality and the associated work.

Chapter 4 describes comprehensively the approaches, techniques and technologies employed in this work. This chapter presents the research activities leading to the implementation of the framework such as design considerations, architecture and implementation limitations for the federated identity and resource (service) access mode. This chapter also explains the four phases of the research methodology adopted in the thesis which include: (1) The framework foundation, (2) The design of the framework, (3) The test-bed prototype implementation, and (4) The assessment of the framework.

Chapter 5 presents the system design, architecture and system implementation. It explains the design of the system which includes the proposed system use-case, activity, class, the sequence diagrams, packages and classes and flowchat of algorithms. It then explains the architecture of the proposed system which covers: (1) The different entities or stakeholders of the system, (2) The PUSTPF, and (3) The comprehensive PUSTPF protocol architecture.

Chapter 6 provides the experiment outcomes. It describes the experimental desgin, test-bed technologies and implementation details, such as to demonstrate how the IMA based mutual attestation protocol is integrated in the Shibboleth framework. It also presents the TMAP proof of concept implementation result analysis obtained for the native true WSSO scheme, which includes: (1) The security and trustworthiness test results and (2) The TMAP performance results. This chapter also presents the PUSTPF test-bed result analysis. The results obtained from this test-bed experiments are: (1) The integrated user basic AuthN mechanism and the Home Domain machine's mutual attestation results, (2) The Home Domain client's and IdP machine platforms performance measurement and benchmarking, (3) The security and trustworthiness testing with the help of the home domain mutual attestation scenarios, and (4) The mutually attested machines' platform privacy conservation results. This chapter also provides: (1) Comparisons between the emergent and practicable frameworks, (2) Comparison of mutual attestation performance analysis and (3) Compression of the proposed works with the existing works.

Chapter 7 is the summary of the research work. This chapter presents the research conclusion, limitations and the potential future works.

1.10 Summary

This chapter presents the introduction, illustrates the motivation scenario, and the related problem statement. It then highlights the research objectives and the research scope. The research questions, research methodology, research activities, and research contributions are also discussed.

CHAPTER 2

BACKGROUND STUDY

2.1 Introduction

This chapter discusses the fundamental background study of federated identity and resource (service) access management systems. The identity concept in a virtual environment is discussed in the second section. The third section describes the IAM related basic processes and the evaluation of the IAM systems. The types of WSSO authentication schemes are explained in the fourth section. The fifth section explains the FIAM mode and associated FIAM elements, FIAM Pros, FIAM standards and presents a comparison of the federated and the proxy true WSSO schemes. The identity masking schemes and their practicability in a real environment is discussed in the section six. The seventh section describes Trusted Computing and its related concepts and technologies. The IMA, particularly such as what integrity is, the integrity objective and the architectural design of the integrity measurement are discussed in the eighth section. The ninth section summarizes this chapter.

2.2 Identity in a Virtual Environment

To access online resources (services) such as e-university resources, e-government and e-health services, users need to have an identity. The electronic virtual identity is assigned to a person or device upon successful registration with an organization’s authority [109], [128].

2.3 Identity and Access Management

To manage the user's virtual identities and allow access to the multiple services of an organization in a virtual environment, the diverse nature of IAM modes has been developed. Usually, each IAM system is a combination of a group of guidelines, processes and technologies [150]. The purpose of IAM is to create, manage and revoke user identities, as well as put into effect the organization's policies, which are enforced on each user after successful authentication and authorization. The basic processes of IAM (Figure 2.1) are [19]:

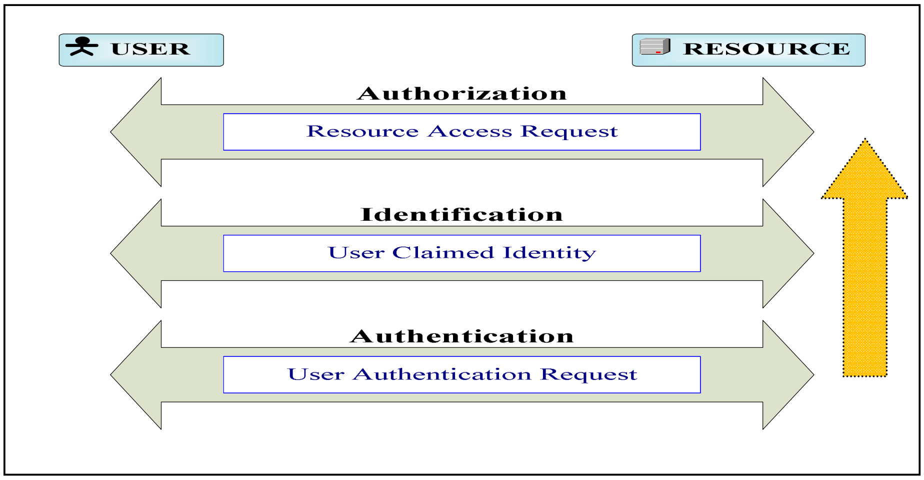


Figure 2.1: IAM basic processes

- *Authentication*: The AuthN process ascertains that the user is certainly the one who or what he/she/it claims to be and consists of three main phases such as: (1) presenting the credentials, (2) the claim, and (3) conceding the privileges. A user AuthN scheme maybe be a single factor or two factors such as something the user knows (i.e., secret) or possess (i.e., smartcard with secret). The passwords and digital certificates are the most common user AuthN mechanisms used in a virtual environment.

- *Identification*: Identification of a user is a single-time action and depends on two dimensions: (1) The user's information (e.g., the person's name, date of birth etc.) is assembled and is associated with the identity and (2) The certainty level with which the identifying attributes are added to a user's identity.

- *Authorization*: In the AuthR process the FD SP/RP organization fully depends on a user's HD organization for a valid attribute derivation. The FD SP then, on the attribute resolution, decides whether the HD user should be granted or not to access a protected resource (service) sited at the FD.

2.3.1 Identity and Access Management (IAM) Modes

IAM models have evolved and consequently more novel models have appeared [19], [20], [21]. The security, trust and privacy challenges in IAM modes are given in Table 2.1. Each model is described afterward.

Table 2.1: Security, trust and privacy in IAM modes

IAM Modes	**Trust**	**Platform Security**	**Privacy in Foreign Domain**
	Trust Association	IDtheft	User Attributes Association
Isolated	Strong	Low	Low
Centralized	Weak	Vulnerable	Vulnerable
User-centric	Strong	Vulnerable	Low
Federated	Strong	Vulnerable	Low

2.3.1.1 Isolated Mode

The isolated mode is the mode in which each isolated domain operates independently from other domains. In the isolated mode, threats related to user identity theft and privacy (e.g., association of the end user' attributes to the same user distinct identifiers) are negligible because these attributes are in the control of a solitary organization [4], [10].

2.3.1.2 Centralized Mode

In the centralized mode, a single user's credential and identifier are used by every SP (Microsoft.Net Passport [22], [147], [148], [149]). The centralized mode overcomes the issue of the inconvenience to link isolated systems. However, the security and privacy elements in the centralized mode are not promising because of the centralized control over the user's identity.

2.3.1.3 User-centric Mode

In this mode, the user is in power as to what information is to be disclosed and what not to be disclosed to the SPs [23]. In this mode, the problem of multiple credential and identifier management is eased and improved by storing them in a tamper resistant hardware device.

2.3.1.4 Federated Mode

In the federated mode [24], the HD is responsible for user registration and AuthN while the SP is responsible for resource (service) AuthR. The advantages offered by the federated mode are diversified well over any other kind of modes. The FIAM mode overcomes the issues in previous modes such as: (1) The inefficiencies in the isolated mode, and (2) The security and privacy issues in the centralized mode.

2.4 Web Single Sign-On (WSSO) Authentication Schemes

The web AuthN schemes are also known as Web Single Sign-On (SSO) AuthN schemes. The acronym WSSO will be used in the rest of this thesis. The two types of WSSO schemes are [3]: (1) the native true WSSO and (2) the proxy true WSSO.

2.4.1 Proxy True Web Single Sign-On

In the proxy true WSSO, the HD IdP acts as a broker between HD users and the FD SPs. This scheme allows the HD user to authenticate only once, via the SSO protocol, to the HD entity IdP until a session has expired. This scheme is described in Figure 2.2.

How the proxy true WSSO differs from the native true WSSO include:

- *Trusted Third Party (TTP) Involvement*: The main entity in proxy true WSSO scheme is the TTP entity (i.e., IdP). This enity is responsible for the home domain users AuthN, produce and dispatch the users attribute to the foreign domain SP.

- *Trust Association*: In proxy true WSSO scheme the HD users and FD SPs have a strong trust in the HD IdP. In this scheme: (1) the HD user trusts the HD IdP for AuthN, (2) the FD SP trusts the AuthN and AuthR assertions it has received from the HD IdP.

- *Open Source Package Availability*: The key proxy true WSSO schemes/ federated identity and resource (service) access schemes (e.g., Shibboleth etc.) are designed on open source and standard technologies.

- *Privacy Preservation Features*: The proxy true WSSO scheme strongly supports the privacy features [133]: (1) Pseudonymity and (2) Unlinkability. The pseudonymity means that the identity does not include any user's Personal Identifying Information (PII), while unlinkability means that the SPs are unable to work out which pseudonym belong the same user [25], [26].

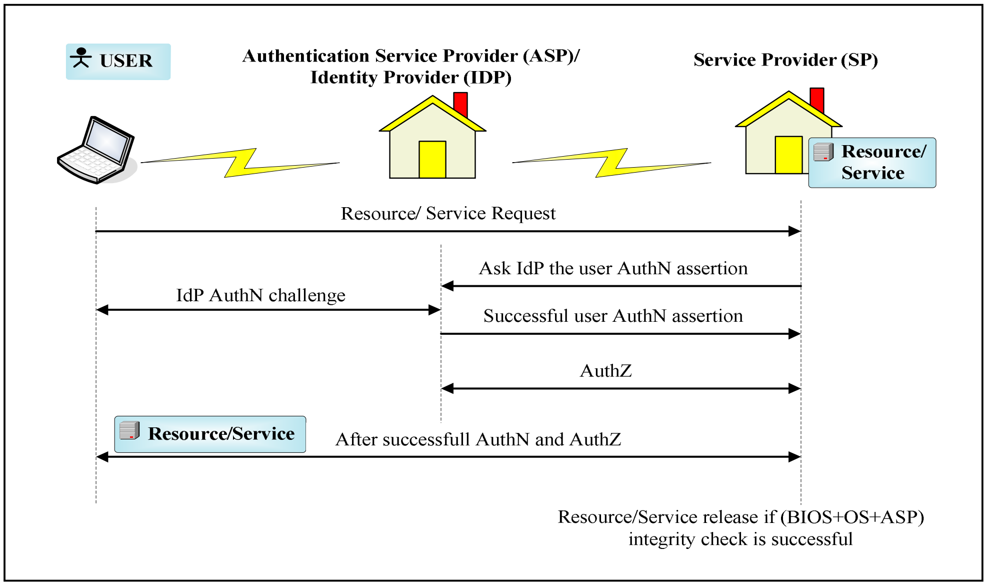

USER
Authentication Service Provider (ASP)/
Identity Provider (IDP)
Service Provider (SP)
Resource/
Service
Resource/ Service Request
Ask IdP the user AuthN assertion
IdP AuthN challenge
Successful user AuthN assertion
AuthZ
Resource/Service
After successfull AuthN and AuthZ
Resource/Service release if (BIOS+OS+ASP)
integrity check is successful

Figure 2.2: Proxy true WSSO scheme

2.4.2 Native True Web Single Sign-On

The native true WSSO consists of two main entities: (1) a user machine which is set with a Trusted Platform Module (TPM) and an Authentication Service Provider (ASP) and (2) an SP. The SP validates the client machines, trusted state on the basis of its platform security credentials. In this scheme the ASP's role is shifted to a trusted element (shown in Figure 2.3).

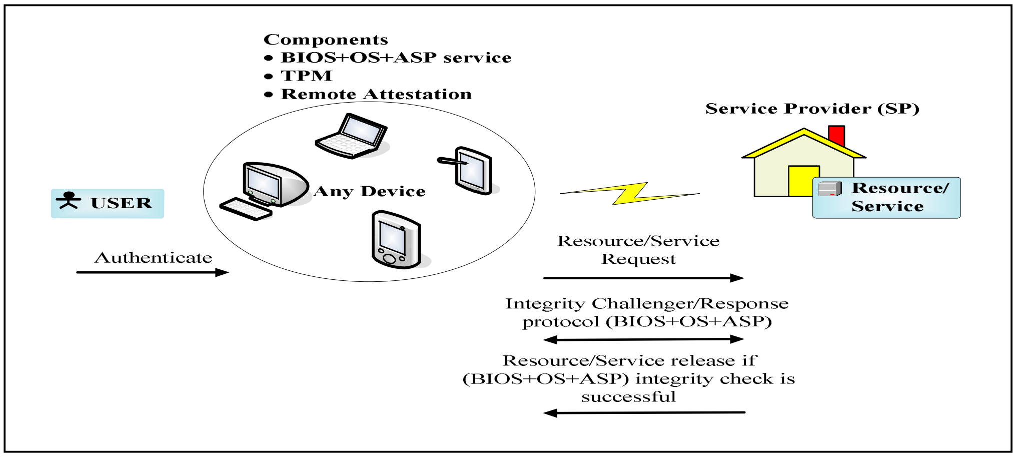


Figure 2.3: Native true WSSO scheme

The proxy and native schemes comparison is given in Table 2.2. How the native scheme is different from the proxy true WSSO are:

- *TTP Elimination*: In the native true WSSO scheme the user no longer relies on an external entity such as an IdP /ASP for AuthN.

- *TPM as IdP*: The TPM plays the role of an IdP. The challenger (e.g., server) validates the client machine's TPM legitimacy by checking that the received quote is really signed by a legitimate TPM or not.

- *Trust Association*: In this scheme, the SP trusts the PrivacyCA chosen by a user for certifying the Attestation Identity Keys (AIKs) and the client machine platform security credential measurement.

- *Open Source Package Availability*: The unavailability of open source packages is the biggest practical constraint of a native true WSSO scheme adoption in federated identity and resource (service) access schemes.

Table 2.2: Security, trust and privacy in WSSO schemes

WSSO Schemes	TTP	Trust Association	Trusted Computing Mutual Trust Link	Security IDtheft	Privacy Protection
Proxy True WSSO	Yes	Strong	No	Vulnerable	Low
Native True WSSO	No	Weak	No	Vulnerable	High

2.5 Federated Identity and Access Management (FIAM)

A FIAM mode is constructed on the notion of the identity federation [129]. The main purpose of the identity federation is the linkage of the HD user identities at a variety of FD SPs (Figure 2.4). An identity federation is a suite of three elements [158]: (1) business contracts (e.g., policies- method of AuthN, attributes and resource access etc.), (2) standards (e.g., OSAIS and SAML, Identity-FF, Web Service-Federation, and Identity-Web Service Federation) and (3) technologies (e.g., Shibboleth, Liberty Aliiance, Card Space and OpenID). The objective of these elements is to work together to enable an assemblage of FD SPs to identify the HD end user identifiers and the rights from other FD SPs inside of the group [21].

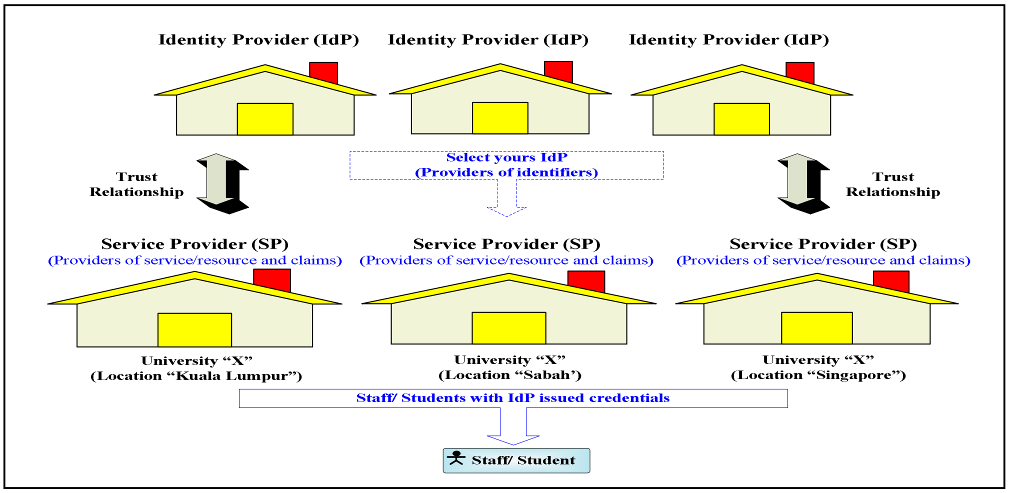


Figure 2.4: Federated identity and access management mode

2.5.1 Elements of Federated Identity and Access Management

The basic three main elements taking part in FIAM schemes are:

- *The Home Domain User*: In the FIAM mode, the HD user is also known as an HD subject/ principal.

- *The Home Domain Identity Provider*: In the FIAM mode, the IdP entity is also known as an Asserting Party (AP). The purpose of this entity is to authenticate the HD users and share their AuthN and AuthR assertions with the FD SP.

- *The Foreign Domain Service Provider*: In the FIAM mode, the SP entity is also recognized as a Relying Party (RP) and trust the HD IdP asserting information about an HD user. The SP then uses this asserting information to carry out the resource (service) access decision.

2.5.2 Pros of Federated Identity and Access Management

The Pros of the FIAM to the different stakeholders are:

- *Web Single Sign-On as AuthN Service*: The WSSO AuthN service permits the HD users to shift among distinct FD SPs.

- *Cut-down the Management Cost*: The FIAM reduces the management cost by transferring it to an organization who is a member of the federation. For instance, the SPs are free from the headache of users credential management.

- *Enhanced Scalability*: FIAM enhances the HD IdP and FD SP scalability such as: (1) The HD IdP can easily add a new user and at the same time revoke the existing user accounts without disturbing the HD basic infrastructure, (2) New FD SPs can easily join the federation and (3) It allows resource (service) access to a much greater number of users.

- *Element Association*: The HD IdPs are in close association with the HD users and FD SPs. This association has two pros: (1) The promotion and retail of extra resources (services) and (2) The ease of the billing acquisition of the FD SPs that they uphold.

- *Reduce Information Redundancy*: The roles of the HD and FD organizations in a federation are clearly stated: (1) The FD SP does not maintain the users' registration and AuthN information and (2) The HD IdP does not need to be anxious about the resource (service) AuthR information.

- *Privacy Conservation*: The HD conserves the user's identity privacy at the FDs.

2.5.3 Standards of Federated Identity and Access Management

To exchange the user's related information between the HDs and FDs in Inter domain scenario, the major FIAM schemes use the standard SAML protocol [17], [18]. Shibboleth is a FIAM mode standard [12], [13] which uses SAML protocol. These two standards are discussed in the following sub-sections:

2.5.3.1 Security Assertion Markup Language (SAML)

The OASIS (Organization for the Advancement of Structured Information Standards) is a non-profit international organization found in 1993 under the Standard Generalized Markup Language (SGML). The objective of the OASIS establishment is to promote the expansion and adoption of open standards for security and Web services. It is the OASIS which defines the SAML standards and security [18]. The SAML specification suggests a variety of security and privacy mechanisms [27] described below:

- Security in SAML
  - Integrity and confidentiality of messages may protected by using the HTTP over SSL 3.0 (such as HTTPS) or TLS 1.0 [6].

- Privacy of the SAML
    - User identity privacy is protected by of Pseudonymity at the FDs.
    - One-Time Identifier (transient identifiers) conserves user privacy.

The SAML uses XML [111] for exchanging user information among the HD IdP and FD SP. In other words, the SAML builds on the XML framework for exchanging user AuthN and AuthR data (information) in Inter-domain scenarios. The two SAML versions are: (1) SAML 1.0/ 1.1 [31] and (2) SAML 2.0 [28]. The SAML protocol can be divided into several main components as:

- *Assertions*: The assertion contains user security information placed between these tags: <saml: Assertion> ... </saml: Assertion>. The SAML assertions consist of three basic statements: (1) Authentication statement, (2) Attribute statement and (3) Authorization decision statement.

- *Protocols*: The HD and FD communicate with each other for the assertions through the SAML protocols. The SAML is consisting of the main protocols such as: (1) Authentication request protocol, (2) Attribute query protocol, (3) Authorization decision query protocol and (4) Single sign-out protocol.

- *Metadata*: The SAML makes is easy for the HD and FD to encode and present their essential configuration data using metadata files [29].

- *Bindings*: Bindings are used to map the SAML protocol messages onto the lower network communication protocols to transport the SAML assertions between the HD and the FD [30].

- *Profiles*: The profiles deal with the question of how SAML assertions, protocols, and bindings combine to achieve an SSO in between the HD and FD. Some examples of profiles are web browser WSSO and single sign out profiles.

2.5.3.2 Shibboleth

Shibboleth is a standard and complete open source FIAM software package. Shibboleth uses a standard SAML protocol for AuthN assertion and attributes exchange between HD and FD. The well-known body working on the identity federation is Internet2 with Shibboleth. The aim of the Internet2 project was to build federation of identities for academic institutions and their partners. The roles, features, pros and working of Shibboleth are described below:

a) Roles of Shibboleth

The four roles of Shibboleth architecture are [12], [13]:

- *Home Domain Principal (or User)*: The resource (service) consumer.

- *Home Domain Identity Provider*: The HD IdP architecture is drawn from SAML. In Shibboleth, a user AuthN and an attribute assertion are carried out in accordance to the SAML protocol specification [31], [32], [33].

- *Foreign Domain Service Provider*: The module “mod_shib” which is an Apache web server “plug-in” controls access to a protected resource. The Shibboleth FD, which consists of a daemon “shibd” and Apache module “httpd”, listens to AuthR requests from a web server.

- *Discovery Service*: Through the DS HD users can select their HD IdP to authenticate with. The FD, upon receiving a resource request, redirects a user to the HD to select the preferred IdP through the DS.

b) Features of Shibboleth

The main features of Shibboleth are:

- The HD IdP is responsible for users’ AuthN through any in-place AuthN method.

- The HD of an organization is the owner of its users’ identity information.

- The FD SP trusts the HD IdP for assertions, while the HD user trusts the HD IdP for AuthN and privacy conservation.

- The FD in a federation is free from the burden of user administration and the HD is free from the AuthR decisions.

c) Pros of Shibboleth

The pros of Shibboleth are described below:

- The Shibboleth AuthN mechanism has numerous advantages: (1) WSSO AuthN across multiple FDs, (2) Each organization may adopt different AuthN mechanisms and (3) It eliminates to authenticate a subject at every remote SP instance.

- In the Shibboleth architecture the HD IdP releases limited amounts of user information to the FD SPs for AuthR decisions which certainly conserves the users' privacy.

- Shibboleth provides flexible AuthR management which makes use of users or group identifiers. This enables the FD SPs to: (1) Leverage multi-grained access control, and (2) Provide resource access constraint on the basis of the users' attributes.

d) Workings of Shibboleth

Shibboleth offers resource or service access in Inter and Intra domain scenarios. The HD IdP and FD SP use the metadata files, with many others, to exchange the AuthN and AuthR data. These metadata files are usually XML files. The purpose of these metadata files is to precisely identify the providers' domains. Shibboleth works (Figure 2.5) as below:

A HD end user requests a protected resource sited at the FD SP (step-1). To access a protected resource the FD SP requires the HD IdP user AuthN and AuthR information. Therefore, the HD end user is redirected to the "Discovery Service (DS)" to choose his/her HD IdP entity using a Graphical User Interface (GUI) (step-2). The HD IdP presents a login page to the user, which depends on an AuthN mechanism. The HD user submits his/her basic AuthN credential that he/she acquired from the HD IdP in the registration process (step-3, 4 and 5).

At this point, if a HD user is successfully authenticated, the HD IdP then establish a session (S) and create an authentication handle (H) for the authenticated HD user (step-6, 7 and 8). The "H" communicates by a user agent (browser) and passes on the "H" to the FD (step-9). The FD SP employs this "H" to ask for an HD end user's "attributes" from the HD IdP machine (step-10). After fetching the "attribute" from the HD machine, it then is sent to the FD SP (step-11). The requested attribute is passed through a sequence of steps such as: (1) Pulling the attribute as found in a system of catalog such as "LDAP", (2) Assigning specific protocol encoders, and (3) Finally, it is ready to be filtered for the propose of releasing (step-12).

Shibboleth HD IdP machine does not keep any HD user's associated attributes. For this purpose, it depends on the outer data storehouses such as the LDAP storehouse. The IdPs release an authenticated user attributes to the FDs for the purpose of AuthR, associated to the FDs' SP policies (i.e., who can access a protected resource) (step-13 and 14). The aim of these policies that the FDs are interested in what types of attributes about a particular home domain authenticated user, in the form of a metadata file, for an access control policies. The HD IdP employs the "Attribute-Filter" which is an XML file that contains attribute descriptions such as the attributes that are released to the FD. The information requested for AuthR is passed to the web service application in the HTTP header form such as via the Apache mod_shib adapter.

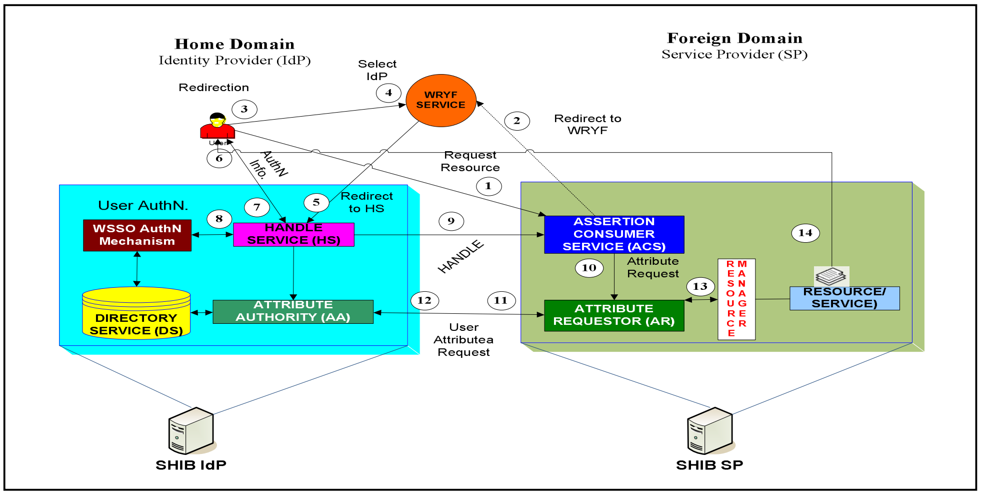


Figure 2.5: Shibboleth architecture

2.5.4 Federated Mode and Web Single Sign-on Relationship

In the FIAM mode, the HD IdP is responsible for collecting and maintaining the integrity of a person’s virtual identity information and the FD SP for resource (service) access which the HD users are of interest in using it. The user’s virtual identity information may be any electronic information which is possibly associated to a real physical person to identify this particular user [34]. The purpose of the FIAM mode is to manage and control the releases of users’ personal information in any type of electronic communication in a federated environment [35].

Whereas, WSSO is an AuthN service, particularly the proxy true WSSO, used by the FIAM to authenticate the HD user and then exchange the AuthN and AuthR information with the FD SP. The SAML [17] works as a main inter-domain protocol to exchange the AuthN and AuthR information between the HD and FD. Shibboleth, an example of a federated mode, which will be discussed next, provides the HD users access to multiple resources (services) sited at FDs via the WSSO facility. These shows that the proxy trues WSSO and the federated mode are related concepts because they are associated with one another.

2.6 Identity Masking Schemes

In a virtual environment, every subject (user) has some kind of credential (e.g., username and/or password) to be authenticated and identified by for accessing resources (services). To access a protected resource (service) the users must demonstrate the ownership of these credentials to the interested parties (e.g., IdP, ASP or SP which is linked to the trust association between the users and the parties). The purpose of the identity masking scheme is to mask the resource (service) consumer identity at different parties (i.e., maybe internal or external to the user). The two existing identity masking schemes are: (1) the multi-show and (2) the extended one-show (in which identity masking is achieved via a blind signature [36].

In the multi-show scheme, the possession of credentials can be demonstrated a random number of times without being linked and would not compromise the user’s identity anonymity. While, in the extended one-show scheme, the user’s identity anonymity and unlinkability is achieved through the adoption of a blind-signature. In emergent unified security, tust and privacy framework (section 4.3.1.1), the focus is on the blind signature practicality in FIAM mode. The public key digital signature is an example of a one-show credential scheme; its cons, blind signature and advantage over the public key digital signature, and how the user identity privacy is conserved in the one-show scheme through a blind signature scheme is discussed in the following two sub-sections:

2.6.1 Public Key Digital Signature Scheme

The conventional public key digital signature scheme consists of a: (1) private signing function (S), and (2) public verifying predicate (V) [90]. The private signing function (S) is known only to the signer. Whenever a message (m) is sent, it is first provided to the signer to get a signature on this message (m), e.g., the S (m). Next, the verifier performs a validation procedure to check the V (m, S (m))’s validity. The main problem in this scheme is:

- It cannot produce a signature (S) on the message (m), such as S (m), without the knowledge of the signer. This issue certainly raises a mesaage (m) privacy concen because the signer can know the message (m) detail.

2.6.2 Blind Signature Scheme

The blind signature scheme [36] is based on an RSA digital signature [37]. The blind signature is the enhancement of a conventional system with blind and unblind functions (e.g., BF and BF-1). In this scheme, whenever a user wishing to sign a message (m) he/she first blinds the message using the blind function (e.g., BF (m)). Next, the user sends the blinded message to the signer to get a signature on it. After receiving the blind message, the signer signs it and returns the S(BF (m)) to the user. In the end, the user unblinds the message using the unblind function to obtain the signature on the message (e.g., BF-1(S(BF (m))) = S(m)). The advantage of this scheme over the scheme in section 2.6.1 is:

- The blind signature scheme is used to sign data (e.g., electronic data). This data can later be authenticated in a manner without disclosing a user's identity. In the blind signature, the signer: (1) has no knowledge of what is in the messages they signed, or (2) about the signatures that were obtained by the receivers for their messages.

2.7 Trusted Computing (TC)

In the year 2000, the trusted computing or trusted platform step was initiated by the Trusted Computing Platform Alliance (TCPA) [38]. It is recognized as TCG [14]. The TCPA's basic notion was to introduce or place trust into the computing platforms (Figure 2.6) through an embedded TPM chip [16]. The Core Root of Trust for Measurement (CRTM) is the first component to run during the boot process. This CRTM may be physically located inside the TPM or externally. However, in either case, it is always in a trusted state according to the TCPA specification.

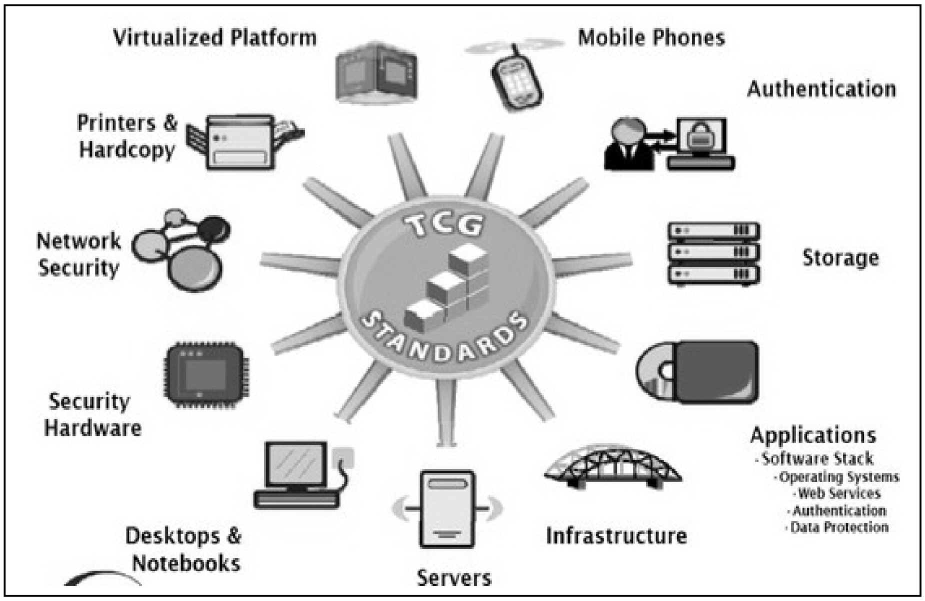
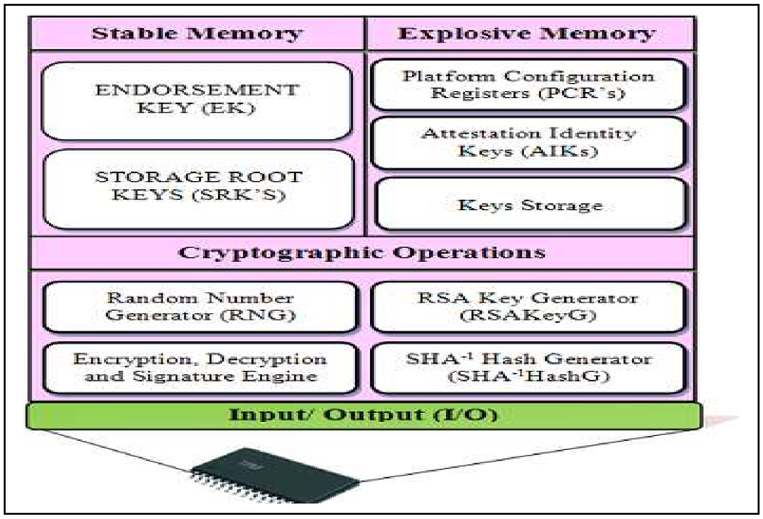


Figure 2.6: Trusted computing group standards [14]

Another reason behind the adoption of trusted computing is of the conventional security mechanism’s continuous failure to protect computer systems against malevolent software. The TCG notions related to this work are discussed in sub-section 2.7.1 to 2.7.4.

2.7.1 Trusted Platform Module (TPM)

TPM is a small coprocessor chip that performs mixed security functionalities (Figure 2.7): (1) private key protection like a smartcard, (2) RSA key generation, (3) signature generation etc. The TPM also has many components which consist of among others: (1) Endorsement Key (EK)-a manufacturer built-in key to uniquely identify a particular machine TPM, and (2) Attestation Identity Keys (AIKs) (pseudonym keys)-which helps to protect platform built-in key privacy.



Stable Memory
Explosive Memory
ENDORSEMENT KEY (EK)
Platform Configuration Registers (PCR`s)
Attestation Identity Keys (AIKs)
STORAGE ROOT KEYS (SRK’S)
Keys Storage
Cryptographic Operations
Random Number Generator (RNG)
RSA Key Generator (RSAKeyG)
Encryption, Decryption and Signature Engine
SHA^{-1} Hash Generator (SHA^{-1}HashG)
Input/ Output (I/O)

Figure 2.7: Trusted platform module

In addition to the TPMEK and TPMAIK, each TPM also contains Platform Configuration Registers (PCRs). The main function of PCRs is to verify the anticipated machine (i.e., target) platform configuration to the challenger machine. Each PCR register is made of 20 bytes (160bits) broad which holds a specific machine hardware & software “condition” hash (i.e., using SHA-1 scheme) digest. For instance, PCRs are competent to accumulate a wide-range of entities such as BIOS, BOOT LOADER, Kernel and Application measurement (i.e., cryptographic hashes) using the SHA-1[39]. PCR_EXTEND can be used for PCR manipulating. This is a two step operation given as:

- In step-1, the current VALUE (i.e., measurement value of component) will be accrued in the PCR, and then the hash of this VALUE is added on to the existing PCR VALUE.
- In step-2, the ensuing structure of the SHA-1 is recorded in a similar PCR.

Later in the validation process, the accrued measurements are presented to the challenger. This is to confirm that a machine's platform trustworthy configuration is as proof to which the TPM belongs.

2.7.2 Trusted Boot

The trusted boot supports the remote machine platform attestation (AuthN). On a target machine (the machine which wants to attest its platform integrity states to the challenger), the first trusted base is the "immutable base", the CRTM, which may reside in BIOS and is always trusted. Whenever the target machine is powered on the control, it is always transferring to the "immutable bases" and is the first piece (i.e., CRTM) to be executed. The "immutable base" then measures itself and then the BIOS and stores the computed SHA-1 result over the BIOS content in the TPM. This mechanism is then utilized recursively, creating a boot time chain root of trust as shown in Figure 2.8, to the next component until the OS has been loaded.

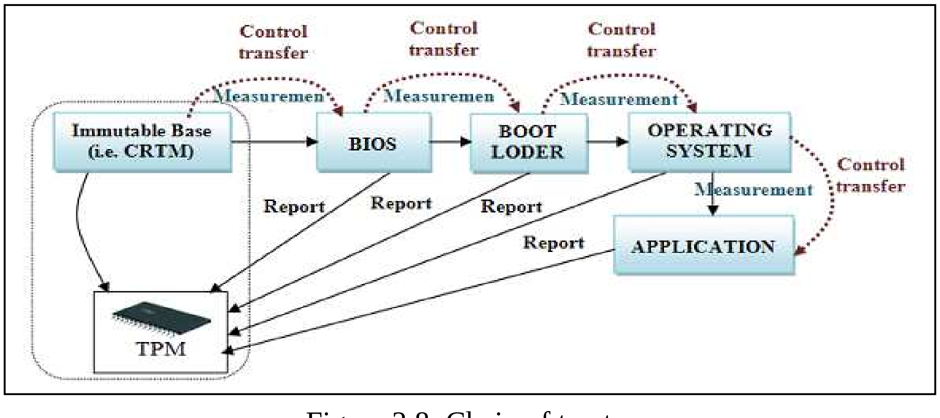


Figure 2.8: Chain of trust

2.7.3 Privacy Certificate Authority (PrivacyCA)

The PrivacyCA is a trusted TTP. The PrivacyCA needs to have knowledge about each organization machine's valid $\mathrm{TPM}_{\mathrm{EK}}$ (i.e., $\mathrm{TPM}_{\mathrm{EK}}$), an RSA key pair, and the public part (i.e., $\mathrm{TPM}_{\mathrm{EKpublic}}$) of the keys. The $\mathrm{TPM}_{\mathrm{EK}}$ key cannot be used due to privacy and that is the reason to authenticate this particular TPM to an entity called a verifier. So to protect the EK privacy, the pseudo (i.e., AIK and RSA key pair) key is used to obtain a certificate for the newly generated $\mathrm{TPM}_{\mathrm{AIK}}$. Through this certificate, the TPM then proves its authenticity to the verifier with respect to the AIK.

2.7.4 Attestation

In general, attestation is analogous to signing a peace of document, such as degree, certificate etc. The signing process represents the authenticity and geniuses of a respective document that it is issued to 'X' by the authority 'Y'. However, to attest a computing platform electronically and remotely, the TC novel concept called the "Remote Attestation (RA)" is utilized. The remote attestation technique is used to affirm two things: (1) That the remote machine contains a genuine TPM and (2) Its corresponding machine platform integrity is not compromised. The TCG remote attestation scheme is discussed in sub-section 2.7.4.1 to 2.7.4.2.

2.7.4.1 Remote Attestation

Remote attestation is a technique through which a client machine validates its platforms (i.e., hardware and software) measurement to a remote machine (also known as a challenger). The main objective of this technique is to let the remote machine ascertain the degree of trust in the attestor machine on the basis of the measurement health status. The remote machine platform attestation architecture is made of two key elements: (1) the IMA [15] or any other attestation scheme and (2) the remote machine platform attestation protocol. The two main entities taking part in a remote attestation scheme are: (1) a challenger machine, and (2) a target machine.

In remote attestation, the challenger is an entity that challenges the target machine for its platform authenticity. The protocol used among the challenger and target machines is called the integrity challenge-response protocol. In a simple remote attestation process, the challenger (e.g., a server machine) challenges a remote target (e.g., a client machine). The target machine then collects the requested component's integrity and returns it to the challenger. The integrity measurement process between the target and the challenger machines is depicted in Figure 2.9.

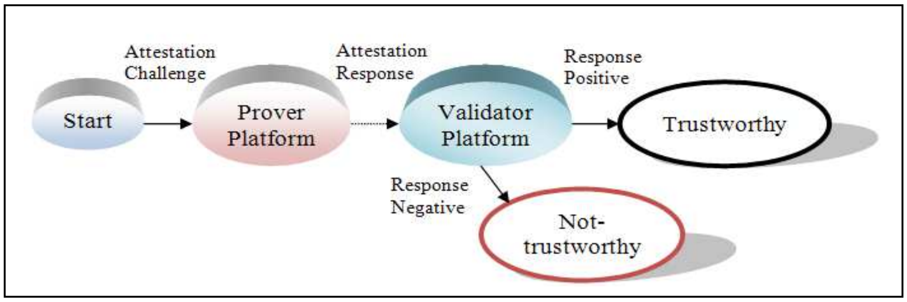


Figure 2.9: Remote attestation technique

The machine platform integrity process consists of a series of measurement steps starting with the CRTM. The CRTM consists of a boot-strapping process which measures the next component (e.g., boot-loader, OS, applications etc.) in the chain and adds the measurement value in to the TPM. In this manner, each component is measured and the measurement is added to the TPM before it is executed. The malevolent software cannot conceal its existence in a machine platform. The reason for this is that after storing the values in a TPM, it cannot be rolled back until the platform is rebooted. The IMA based attestation mechanism, which is the most practicable attestation mechanism, is utilized in this work and is discussed below.

2.7.4.2 Mutual Attestation

The traditional remote platform attestation approach supports only the remote machine attestation (or remote AuthN) of a specific target machine. The problem with this approach is that it is unable to carry out the attestation (or check the authenticity) of a challenger machine. The lack of mutual attestation in between communicating machines may bring along several machine platform security and trust issues. The work presented in this thesis solves the existing issues by extending the remote platform attestation protocol into a mutual machine attestation protocol.

This mutual attestation technique allows the communicating machines: (1) to build a mutual trust and (2) to validate each other's machine platform trustworthiness' state. The client and the server machines in a federated research collaboration scenario (section 1.2) will use this mutual attestation protocol to prove each other's platform trustworthiness. The flow of the mutual attestation technique is given in Figure 2.10.

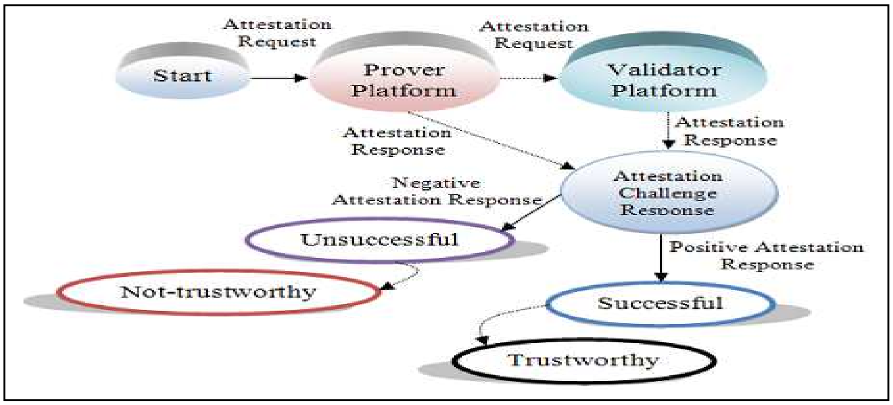


Figure 2.10: Mutual attestation flow

2.7.5 Open Source Trusted Computing (TC) for JAVA

The trusted computing for Java platforms was initiated by the Institute for Applied Information Processing and Communications (IAIK) [113], Graz University of Technology, supported by the European Commission as part of the OpenTC project. The objective was to develop trusted computing services for Java software developers. The main packages developed by IAIK for Trusted Computing for the Java Platforms are [40]: (1) jTSS-TCG Software Stack, and (2) jTPM Tools etc. These packages are described subsequently in the followings:

- *jTSS*: The jTSS package is compliant with the TCG software stack which is the key element of Trusted Computing platforms. The key features supported by jTSS are the TSS Device Driver Library (DDL), TSS Core Services and TSS Service Provider (SP).

- *jTPM*: The jTPM Tools consist of command line tool sets which represent communication with the TPM and TSS. The jTPM Tool package includes taking and clearing the TPM ownership, releasing of current PCR registers, Extending the PCRs and the creation of quotes etc. The basic requirements for utilizing jTPM Tools are: (1) Real TPM hardware chip or TPM emulator, (2) Root level (administrative) permission consent, (3) Sun Java ver. 5 or above, (4) Running of the TCG compliant jTSS on a machine, and (5) TC cert bundled in jTPM Tools 0.7.

2.8 Integrity Measurement Architecture (IMA)

In future federated identity and resource (service) access systems, mutual trust formation among communicating stakeholders/entities is becoming highly imperative because of the emerging security threats (e.g., Malwares and Trojans) to the computing platform.

2.8.1 What is an Integrity

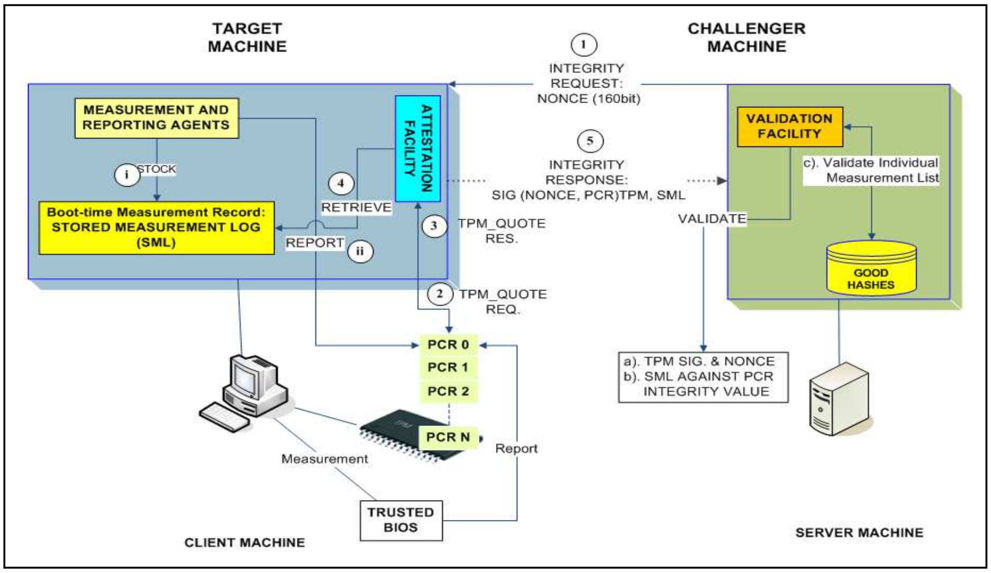
In general, integrity refers to a particular program binary possession which represents this particular program's secured state or the defense capability against unauthorized alteration. For some programs' integrity, contagion means the program association with the low integrity data [139] and to others, certification of the high integrity data [140]. However, in practice, both meanings have their own limitations [139], [140]; such as, in the first case, the programs frequently run the low integrity data devoid of being infected and in the second case, the application certification can become extremely costly [15]. In addition to the above, IBM introduced the IBM 4758 in which the program integrity is ascertained by means of the program code [141]. However, the disadvantage of the IBM 4758 is the unavailability or impracticality of these aspects in current machines [15]. To overcome such challenges, Sailor et al. [15] introduced "IMA", a more flexible and practical integrity measurement scheme for computing platforms.

2.8.2 The Integrity Objective

The main goal of the integrity is to demonstrate to the challenger machine that a program integrity running on a target machine is sufficient to make use of a resource (service) provided by the challenger. The program integrity is a binary possession. Therefore, utilizing the program integrity, the challenger can detect the unauthorized alteration of a program.

2.8.3 The Architectural Design of the Integrity Measurement

The overall architectural design of the integrity measurement (Figure 2.11) consists of three fundamental units: (1) The Integrity Measurement Acquiring, (2) Integrity Challenge/ Response and (3) Integrity validation. These units are described in the sub-section 2.8.3.1 to 2.8.3.3:



TARGET MACHINE
CHALLENGER MACHINE
1
INTEGRITY REQUEST: NONCE (160bit)
MEASUREMENT AND REPORTING AGENTS
ATTESTATION FACILITY
VALIDATION FACILITY
5
INTEGRITY RESPONSE: SIG (NONCE, PCR)TPM, SML
c). Validate Individual Measurement List
i
STOCK
4
RETRIEVE
Boot-time Measurement Record: STORED MEASUREMENT LOG (SML)
3
TPM_QUOTE RES.
VALIDATE
REPORT
ii
GOOD HASHES
2
TPM_QUOTE REQ.
PCR 0
PCR 1
PCR 2
PCR N
a). TPM SIG. & NONCE
b). SML AGAINST PCR INTEGRITY VALUE
Report
Measurement
TRUSTED BIOS
CLIENT MACHINE
SERVER MACHINE

Figure 2.11: Integrity measurement architecture

2.8.3.1 Integrity Measurement Acquiring Phase

The IMA is based on the boot time measurement. In the boot time measurement, whenever an IMA and TPM enabled machine is powered on, the machine takes the measurement of all the loaded executables till the last application is loaded. The integrity measurement notion steps are: (1) First, the BIOS measures (i.e., the components integrity) the Boot-loader, (2) The Boot-loader then measures the OS kernel and (3) The OS kernel then measures the loaded applications. The IMA actually maintains the loaded components history or record in an in-kernel ML. The ML holds the text report and the loaded measurement components' corresponding hash entries. The measurement is put into practice as a 160bit outcome calculation in the form of a SHA-1 hash function. The hash function is applied to the files during the booting of the system. The file actually holds the data or executables which are loaded during the booting process. To protect the in-kernel ML, the TPM is used to keep the integrity value over the in-kernel ML. The TPM protects the in-kernel ML integrity and the challenger can sense the unauthorized alteration to the in-kernel ML during the integrity measurement validation step. The TPM uses the Platform Configuration Registers (PCRs), which are the protected data registers, to keep the integrity validation value over the total number of measurements taken during the booting process. Therefore, the TPM PCR (e.g., 10 or 13) aggregates any measurement using the TPM_EXTEND function. For instance, if j measurements have been taken (i.e., M1, M2, M3, M4…Mj) then the aggregation of these in selected the PCR are: SHA-1(…SHA-1(SHA-1(SHA-1(SHA-1(0||M1) ||M2) ||M3) ||M4)…||Mj).

2.8.3.2 Integrity Challenge/ Response Phase

The challenging party uses the integrity challenge/response protocol to retrieve the current ML and TPM_QUOTE. The steps involved in the integrity retrieval and the validations are described here:

- The challenger machine creates a 160-bit non-inevitable nonce and pushes it in the integrity challenge request message towards the target machine.

- After receiving the attestation challenge request, the target machine prepares the response which includes two outcomes: (1) Signed TPM_QUOTE and (2) Ordered ML. First, the target machine loads the AIK, a 2048-bit RSA key, into the TPM chip which is known only to this TPM. For the TPM AIK public part, a certificate is obtained from the PrivacyCA. The generated signature actually associates the PCR signature to the particular machine TPM. The TPM_QUOTE is generated by signing the chosen PCR and nonce received in the integrity challenge request with the TPM AIK key private part. The ordered ML and the signed TPM_QUOTE are then returned to the challenge machine as an integrity challenged response.

2.8.3.3 Integrity Validation Phase

To validate the target machine's integrity, the integrity validation unit consists of the following steps:

- The nonce validation is received in the integrity challenge response. This is to ensure that the QUOTE is not a replay attack by an infected machine.

- The TPM_QUOTE signature validation determines two things: (1) The honesty of the QUOTED PCR values and (2) The QUOTING TPM authenticity that it is really the one on the attested machine.

- The Boot aggregate computation and evaluation is added to the initial measurement of the ML. This is to ensure that the boot aggregate is not tampered with. In addition to that, individually, each and every measurement in the ML is validated against known good hashes in the Data Base.

- To validate that the MLs have been tampered or not, the ML PCR value is virtually recomputed. This process is started with the initial measurement of the ML until the ML is consumed. The outcome of the virtual PCR value is then compared with the signed TPM PCR value (i.e., PCR-10). If they do not match then the target machine attestation fails.

2.9 Summary

This chapter presents the background study which covers the nature of the technologies and notions that are used in this work. This chapter consists of nine sections. The first section provides the introduction. The identity in the virtual environment is discussed in the second section. The identity and access management, related basic processes and distinct modes are presented in the third section. The difference between the WSSO authentication systems is discussed in the fourth section. The fifth section coveres the FIAM mode related elements and standards, and a comparison between the federated mode and the proxy true WSSO. The identity masking schemes and their practicability are discussed in the sixth section, particularly the blind signature scheme. The trusted computing and integrity measurement architectural notions and technologies are discussed in the seventh and eight sections, respectively. These sections also explain how IMA and TPM integration can bring mutual trust and security among communicating machine platforms. This chapter is summarized in the ninth section.

CHAPTER 3

LITERATURE REVIEW

3.1 Introduction

This chapter provides the pertinent allied research work. The first section presents the introduction of the chapter. The literature review related to this work is presented in the second section. The overall literature covering the related work to this research area described in sequence in the five sub-sections. The first sub-section describes the IAM mode failures and advancements. The trust and security in FIAM is described in the second sub-section. The unification of security, trust and privacy is discussed in the third sub-section. The fourth and fifth sub-sections, respectively explains the practicability of the identity masking scheme and the remote attestation techniques. The analysis of the past works, and the security, trust and privacy unification is discussed in the third section. The fourth section summarizes this.

3.2 Literature Review

3.2.1 Identity and Access Management (IAM) Mode

The IAM concept is not new. Many offline IAM modes have been operating for decades. The examples of common and widespread offline IAM modes include country passports, National Identity Cards (NICs) and driving licenses issuers. Different online IAM modes are used. Online IAM modes evolved from isolated to federated form and are classified into five basic modes: (1) Isolated, (2) Centralized

(3) Distributed, (4) User-centric and (5) Federated [19], [21]. In online IAM modes, the security, trust and privacy challenges involve are: (1) User identity credential security concerns (i.e., safe or not against online threats), (2) Trust in the user (client) and home domain server (IdP) machine platforms, that they are really in trustworthy state (i.e., the users credentials are protected), and (3) User identity privacy concerns (e.g., some one knows the real identity and which resources (services) are being accessed). This section presents the reasons why the earlier IAM modes (i.e., isolated, centralized and distributed) were failed and the needs for advanced IAM modes.

Table 3.1: IAM modes and technology aspects

Technology Aspects	**IAM Modes**			
	Federated	**User Centric**	**Centralized**	**Isolated**
AuthN.	IdP	Through IdP	Single major record	Every occasion
AuthR.	SP	SP	centralized	Every occasion
Identity locality	IdP	User selected IdP	Stored in a major Record	Detach SPs records
Trust Association	Strong at IdP	Account divisions	User dependent on SP for security and privacy	Strong
Linkability	Not linkable	Not linkable	Linkable	Not linkable
Anonymity	Anonymous	Anonymous	Not anonymous	Not across
Susceptible/ Vulnerabile	Dependent on the IdP	No protection after revealing the information.	Masquerade of user's personal info.	Fewer inducement

a) Isolated Mode

The reasons for the isolated mode (section 2.3.1.1) failure are [4], [10]: (1) Concern of credential burden which introduced inconvenience to the users, (2) Users failed to remember passwords, (3) Users cannot access the resources (services) sited in another domain. The strength of an isolated mode that they keep the users information in isolated way. Therefore, the isolation makes harder to combine the users information for the purpose of precise users information matching. The different user AuthN mechanisims and policies also makes harder to achieve the end user expediency. The user information privacy is protected because it not leaves the organization boundries.

b) Centralized Mode

In Passport, an example of a centralized IAM mode (section 2.3.1.2), all user information are stored in a central repository. So in case of any malevolent activity nothing will be protected [41], [147], [148]. The two main security problems in

Passport wallet via Hotmail are identified in [42], [43], [136] as: (1) reusing of cashed AuthN information and (2) the cross site scripting bugs. In addition to those, the main reasons of Passport failure, given in the following, are relative to its "trust model".

- First, Microsoft participates in every transaction between company "X" and company "Y".

- Second, the users attribute privacy concern: (1) Microsoft is in authority to access each and every user attribute and (2) the lack of the fine grained polices (e.g., user attributes releasing policies).

c) Distributed Mode

In distributed mode the users identity information exchange across a single or multiple trust domains (i.e., witnin between different domains or a single domains). The distributed mode is an alternative solution to the centralized mode. However, the user identity information privacy issue is associated with the design, execution and distributed identity mode active life. In distributed mode the "globally unique identifier" was introduced such as in X.509 based PKI systems. However, the problem with the "globally unique identifier" is the privacy concern becuase everyone knows everyone else's "globally unique identifier" [43]. The knowledge of the "globally unique identifier" certainly raises user identity privacy concerns.

d) User-Centric Mode

Traditionaly a user handles his/her multiple personal identity credentials and indentifiers by memorizing the credentials (e.g., password) or recording their identifiers on a paper. However, increases in the number of these credentials and identifiers causes the credentials and identifiers management issue. To solve the issue of multiple user credentials and indetifiers issue Josang and Pope [23] introduced a user-centric mode (section 2.3.1.3) which let the users store their identifiers and credentials from different SPs in a single tamper resistant hardware device. They merged a Personal AuthN Device (PAD) with an isolated mode. The Privacy and Identity Management (IdM) for Europe (PRIME) [44] and OpenID [45] discussed in [46] are examples of a user-centric mode. The user-centric approach solves the user credentials management issue. However, lacking of platform trust in the user-centric approach may raise security concerns.

e) Federated Mode

The concerns discussed above related to earlier IAM modes illustrate the needs to develop a new IAM mode. In such a new IAM mode, the dispersed HD organization should be responsible but not the users [43]. This has lead to the federated modes (section 2.3.1.4) which is based on a proxy true WSSO AuthN scheme [3] or WSSO [47], [134], [135]. The examples of federated mode are (e.g., Passport, Athens, Shibboleth [13] and CardSpace [48], [49]). The identity federation technology such as SAML [17] fulfills at least some of the Organization for Economic Co-operation and Development (OECD) privacy guidelines by [50], [123]: (1) Allowing the HD IdP to create an arbitrary identifier for each user and (2) The attribute assertions should be short lived (i.e., added by the HD IdP to the FD SP). The property "short lived" of an attribute reduces, in the end of each transaction, the FD SP control over the attributes.

3.2.2 Trust and Security in Federated Identity and Accesss Management

Trust in general has been studied in a range of numerous areas and thus a lot of dissimilar trust definitions may exist in the literature. These trust definitions, in dissimilar areas, may mean a bit different to each person and the environment it is used in. Therefore, this section first describes how trust is observed and exists in different areas as well as in personal interaction with, the focus on trust in the FIAM mode.

Trust in computer science signifies: (1) Peer quality measurement in Peer-to-Peer (P2P) schemes [51], [151], (2) Web services trust in World Wide Web [152] and distributed environment [130], (3) Inspiration in the online transactions and recommender schemes [52], (4) AuthN technique type/digital signature names [53]. Trust, in literature, has also been defined in terms of: (1) The sensation of security (e.g., the subject's feelings that he/she is secure in an action/task) [54], [55], [121], [131], [132], (2) Solace [56], [57] lack of fear. Gove stated that for trust formation it is necessary that the subject who confronts a threat must be eager to depend on another entity with the “sensation of security” or the recommendation of the previous entity [58]. This means that trust is important for someone to feel secure in an action or a task.

To some, trust signifies the level of confidence of a party in someone (or something) [59]. Jøsang et al. illustrated this trust definition clearly and completely comprised the trust fundamental constituents such as: (1) Reliance on the TTP, (2) TTP trustworthiness, and (3) Associated threats to the TTP if it is functioning malevolently [21], [60].

Trust in existing FIAM modes is achieved in different ways such as: (1) Liberty Alliance (LA) [61], [145] defines a Circle of Trust (CoT) to which the SPs and IdPs adhere by signing a business agreement in order to support secure transactions among CoT members [24], (2) Trust association similar to CoT is absent in the OpenID [62], [143], [144] because trust is shifted to the social echelon as of the application echelon [63].

Table 3.2: Federated WSSO approaches and related aspects

	Federated WSSO Approaches				
	OpenID	**Microsoft Passport**	**Liberty Alliance**	**SAML**	**Shibboleth**
WSSO facility	Yes	Yes	Yes	Yes	Yes
Attribute swapping	Yes	Yes	Yes	Yes	Yes
Dissemination type	Obtainable	Not freely Obtainable	Obtainable	Obtainable	Obtainable
Security	OpenID reliant	Microsoft reliant	IdP and SP dependent	Yes	IdP and SP dependent
Trust Establishment	OendID provider	Centralized policy	Business and Lawful accords	Agreements	Agreements
Privacy	Not entirely	Not entirely	To some extent	Yes	Yes
Scalability	High	Petite federations	High	High	High

Trust absence in the FIAM mode may raise security concerns [66], [67], [132] because the FIAM mode connects previously isolated collections of user identity information. Therefore, if a user’s account at his/her HD IdP were successfully phished then the attacker would also have the opportunity to access other associated FD SPs [64]. Also, the security concerns [65], [122], [137], [138] are discussed in a

Web Service (WS)-enabled FIAM: (1) Where the receiver of a message may not be an ultimate destination and (2) Improper security measures may result in the unauthorized access to a user's personal information which will lead to the violation of privacy [4], [5].

To overcome such security concerns, the TCG [14], [125], [127] introduced a hardware based security and trust solution. The TCG defines the trust as, “Trust is the anticipation that a device will behave in a particular manner for a specific purpose” [68]. Alam et al. further make clear the terminologies “specific purpose” and “particular way” [69]: (1) The term “particular manner” is concerned with the question of how a task is expected to be performed and (2) “specific purpose” refers to a particular task or scenario, e.g., usage of an object, web service access, or some computational activity.

Therefore, in this work the trust definition in a federated environment is derived from the TC notion. The TC trust definition is precise to the trust in machine platforms (i.e., BootLoader, OS and Applications) integrity. The TC defines the trust as “Trust in a device platform is the expectation that a device will behave honestly and faithfully to carry out a designated function. A device can be any pervasive device, a desktop (i.e., client) and a server (e.g., IdP or SP) equipped with an entity (i.e., TPM- a tamper-resistant piece of hardware security solution). So, in federated research collaboration scenario (section 1.2) the home domain target’s and the challenger’s machine platforms are said to be mutually trustworthy if both of the machine platforms integrity is mutually validated.

3.2.3 Security, Trust and Privacy Unification

Watanabe and Tanaka [70] proposed a federated AuthN scheme using a cellular phone: (1) to improve ID assurance and (2) to secure the AuthN in OpenID. They solved the security and privacy problems in the current OpenID scheme. However, their solution did not include the communicating machine’s platform integrity measurement. Therefore, the user will be unaware as to whether he/she is interacting with honest or dishonest FAPs or OPs.

Lutz and Campo [71], introduced the identity token concept in Multi-domain-Federations (MdFs). The purpose of this work was to bridge the gap between security (e.g., the identity of the user can be detected definitely), and privacy (e.g., minimal information sharing about the user). To protect the user's privacy in the MdF, the user in the FD is visible only with pseudonymity. This guarantees that no private information will be kept at the FD. They examined the user identity security and privacy but not the trust in machine platforms.

Dey and Weis [72] identified a key privacy concern in the OpenID federated login scheme. The concern is that the IdPs could possibly link the user's identity and track his/her visit across multiple sites. In a proposed solution, they solved the problem by pseudonymity and unlinkability which are achieved through a blind signature [36] scheme. The concerns in their scheme are: (1) The token storage and (2) The absence of client and the IdP machine platforms mutual attestation.

The idea behind the Shibboleth design is to ease the formation of federations and collaborations [150] between the participating organizations [13]. The advantage of Shibboleth over other FIAM modes is the privacy preservation [43]. However, the issues in Shibboleth are: (1) A lack of integration of the user AuthN mechanism with that of the HD client and IdP machines' mutual attestation, (2) The absence of mutual trust formation between the HD clients and the IdP machine and (3) The FD resource AuthR decision not made on the basis of the HD client and the IdP machines' successful mutual attestation trusted attribute.

Hacket et al. [159] identified several significant security and privacy concerns in BrowserID and WebID federated identity systems that need to be addressed in a cohesive way. Singh et al. [73] introduces privacy, trust and policy based AuthR framework for web and grid services in distributed environment. Kungpisdan [74] presents a framework for agent based SET mobile set payment. Their proposed solution is practicable because it is fully well-matched with the existing SET payment infrastructure.

In addition to the above, Mond [75] discusses trust, privacy and security issues in peer-to-peer environments. Karnouskos et al. [76] discusses how security, trust and privacy concerns are tackled in the Secure Mobile Payment Service (SEMPOS) project from the technology and business model perspective. Whereas, Rooy et al. [77] discusses privacy and trust concerns in the future Internet because of unseen users' data collection.

The above discussion shows the previous steps that have been taken to bridge the gap between SP (Security and Privacy). However, to build a unified STP framework that combines some STP related aspects (in a single framework) still does not exist. Solving the STP puzzle in the future FIAM mode is an interesting but hard challenge. The construction of PUSTPF for FIAM mode is discussed in the chapter 4 (section 4.2.2). For this, first STP threats are described in federated identity and resource access scenario through a threat model. The threat model, given in section 4.2.2, covers: (1) Weak AuthN, (2) The absence of a mutual trust formation among the HD clients and the IdP machines' platforms and (3) The HD clients and IdP machine platform privacy-conservation in mutual attestation protocol at the FD SPs. On the basis of the identified threats two different holistic frameworks (section 4.3.1) are contributed: (1) The EUSTPF (section 4.3.1.1) which does not include a TTP and (2) The PUSTPF (section 4.3.1.2) which includes a TTP.

3.2.4 Practicability of Identity Masking Schemes

This section describes the Identity Masking Schemes (IMSs) practicability in real environment and particularly in a USTPF.

The identity masking schemes are anonymous credential schemes. The purpose of identity masking is to mask the original user's identity with a pseudo identity. In early non-anonymous credential schemes such as in the Public Key Infrastructure (PKI) [79], [108], the concern was about the user identity privacy (section 2.6.1). To overcome such an issue, anonymous credential [80], [81] or identification schemes [82], [83] were introduced.

Chen [82] proposed one of the earlier anonymous schemes. The proposed scheme offers the echelons of effectiveness vital for practicable systems. However, the Chen's scheme suffered of privacy issue because of the participation of a third entity in the consumer’s registration activity. Damgård [84] proposed a complexity theoretic primitive scheme, which is based on the “one-way function” and “zero-knowledge proofs”, but the practicability of Damgård is low. The general credential scheme [83] has low practicability as well because it is contrasted upon the “one-way function” and “zero-knowledge proofs”. Brands [85] introduced the “Certificate based scheme”. However, the major drawback of this scheme is that all participating issuer organizations must concur on pertinent “set of security” parameters. Therefore, to build such a scheme in reality is not any easy task. As well, the other anonymous credential systems [86], [87], [88], [89] are also not yet possible to be implemented in daily practices [72]. The main issue related to such schemes in a practical implementation scenario is the underlying mechanism complexities that it is created on.

Diffie and Hellman [90] introduced “a private signing function” such as (Pvt-SF)”. The (Pvt-SF)” is only known to the signer’s authority whereas the “public verifying predicate” such as (Pub-VP) is in the knowledge of the validator. However, to be in this scheme’s signature generation without the knowledge of the signer’s authority is not possible. To overcome the aforementioned issues, Chaum [36] introduced a “Blind Signature (BS)” which is a practicable signature generation scheme. The BS improves the user’s privacy concerns by using a “Blinded Function (BF)” and an “Unblinded Function (BF-1)”. The EUSTPF (given in section 4.3.1.1) is based on Chaum scheme. However, the EUSTPF is not practicable (section 1.8) because of the unavailability of an open source packages [74], [78].

3.2.5 Practicability of Remote Attestation Techniques

Remote attestation in computing and mobile machine platforms is an emerging research and distinct attestation techniques have been developed. The selection of the attestation technique depends on: (1) The practicality of the scenario, (2) Open source technology availability, (3) Privacy concerns, (4) The possibility of putting it into practice, (5) The complexity of the attestation scheme and (6) The machine platform measurement management.

The IMA based remote attestation technique relies on the machine platform measurement configuration (or binaries), Figure 3.1 (given on the next page), developed by the IBM T.J Watson Research Center [15]. The IMA is a most practicable attestation technique due to fewer complexities and the availability of open source technologies.

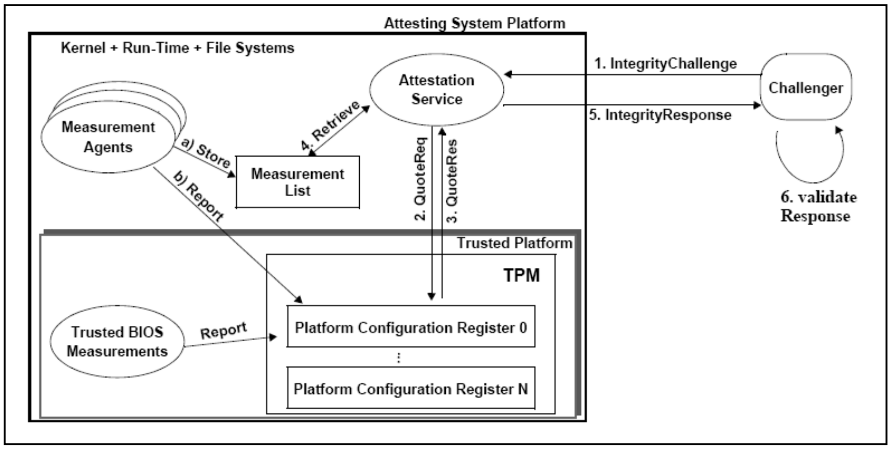


Figure 3.1: Integrity measurement architecture attestation scheme [15]

The Direct Anonymous Attestation (DAA) [160] (Figure 3.2) was proposed to overcome the issue of the recurring process of obtaining the AIKs' certification from the PrivacyCA in the TPM1.1b version. Through the DAA scheme, the users can obtain the AIK certification without the presence of the PrivacyCA. The DAA scheme [91], [92] is formed upon cryptographic schemes: (1) group signature, (2) credential systems and (3) identity escrow. The new TMP1.2 specification supports the DAA instead of the earlier TPM1.1b. The DAA protocol consists of two stages [91], [92]: (1) the enlisting stage and (2) the signing stage.

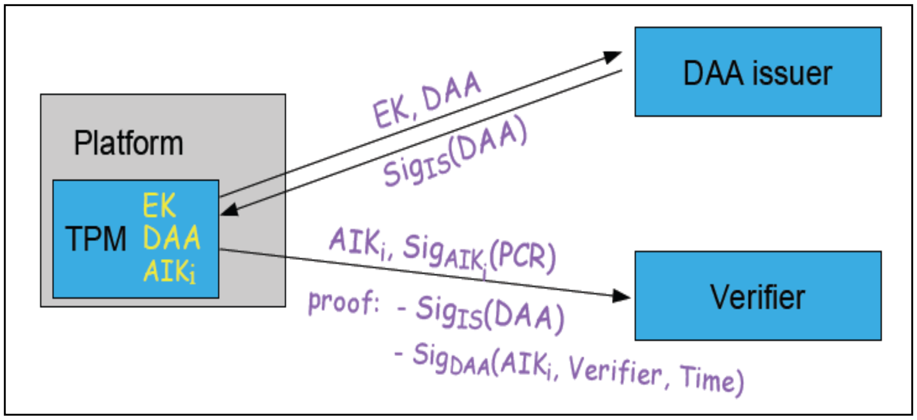


Figure 3.2: Direct anonymous attestation scheme [160]

The PBA [93] probably overcomes the security and privacy issues in the external networks introduced by the binary integrity measurement scheme. Distinct attestation techniques were designed [94] which can utilize the PBA such as: (1) The Delegation Based Attestation (DelegationBA), (2) Derivation Based Attestation (DBA) and (3) Enforcement Based Attestation (EBA). The PBA scheme is new and it still has many challenges to overcome [94].

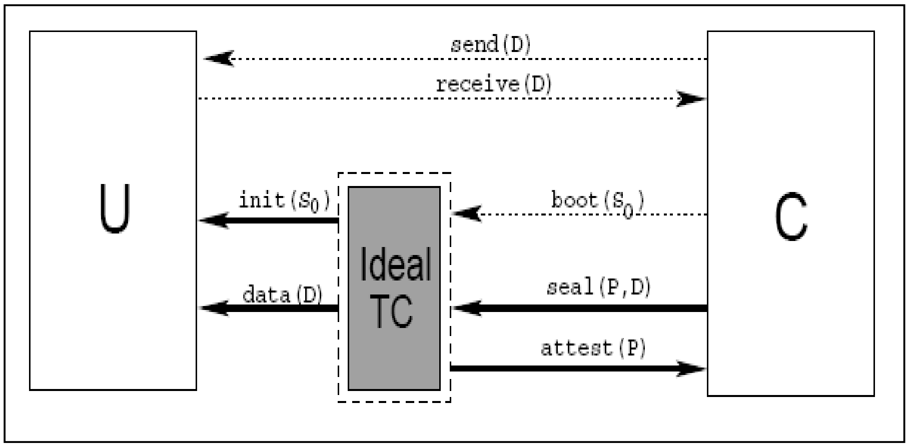


Figure 3.3: Property based attestation scheme [93]

Garris et al. [95] discusses the design and implementation of a trustworthy kiosk computing prototype. The proposed prototype is based on two protocols: (1) The first protocol allows a mobile device owner to establish trust on a public computer kiosk before revealing any personal information to the kiosk and (2) The second protocol allows a kiosk owner to verify that the kiosk is running approved software. The concerns in this scheme are: (1) The kiosk cannot assess the mobile device platform authenticity (or trustworthiness) and (2) Kiosk platform privacy concerns.

A trustworthy AuthN scheme was introduced by [96], [97] of which: (1) was designed on the OpenID concept and (2) makes use of a remote attestation technique to measure, report and validate the integrity of a target machine. However, the OpenID concept is different from other IAM systems. The OpenID IdPs issue “global identifiers” to their users through which the users then login to any SP. In the rest of the transactions, the IdPs and SPs then refer to the user this “global identifier”. Therefore, the problem of using a “global identifier” OpenID does not support any anonymity or unlinkability [98], [142].

Ali and Nauman [99] proposed trust-aware web server architecture for enforcing access control policies based on a client integrity state which they called Integrity Based Access Control (IBAC). The limitations in this scheme are: (1) The client machine platform privacy-concern and (2) The client side cannot assess the web server platform trustworthiness.

Pashalidis and Mitchell [2] proposed a theoretical scheme that eliminated or restricted the role of the ASP. The main contribution in this work is the design of an integrity/challenge response based SSO protocol for the local true SSO scheme. The concerns in their scheme are: (1) The architecture complexity, (2) The client platform privacy concern and (3) The absence of the machine platform mutual attestation.

Ali et al. [100] integrated a remote attestation technique in a FIdM to strengthen the client machine security. As the proposed scheme is constructed on a remote attestation protocol the main concerns are: (1) The client cannot assess server platform integrity, (2) The resource access not linked with mutual attestation result.

Sailer et al. [115] propsoed the mutual attestation and showed how at some point, such as in the areas of Software as a Service (SaS), Grid and Cloud computing, the bidirectional trust formation using TC attestation technique may play a major role. Shane et al. [116] explained how to enable protection against “crimeware” threats such as rootkits, worms, keystroke-loggers, viruses and Trojans in open environments using a TC technology. Zhan et al. [117] proposed a Trusted Grid (TG) model utilizing the TC technology based trust and security solutions. They demonstrated “how to construct a trustworthy sub-domain” for Grid Environment (GE) through TC mutual attestation scheme. Cáceres et al. [118] presented mutual attestation scheme for mobile devices.

The concerns related to federated resource (service) access are discussed in section 4.2.2. These concerns include: (1) The user AuthN and, AuthR and (2) the trust formation between communicating machines. To access federated resources (services), the concerns associated with the traditional certificate based AuthN scheme are: (1) The key pair estimation or calculations through modern software based attacks, (2) The storing of a private key on a user system which probably increases the vulnerability of theft. The TPM provides: (1) protection of private keys through pseudonyms and (2) machine platform mutual attestation.

Bringing together the strengths of trusted computing technology based mutual attestation protocol and of Shibboleth to tackle security, trust and privacy concerns in a unified way, also practicability of the end product is an important concern.

In addition of addressing the issue of security, trust and privacy unification in a federated environment the practicability issue needs to be solved too. Therefore, to achieve this in federated scenario the major modification is carried out in the HD organization, minor changes in the FD SP application level, and the SP modules are unchanged. The PUSTPF is compatible with the current FIAM system infrastructure (e.g., Shibboleth), whereas the EUSTPF is not [74]. The practicability scope is restricted to the framework scalability, flexibility and simplicity – login convenience [73], [74].

3.3 Past Works Comparison

The past works consists of: (1) related works with the author's last name and the publication year (given in rows) vs. (2) the STP and attestation scheme columns.

Table 3.3 (given on the next page) shows the past works have considered the STP unification in different styles (i.e., some focus on the integration of the SP, while others focus on the integration of the ST). Not many of the works focus on the STP unification in a composite mode. Therefore, if server gets infected by a malevolent activity then the corrupted server might misuse (i.e. the adversary) the user's credentials to gain personal benefits.

Table 3.3: The comparison of past works realted to the security, trust and privacy

Past Works	Security	Trust	Attestation Scheme	Privacy
Cantor et al. 2005 [12]	They provides communication link security, but missing TC based security and mutual trust formation		Not included	Shibboleth provide privacy protection at the FDS
Dey et al. 2010 [72]	This scheme missing TC provided security and trust solutions		Not included	The user privacy is protected at the HD and FD
Ali et al. 2010 [100]	The TC security and trust solutions are used to protect only the client machine platform integrity but not the IdP		Remote Attestation	The client platform privacy is not protected at the server
Klenk et al. 2009 [96]	TC solutions are used to harden the client machine platform security and bring trust in it		Remote Attestation	The client platform privacy is not protected
Leicher et al. 2010 [97]	The client platform embedded TC based security and trust notions		Remote Attestation	The client platform privacy is not conserved at the OpenID provider
Lutz et al. 2006 [71]	Missing TC provided bidirectional security and trust solutions		Not included	The user identity privacy is protected via IDToken
Watanabe et al. 2009 [70]	The scheme not included TC mutual security and trust nations		Not included	Not discussed
Pashalidis et al. 2003 [2]	The client platform integrity protected via TC provided solutions		Remote Attestation	The client platform privacy is not protected at the SP
Garriss et al. 2007 [95]	The mobile platform user can find that he /she connected to a secured and trusted kiosk, but kiosk cannot		Remote Attestation	Kiosk platform privacy is not protected
Ali et al. 2009 [99]	The client machine implanted with the TC notions		Remote Attestation	The client platform is not conserved
Sailer et al. [115]	Incorporated TC bidirectional security and trust establishment in enterprise scenario		Mutual Attestation	The client and server privacy against each other is not protected
Zahn et al. 2007 [117]	TC bidirectional security and trust solution is provided in the sub-domain		Mutual Attestation	The platforms privacy is not conserved
Cáceres et al. [118]	Introduced TC mutual security and trust notion mobile platforms		Mutual Attestation	The privacy of the interacting platforms is not protected
Khattak et al. [74], and [75]	Embedded TC Bidirectional security and trust solution		Mutual Attestation	The user privacy is protected, but the client and server platform privacy is not protected

3.4 Summary

This chapter presented the critical review of the associated works pertinent to this research. The introduction of the chapter was provided in the first section. The second section discussed the literature review related to this work. The literature review covered different IAM modes and the advancement of these modes to show why the early IAM systems failed. It also presented how trust and security are managed in the existing FIAM mode, the unification of security, trust and privacy, and the practicability of the identity masking schemes in the real environment. This chapter also explains different machine platform attestation techniques, the related challenges of the existing attestation techniques and the practicability of the attestation technique such as the implementation possibility in the real environment or experiments/ test-beds. The comparison of the past works and the security, trust and privacy unification were discussed in the third section.

CHAPTER 4

RESEARCH METHODOLOGY

4.1 Introduction

Constructing a framework is cyclical, which consists of repetitive constructions and examination steps prior to the achievement of the end design artifact [101]. The work presented in this thesis is also cyclic in nature, which consists of several phases to refine the process of the framework construction. The research methodology established for this work consists of four main phases (Figure 4.1). The framework foundation phase is discussed in the second section. The third section describes the design of the hypothesis. The test-bed prototype implementation phase is presented in the fourth section whereas the assessment of the framework is discussed in the fifth section. The chapter is summarized in the sixth section.

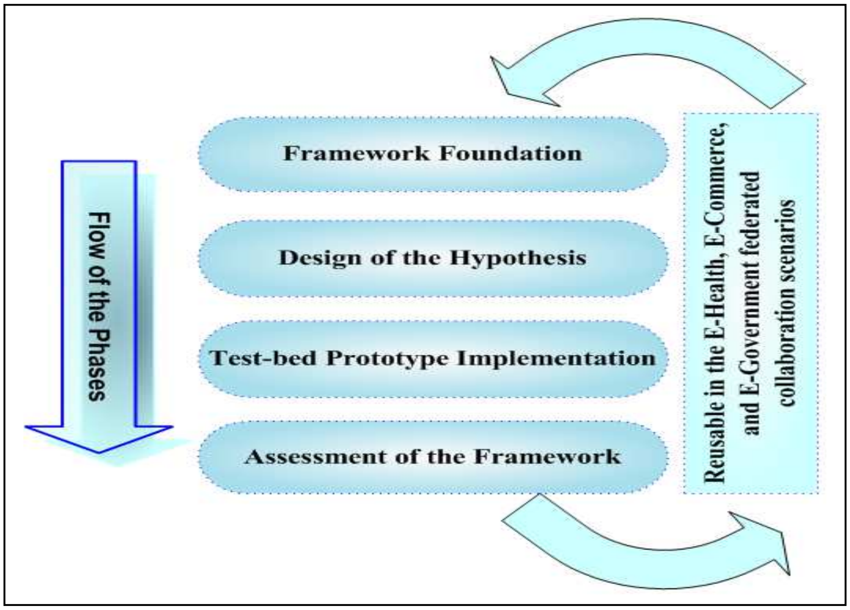

Flow of the Phases
Framework Foundation
Design of the Hypothesis
Test-bed Prototype Implementation
Assessment of the Framework
Reusable in the E-Health, E-Commerce, and E-Government federated collaboration scenarios

Figure 4.1: Research methodology phases

4.2 Framework Foundation

This phase forms a base for the designing of a PUSTPF. This phase is divided into four sub-phases. The first sub-phase presents the pros and cons of the four basic procedures of the FIAM mode. The threat model for the federated identity and access mode sub-phase explores the flaws in the four basic procedures. The third sub-phase presents the proposed PUSTPF aspects whereas the machine platform mutual attestation technique selection is discussed in the fourth sub-phase. Each of these sub-phases is discussed in sequence:

4.2.1 Federated Identity and Resource (Service) Access Scenario

In this work, Shibboleth is chosen to construct the PUSTPF which uses the proxy true WSSO design to access resources (services) in the Inter-domain scenario. Shibboleth fulfills the pros of each of the four basic federated identity and resource (service) access procedures [43]. However, Shibboleth does have the major cons in the four basic procedures. They are described in 4.2.1.1 to 4.2.1.4.

4.2.1.1 Home Domain end User Authentication

The main concern in all the FIAM modes', end user AuthN mechanisms, is that they are used only to validate the end user identity or authenticity. The cons in Shibboleth end user traditional AuthN mechanism are:

- Current AuthN mechanisms are lacking in identifying and confirming the HD end user and the IdP machines platform mutual integrity state or authenticity.

- They are highly vulnerable to advanced ID theft threats.

- In HD, for security reasons, the end user puts all of his/her trust in the HD IdP for AuthN and needs to share all of his/her private information with the HD IdP. So, if the HD IdP is not in a trusted state any more (e.g., running with a malicious activity), platform security as well as user data privacy may be jeopardized.

4.2.1.2 Home Domain end User Authorization

In all existing federated scenario the cons of the AuthR process:

- In the FD, the resource (service) AuthR decisions are carried-out on the basis of the user attributes which are generated from the AuthN mechanism. However, it is not on the HD client and IdP machines platform successful mutual attestation attribute.

4.2.1.3 Trust Binding

In existing FIAM the cons of the trust binding:

- There is a complete lack of TC based mutual trust formation between the HD IdP and the client machine platform.

4.2.1.4 Identity Masking

Practical identity masking techniques could solve the end user privacy problems but the cons of this scheme are:

- Key existing identity masking schemes are impractical.
- Moreover, the practical identity masking scheme (i.e., blind signature) protect the end user’s identity privacy.
- Shibboleth a standard FIAM solution conserves the HD end user identity privacy in the FD via an alias.

4.2.2 Threat Model for Federated Identity and Access Mode

The cons of the four basic processes, discussed in section 4.2.1, clearly demonstrate the flaws and vulnerabilities of the FIAM mode in Shibboleth. The flaws and vulnerabilities can be illustrated through a threat model, given in Figure 4.2. It explains how these flaws and vulnerabilities could possibly lead to STP risks. The threat model is precise to the federated identity concerns in the Inter-domain resource (service) access described below.

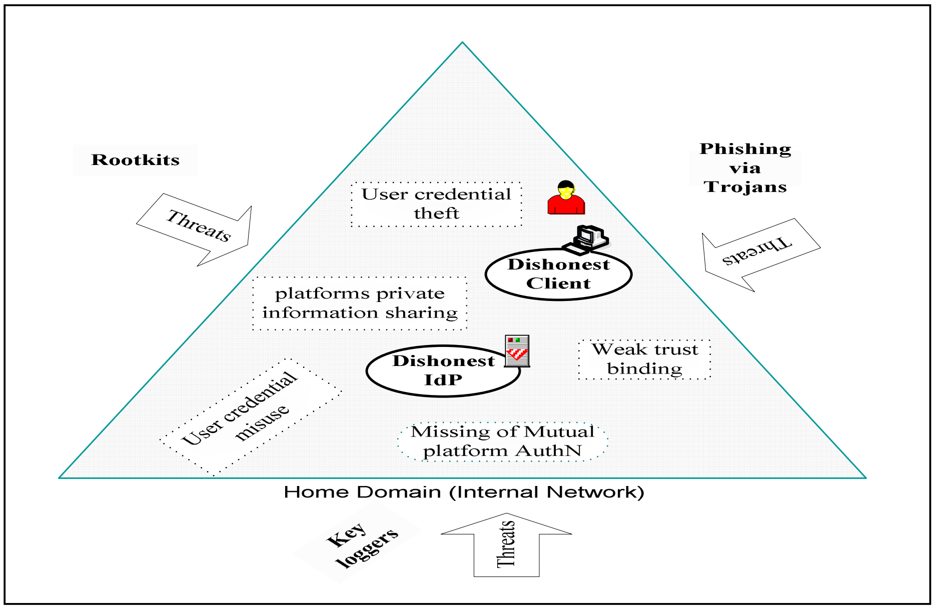


Figure 4.2: Federated identity threat model

4.2.2.1 Threat-1: Identity (ID) Theft

The end user's machine (i.e., client) is probably easily infected by a variety of ID theft threats such as Phishing attacks by Trojans and rootkit attacks, etc. So, in the case of any successful attack, the invader could possibly use the acquired credentials to gain personal advantages.

4.2.2.2 Threat-2: Dishonest Identity Provider

The HD IdP has to always be present online and is in charge of authenticating the end user and exchanging the authenticated user AuthN and AuthR claims with the FD.

Therefore, a potentially dishonest (i.e., malevolent) HD IdP could misuse the user credentials if it is infected with any malicious activity.

4.2.2.3 Threat-3: Weak Trust Binding

The lacking of mutual trust between the communicating machines' platforms leads to security threats. So, HD IdP and client machines infected by any malevolent activity causes the trust ambiguity (e.g., whether the HD IdP and client machines are trustworthy to converse or not). In addition to that, the FD does not have any knowledge regarding the HD IdP and client machines' platform trustworthiness and the resource AuthR in the FD is not on the basis of a trusted attribute.

4.2.2.4 Threat-4: Attested Platforms Privacy Anxiety

A mutually attested machine platform possibly raises platform measurement privacy risks.

4.2.3 Security, Trust and Privacy Aspects Specification

On the basis of the four basic federated resources (services) access procedures and threats discussed in section 4.2.1 and 4.2.2, respectively, this section discusses the PUSTPF chosen STP aspects/ characteristic (see section 4.2.3). The chosen STP aspects signify the STP unification to achieve the research objective. The final product, PUSTPF prototype, is then evaluated on the basis of the following chosen STP aspects.

4.2.3.1 Security Aspects

The PUSTPF security aspects are:

- The Fusing of the end user AuthN mechanism with the machines mutual attestation technique.

- To bring in preemptive threat detection capability because the HD machines have no anti-threat capability against Trojans and rootkit attacks.
- The prevention of the attested platforms security credentials traceability.

4.2.3.2 Trust Aspect

The PUSTPF trust aspect is:

- The introduction of TMAP in FIAM to establish bidirectional mutual trust among interacting entities.

4.2.3.3 Privacy Aspects

The PUSTPF privacy aspects are:

- The mutually attested machines platform credentials privacy conservation.
- The mutually attested machines anonymity and unlinkability curing.
- Linkage of the resource (service) access in FD with a Trusted-attribute.

4.2.4 Mutual (Bidirectional) Attestation Technique Selection

The literature review on the attestation schemes assist in the selection of the most practicable attestation scheme. On the basis of the attestation sechemes pros and cons IMA is selected to utilize it in the PUSTPF testbed prototype experiment. The pros and cons of each scheme described in 4.2.4.1 to 4.2.4.3:

4.2.4.1 Integrity Measurement Architecture (IMA)

The pros of the IMA [15] given below shows that the IMA practicality is higher than any other attestation scheme. In addition to that, from the perspective of the HD (Internal) network, in the PUSTPF prototype, all cons of the machine platform binary/

integrity measurement most likely turn into the pros [102]. The pros and cons of the IMA scheme are:

- The Pros:
  - Open source tool availability (e.g., TC for the Java Platforms [40]).
  - The PrivacyCA in the TTP based attestation scheme provides a higher level of privacy than that provided by the DAA scheme [103], [104].
- The Cons:
  - There are platform measurement privacy concerns.
  - There are software monopoly concerns.
  - If the entity challenger is located out of the internal network, then the holding of the HD clients' trusted certificates by the challenger generates the trust uncertainties such as how can the external entity be trusted when the user is not in a direct trust association.

4.2.4.2 Direct Anonymous Attestation (DAA)

The pros and cons of the DAA scheme are [91], [92]:

- The Pros:
  - In the DAA scheme, the target (client) machine anonymously validates itself to, the verifier, show the possession of a DAA certif. acquired from the DAA issuer.
  - It accomplishes "random base" DAA signature linkability (e.g., if identical TPMs, for the identical verifier, two random based signatures cannot be associated with each other.
  - The key privacy feature of the DAA scheme is that it minimized the role of a TTP.

- The Cons:
    - Highly intricate.
    - The unlinkability issue – Embedding of covert identifying information into the DAA certif. from the DAA Issuer can link the TPM’s transactions to locate who the DAA certif. belongs to.

4.2.4.3 Property Based Attestation (PBA)

The PBA [93] scheme is still in the early stages and applicability in practical distributed scenarios has several challenges that still need to be addressed [94]. The pros and cons of the PBA scheme are [94]:

- The Pros:
    - The properties probably conceal machine platform susceptibilities by not unveiling the machine platform measurement details.
    - The properties preserve the attested machine (target) privacy at the verifier (challenger) by not disclosing the identity of the attested machine measured pieces.
    - Writing meaningful Access Control (AC) - policies on the basis of the machine properties is an added advantage.
- The Cons
    - The machine platform property scope clarification is intricate because the platform consists of diverse kind of properties. For instance, to one person, property in a platform may be any feature (or behaviour) of a particular tangible or intangible component, while to others, the whole platform may be a property.
    - Attested machine platform property requirement consideration is important because the platforms of these properties may be seen as components, for instance, security allied properties. So, if a

property consists of several sub-components such as confidentiality (i.e., confidentiality is probably achievable by an SSL [6] in a transit or via encryption in a storage component etc.) then probably it would reveal extra information).

- There are limitations of the software application specific properties. For instance, unavailability of several general types of application specific properties that are possibly described for the software in every OS in every kind of hardware.
- In real time systems, the issue of property certification is important. For instance, the faith in a client's machine configuration state is that it complies with the configuration state when it is tested and certified in the lab.
- The issue of the properties' reliance on each other is a concern. For instance, property "A" depends on another property "B" for its functionality. This issue arises when the previous property "A" is not independent for its functionality from the latter "B".
- There are issues of property certificates being revoke or shared.

4.3 Design of the Hypothesis

The design of the hypothesis phase presents the potential design and development methodology used in this work. The design and development methodology is a collection of actions and techniques which examines the procedures, difficulties and complexities in order to reduce the design and development requirements [105]. In the development of the PUSTPF, the Rational Unified Process (RUP) is used as a software product development methodology.

The architecture designs of PUSTPF and EUSTPF are discussed in the section 4.3.1. The answer of the question that why PUSTPF is chosen as final design architecture is explained in 4.3.1.1 to 4.3.1.2

4.3.1 Possible USTPF Design Architectures

The two different kinds of design architecture are possible:

4.3.1.1 Emergent USTPF / Non-Trusted Third Party Solution

A theoretical EUSTPF for federated identity and resource (service) access is proposed in Figure 4.3. The EUSTPF consists of: (1) A blind signature [36] and (2) A TMAP. The beauty of this scheme is that the client and the RP/SP machine platforms must be carried out through the machine platform mutual attestation protocol. So if: (1) the client machine platform integrity check fails, then the SP/RP will not release a resource to the client and (2) if the RP/SP machine platform integrity check fails, then the client will not release the acquired access token to the RP/SP.

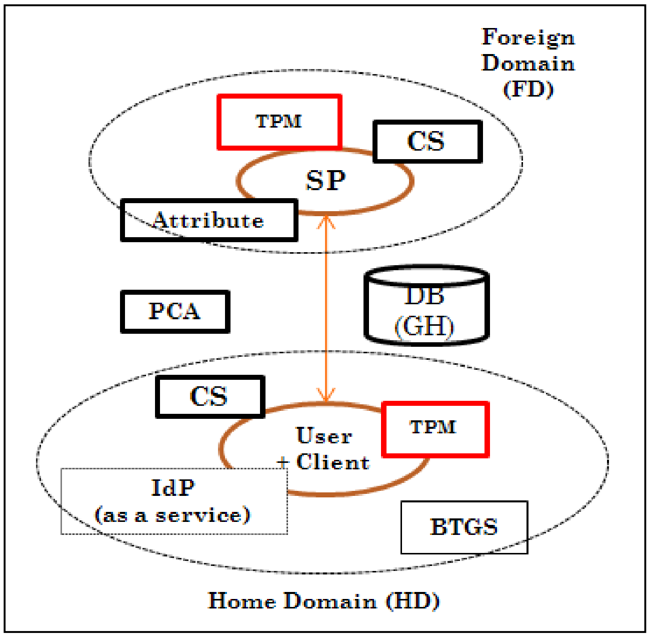


Figure 4.3: EUSTPF - With-out a trusted third party

However, due to impracticability and mutual platform privacy concerns the EUSTPF is not suited for federated resource (service) access. Therefore, such limitations of the EUSTPF lead this research to construct a PUSTPF (section 4.3.1.2) for FIAM which covers overall research objectives and fulfills all STP selected aspects.

4.3.1.2 Practicable USTPF/ Trusted Third Party Solution

As discussed in the previous section that the EUSTPF has own limitations and does not complies to the chosen research objective Therefore, third party based open source and standard FIAM solution is selected to build a PUSTPF. The main entities in trusted third party solution are HD user and IdP, and FD SP. The methodology is given in section 4.3.2, whereas the PUSTPF design, architecture and implementation will be discuss in chapter 5.

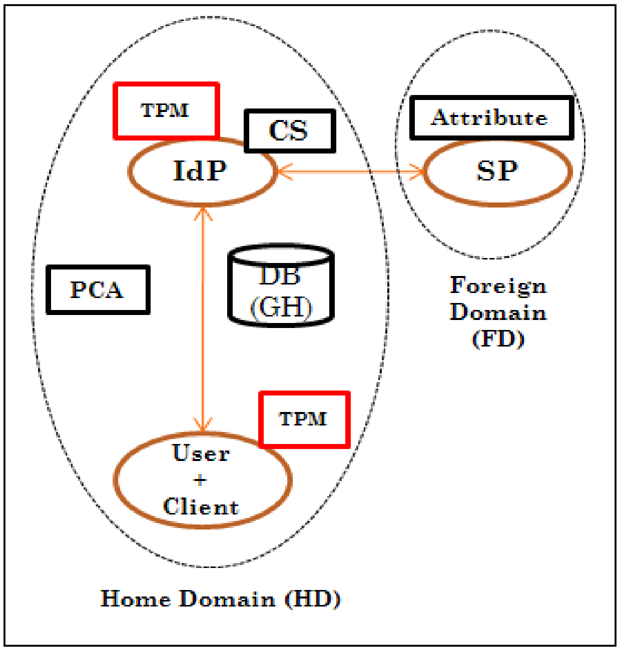


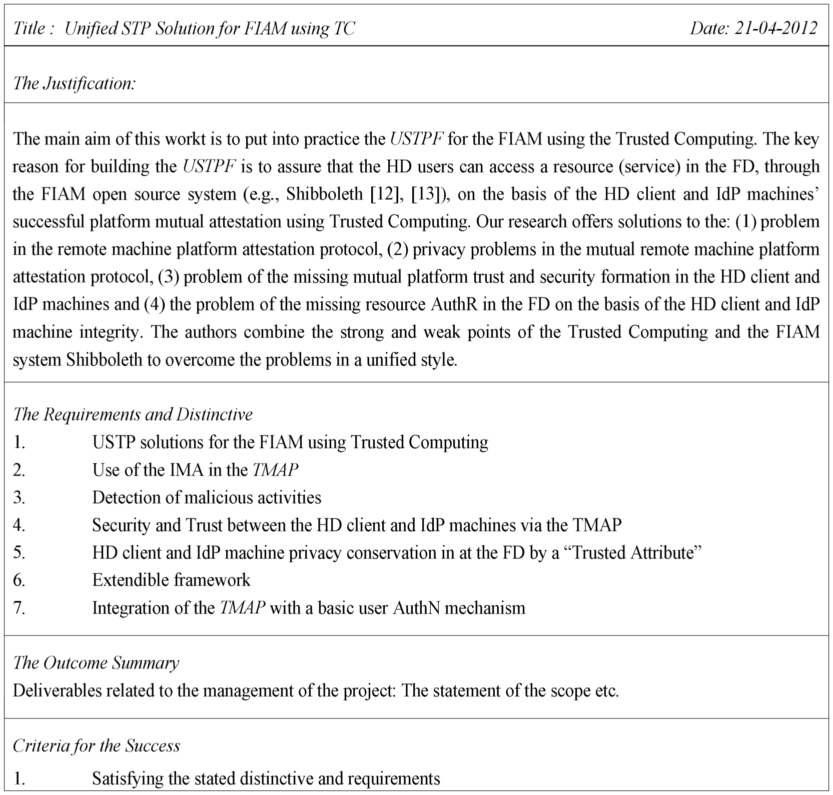
Figure 4.4: PUSTPF - With-a trusted third party

4.3.2 Rational Unified Process (RUP)

The Rational Unified Process [110], [124] is a software engineering process which consists of techniques and phases to steers the software product developer in his/her activities [106]. The four main phases of RUP are:

- *The Initiation: Inception* - The inception is a phase of the project initiation. The aim of this phase is the comprehension of the early requirements gathering. The most significant deliverables in the end of the inception phase is the "statement of the scope (Table 4.1)". Chapter 5 discusses the system design, architecture and implementation.

Table 4.1: Statement of the scope



Title : Unified STP Solution for FIAM using TC	*Date: 21-04-2012*
The Justification:	
The main aim of this workt is to put into practice the *USTPF* for the FIAM using the Trusted Computing. The key reason for building the *USTPF* is to assure that the HD users can access a resource (service) in the FD, through the FIAM open source system (e.g., Shibboleth [12], [13]), on the basis of the HD client and IdP machines' successful platform mutual attestation using Trusted Computing. Our research offers solutions to the: (1) problem in the remote machine platform attestation protocol, (2) privacy problems in the mutual remote machine platform attestation protocol, (3) problem of the missing mutual platform trust and security formation in the HD client and IdP machines and (4) the problem of the missing resource AuthR in the FD on the basis of the HD client and IdP machine integrity. The authors combine the strong and weak points of the Trusted Computing and the FIAM system Shibboleth to overcome the problems in a unified style.	
The Requirements and Distinctive 1. USTP solutions for the FIAM using Trusted Computing 2. Use of the IMA in the *TMAP* 3. Detection of malicious activities 4. Security and Trust between the HD client and IdP machines via the TMAP 5. HD client and IdP machine privacy conservation in at the FD by a "Trusted Attribute" 6. Extendible framework 7. Integration of the *TMAP* with a basic user AuthN mechanism	
The Outcome Summary Deliverables related to the management of the project: The statement of the scope etc.	
Criteria for the Success 1. Satisfying the stated distinctive and requirements	

- *The Touching On: Elaboration* - After the collection of the project requirements at the high level in the initial phase, the elaboration stage mainly focuses is on the collection of the project requirements at the lower level. The requirements collected at the lower level are captured using "use case models". The elaboration phase's main deliverables are the use case models. This phase assists to identify the project related use cases and the actors. The descriptions of the majority of the use cases have been constructed in Chapter 5.

- *The System Design and Execution: Constriction* - In the inception and elaboration phases after adequate requirements have been captured, the system design and implementation may start, the third phase construction is to be carried out. The main purpose of this phase is the software system development to the level of deployment. The main activities carried out in this phase use sequence plus activity diagrams as a manual in the system design, system coding, interconnection of different devices (computers using switch via LAN), attestation collector (daemons) and validation agents' (modules') development and integration.

- *The End: Transition* - The RUP's first three phases are the heart of the methodology for the software product development. Transition is the end phase of the RUP, the main focus of this phase is the testing of the final software product.

4.4 Test-bed Prototype Implementation

4.4.1 Verification of the Hypothesis

The test-bed prototype experiment is used in this work as the verification of the hypothesis instrument. The test-bed experimentation requirements are categorized in the followings (4.4.1.1 to 4.4.1.3):

4.4.1.1 The Pre-requisites Requirements

The declarations of the requirements given below were utilized as the prototype construction prerequisite requirement declaration.

- The Hardware-based Security chip Ownership such as The HD clients' and IdPs' TPM chips must be enabled, switched-on, and owned by the HD administrator (e.g., IT support department of the HD organization).

- The HD administrator should generate the AIKs for the HD clients and IdP set TPM chips, and enroll these AIKs with an HD trusted PrivacyCA.

- The HD clients' and IdPs' machines have to be configured with the IMA configuration.

- The HD clients' and IdPs' machines must have the attestation collector and validator agents. Each of these agents further are sub-divided into sub-agents, for instance the AttestationCollectorAgent's are divided into the AttestClientSML, AttestClientPCR, AttestIdPSML and AttestIdPPCR, while the AttestationValidationAgent's are divided into the Validation of the Received SML (VRSML), Validation of the Received PCR (VRPCR), Validation of the Received Nonce (VRN) and Validation of the Received Certificate (VRCertificate).

4.4.1.2 The Hardware (H/W) Requirements

The Test-bed prototype development requires the following hardware apparatus:

- Three machines: (1) the client, (2) IdP and (3) SP are arranged. The client and IdP machines must have been implanted with a "TPM" security hardware chip. The client and IdP machines possess 1GB and 2.5GB RAM, 2.00GHz and 1.83GHz CPU, and each has a 40GH HDD, respectively.

- 10/100Mbps switch and category.5e UTP cables to connect the client, IdP and SP machines.

4.4.1.3 The Software (S/W) Requirements

To construct a PUSTPF for the FIAM, the HD and FD entities software requirements are:

- Machine 1- The home domain identity provider:
  - Installed Ubuntu (9.10) Linux OS
  - Installation of Java (e.g., jdk 1.6)
  - Apache (Tomcat) installation
  - Setting Light Weight Directory Access Protocol (LDAP)
  - Setting Java AuthN and AuthR Service (JAAS) etc.

    - Compilation - Linux Kernel ver. 2.6.35 with IMA
    - Using TC for the Java (tm) platform (i.e., jTSS, jTPM Tools and PrivacyCA setting)
- Machine 2:- The foreign domain service provider:
    - CentOS 5.6 Linux OS
    - Log4ccp package
    - Xerces-C package
    - XML-Security-C package
    - XML Tooling-C package
    - OpenSAML-C package
    - Apache 2 (httpd) web server etc.
- Machine 3: The home domain client (user system)
    - Ubuntu (9.10) Linux OS installation
    - Java (e.g., jdk 1.6) installation
    - Compilation - Linux Kernel ver. 2.6.35 with IMA

4.4.2 Data Requirement and Analysis Approach

In this research experimentation is used to acquire the data (section 6.2.2.1). The obtained data (i.e., AuthN data, mutual attestation data and performance data) then investigated and discussed (Chapter 6 mention here). In the experiment Shibboleth (i.e., IdP and SP), jTSS, jTPM, and IMA are used to construst a PUSTPF. Since Shibboleth IdP, jTSS, jTPM and IMA all are Java based consequently Java is the optimum option use to construct the machine platform mutual attestation. The detail such as “how this is achieved?” discussed in Chapter 5.

4.4.2.1 Reasons to Choose Test-bed Experiment

The particular reasons for choosing the test-bed experiment are given below:

- The majority of the past and current works related to federated resource access and trusted commuting in a web service environment were carried out using the experiments to validate their concepts. Please refer to Table 3.3 (section 3.3) the past works analysis in Chapter 3.

- The formalization of the PUSTPF is hard to model [107]. This is due to "formal methods have limitations in that they become too complex when applied to real systems because these systems generally are too complex to be practically modelled with formal methods".

4.4.2.2 Performance Measurement

The analysis of the mutual attestation performance measurement (i.e., relationship among the number of measurements and the attestation time) discussed in the section 6.2.1.2 (EUSTPF) and section 6.2.2.1 (PUSTPF). The comparisions of the mutual attestation performance (i.e., (i) client and server vs HD client and IdP and (ii) IMA based attestation scheme comparision) disucssed in the section 6.2.4.

4.5 Assessment of the Framework

The result analysis and discussion of the acquired results from the proof of concept implementation and test-bed experiments are presented in Chapter 6. The précis of the result analysis and discussion which will be discussed in Chapter 6 are:

- TMAP proof of concept result analysis

- Practicable framework test-bed result analysis

- Comparison between the emergent and practicable schemes

- Mutual attestation protocol performance comparison

- Comparison of proposed framework with existing works

4.6 Summary

The framework foundation phase was discussed in the second section of this chapter. It explored the pros and cons of the four basic procedures, the threat model, the chosen STP aspects and the available existing attestation schemes' pros and cons. The third section described the design of the hypothesis phase which covers the PUSTPF test-bed prototype development methodology. The test-bed prototype implementation phase presents the requirements in the fourth section. It described the verifification of the hypothesis, data requirement and the analysis approach. The fifth section lists the assessment of the framework discussed in Chapter 6.

CHAPTER 5

DESIGN OF THE SYSTEM ARCHITECTURE

5.1 Introduction

This chapter covers the the architecture design of the system. The system architecture in second section presents the different entities (or stakeholders), PUSTPF, and comprehensive PUSTPF protocol architecture. The third section focuses on the system architecture design of the PUSTPF functionalities through use-case, activity, class, sequence diagrams, packages and classes and algorithms in the form of flowchars. The chapter is summarized in the fourth section.

5.2 System Architecture

Establishing secure hardware root based mutual trust among different stakeholders in a Federated Identity and Access Management (FIAM) is a challenging task. FIAM provides a complete security model to cater different security aspects of online resource management, such as identification, AuthN, AuthR and auditing. However, some important aspects of security are still missing in the current FIAM frameworks. Establishment of mutual trust among the stakeholders involved in an online transaction need to be addressed in such a manner that will not only cater to the problems of unequivocal identification, AuthN, and AuthR but will also address the challenges of verifying integrity and privacy of the platforms involved. Therefore, lack of mutual trust in federated identity and resource (access) systems can possibly lead to several security threats. However, in federated identity and resource (service) access environment sharing machines platforms integrity lead to the machine platforms measurements (security credentials) privacy concerns in the Inter-domain scenario.

This section presents the PUSTP system architecture for federated identity and resource (service) access environment (Figure 5.1). The scheme comprehensive architecture details are explained in the sub-section 5.3.1 to 5.3.3 It leverages the integrity measurement mechanisms in the hardware rooted based security and mutual trust among different stakeholders. The Shibboleth (Figure 2.5) [12], [13] which is a FIAM Systems provides a standard base, complete open source implementation of proxy true WSSO scheme within or crossways in different security domains. In other words, it provides a complete implementation of FIAM features. Shibboleth is used to protect sensitive resources from unauthorized access across organizational boundaries. In proposed architecture for PUSTPF prototype Shibboleth will be utilized as test-bed target architecture. The rationale behind selecting the Shibboleth as target architecture can be explained from the following sscenario:

Generally in FIAM, a HD client machine requests certain resource or service from a FD SP. The FD SP machine and the HD client machine are assumed to have no former trusted association between them and the FD SP machine entails some sort of AuthN information about the HD client machine before making any access decision. The FIAM architecture is designed in such a manner that each and every HD user is associated with a single or multiple HD IdPs machine. Since the HD IdPs machines are inside the user's organizational structure, the user has trust over the HD IdPs machine, which authenticates the users. The HD IdPs machine authenticates the users on behalf of the FD SP and provides the FD SP with AuthN information associated with the HD user. Thus, the FD SP machine makes access decisions to allow or deny access to a particular resource based on the AuthN information from the HD IdP and HD organizational policies.

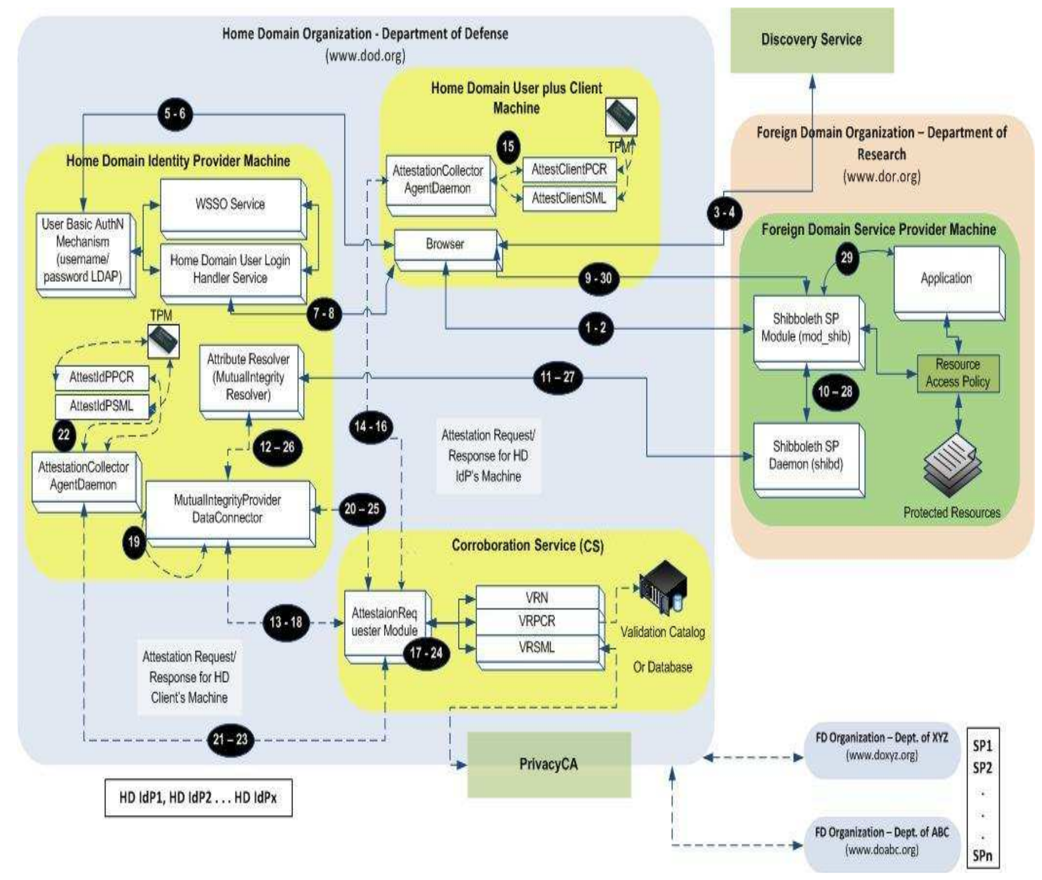

Home Domain Organization - Department of Defense
(www.dod.org)
Discovery Service
Home Domain User plus Client Machine
TPM
15
AttestationCollector AgentDaemon
AttestClientPCR
AttestClientSML
Browser
5 - 6
Home Domain Identity Provider Machine
User Basic AuthN Mechanism (username/ password LDAP)
WSSO Service
Home Domain User Login Handler Service
7 - 8
TPM
AttestIdPPCR
AttestIdPSML
22
AttestationCollector AgentDaemon
Attribute Resolver (MutualIntegrity Resolver)
12 – 26
MutualIntegrityProvider DataConnector
19
20 – 25
14 - 16
Attestation Request/ Response for HD IdP's Machine
Foreign Domain Organization – Department of Research
(www.dor.org)
3 - 4
Foreign Domain Service Provider Machine
29
Application
9 - 30
1 - 2
Shibboleth SP Module (mod_shib)
Resource Access Policy
11 – 27
10 – 28
Shibboleth SP Daemon (shibd)
Protected Resources
Corroboration Service (CS)
AttestaionReq uester Module
17 - 24
VRN
VRPCR
VRSML
Validation Catalog
Or Database
13 - 18
Attestation Request/ Response for HD Client's Machine
21 – 23
PrivacyCA
HD IdP1, HD IdP2 . . . HD IdPx
FD Organization – Dept. of XYZ
(www.doxyz.org)
FD Organization – Dept. of ABC
(www.doabc.org)
SP1
SP2
.
.
.
SPn

Figure 5.1: The PUSTP system architecture

In the above architecture description, the HD IdP and FD SP both play their own respective roles. That is, the FD SP delegates the AuthN mechanism to the HD IdP that provide the HD user AuthN information to FD SP. This way, the FD SP does not need to manage the HD user's credentials as the number of users may vary from organization to organization. The FD SP only utilizes this information in order to make access decision and apply organizational security policies. Therefore, the current FIAM address the issues of HD user AuthN, AuthR and auditing in a sophisticated and widely accepted manner. However, the problems related to the verification of mutual integrity of different platforms involved in this architecture remain unsolved.

Consider the scenario where a HD user's machine that gain access to a protected and sensitive resource is compromised or not in a trusted state at that particular instance. This is certain that such un-trusted or compromised platforms may lead to several security threats in a highly sensitive organization. Furthermore, the HD IdP machine authenticates the user and sends the AuthN information i.e., HD user credentials to the FD SP for making access decision. What if the HD IdP machine platform is tampered with a malware or in an un-trusted state at the time of AuthN? The infected HD IdP machine platform can steal user credentials and may misuse these credentials at a later time by accessing the sensitive resource. The FD SP also needs some sort of assurance that both the HD client and IdP machines platforms are in a mutual trusted state when some sensitive resource is accessed. This assurance can be provided by employing mutual attestation technique in the current FIAM solutions. Therefore, verifying mutual integrity of the communicating machines platform via mutual attestation protocol is one of the important security aspects in the FIAM that needs to be addressed. In addition to that using mutual attestation protocol in FIAM may raise machines platforms privacy concerns. Therefore, to conserve the HD client and IdP machines platforms integrity privacy at the FD SP and the HD IdP machine platform integrity privacy to conserve at the HD clients machines needs careful considerations. Therefore, the disused STP challenges lead this research to develop a PUSTPF for federated environment which at least combine some of STP aspects specified in chapter 4 section 4.2.3. The subsequent sections describe the PUSTPF architecture and the implementation details of mutual attestation technique in the Shibboleth architecture.

5.2.1 Proposed System Architecture

There are a number of stakeholders involved in PUSTPF architecture for FIAM. They provide: (1) Integrated AuthN (e.g., user basic AuthN mechanism integration with the TMAP), (2) Mutual trust and security formation through TMAP, (3) Mutually attested machines privacy conservation in TMAP and (4) Resource or service AuthR decisions on the basis of the HD clients and IdP machines platforms’ successful mutual integrity verification. Each of these stakeholders are described below.

5.2.1.1 Home Domain Stakeholders/Entities

The HD consists of two main entities: (1) Home Domain client machine and (2) Home Domain identity provider machine. In a federation there may be many HDs and each HD may have many clients, and each single IdP has embedded with the TPM and IMA configuration. The details of both entities are given here:

a) Home Domain User and Client Machine

In PUSTPF architecture the HD user has to known the username/password issued to him/her by the HD organization and this user has also assigned a client machine platform whose integrity is verified by the HD IdP at time when the client request to access a protected resource residing at the FD SP. In order to verify integrity of the client machine platform it is mandatory that the platform must be a TPM-enabled system. To leverage the capabilities of the TPM in a broader spectrum, there are some prerequisites that need to be fulfilled by the HD DoD organization prior to deploying the architecture. The prerequisites are:

- The TPM residing at the HD client machine platform must be enabled, activated and owned by the user and/or by the system administrator of the HD organization where the client works (such in the FIAM scenario www.dod.org).

- Each HD client machine platform needs a specific Attestation Identity Key (AIK) that needs to be created after taking the ownership of the TPM.

- According to TCG specification [68] the client's machine AIK must be registered with a PrivacyCA so that the TPM authenticity can be accomplished. (This can be done by deploying own PrivacyCA and registering the AIK with it).

- In order to use the AIK in an attestation scenario, the AIK secret value (generated at time of AIK creation) must be accessible to the Daemon-Attestation_Presenter named here Attestation Collector Agent (ACAgent) Daemon running on the client machine.

Each prerequisites is performed only once during the entire process of deployment. The procedure on how to accomplish these prerequisites will be described in detail in the later sections.

b) Home Domain Identity Provider Machine

The Home Domain IdP machine is one of the major entities in PUSTPF architecture that performs user AuthN on behalf of the FD SP machine. The process the HD user credentials in such a manner that it releases attributes related to the HD user that is to be used by the FD SP machine in resource or service decision making process. Since the FD SP machine protects sensitive resources and it will releases these resources to the HD client machine only in a scenario where the FD SP machine can build a trust association with the HD client machine as well as the IdP machine. In order to build the trust among the entities involved (e.g., HD client and IdP, and FD SP machines) there must be an entity that is capable to perform mutual integrity measurement of the HD client and IdP machines platforms on behalf of FD SP machine. The FD SP machine can also perform the mutual integrity measurement of these platforms directly but it will give rise to several privacy issues.

Firstly, in a federated identity and resource or service access environment there are a large number of HD clients machines and the FD SP machine has to provide them with different type of resources (e.g., sensitive and protected resources or normal and unprotected resources). The FD SP machine has to maintain its own security mechanisms and AuthR policies rather than attesting the integrity of each and every HD client machine with-respect-to their HD IdPs machines, therefore, this is not a feasible solution. Secondly, the FD SP machine usually resides outside the organizational boundaries of the HD client machine (e.g. in PUSTPF scenario both client and IdP machines are members of the HD department of defense organization, whereas SP machine sited at FD department of research organization). If the HD client and IdP machines reveal their platform security credentials that are required for integrity measurement, their privacy will no more be conserved.

Keeping in mind the above mentioned problems, the FD SP machine must delegate the process of mutual integrity measurement to some other entity. In the current Shibboleth architecture the HD IdP machine can perform AuthN of the user but cannot assure the integrity of the client's platform. Since in the PUSTPF architecture, the integrity of both the HD client and IdP machine platforms are validating therefore a better option for the mutual integrity measuring entity should be the HD IdP machine. The rationale behind this option is that in a federated environment almost every HD organization has its own IdP machine to authenticate their HD users and the HD users already are aware of the HD IdP machine. For AuthN, the HD clients machines already release their credentials to the HD IdP machine, so the possibility of releasing information related to their machines platform configuration to the HD IdP is much higher and the user is in strong trust association with his/her HD IdP.

c) Corroboration Agent/ Service

Corroboration Service is a specialized entity that actually performs the mutual attestation on behalf of the HD IdP and client machines. The HD IdP machine initiates the process of integrity verification regardless of the target machine platform (e.g., client or IdP) and the CS performs the mutual integrity verification. This means the primary challenging party is the CS. The CS can be a dedicated entity within the HD organizational boundaries of the user or it can be an internal part of the IdP machine platform. For the realization of PUSTPF proposed architecture, the CS is defined as integral part of the HD IdP machine but for the sake of clarity, it is termed as a separate entity other than the HD IdP. In general FIAM, there is no concept of using CS as it is a specialized entity for performing mutual attestation only but in PUSTPF, CS plays an important role by verifying mutual integrity of the HD client and IdP machines platforms. In addition to that, the CS entity asset in this work is to conserve the HD machines privacy because it is the part of HD IdP and is responsible for performing the HD machines mutual attestation.

5.2.1.2 Foreign Domain Stakeholders/Entities

In FD and particularly in FIAM, SP is one of the core entities that provide users with different type of resources and services. The SP protects sensitive resources and is responsible to enforce organizational policies to make access decision for the resources. Since this work is proposing to verify the mutual integrity of machine platforms used in PUSTPF architecture, so the SP is extended with limited changes to incorporate the integrity measurement mechanism. The changes made to the SP are only limited to the application end of the SP and the core modules of the SP are not modified.

Access decision to grant or deny access to a particular resource is based on the organizational policies and the attributes that are returned by the IdP about the user after successful AuthN is performed. The web server running on the SP is responsible for enforcing these policies and the decision making to allow or deny access to a particular resource. In PUSTPF architecture, an application is created that protects certain sensitive resource to be accessed by a client. The application is responsible to release the resource to the client after checking the AuthN and the integrity verification attributes released by the IdP. Since in this work the mutual attestation protocol performing the attestation of HD both the client and IdP machines platforms, in order to build a trusted relationship among the entities, so the application at the FD SP shall check whether the mutual attestation attribute resulted in true before releasing the resource.

5.2.1.3 Discovery Service (DS)

Generally, in a FIAM system each and every participant organization may have their own separate HD IdP machine that performs AuthN of the users on behalf of the FD SP. Thus, in a federation there may have multiple numbers of IdPs machines to facilitate HD user’s AuthN in its organization boundaries. Each and every FD SP machine must also be registered with these IdPs machines in a federated environment. The DS is a specialized entity in the Shibboleth architecture that is used to select a particular IdP machine for AuthN among the available IdPs machines in a federation. When a user access a resource at the FD SP, the FD SP redirects the HD user to the DS, which presents an interface to the HD user to select their particular HD IdP machine. Once the HD IdP machine is selected the user is redirected to that selected HD IdP machine for AuthN by the FD SP machine. Since, the DS plays the role of selecting HD IdPs machines only; there is no need to make any changes to it in order to make it work with the proposed architecture. The proposed architecture and the modifications need to be performed in order to incorporate machines mutual attestation mechanism in a federated identity and resource or service access environment.

5.2.2 Practicable Unified Security, Trust and Privacy Framework

Federated Identity and Access Management (FIAM) mode provides WSSO features that are designed specifically to address the issues in secure online resource sharing and collaboration among different HD and FD organizations. Shibboleth is one of the most significant and widely adopted FIAM modes that provide user’s AuthN, AuthR and auditing in a privacy conserving manner. However, a major limitation of almost all of the current FIAM in general, and Shibboleth in specific, is that there is no mechanism available for assuring the HD communicating machines mutual trustworthiness and security in the online resource sharing system. A HD user accessing a sensitive resource sited in a FD may be trustworthy. However, the HD client machine platform and the HD IdP machine that the user used to authenticate with and provide AuthN and AuthR information to the FD to access the resource or service might probably be tampered with some malware or Trojans. This may compromise the sensitive resources. This is a major issue in the process of establishing mutual trust among different entities in security critical organizations.

To alleviate these problems, a solution is proposed to leverages the features of FIAM (e.g., SSO, password management, AuthR, auditing privacy etc.) and to incorporate mutual attestation technique to verify the integrity of the HD stakeholder’s machines platforms (e.g., client and IdP) involved in online collaboration among different organizations. Furthermore, in PUSTPF architecture the concept of machines mutual attestation is proposed to not only verifies the integrity of the HD client’s machine platform but will also verify the integrity of the HD IdP’s machine platform. This will build a secure and trustworthy base for the FD SPs to share sensitive resources with the HD users in a secure and trusted environment. For the realization of mutual integrity verification mechanism in the architecture, the IMA [15] (section 2.8.3) is implemented as a proof-of-concept. To understand the proposed architecture and the overall system it is necessary to understand the information flow in the Shibboleth system. The information flow in Shibboleth architecture is explained with a simple scenario in section 2.5.3.2.

How the PUSTPF architecture incorporates the machine mutual attestation mechanism in the federated environment particularly in the HD (Figure 5.1). In addition, what changes are made to the entities involved in the FIAM system to incorporate the mutual attestations are also discussed.

In essence, the incorporating integrity measurement in PUSTPF architecture can be categorized in to two different phases: (1) Integrity Measurement phase and (2) Validation phase. In the first phase the trust tokens are calculated and processed and in the later phase validation is performed.

5.2.2.1 Mutual Integrity Measurement Phase (MIMP)

The modifications carried out in the HD organization for mutual integrity measurement is discusse here:

a) Home Domain Client Machine Modification

The HD client's machine platform is one of the most important entities to be trusted in a federated identity and resource or service access environment because sensitive resources and services are meant to be released to the HD user. The HD user may be trusted but his/her client machine platform may not. Therefore, a solution is proposed that will validate the mutual integrity of HD client's and IdP machines platforms prior to releasing any sensitive resource to it. As discussed earlier, that in order to verify the mutual integrity of both HD client's and IdP machines platforms, it is mandatory that the both HD machines platforms must be TPM-enabled. The HD client machine platform modification is presented in this section whereas the HD IdP will present in next section. For machine mutual integrity in this work IMA is implemented as a proof-of-concept implementation. Since IMA measures each and every executable loaded for execution and configuration files on the HD target machine platform (client's machine platform in this case) and then report these measurements to the challenging party (CS) located at the HD IdP machine, the HD client's machine platform need to be modified to perform the measurement and report it for verification. For this purpose, a Java based ACAgent_Daemon is developed that is responsible for performing the following tasks:

- It listens to the incoming attestation request from the CorroborationAgent (CAgent) and responds accordingly.

- It requests the TPM to execute a quote operation over the PCR values, which store the measurements calculated by IMA and the received nonce from the challenger. The quote operation means that the TPM digitally sign the PCR10 value by its AIK. How this AIK is created will describe in the implementation section.

- During attestation process it reads the Stored Measurement Log (SML) for reporting to the challenger.

- It processes the incoming SAML requests and outgoing SAML responses during the attestation process.

The ACAgent_Daemon consists of different attestation collectors that perform the above mentioned tasks. The ACAgentt has its own operations to perform (Figure 5.1). During the attestation process, when the ACAgent_Daemon receive an attestation request from the challenger, it first calls the AttestClientPCR. The AttestClientPCR is responsible for performing the TPM quote operation. The attestation request contains a nonce (i.e., a random number), the AttestClientPCR takes this nonce and send it to the TPM to perform the quote over the nonce and PCR10 (in this work PCR10 is used by IMA describe in Section 2.8.3). The AttestClientPCR encodes the quote as an XML node and populates SAML response.

Afterwards, the ACAgent_Daemon initiates the AttestClientSML to read and extract the SML from the file system. IMA stores the SML in a specialized file system on the hard disk. The file system is known as securityfs. The AttestClientSML extracts the SML from securityfs and encodes it as XML node. For the sake of pithiness, the quote over PCR10 and SML is termed as trust tokens. Once the attestation collector agents collect these trust tokens, the ACAgent_Daemon encodes these tokens into a single SAML response. The resulting SAML assertion is known as attestation response. The attestation response is returned to the challenging party i.e., CS for the process of verification.

b) Home Domain Identity Provider Machine Modification

The Shibboleth Identity Provider is, part of the HD organization, and its implementation is primarily based on *Java Servlets* which are built on top of the Spring Framework. The HD IdP machine consist of a number of components that are defined as interoperable XML data structures. The HD IdP machine uses these XML configurations to resolve different attributes related to a HD user. The *LoginHandler* component of the HD IdP machine is responsible to define the AuthN mechanism (e.g., username/password using LDAP in this work). First the HD user is authenticated using the above mentioned mechanism then the requested attributes related to the HD user are released.

The IdP uses different AttributeResolvers to retrieve different attributes related to a particular authenticated user. Since the HD IdP machine does not store user's credentials and attributes, it just make use of these credentials and attributes that are stored in separate data stores e.g., RDBMS or LDAP. The AttributeResolver uses a specialized component known as DataConnectors to communicate with these data stores. Depending upon the nature of each data store, the DataConnectors are responsible for communicating with the data store in such a format that is understandable by the particular data store. Furthermore, the DataConnectors are also responsible for mapping the returned attributes from the data stores to such a structure that is understandable by the AttributeResolvers.

Shibboleth framework provides different default DataConnectors for communicating with different data stores (e.g., LDAP connector, Relational Database connector etc.). In the architecture (Figure 5.1), for the AuthN mechanism, the LDAP connector is used to communicate with the underlying LDAP server but the default AttributeResolvers and DataConnectors (DCs) cannot be used to accommodate the integrity verification attributes related to both the HD client and IdP machines platforms. For this reason, own AttributeResolver is developed which is known as MutualIntegrityResolver (MIR) and a DC known as MutualIntegrityProviderDataConnector (MIPDC). The MIR communicates with the

MIPDC to populate the MutualPlatformIntegrity (MPI) attribute that contains the mutual attestation result performed for HD client and IdP machines platforms.

As mentioned earlier, that in order to achieve a high degree of mutual trust among the entities involved in federated identity and resource or service access environment, a mutual attestation technique is proposed that will first perform the integrity verification of the HD client's machine platform. If the HD client's machine platform is in a trustworthy state then the integrity verification of the HD IdP's machine platform will be performed. If both the HD entities mutual attestation resulted in true – assuring that both platforms are in trustworthy state – only then the resource will be released to the HD user. To achieve this, the MIPDC calls the CS and sends the HD client's machine IP address to the CS to perform its integrity verification. If the attestation result is true then the MIPDC calls the CS again to perform the attestation of the HD IdP machine. The CS collects both the HD entities attestation results and returns it back to the MIPDC. The connector then populates the MPI attribute with the result and returns it to the MIR.

In PUSTPF architecture the actual challenging party (that initiates and performs the attestation) is the CS. The CS may reside as a separate entity or as part of the HD IdP machine platform. In order to perform the HD machines mutual attestation process, the CS requires the trust tokens be returned by the HD client and IdP machines respectively. The previous section described how these trust tokens are collected at the HD client's machine platform. The mechanism to collect trust tokens for the HD IdP machine platform is the same. The same ACAgent_Daemon is used to listen to the attestation request from the CS and respond with the trust tokens to HD IdP machine. The AttestIdPPCR and AttestIdPSML are used to collect the trust tokens. This concludes the measurement portion of IMA in the architecture (Figure 5.1). The verification of these measurements is performed at the CS located at HD IdP machine to assure that the HD target machine platform is in a trustworthy state.

5.2.2.2 Validation Phase (VP)

In this phase of the architecture (Figure 5.1), the CS plays its role for performing verification of the HD client and IdP machines platforms measurements to assure the trustworthiness and security of the HD machines platforms. The machines mutual attestation process starts after the successful AuthN of the HD user is performed by the HD IdP machine. After HD user AuthN, the HD IdP machine has to release HD user's attributes to the FD SP. Since in PUSTPF architecture the mutual machines integrity validation of the involved HD machines platforms is carried-out, the MIPDC residing at the HD IdP machine sends an attestation request (including HD client's machine IP address) to the CS to start the HD machines mutual attestation. The AttestationRequester_ Module at the CS generates a random number i.e., nonce and include this nonce in an attestation request. It then sends this request to the HD client machine (since the client's machine platform attestation is performed). The rationale behind including nonce in the attestation request is to ensure the freshness of the attestation request. This will counter to several replay attacks.

The ACAgent_Daemon running on the HD client's machine platform generates an attestation response as discussed in the section 5.2.2.1 (a). The attestation response is returned to the CS and it contacts different specialized Attestation Validation Agent (AVAgent) components to ensure the trustworthiness of the trust tokens and thus ensuring trustworthiness and security of the HD target machine platforms. The AVAgent uses different types of validation components according to the nature of the trust tokens namely ValidationofReceivedNonce (VRN), ValidationofReceivedPCR (VRPCR), ValidationofReceivedSML (VRSML) and ValidationofReceivedCertificate (VRCertificate). In proposed mutual attestation technique for federated identity and resource or service access the validation components first asses the HD client machine platform security and trustworthiness. If successful then it will asses the HD IdP machine platform security and trustworthiness. The whole validation process is described in (a) to (d) below:

a) Validation of Received Nonce (VRN)

The nonce received in the attestation response is checked against the nonce which is sent by the CS to the HD client machine in the attestation request. If the received nonce matches the one which is sent by the CS, then it assure the freshness of the nonce and hence the attestation response is legitimate and not a reply attack.

b) Validation of Received PCR (VRPCR)

The attestation response includes a quote over the PCR10. The quote is performed by the TPM. The PCR10 contains aggregate value of all the hashes measured by the IMA and signed by the TPM. The VRPCR utilizes the SML included in the attestation response to re-calculate the aggregate value of the PCR10. It then matches the received PCR10 value with its re-calculated aggregate PCR10 value. Therefore, if received PCR10 value is identical to the calculated aggregate PCR10 value then this means that SML values are not altered/ tampered (i.e. dishonest) by a malevolent software or man-in-the-middle attack. This way, if an eavesdropper somehow manages to modify the SML in the attestation response, the value of the PCR10 (signed by TPM) will be different from the one which is re-calculated by the CS using the same SML used by the TPM.

c) Validation of Received SML (VRSML)

The third and final check at the CS is performed when the CS calls the VRSML for validation process. The VRSML extracts the SML from the attestation response and iterates through all the entries in the SML. It checks whether the hashes in the SML that represents the executables and configuration files loaded for execution on the target platform is known-good hashes or not. To realize this feature, the CS creates and manages a database of known-good hashes. The VRSML matches each and every hash in the SML against the database. If all of the hashes successfully matched to the known-good hashes in the database then it is considered that the executables running on the HD target platform are trusted and hence the platform is considered as trustworthy and secure.

However, maintaining such a database is not a trivial task. The CS needs to be aware of the executables and configuration files of all the platforms that are to be verified. In PUSTPF architecture, the CS only performs the integrity verification of the HD target platforms within the boundaries of an organization; therefore the CS requires keeping the record only for a limited number of executables and configurations.

d) Validation of Received Certificate (VRCertificate)

The CS also uses a specialized component for validating the certificates used in the attestation process. During the attestation process the TPM residing at the HD target platform (i.e., client or IdP) creates an AIK for signing the PCR value (known as quote operation). The TPM uses its Endorsement Key (EK) in the AIK creation. The private portion of the EK is only known to the particular TPM and it never leaves the TPM. The public portion of the EK is registered with a certification authority known as PrivacyCA. This registration is done by the owner of the TPM. When the TPM is asked to perform the quote operation over the PCR values during the creation of attestation response, the TPM sends the public portion of the EK and AIK to the PrivacyCA to confirm the validity of these keys. Since the PrivacyCA is assumed to know the public portion of the TPM, it checks the particular EK against its certificate list and if found, it signs the AIK with the public portion of the EK. All this procedure is done only to ensure that a genuine TPM signs the attestation response and not a fake one.

The TCG has provided a standard implementation of the PrivacyCA. It can be an authorized Certification Authority or an implementation of the standard PrivacyCA for an organization or a federation. In proposed architecture, since there is no authorized PrivacyCA available, the standard implementation of the PrivacyCA is used. During the validation phase the CS performs the certificate validity check in order to confirm that the attestation response that the CS received was from a genuine TPM and was not faked.

Once all the validators have performed their respective of validation, the CS then combined the results and send it to the MIPDC residing at the IdP of the HD organization. In this work, the mutual attestation of the HD client machine as well as IdP machine platforms are carried out, the MIPDC checks the attestation result of the HD client machine. If the HD client machine attestation resulted in true (i.e., the HD client’s machine platform is in a trustworthy and secure state), the MIPDC sends a new attestation request to the CS (including HD IdP’s machine IP address) to perform integrity validation of the HD IdP’s machine platform. The same process of integrity verification is followed by the CS for validating the integrity of the HD IdP’s machine platform as discussed above for the HD client’s machine platform.

When both the attestation processes mentioned above are completed the MIPDC again checks the attestation result from the CS and if both the results are true (i.e., both the HD client and IdP machines platforms are in a trusted state), it passes the true result to the MIR. If any of the attestation resulted in false it will make the HD client and IdP machines mutual attestation to false, thus the MIPDC will pass false in the result to the MIR. The MIR populates the MPI attribute with the result as true or false depending on the result of the mutual attestation. The HD IdP machine then releases the MPI attribute along with any other attributes that are mentioned in the AttributeReleasePolicy of the HD IdP machine. Finally, the HD IdP machine releases these attributes to the FD SP, which verifies the released attributes against its access decision or organizational policies and makes an access decision to allow or to deny access to a particular resource by the HD client machine.

5.2.3 Comprehensive System Architecture Protocol

The section 5.2.1 to 5.2.2 describes the PUSTPF architecture for federated identity and resource or service access using mutual attestation technique. Those sections also described the modifications needed to be made to incorporate IMA based mutual attestation technique in the Shibboleth framework. This section describes the information flow among different entities in PUSTPF architecture. Figure 5.1 (section 5.2), shows the comprehensive architecture of the framework and the information flow among the entities.

The proposed architecture steps shown in Figure 5.1 described below:

1. The user located in the HD organization (www.dod.org) opens a browser and request a resource access sited at the FD organization (www.dor.org). After receiving the request the FD SP then checks to see if the requested resource is an unprotected entity. If yes, then it simply releases the resource to the user (Step 1 & 2).

2. In the second case if a resource is a shielded resource then the FD organization SP forwards the HD organization user’s machine browser to the DS in order to

select his/her HD IdP machine (in case of multiple IdPs) and to get authenticated. The user selects his HD IdP machine and the DS redirects the user's machine browser to the selected HD organization IdP (Step 3 & 4).

3. The WSSO AuthN service at the HD IdP machine brings a HD user to a selected HD organization IdP log-in portal. The HD user then enters his/her basic AuthN credentials (username/password) and the HD organization IdP machine then verify the user entered credentials against the entries in the LDAP (Step 5 & 6).

4. Upon successful AuthN the WSSO service contacts the Login Handler Service residing at HD IdP machine to create a session for a HD user and sends a handler to the user's machine browser. This handler is used by the WSSO service for providing WSSO features for this particular user. If any other FD SP in the federation request the HD IdP to authenticate this particular user, the WSSO service at the HD IdP checks for any pre-existing session and handler for this user and then performs the next step without prompting the user with login page. In this way, during the session time-period the HD IdP shall provides WSSO AuthN for the authenticated user across different organizational boundaries (i.e., different FD SPs) (Step 7 & 8).

5. The user's browser sends this handler to the mod_shib module residing on the FD SP in order to proof that his AuthN is successful (Step 9).

6. Since FD SP organization requires different attributes about a user, the mod_shib module sends a request to the Shibboleth daemon (shibd) running on the FD SP to request the attributes about a particular user from the HD IdP machine (Step 10).

7. The shibd sends a request to the Attribute Resolver module at the HD IdP machine to release the attributes related to the HD client and IdP machines mutual attestation. (This request contains the authenticated user's AuthN handler in order to know which HD user's client's and IdP machine attributes are requested) (Step 11).

8. The Attribute Resolver checks its configurations and collects some of the attributes (e.g., name, common name etc.) about the HD client machine from the underlying LDAP server using the default DataConnectors. Since in the architecture, the mutual integrity validation of the HD client and IdP machines (i.e. HD target machines) platforms. So for this purpose a new MIPDC is created in this research work. The Attribute Resolver contacts the MIPDC for the required MPI attributes (Step 12).

9. The MPI attribute needs the mutual attestation to be performed. The MIPDC first send an attestation request to the CS to perform attestation of the HD client's machine platform (Step 13).

10. The AttestationRequester_Module residing on the CS (i.e. Part of a HD organization IdP machine) then generates a nonce and creates an attestation request to send to the ACAgent_Daemon running on the HD client's machine platform (Step 14).

11. The ACAgent_Daemon at the HD client machine initiates the AttestClientPCR and AttestClientSML to collects the trust tokens (PCR10 & SML) from the HD client machine TPM. The HD client machine TPM performs the quote operation over PCR10 value. The ACAgent_Daemon collects these tokens and sends it back to the AttestationRequester_Module at the CS (Step 15 & 16).

12. Upon receiving the trust tokens for the HD client machine, the CS initiates the ValidationofReceivedPCR and ValidationofReceivedSML to validate the trust tokens respectively. The CS also checks the certificate validity by contacting the PrivacyCA and uses its validation database to verify the hashes in the SML against the hashes in the verify database. In addition to that CS also performs the ValidationofReceivedNonce by comparing the received nonce that it sent in the earlier attestation request message. (Step 17).

13. The CS combines the validated results and encodes it as XML node. This result is then returned to the MIPDC at HD IdP machine (Step 18).

14. In this work the mutual attestation of the HD client and IdP machine platforms is carried out, then the MIPDC shall checks the attestation result of the HD client machine. If the result is false (i.e., HD client's machine platform is not in a trustworthy state) then the Steps 20 – 25 (Figure 5.1) will not be processed (Step 19).

15. If the result is true (i.e., HD client's machine platform is in a trustworthy state) then the MIPDC again sends an attestation request to the CS to mutually verify the integrity of the HD IdP's machine platform (Step 20).

16. The AttestationRequester_Module residing on the CS part of the HD IdP machine generates a nonce and creates an attestation request to the ACAgent_Daemon running on the HD IdP's machine platform (Step 21).

17. The ACAgent_Daemon at the HD IdP machine initiates the AttestIdPPCR and AttestIdPSML to collects the trust tokens (PCR10 & SML) from the HD IdP machine TPM. The HD IdP machine TPM performs the quote operation over PCR10 value. The ACAgent_Daemon collects these tokens and sends it back to the AttestationRequester_Module at CS part of the HD IdP machine (Step 22 & 23).

18. Upon receiving the trust tokens for the HD IdP machine, the CS initiate the ValidationofReceived PCR and ValidationofReceivedSML to verify the trust tokens respectively. The CS then also checks the certificate validity by contacting the PrivacyCA and uses its validation database to verify the hashes in the SML against the hashes in the verify database. In addition to that CS also performs the ValidationofReceivedNonce by comparing the received nonce that it sent in the earlier attestation request message. (Step 24).

19. The CS then combines the validation result and encodes it as XML node. This result is then returned to the MIPDC at the HD IdP machine (Step 25).

20. The MIPDC collects the mutual attestation result and passes this result to the Attribute Resolver at HD organization IdP machine (Steps 26).

21. For HD machines mutual attestation, MIR is created and populates the MPI attribute with the mutual attestation result and sends this attribute to the Shibboleth Daemon (shibd) running on the FD SP machine (Step 27).

22. The shibd passes the attributes to the FD Shibboleth SP module (mod_shib) in order to apply its resource access policies and check the attributes values against its organization policy (Step 28).

23. The mod_shib is responsible to apply organizational resource access policies according to the value of the attributes (e.g., MPI is true or false). It contacts the application that is protecting the sensitive resource and makes the decision to allow or to deny the resource to the HD client machine on the basis of HD client and IdP machines successful mutual attestation result (Step 29).

24. According to the access decision in step 29, the resource is either released to the HD user or the access is denied for this particular resource (Step 30).

5.3 System Architecture Design

This section describes PUSTPF functionalities through different diagrams such as use-case, activity, class, sequence diagram, packages, and algorithms flowcharts (5.3.6):

5.3.1 Use-Case Diagrams

The use-case diagram is used in capturing the system functional requirements. The functional requirements are the behaviors of the system. The behavior maybe a single service and a function (or group of services and tasks) that a system is obligatory to carry out. The use-cases described below explore a number of groups of funtions of the system:

5.3.1.1 Use-case Diagram-1

Figure 5.2 illustrates the HD user AuthN using basic AuthN mechanism with the HD IdP machine. The SP role to protect a resource or service it provided.

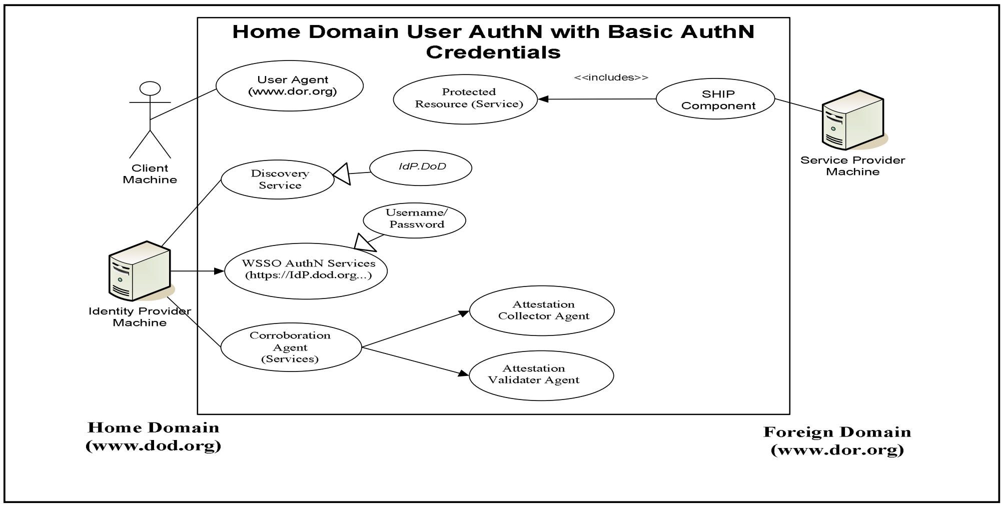

Home Domain User AuthN with Basic AuthN Credentials
Client Machine
User Agent (www.dor.org)
Protected Resource (Service)
<<includes>>
SHIP Component
Service Provider Machine
Discovery Service
IdP.DoD
Username/ Password
WSSO AuthN Services (https://IdP.dod.org...)
Identity Provider Machine
Corroboration Agent (Services)
Attestation Collector Agent
Attestation Validater Agent
Home Domain (www.dod.org)
Foreign Domain (www.dor.org)

Figure 5.2: Use-case diagram-1

5.3.1.2 Use-case Diagram-2

Figure 5.3 shows HD client and IdP machines mutual attestation. This step will execute only if the HD user is successfully authenticated to the HD IdP using his/her basic AuthN credentials.

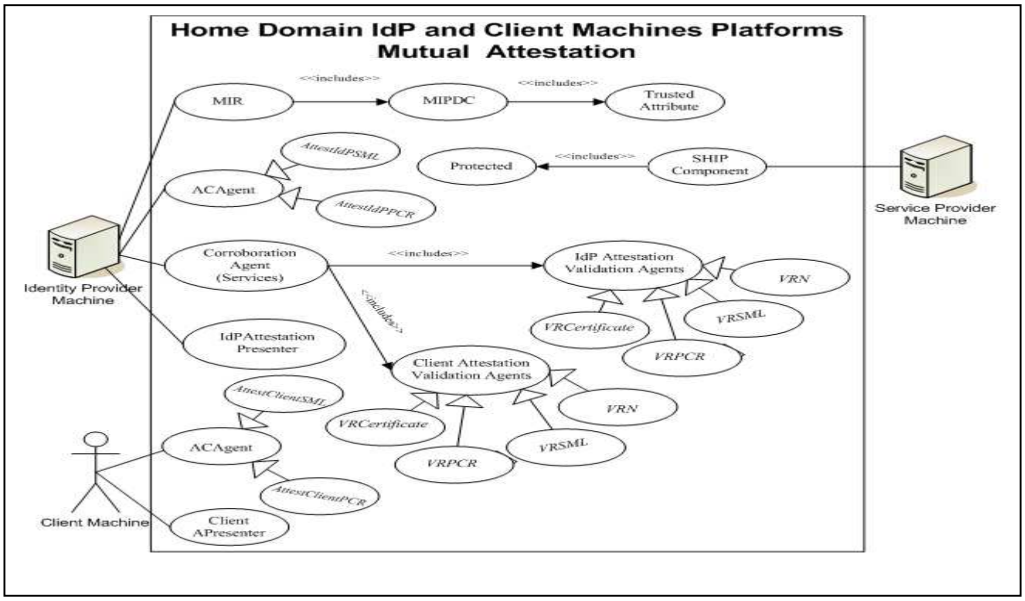


Figure 5.3: Use-case diagram-2

5.3.1.3 Use-case Diagram-3

The use-case diagram given in Figure 5.4 shows the FIAM mode functionalities that the HD and FD organization may carry out.

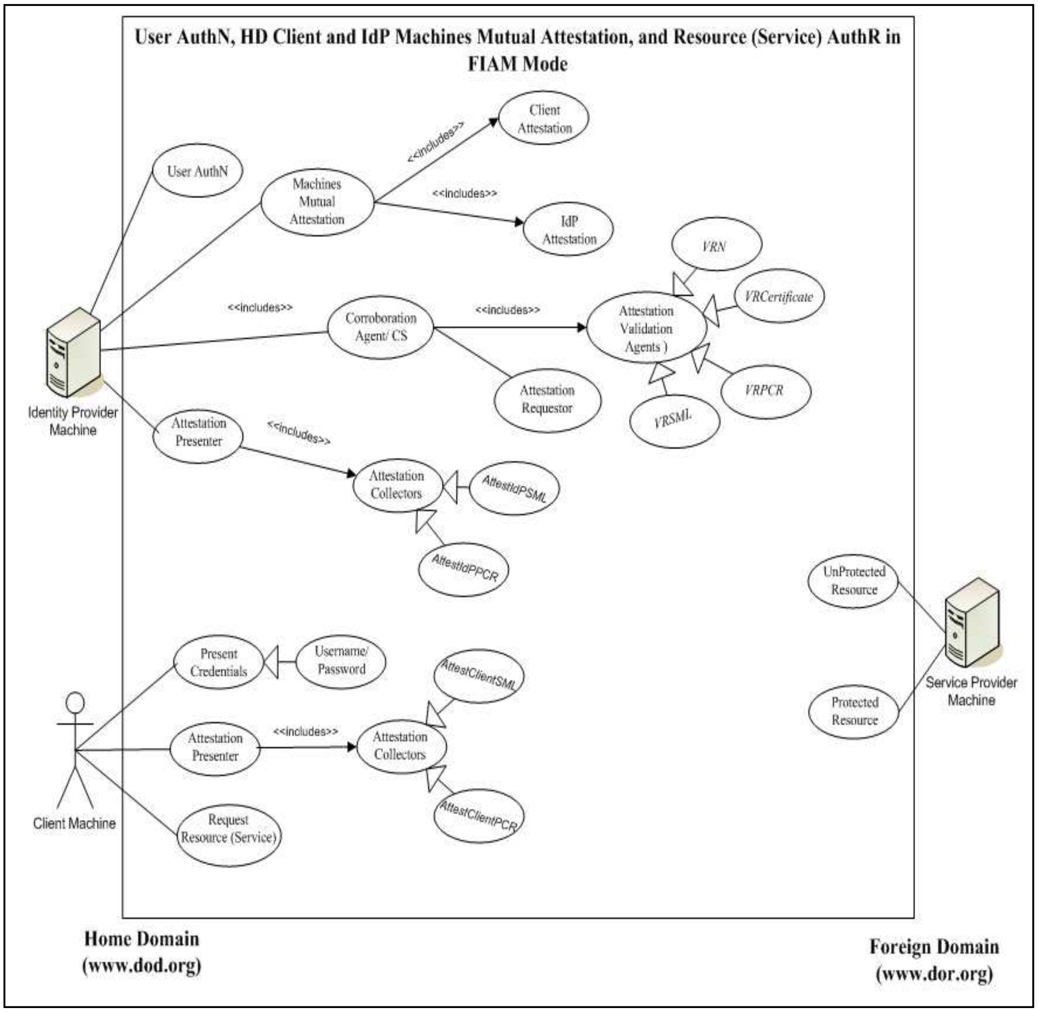


Figure 5.4: Use-case diagram-3

5.3.2 Activity Diagram

An activity diagram represents the system behaviors or activities. The activity diagram for the proposed scheme is given in Figure 5.5. The user may access two different resources (services): (1) the protected resource (services) which requires the HD user AuthN, and the HD client and IdP machines mutual attestation processes' successful execution. (2), the unprotected resource (service) may not require user AuthN and AuthR processes execution. The core of the framework is Mutual Integrity

Provider Data Connector (MIPDC). The MIPDC has three major functions: (1) Extractions of the HD client's and IdP's machines IP addresses, (2) Initiating the attestation process by sending the HD client's and IdP's machines IP to the CS/CAgent and (3) Attestation result collection from the CS and encoding it to populate the MPI attribute.

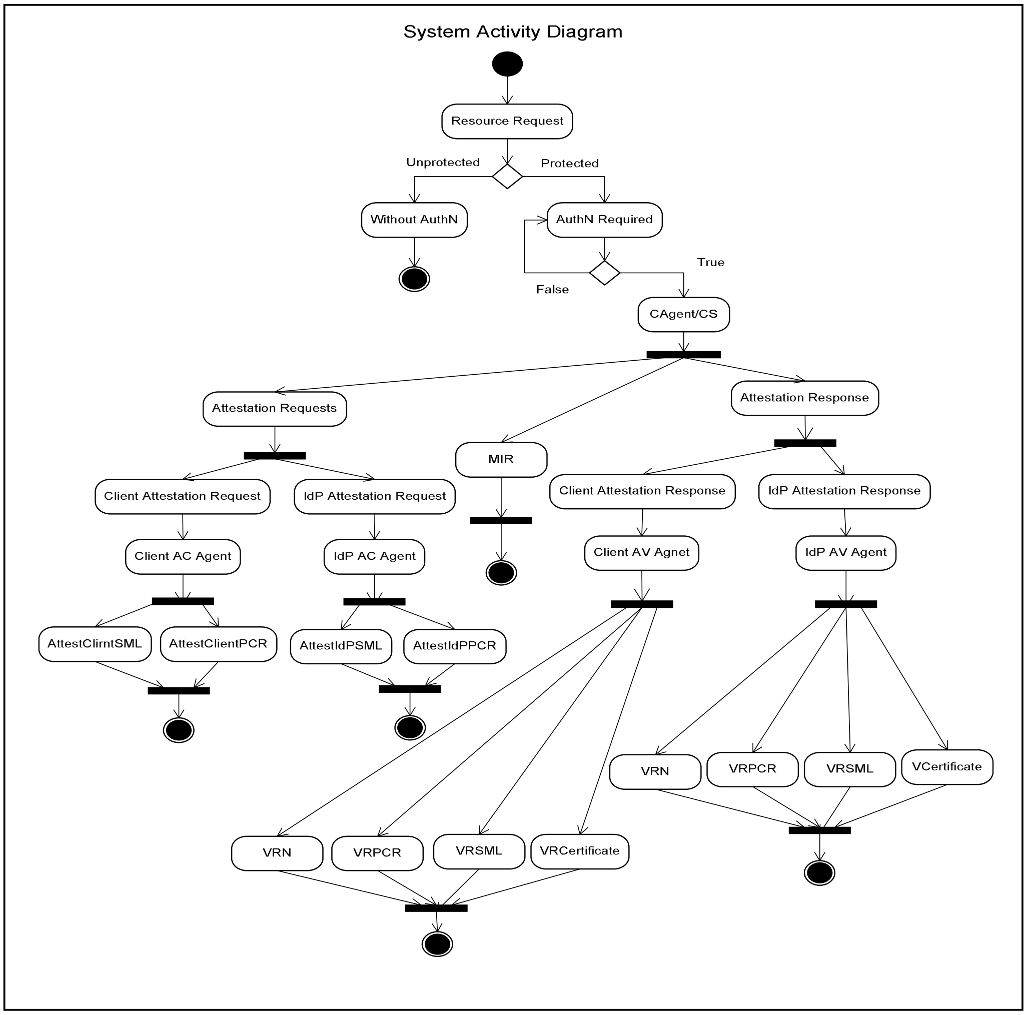


Figure 5.5: Activity diagram

5.3.3 Class Diagram

Figure 5.6 shows the interactions of different classes of the system architecture (section 5.2) and relationship among each other. The MIPDC class has five major roles: (1) Extortion of the HD client machine IP address from requests accessing a protected resource at the HD IdP machine, (2) Kick off the requested target machine attestation procedure by forwarding the target machine IP to the CS, (3) Collects the HD targets machines attestation outcome, (4) Carry out the HD IdP machine attestation and (5) Encoding to populate the attribute-MutualPlatformIntegrity (MPI).

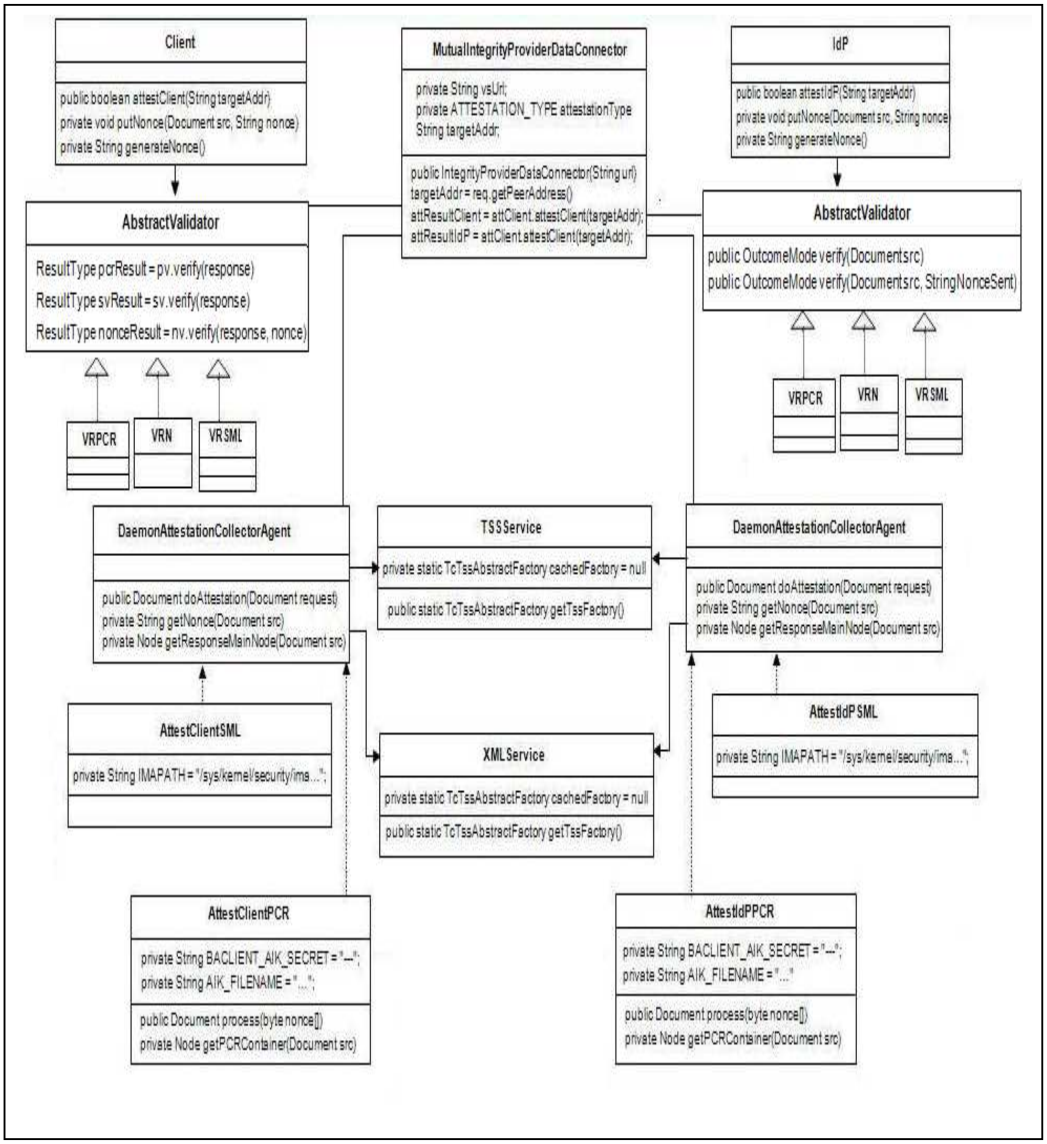

Client
public boolean attestClient(String targetAddr)
private void putNonce(Document src, String nonce)
private String generateNonce()
MutualIntegrityProviderDataConnector
private String vsUrl;
private ATTESTATION_TYPE attestationType
String targetAddr;
public IntegrityProviderDataConnector(String url)
targetAddr = req.getPeerAddress()
attResultClient = attClient.attestClient(targetAddr);
attResultIdP = attClient.attestClient(targetAddr);
IdP
public boolean attestIdP(String targetAddr)
private void putNonce(Document src, String nonce
private String generateNonce()
AbstractValidator
ResultType pcrResult = pv.verify(response)
ResultType svResult = sv.verify(response)
ResultType nonceResult = nv.verify(response, nonce)
AbstractValidator
public OutcomeMode verify(Document src)
public OutcomeMode verify(Document src, String NonceSent)
VRPCR
VRN
VR SML
VRPCR
VRN
VR SML
DaemonAttestationCollectorAgent
public Document doAttestation(Document request)
private String getNonce(Document src)
private Node getResponseMainNode(Document src)
TSS Service
private static TcTssAbstractFactory cachedFactory = null
public static TcTssAbstractFactory getTssFactory()
DaemonAttestationCollectorAgent
public Document doAttestation(Document request)
private String getNonce(Document src)
private Node getResponseMainNode(Document src)
AttestClientSML
private String IMAPATH = "/sys/kernel/security/ima...";
XML Service
private static TcTssAbstractFactory cachedFactory = null
public static TcTssAbstractFactory getTssFactory()
AttestIdP SML
private String IMAPATH = "/sys/kernel/security/ima...";
AttestClientPCR
private String BACLIENT_AIK_SECRET = "---";
private String AIK_FILENAME = "...";
public Document process(byte nonce[])
private Node getPCRContainer(Document src)
AttestIdPPCR
private String BACLIENT_AIK_SECRET = "---";
private String AIK_FILENAME = "..."
public Document process(byte nonce[])
private Node getPCRContainer(Document src)

Figure 5.6: Class diagram

5.3.4 Sequence Diagram

Figure 5.7 illustrates the scheme (section 5.2) sequence diagram in which a user requests a protected resource sited at the FD organization. It also shows the necessary operations which are carried out before the release of a protected resource to the HD organization user.

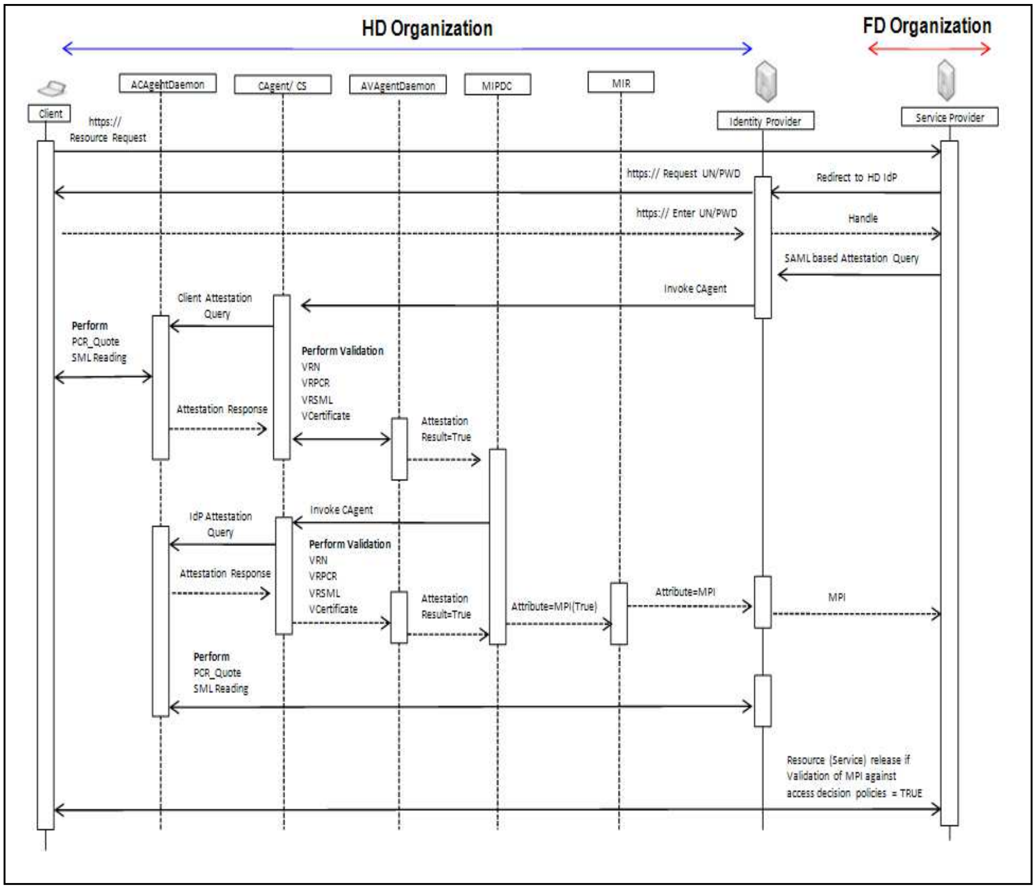


Figure 5.7: Home domain client and idp MA sequence diagram

5.3.5 Packages and Classes

5.3.5.1 Home Domain Client Machine Attestation Collector

The package *pustpf.ima.net.client* consists of *attetstaioncollector* sub-package. The sub-package consists of *AtesstClientSML* and *AttestClientPCR* classes (5.2.2.1 (a)). The agent *DaemonAttestationCollectorAgent* listens to the HD IdP machine attestation request. The *TssService* and *XmlService* provide the trusted computing and Xml related services in attestation process.

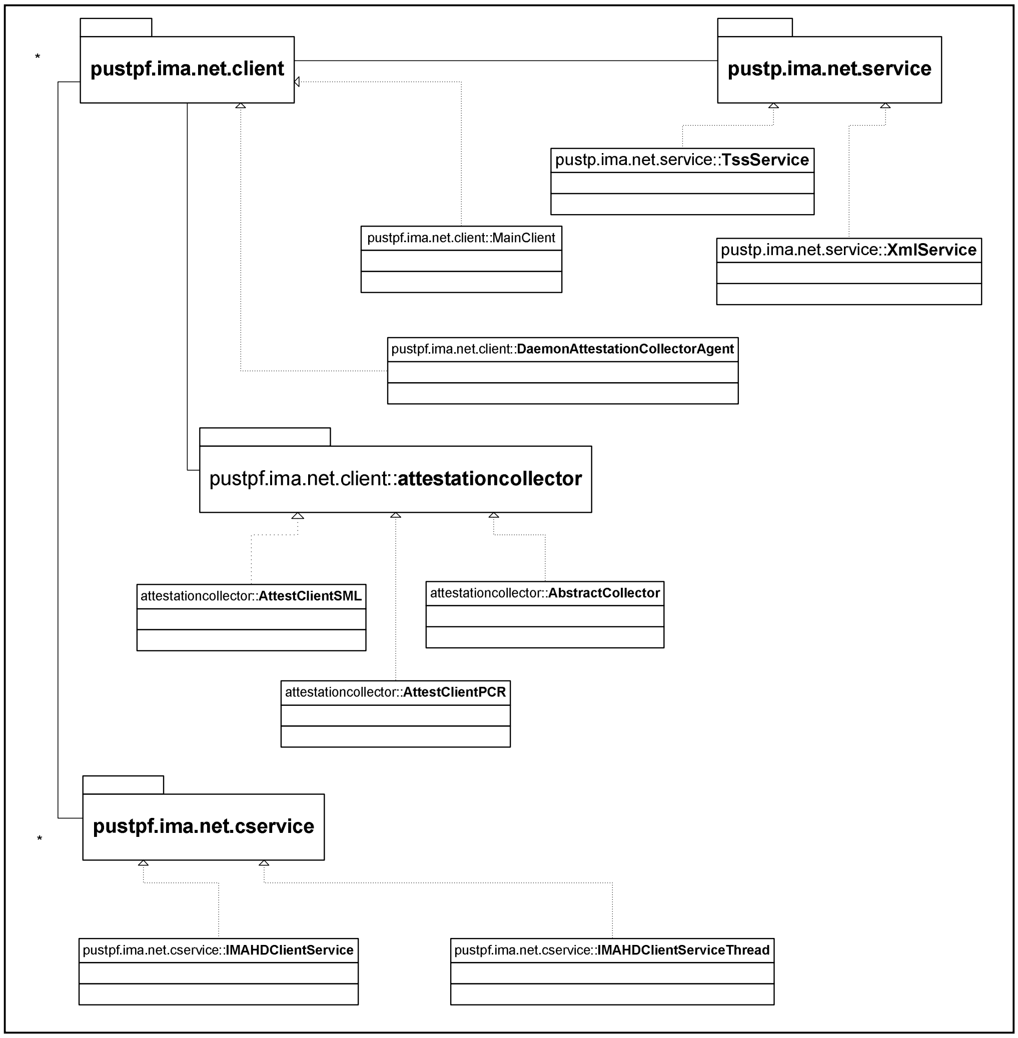

*
pustpf.ima.net.client
pustp.ima.net.service
pustp.ima.net.service::TssService
pustpf.ima.net.client::MainClient
pustp.ima.net.service::XmlService
pustpf.ima.net.client::DaemonAttestationCollectorAgent
pustpf.ima.net.client::attestationcollector
attestationcollector::AttestClientSML
attestationcollector::AbstractCollector
attestationcollector::AttestClientPCR
pustpf.ima.net.cservice
*
pustpf.ima.net.cservice::IMAHDClientService
pustpf.ima.net.cservice::IMAHDClientServiceThread

Figure 5.8: HD client machine attestation collector package

5.3.5.2 Home Domain Identity Provider Attestation Collector

The package *pustpf.ima.net.identityprovider* consists of *attetstaioncollector* sub-package. This sub-package further consists of *AtesstIdPSML* and *AttestIdPPCR* classes (5.2.2.1(b)). The agent *DaemonAttestationCollectorAgent* listens to the HD IdP machine attestation request. The *TssService* and *XmlService* provide the trusted computing and Xml related services in attestation process.

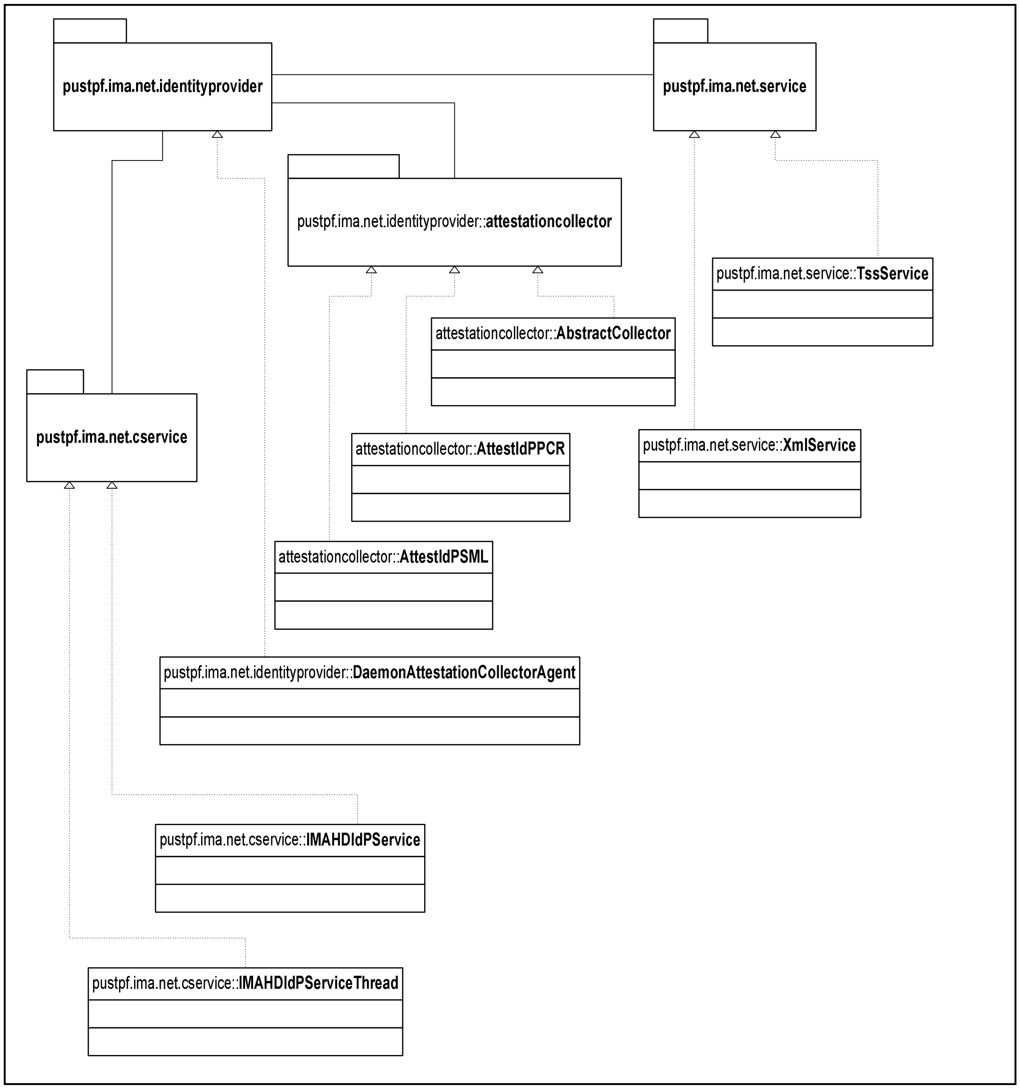

pustpf.ima.net.identityprovider
pustpf.ima.net.service
pustpf.ima.net.identityprovider::attestationcollector
pustpf.ima.net.service::TssService
attestationcollector::AbstractCollector
pustpf.ima.net.cservice
attestationcollector::AttestIdPPCR
pustpf.ima.net.service::XmlService
attestationcollector::AttestIdPSML
pustpf.ima.net.identityprovider::DaemonAttestationCollectorAgent
pustpf.ima.net.cservice::IMAHDIdPService
pustpf.ima.net.cservice::IMAHDIdPServiceThread

Figure 5.9: HD identity provider machine attestation collector package

5.3.5.3 Home Domain Target Machines Mutual Attestation Validation

The package *tbed.pustpf.ima.net.avagent* consists of a sub-package such as *tbed.pustpf.ima.net.avagent::services*. The *tbed.pustpf.ima.net.avagent* package consists of *Target* and *MutualIntegrityProviderDataConnector* classes. The *tbed.pustpf.ima.net.avagent::services* cosnsist of *XmLServices*, *LogHashSlotin* and *LogHashSlotinInquiryProducer* classes. The *XmlServices* class provides Xml related services, whereas the LogHash classess provides the hashing services in attestation process. The *VRN*, *VRPCR* and *VRSML* components are used to validated the returned nonce, PCR value and SML

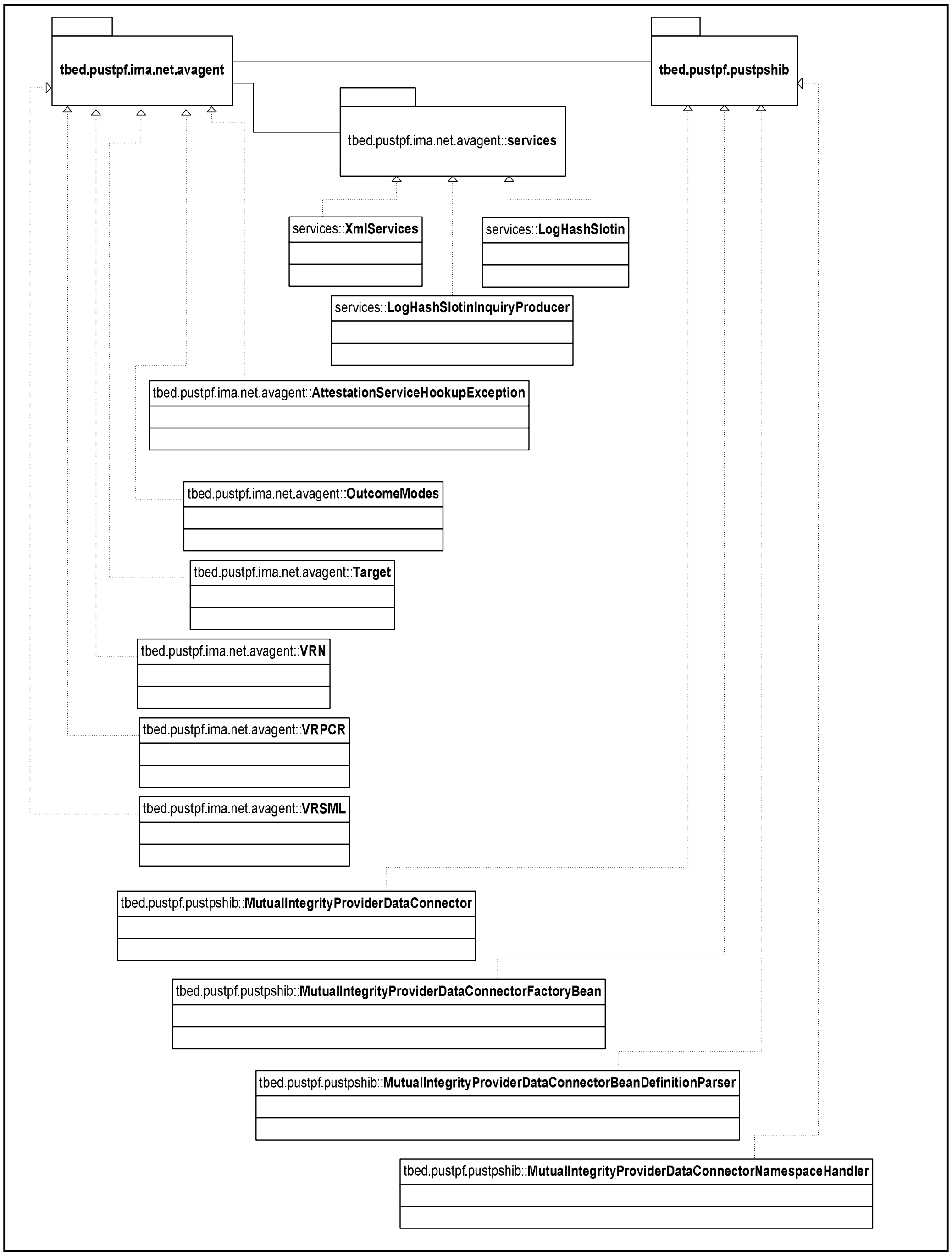

tbed.pustpf.ima.net.avagent
tbed.pustpf.pustpshib
tbed.pustpf.ima.net.avagent::services
services::XmlServices
services::LogHashSlotin
services::LogHashSlotinInquiryProducer
tbed.pustpf.ima.net.avagent::AttestationServiceHookupException
tbed.pustpf.ima.net.avagent::OutcomeModes
tbed.pustpf.ima.net.avagent::Target
tbed.pustpf.ima.net.avagent::VRN
tbed.pustpf.ima.net.avagent::VRPCR
tbed.pustpf.ima.net.avagent::VRSML
tbed.pustpf.pustpshib::MutualIntegrityProviderDataConnector
tbed.pustpf.pustpshib::MutualIntegrityProviderDataConnectorFactoryBean
tbed.pustpf.pustpshib::MutualIntegrityProviderDataConnectorBeanDefinitionParser
tbed.pustpf.pustpshib::MutualIntegrityProviderDataConnectorNamespaceHandler

Figure 5.10: HD target machine attestation validator agent package

5.3.6 Flowcharts

The section 5.3.6.1 to 5.3.6.5 discusses the use-cases, activity and sequence diagrams algorithm in the form of flowchart.

5.3.6.1 Flowchart 1: Use-case Digram -1

Figure 5.11 shows the flow of user basic AuthN process in Shibboleth architecture. The user keyin first his/her basic AuthN credentials (e.g., UN/PWD) to the login screen prompted by the HD IdP. The details of Shibboleth basic user AuthN process explained in section 2.5.3.2 (d).

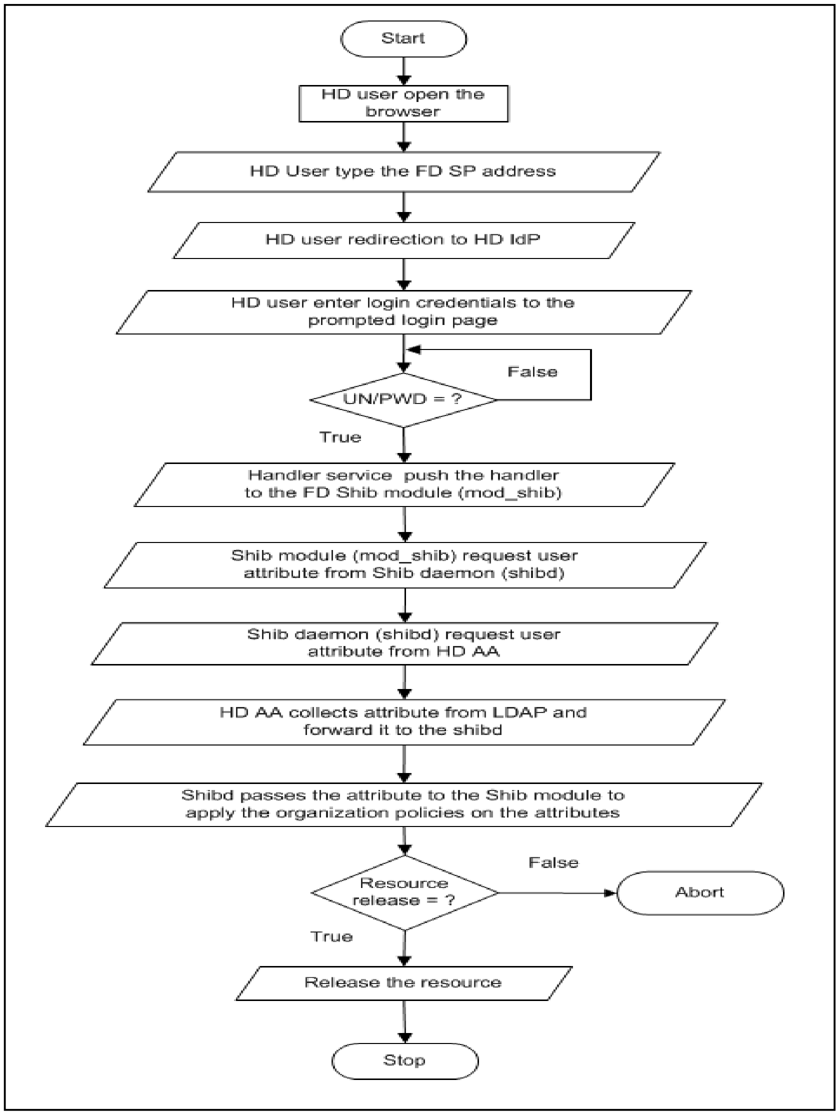

Start
HD user open the browser
HD User type the FD SP address
HD user redirection to HD IdP
HD user enter login credentials to the prompted login page
False
UN/PWD = ?
True
Handler service push the handler to the FD Shib module (mod_shib)
Shib module (mod_shib) request user attribute from Shib daemon (shibd)
Shib daemon (shibd) request user attribute from HD AA
HD AA collects attribute from LDAP and forward it to the shibd
Shibd passes the attribute to the Shib module to apply the organization policies on the attributes
False
Resource release = ?
Abort
True
Release the resource
Stop

Figure 5.11: HD user AuthN with basic AuthN credentials

5.3.6.2 Flowchart 2: Use-case Diagram-2

Figure 5.12 shows the flow of HD IdP and client platforms mutual attestation process. The mutual attestation process executes after the successful run of user basic AuthN process. The details of mutual attestation process explained in section 5.2.2.1.

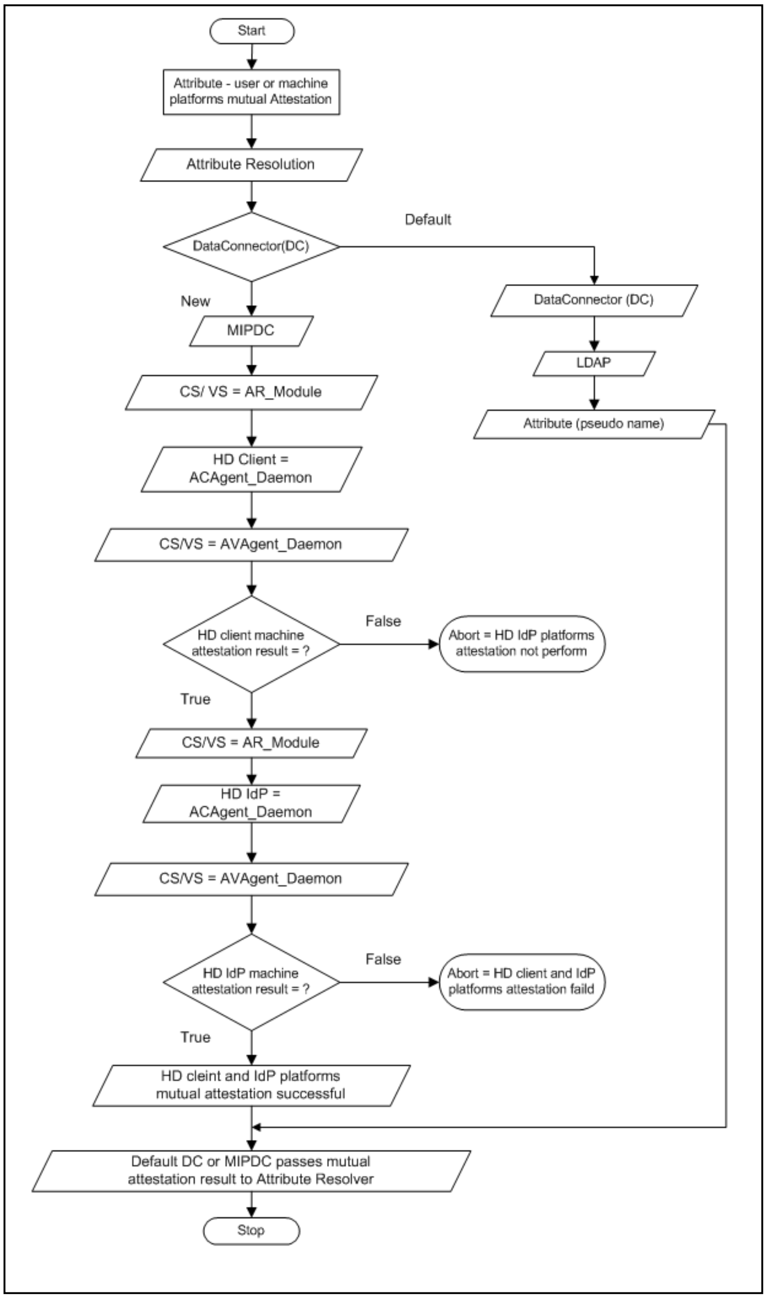

Start
Attribute - user or machine platforms mutual Attestation
Attribute Resolution
DataConnector(DC)
Default
New
DataConnector (DC)
MIPDC
LDAP
CS/ VS = AR_Module
Attribute (pseudo name)
HD Client = ACAgent_Daemon
CS/VS = AVAgent_Daemon
HD client machine attestation result = ?
False
Abort = HD IdP platforms attestation not perform
True
CS/VS = AR_Module
HD IdP = ACAgent_Daemon
CS/VS = AVAgent_Daemon
HD IdP machine attestation result = ?
False
Abort = HD client and IdP platforms attestation faild
True
HD cleint and IdP platforms mutual attestation successful
Default DC or MIPDC passes mutual attestation result to Attribute Resolver
Stop

Figure 5.12: HD IdP and client machine platforms mutual attestation

5.3.6.3 Flowchart 3: Use-case Diagram-3

Figure 5.13 shows the flow of the HD user AuthN and HD IdP and client machine platforms mutual attestation processes. The details of the HD user AuthN by basic AuthN credentials and HD client and mutual attestation processes explained in section 5.2.3.

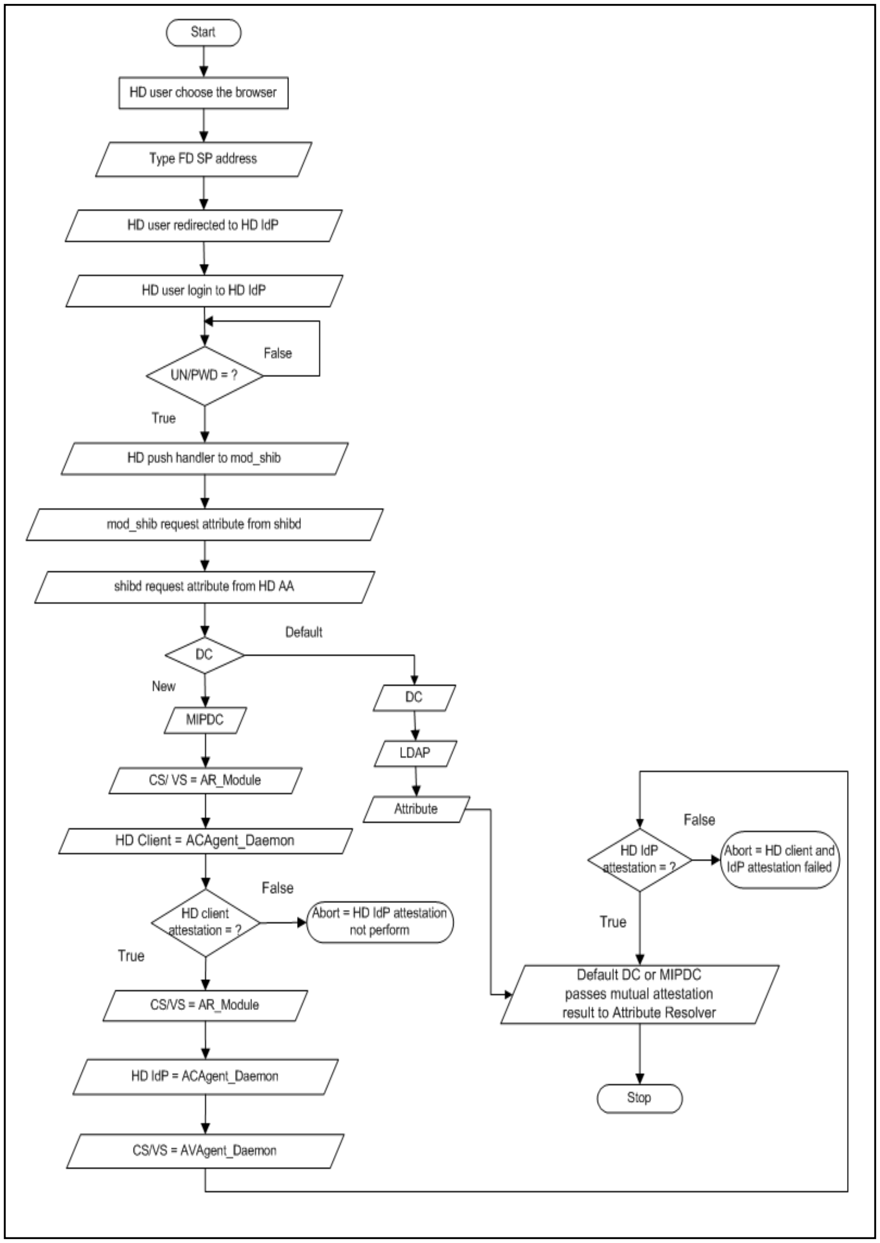

Start
HD user choose the browser
Type FD SP address
HD user redirected to HD IdP
HD user login to HD IdP
False
UN/PWD = ?
True
HD push handler to mod_shib
mod_shib request attribute from shibd
shibd request attribute from HD AA
Default
DC
New
MIPDC
DC
LDAP
Attribute
CS/ VS = AR_Module
HD Client = ACAgent_Daemon
False
HD client attestation = ?
Abort = HD IdP attestation not perform
True
CS/VS = AR_Module
HD IdP = ACAgent_Daemon
CS/VS = AVAgent_Daemon
False
HD IdP attestation = ?
Abort = HD client and IdP attestation failed
True
Default DC or MIPDC passes mutual attestation result to Attribute Resolver
Stop

Figure 5.13: HD user AuthN, client and IdP mutual attestation processes

5.3.6.4 Flowchart 4: Activity Diagram

Figure 5.14 shows the flow of the HD user AuthN, HD IdP and client machine platforms mutual attestation and AuthR processes. The details of the HD user AuthN by basic AuthN credentials, HD client and mutual attestation and AuthR via MPI attribute explained in section 5.2.2.1.

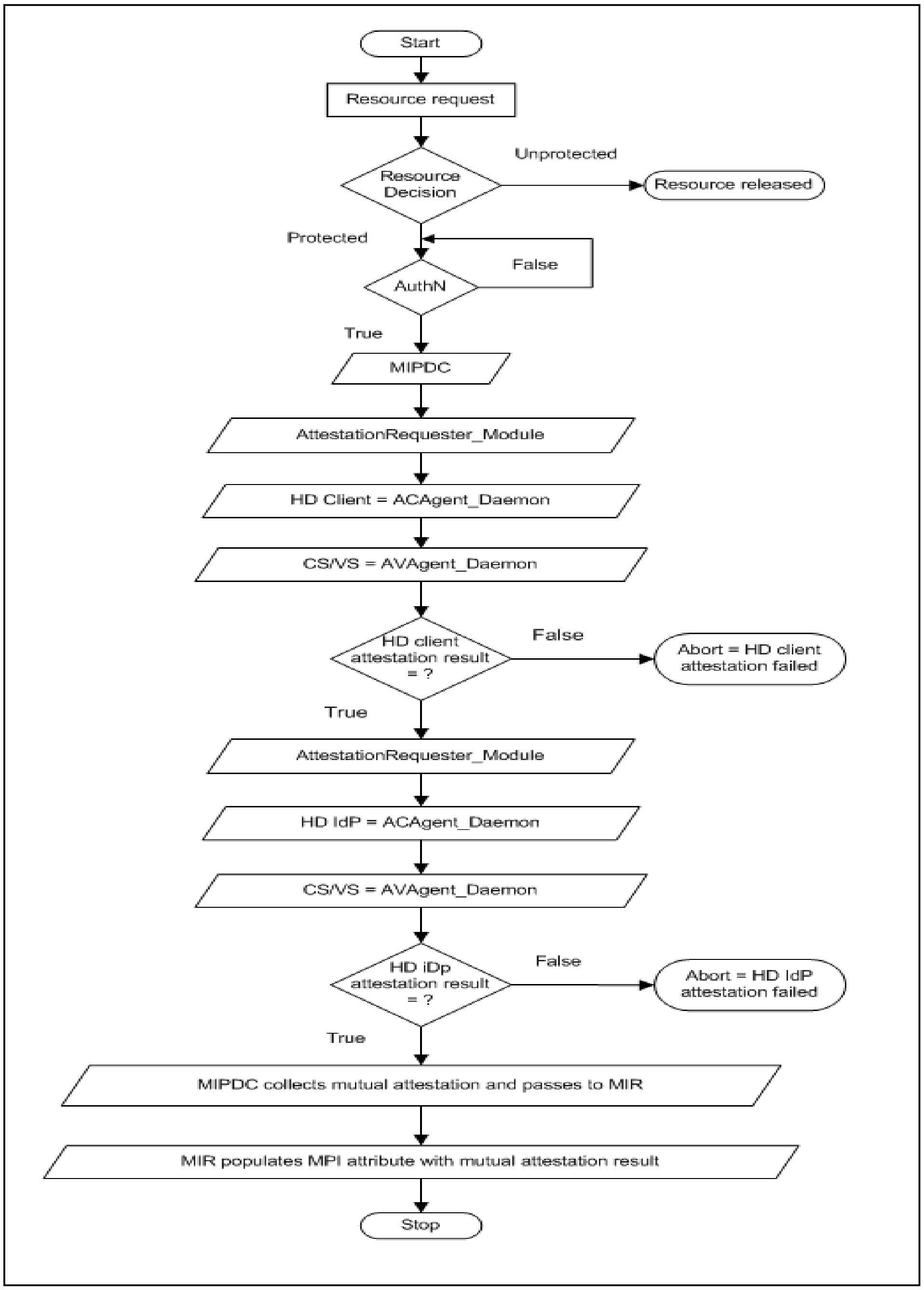

Start
Resource request
Resource Decision
Unprotected
Resource released
Protected
False
AuthN
True
MIPDC
AttestationRequester_Module
HD Client = ACAgent_Daemon
CS/VS = AVAgent_Daemon
HD client attestation result = ?
False
Abort = HD client attestation failed
True
AttestationRequester_Module
HD IdP = ACAgent_Daemon
CS/VS = AVAgent_Daemon
HD iDp attestation result = ?
False
Abort = HD IdP attestation failed
True
MIPDC collects mutual attestation and passes to MIR
MIR populates MPI attribute with mutual attestation result
Stop

Figure 5.14: HD user AuthN, client and IdP mutual attestation and AuthR process

5.3.6.5 Flowchart 5: Sequence Diagram

Figure 5.15 shows the flow of the HD organization and FD organization processes such as: (1) HD organization user AuthN and IdP and client machine platforms mutual attestation and (2) FD organization AuthR. The details of the HD organization and FD organization processes explained in section 5.2.3.

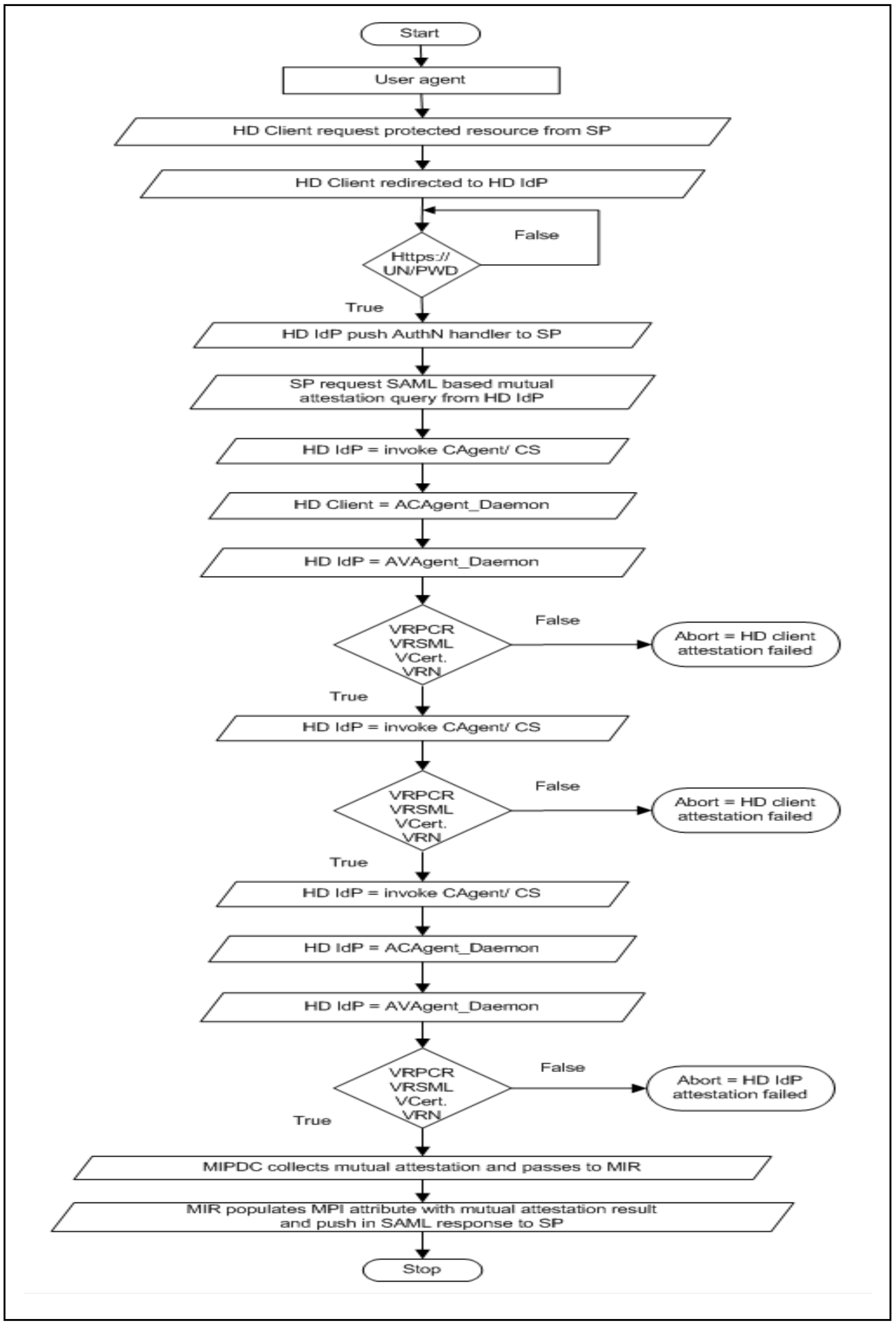

Start
User agent
HD Client request protected resource from SP
HD Client redirected to HD IdP
Https:// UN/PWD
False
True
HD IdP push AuthN handler to SP
SP request SAML based mutual attestation query from HD IdP
HD IdP = invoke CAgent/ CS
HD Client = ACAgent_Daemon
HD IdP = AVAgent_Daemon
VRPCR VRSML VCert. VRN
False
Abort = HD client attestation failed
True
HD IdP = invoke CAgent/ CS
VRPCR VRSML VCert. VRN
False
Abort = HD client attestation failed
True
HD IdP = invoke CAgent/ CS
HD IdP = ACAgent_Daemon
HD IdP = AVAgent_Daemon
VRPCR VRSML VCert. VRN
False
Abort = HD IdP attestation failed
True
MIPDC collects mutual attestation and passes to MIR
MIR populates MPI attribute with mutual attestation result and push in SAML response to SP
Stop

Figure 5.15: HD organization and FD organization processes sequence diagram

5.4 Summary

In this chapter, the system architecture desgin is presented. The system architecture which includes different entities or stakeholders and comprehensive PUSTPF protocol architecture discussed in the second section. The architecture desgin of the proposed scheme was explained through different use-case, activity, class, sequence diagrams. This chapter also provided the diagrams corresponding algorithms in the form of flowchats.

CHAPTER 6

EXPERIMENTAL DESIGN, RESULTS AND DISCUSSION

6.1 Introduction

This chapter explains the experimental desgin, experiment outcomes obtained from the proof of concept for the trustworthy mutual attestation protocol and the practicable cohesive security, trust and privacy framework test-bed prototype. The second section describes the experimental desgin and explain how the IMA [15] based mutual (bidirectional) attestation protocol is integrated in a standard FIAM system (such as the Shibboleth framework). The third section presents the experiment outcomes. It is divided further into five sub-sections. The first sub-section describes the proof of concept for the trustworthy mutual attestation result analysis. The practicable cohesive security, trust and privacy framework test-bed result analysis is provided in the second sub-section. The third sub-section offers a comparison analysis of the emergent and practicable unified frameworks. The analysis of the mutual attestation performance comparison is presented in the fourth sub-section. The fifth sub-section describes the comparative analysis (such as security, trust, privacy and practicability) of the PUSTPF with the other works with the existing works. This chapter is summarized in the fourth section.

6.2 Experimental Design

In realizing PUSTPF for federated identity and resource or service access different technologies are used. In the experiment of the PUSTPF two different technologies are used in a broader spectrum. For user AuthN and AuthR in this Shibboleth an open source framework was used to provide a secure WSSSO mechanism for AuthN and AuthR. To establish mutual trust among different entities in a federated identity and resource access scenario, in this work TCG-based IMA is used in mutual attestation protocol to measures and verifies the integrity of different platforms. The Shibboleth framework and IMA both depends on a variety of underlying technologies that need to be implemented and configured. In this section how these technologies are used and implemented in the prototype are discussed. In addition to that how some of the important components of the Shibboleth framework are implemented and configured are also discussed. The core of the proposed model is measuring the mutual integrity of the HD target platforms (i.e. client and IdP machines) for the establishment of trust. Therefore the implementation and configuration of the IMA and its underlying technologies with respect to the Shibboleth framework are discussed.

The implementation and configuration of two important and related entities of the Shibboleth framework (e.g., HD IdP and FD SP) are also discussed in this section. The Shibboleth framework is an open source FIAM Solution for federated identity and resource or services access which is widely adopted in universities, and governmental organizations around the world for online secure resource sharing and collaboration. The source code of the Shibboleth framework is openly available for used or modified for custom implementation. This section describes the Shibboleth IdP and SP installation and configuration process for the prototype implementation and in the later sections will demonstrate how IMA can be incorporated in the Shibboleth framework.

6.2.1 Home Domain Shibboleth Identity Provider Installation

The HD Shibboleth IdP installation steps are described below (the corresponding complete command-line instructions and code is given in Appendix B):

6.2.1.1 Operating System

In PUSTPF test-bed prototype Ubuntu 9.10 OS is used for the HD organization IdP machine platform. The rationale behind selecting this flavor of Linux OS for the HD

IdP machine is that this version can easily support IMA, which will be used for mutual attestation of HD IdP and client machine platforms.

6.2.1.2 Home Domain Identity Provider Pre-requisites

The HD IdP pre-requisites are:

a) Java

The Sun Java 1.6 or later is required to be installed as a prerequisite for the Shibboleth IdP installation. Download the latest version of Sun’s Java (e.g., jdk 1.6), place it in the user’s home directory and install it (Appendix B [1(i)]).

b) Apache Tomcat

Apache Tomcat is used as a Servlet container for the deployment of HD organization IdP. Since Shibboleth IdP is based on Servlet 2.4 specification, so a Servlet container needs to deploy the IdP. For this reason the Apache Tomcat is chosen as the Servlet container (Appendix B [1(ii)]).

6.2.1.3 Home Domain Identity Provider Installation

The installation process is fairly easy. However to use the HD IdP, some post-installation configuration are required. These configurations will be explained in the coming sections. The installation process includes download the latest Identity Provider software package (Appendix B [1(iii)]).

During the installation process the IdP installer will ask for a directory where the IdP will be installed and the IdP name. The default directory used is /opt/shibboleth-idp. The IdP directory is known as IDP_HOME in the rest of the thesis.

6.2.1.4 Home Domain Shibboleth IdP Configuration

a) Apache Tomcat Configuration

The Apache Tomcat acts as the base Servlet container for the Shibboleth IdP. Since the configuration files of the IdP are XML based, the Tomcat must be enabled to parse XML into the HTML or text format. For this reason Xerces and Xalan are used with the Tomcat. The Xerces is a Java parser for XML and Xalan provides an XSLT processor for transforming XML documents into HTML. These two XML libraries come with the Shibboleth IdP installer which is needed to copy these to the TOMCAT_HOME/endorsed folder (Appendix B [1(iv)]).

Apart from the above, the IdP needs some specific memory configurations for its functioning. The Java Virtual Machine (JVM) memory options need to be set by editing the TOMCAT_HOME/bin/catalina.sh file (Appendix B [1(v)]).

b) Setting SOAP Endpoints

Generally, the communication between the Shibboleth IdP and SP is triggered by the user’s browser but in some situations both the HD IdP and FD SP may required to communicate directly without using the user’s browser. The situation may include certain operations such as Attribute Query, Logout, or Artifact Resolution etc. For this reason, the HD IdP needs to use an additional port that is different from the one used by the client because both have different purposes and security requirements. The port is known as Connector in Tomcat’s terminology. Tomcat uses Simple Object Access Protocol (SOAP), an XML based, in order to realize this communication between the HD IdP and FD SP (Appendix B [1(vi)]).

c) Deploy IdP WAR File

The IdP WAR file is the collection of the IdP jar files and other configurations that are deployed in the Tomcat as WAR files. Whenever the Tomcat restarts it restarts all of the deployed WAR files and hence accommodates any changes made to the IdP software. To automatically deploy the IdP WAR file by the Tomcat, this need to use an XML petite bit of code to inform the tomcat about the HD IdP WAR file (Appendix B [1(vii)]).

The Shibboleth HD organization entity, the IdP, is now up and running. However later it will requires some advance level configurations after installing the FD organization entity (i.e., Shibboleth SP).. The HD IdP status now can be checked by accessing the URL from the same platform where IdP is installed and configured. This URL is used to check the status of the IdP: http://idp.dod.org/idp/profile/Status. The URL will show an “OK” page indicating that the IdP is successfully installed.

6.2.1.5 Enabling Communication Security via SSL

In the proposed architecture, Secure Socket Layer (SSL) protocol is used to secure the communication between different entities during online transactions. The communication between IdP and SP or IdP and Client or SP and client need to be secure in order to provide a secure architecture. The SSL enabling the IdP platform to communicate securely with an entity (e.g., client or SP). In essence the IdP needs to have a secure communication with the client to secure the incoming traffic for the login page. Similarly the IdP needs a secure communication link with the SP to secure the process of sending and receiving attribute requests.

In prototype implementation the RSA key pairs are created for generating certificates to make the communication secure for IdP (Appendix B [1(viii)]). Since the IdP is based on Tomcat, for this the Tomcat configuration needs to be modified to reflect the newly created RSA keystore (Appendix B [1(ix)]).

6.2.1.6 Configuration of Authentication Mechanism

In the proposed architecture the HD user is first authenticated and then his/her client and the HD IdP machine platform integrity is mutually verified in order to authorize him/her for accessing a secure and protected resource. The HD user AuthN is performed at the HD IdP end on behalf of the FD SP as mentioned in the previous sections. This section will describe the steps required for configuring the AuthN mechanism for the HD organization IdP machine. The Shibboleth IdP supports different type of AuthN mechanisms. In the prototype implementation username/password is used as a basic user AuthN mechanism via the Light Weight Directory Access Protocol (LDAP).

The configuration of AuthN mechanism is performed out in different sequence such as configuring the LoginHandler first, then installing the LDAP server as well as

LDAP browser and finally setting up the Java AuthN and AuthR Service (JAAS) for the Shibboleth framework.

a) Configuring LoginHandler for Username/Password based Authentication

The LoginHandler is responsible for authenticating users with a username/password pair using the JAAS mechanism. During the AuthN process the LoginHandler provides a login page to the user for entering his credentials, which are then verified against the LDAP server (Appendix B [1(x)]).

b) LDAP Server and Browser

As the HD organization IdP machine does not store any credentials of the users. It uses different data stores to store user's credentials and access these stores when required. In this work, LDAP server is used as the data store to store user's credentials related to the AuthN purposes. For this purpose Apache Directory Server (ApacheDS) is deployed as LDAP server in the prototype implementation for the reason that it is written in Java and provides some very good and advance features (Appendix B [1(xi)])

The ApacheDS runs as a backend service. Therefore a frontend interface is needed to create own directory for HD organization (i.e., users, their usernames and passwords and other credentials like address, phone numbers, and email etc). Thus, Apache Directory Studio is used to create and maintain own directory of users. The Apache Directory Studio is an LDAP browser used for communicating with the backend ApacheDS server (Appendix B [1(xii)]). With the Apache Directory Studio, directory of users in an organization can be created and the users can be created and maintain in the LDAP server.

c) Configuring the JAAS policy

The LoginHandler element ((5.4.1.6)) contains a specific attribute mentioning that the AuthN policy used by the Shibboleth IdP will be JAAS Authn policy as "jaasConfigurationLocation <=> "file <:///> opt/shibboleth-idp/conf/login <.> config"". The IdP login configuration needs to be changed to reflect the fact that LDAP server is used as AuthN data store by the Shibboleth IdP (Appendix B [1(xiii)]).

d) LDAP Data Connector

As the HD organization IdP machine does not store user's credentials or attributes. These are stored in data stores outside the IdP software. The IdP uses specific data connectors for extracting these credentials and attributes from the data stores. Since in this work LDAP used as the data store for storing user's credentials and other information, for this default data connector is used for LDAP. The data connectors are residing at the HD organization IdP machine at the location folder </>opt</>shibboleth<->idp</>conf in the file: attribute<->resolver<.>xml (Appendix B [1(xiv)]).

6.2.1.7 Attribute Resolution and Filtering

The Shibboleth HD IdP entity is responsible for authenticating a user on behalf of the SP. After successful AuthN, the user's browser is given a handler (an AuthN token) and is redirected to the FD SP. The FD SP uses this handler to request attributes related to the login user. Each FD SP requests for attributes according to its requirements for access decision making.

In the proposed architecture the HD client and IdP machines platforms integrity measurement are involved. So the HD client and IdP machines platforms mutual attestation is required to be achieved in proposed architectre. Since the trust tokens that are required for the integrity validation are not stored in the default data stores as the user's credentials, for this it is needed to create own DataConnector to collect the trust tokens and initiate the validation process. In addition to that it is also needed to create own AttributeResolver to populate the custom attribute (i.e., MPI).

To realize the above mentioned data connector and attribute resolver to populate own attribute for the machines mutual platform integrity the following DataConnector, AttributeResolver and attribute are added to the HD organization IdP configuration file known as attribute-resolver.xml. The custom data connector will be discussed in the later section (6.2.4) where the incorporating of the IMA into the architecture is explained.

Data Connector, Attribute Resolver, Attribute Definition and Policies

Apart from the attribute resolution the HD organization IdP machine also enforce some policies to define (Appendix B [1(xv)]) which attributes have to be released to the SP. These policies are defined in the form of rules in the attribute-filter.xml file at the IdP. In this work the releasing of an attribute (i.e., HD client and IdP machines platforms successful mutual attestation) to the FD SP for the resource decision making process so for this purpose customized rule is created.

6.2.2 Foreign Domain Shibboleth Service Provider Installation

The FD SP installation is described in the following sub-section (6.2.2.1 to 6.2.2.7). The corresponding complete command-line instructions and code are given in Appendix B:

6.2.2.1 Operating System

In the prototype implementation of the Shibboleth framework, one of the most stable versions of Linux OS was used CentOS 5.6. The reason for selecting this OS is that the Shibboleth framework officially supports this version of Linux for the FD SP software.

6.2.2.2 Foreign Domain Service Provider Pre-requisites

The FD organization Shibboleth SP is based on number packages that need to be configured and install prior to the installation of the SP software. These packages are used by the FD SP to provide different functionalities. Some of the packages are developed in C/C++ language so for configuring and installing these packages the installation of C and C++ compilers is needed using the CentOS provided installer feature (Appendix B [2(ii)]).

a) log4shib

This package is specifically designed for the Shibboleth FD SP and is a modified version of the log4cpp. This package provides the logging mechanism that is used to maintain logs during accessing a protected resource through Shibboleth SP. This package can be downloaded from its homepage at: “http://www.shibboleth.net/downloads/log4shib/” (Appendix B [2(ii)]).

b) Xerces-C

The FD organization Shibboleth SP configuration is based on XML files. The SP requires a validating XML parser that can be used to parse and validate different XML based configurations. The Apache Software Foundation has developed a C++ based XML-parser namely Xerces - C that is used for processing XML in the Shibboleth framework. It enables the FD SP to: (1) Read and write the XML data and (2) provide a communal library to parse, generate, manipulate and validate XML documents (Appendix B [2(iii)]).

c) XML-Security-C

In the proposed architecture SSL is proposed to use for securing the communication between the FD SP and other entities in federated identity and resource access mode. Therefore, to utilize this feature the FD SP needs to be able to use XML based encryption and digital signatures. The Apache Software Foundation has provided a C++ based library for XML Encryption and Signatures known as XML-Security-C. This library is responsible for processing XML Encryption and Signatures in the Shibboleth framework. This library uses the previously installed Xerces-C parser to provide the digital signature and encryption implementation. XML-Security-C requires OpenSSL to be installed prior to its configuration and installation (Appendix B [2(iv)]).

d) XML-Tooling-C

OpenSAML 2 is responsible for providing a high level interface for processing XML especially for encryption and signing purposes. The OpenSAML 2 itself requires a low-level library for its functioning which is known as XML Tooling-C (Appendix B [2(v)]).

e) OpenSAML-C

The C language implementation of the OpenSAML 2 is known as OpenSAML-C. In the above step the XML Tooling-C library was installed for installing the OpenSAML-C (Appendix B [2(vi)]).

6.2.2.3 Foreign Domain Service Provider Installation

The FD organization Shibboleth SP requires all of the above packages to be installed before installing the SP (Appendix B [2(vii)]).

6.2.2.4 Foreign Domain Shibboleth SP Configuration

a) Apache Web Server Configuration

The FD organization entity (i.e., Shibboleth SP) is based on the Apache web server. So, after installing the Shibboleth SP some minor modifications in the web server is needed to be performed (Appendix B [2(viii)]).

The Shibboleth daemon (shibd) must have to be started independently to execute the apache web server. This daemon is responsible for handling requests coming to the FD organization SP (Appendix B [2(ix)]).

6.2.2.5 Enabling Secure Socket Layer (SSL)

In the HD organization IdP configuration section (6.2.1.5) the details have been given about the using of SSL for securing the communications within the proposed architecture. Among the entities involved are client, IdP and SP. In this section the procedure of enabling SSL security for the FD organization SP machine platform is described.

Since an open source implementation of the SSL naming openssl is available, in the prototype implementation, this openssl is used along with its available libraries to integrate it with the Apache web server. The mod_ssl is used to enable the SSL features in the Apache web server. In the CentOS repositories both of these packages are available (Appendix B [2(x)]) in the CentOS.

In the prototype implementation RSA key pairs are generated to create the certificates to be used for enabling the SSL (Appendix B [2(xi)]). Now the communication with the FD SP machine by any other entity or machine (e.g., client or IdP) is secure and a secure channel is used for accessing the resources protected by the FD organization SP.

6.2.2.6 Metadata for HD IdP and FD SP

The SAML metadata [29] is an integral part of the Shibboleth framework. The metadata is used to identify the organizations HD IdPs and FD SPs in a federation. It is also used to advertise certain capabilities that an FD SP or HD IdP can perform. This section will discuss how to generate metadata for the federation and how to exchange the metadata between the HD IdP and FD SP organizations in order to identify each other as legitimate entities of the federation.

During the HD organization IdP installation process the Shibboleth IdP installer generates a default metadata for the HD IdP and place it at /opt/shibboleth-idp/metadata folder by the name of idp-metadata.xml. The IdP configuration file also reflects this metadata using the <MetadataProvider> tag in a configuration file known as relying-party.xml that is located at the /opt/shibboleth-idp/conf folder. To establish a communication channel between the HD IdP and FD SP, the HD IdP metadata needs to be transferred from to the FD SP and vice versa.

In the prototype implementation, the HD IdP metadata file is copied and placed in a configuration folder used by the FD SP. The Shibboleth SP provides a specialized tool for generating a basic metadata for the FD SP machine platform. The tool is known as metagen.sh and is residing at the SP at location /etc/shibboleth. This is the default location for all the configuration and XML files required by the Shibboleth SP. Using the metagen.sh tool the metadata for FD SP are generated by the sp-metadata.xml file. This metadata is also located in the same configuration folder as mentioned above. This metadata file is then copied and placed in the /opt/shibboleth-idp/metdata folder at the HD IdP machine platform.

Finally, both the HD IdP and FD SP configuration files need to be modified to incorporate the metadata files and their locations. At the HD IdP machine platform the relying-party.xml needs to be modified to include the location for the FD SP metadata by adding some XML code (given in Appendix B [2(xii)]). At the FD SP machine platform the shibboleth 2.xml file located at the /etc/shibboleth folder is the main configuration file. This file needs to be modified to include the location for the IdP metadata (given in Appendix B [2(xii)]).

This way, both the HD IdP and FD SP organizations have each other's metadata files located locally on their platforms respectively and both have added it to their configuration files. Now they can communicate with each other by identifying each other as a member of the same federation. In this manner new FD SPs can be easily added to the federation by adding their metadata information at the organization HD IdP.

6.2.2.7 Protecting Resource via Shibboleth SP

The Shibboleth SP is designed to protect any web based resource or application. The nature of the resource or application depends upon the services provided by an FD SP organization. For example a bank providing financial services to their customers would place financial resources and applications on their SP to make these resources secure and accessed by authorized users with valid mutual platform integrity state. A research organization may put their classified research findings on their SP to protect their resources. Therefore, the implementation of these applications for protecting resources depends upon the nature of services an SP is providing. In the prototype implementation, as proof-of-concept, a PhP based web application (i.e., simple static web page) is implemented and then it is protected through Shibboleth SP. This application or protected resource is sufficient enough to proof the secure access to the resource by taking into consideration the mutual attestation attribute for the access decision making process.

This section presents the procedure of protecting any web based resource through Shibboleth SP. As mentioned earlier (section 6.2.2.4 (a)) that the SP is based on the Apache web server for providing online services. So to protect a resource, a directory has been created by the name secure in the web server resource directory folder. Generally, the Apache web server uses the /var/www/html/ directory for storing its resources. In this work the secure directory is placed in /var/www/html/ at the FD SP platform. In addition to that a simple PHP script is added to the secure directory to show the User Name, Common Name and MutualPlatformIntegrity attributes of the HD IdP and client machines platforms that is accessing the resource. The SAML assertion that is sent by the HD IdP is also displayed. If the HD user is successfully authenticated and also his/her machine platform integrity along with the HD IdP's machine platform integrity is successfully mutually validated then only the user access to these resources will be granted (in the form of displaying the page). If the user AuthN is not successful then the HD IdP machine will display an error page. If the AuthN is successful but either HD client or IdP's machines platform integrity is not mutually validated then the FD SP will not provide access to the resource.

The shib.conf file residing at FD SP at the location /etc/httpd/conf.d/ is responsible for defining and imposing the access control policies. For this purpose an access policy has been defined that will only allow access to an authenticated user if the mutual attestation of the HD client and IdP machines platforms are validated successfully (Appendix B [2(xiii)]).

Generally, not all the resources residing on an FD SP are sensitive and protected through the access mechanism described above. An SP may offer some resources which does not need any protection and security. Such resources can be accessed freely while the same SP can provide protected and secure resources at the same time. For this purpose, in the prototype implementation, another resource has created (simple web page) and placed in another directory that is to be accessed freely without AuthN and validation of the machines mutual integrity platforms. Therefore, another directory was created with the name "unsecure" and placed in the Apache web server resources directory. To inform SP about this directory and its access policy a policy rule is added to the shib.conf file.

Hence, if the URL https://sp.dor.org/secure is accessed by any HD client, it will require successful AuthN and validation of machines mutual attestation process in order to access the secure resource. On the other hand, the URL https://sp.dor.org/unsecure requires neither AuthN nor machines mutual attestation to be performed to access the unsecure resource.

6.2.3 Web Single Sign-On (WSSO) Service

In the proposed architecture, the Shibboleth HD IdP WSSO functionality is leveraged for multiple resources access. The WSSO service is the first contact point at the HD IdP machine platform. When a user is redirected to his selected HD IdP by the DS, the SSO service at HD IdP initiates the AuthN process and a user is presented with the login page. The implementation and configuration details of WSSO service is discussed below (section 6.2.3.1 to 6.2.3.2).

Since Shibboleth framework is based on the SAML v2.0, it provides two different profiles for AuthN [32], [33]. The two profiles are Browser/Artifact profile and Browser/Post profile. The Browser/Artifact profile is mainly used in AuthN in a federated environment. The configurations of these profiles are defined in the SAML v2.0 specification [114]. Since in this work, the Shibboleth framework is using as the baseline framework, the HD IdP uses the Browser/Artifact profile for authenticating the HD users and providing SSO services. The HD and FD metadata configuration for WSSO is described below (section 6.2.3.1 to 6.2.3.2).

6.2.3.1 Home Domain IdP Metadata Configuration

The metadata is the starting point of trust between the HD IdP and FD SP organizations and provides the information about different functionalities that the HD IdP and FD SP are capable of performing (Appendix B [2(xiv)]).

- The information (1) shows that the IdP supports both the SAML v1.1 and SAML v2.0 protocols.

- The configuration (2) shows the end location for the ArtifactResolutionService used in the SSO.

- The configuration (3) explains the actual SSO service running on the IdP. The bindings of different protocols used for SSO with the SSO service are given and the end location URLs are provided where SSO requests are forwarded.

All Location tags used in the configurations are accessed via the Servlet path "/profile" so the Servlet (Tomcat in this work) construct these URLs to be used by the user browser.

6.2.3.2 Foreign Domain SP Metadata Configuration

The FD SP metadata must also be configured to add the SSO configurations. For this reason, the configurations (given in Appendix B [2(xv)]).are added to the SP metadata This configuration informs the HD IdP how and where to push the SAML assertions for SSO through the browser. Generally, the FD SP informs the HD IdP about the end location to use in its request. The HD IdP then uses the above configurations in the FD SP metadata (located at IdP) to validate the end location for the SSO requests. The HD IdP also extracts the information about the version of the SAML used and the binding to be used accordingly.

6.2.4 Implementing Mutual Attestation

The establishment of mutual trust among different entities in a federated identity and resource or services access environment is one of the major security concerns for many organizations to securely share their resources. To establish mutual trust among the entities, in this thesis a PUSTPF architecture is proposed for federated identity and resource or service access to: (1) Integrate of HD user basic AuthN with the mutual attestation technique, (2) mutually validate the HD communicating machines platforms mutual integrity by leveraging mutual attestation technique known as mutual IMA and (3) privacy conservation of HD client and IdP machines in the mutual attestation technique. This section will describe the procedure of using TPM and its keys to maintain and collect mutual trust tokens for the HD target machines platforms using the IMA technique. In addition to that the data connector is created for initializing the machines mutual attestation process after a successful user AuthN process by the HD IdP machine. This section comprises of TPM configuration, IMA implementation and configuration into the HD IdP and client machines platforms, and configuring the underlying Java Trusted Software Stack (jTSS) for communicating with the TPM on both the HD client and IdP machines platforms.

6.2.4.1 Trusted Platform Module (TPM) Configuration

It has already discussed (section 2.7.1) that the TPM is a small cryptographic chip that is used to generate and store secret keys and platform configurations. The TPM may report these configurations to a validator system or challenger machine in order to validate the target or attestor machine platform's integrity. In the proposed architecture, both the HD client and IdP machines platforms must be TPM-enabled in order to facilitate mutual attestation technique. On both the HD machines platforms the TPM can be enabled and activated by the steps:

- Enter to the BIOS settings on system restart and find the Security tab. In the Security tab an option for Enable/Disable TPM will be given. Select to Enable the TPM and restart the system. It may ask for confirmation upon restart about the TPM enable option, confirm it and the TPM is enabled but cannot be used until it is activated.

- Once the TPM chip is enabled the next step is to activate it. The Activate/Deactivate option can be found on the same Security tab in the BIOS settings. Select the Activate option and restart the system. By activating the TPM all of the previous data in the TPM will be vanished and the TPM will be ready to use.

- The current Linux kernels come with built-in device drivers for TPM chip. However, for older versions of kernels these drivers need to be installed.

Since in this work new kernels has used, so no separate drivers for TPM have installed.

- The TPM chip is ready after its ownership is taken by the user or a system administrator in the organization. For using the TPM functionalities and taking its ownership some software is needed that can communicate with the TPM. This software is known as Java Trusted Software Stack (jTSS). The TPM ownership step will discuss in the later sections (section 6.2.4.3).

6.2.4.2 Installation and Configuration of IMA

Integrity measurement architecture is an integral part of the Linux Security extensions that are pre-built in the Linux kernels. Fortunately, kernel version 2.6.30 and onwards has the security extensions like IMA and a few others comes within the kernel source code. However, it requires a compilation process to incorporate it into the kernel and to utilize its functionalities. In the prototype implementation, IMA is used for validating integrity of the HD client and IdP machines platforms. Therefore, Linux kernel version 2.6.35 is utilized for both the HD machines platforms. For this purpose, steps have been performed to compile the Linux kernel for incorporating IMA for both the HD IdP and client machines platforms (Appendix B [3(i)]).

6.2.4.3 Installing Open Source Trusted Computing for JAVA

In this work jTSS [40] is used which is an open source execution of the Trusted Software Stack (TSS) provided by the IAIK [113] to communicate with the TPM in a trusted manner. jTSS is primarily based on the Java language and it provides a trusted API for Java applications to communicate directly with the TPM chip. In the prototype implementation, jTSS is used for communicating with the TPM and creating AIKs (Appendix B [3(ii)]).

The jTSS provides the low-level libraries for communicating with TPM chip. To use these low level libraries some high level tools are required. jTPMTools provides the implementation of such high level tools that can be used with the jTSS to communicate with TPM. The jTPMTools are downloaded and extracted in the same folder as the jTSS was downloaded (Appendix B [3(iii)]).

Some external libraries are also required by the jTSS to perform its functionality. These libraries are downloaded from TrustedJava homepage and placed in the ext_libs folder of the jTPMSTools software. Similarly, the ext_libs folder in the jTSS is copied and placed in the ext_libs of the jTPMTools. The jTPMTools can now be used with the TPM.

- *Taking TPM Ownership*: When the TPM is enabled and activated the next step is to take its ownership. This is done using the jTPMTools. The jTPMTools provide an executable script by the name jtt.sh that is used to perform different tasks like take ownership, creating AIKs and PCR read etc. The TPM ownership is necessary to create AIKs that are required by the IMA for machines mutual attestation in this work. The command to take TPM ownership is given in Appendix B [3(iv)]).

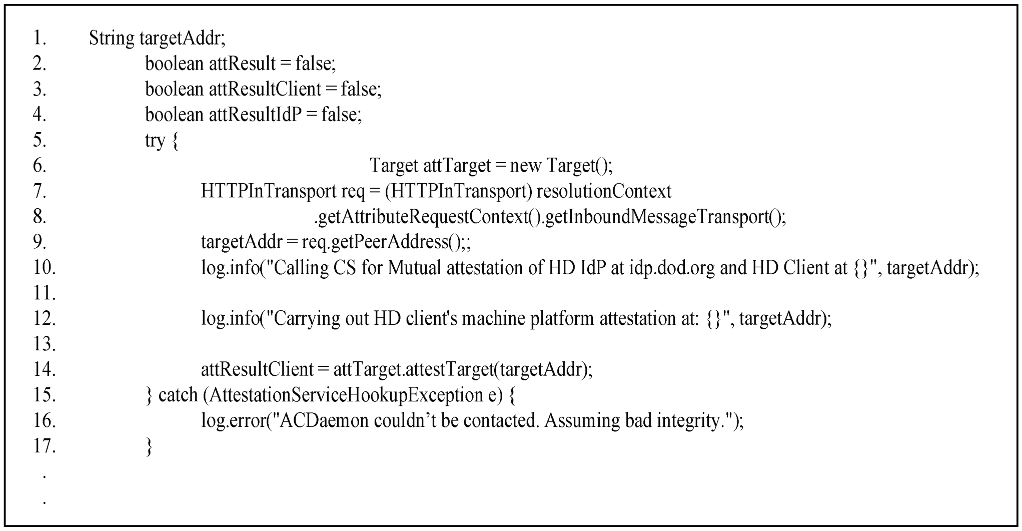
- *Creating Attestation Identity Key (AIK)*: The IMA uses AIK for performing the quote operation over the PCR10 value by the TPM. For IMA to function properly AIK creation is mandatory. The AIKs are created in HD organization on both the client and IdP machine platforms (Appendix B [3(v)]).

6.2.5 Implementation of IMA in Mutual Attestation

In the previous sections (5.2.2.1 to 5.2.2.2) the procedure of initializing and performing mutual attestation is described in detail. In the implementation, when the HD IdP machine receive request for the MPI, the data connectors residing at the HD IdP machine initialize the mutual attestation process by sending a request to the CS to perform the client’s machine platform validation. Afterwards, it performs the HD IdP’s machine platform validation (in case HD client’s machine platform integrity resulted in trusted state) and then encode the result by using MIR that populates the MPI attribute with the final result.

6.2.5.1 Integrity Measurement and Reporting Phase

A data connector naming MIPDC is created to perform three tasks. Firstly, it extracts the HD client's machine platform IP address from the request coming to the HD IdP machine. Secondly, it initiate the process of attestation by sending the HD client's machine IP to the CS for attestation and thirdly, it receives the attestation result from the CS and encode it to populate the MPI attribute. The code snippet shows the extraction of the IP address from the request is given in Figure 6.1.



```
String targetAddr;
        boolean attResult = false;
        boolean attResultClient = false;
        boolean attResultIdP = false;
        try {
                                        Target attTarget = new Target();
                HTTPInTransport req = (HTTPInTransport) resolutionContext
                                .getAttributeRequestContext().getInboundMessageTransport();
                targetAddr = req.getPeerAddress();;
                log.info("Calling CS for Mutual attestation of HD IdP at idp.dod.org and HD Client at {}", targetAddr);

                log.info("Carrying out HD client's machine platform attestation at: {}", targetAddr);

                attResultClient = attTarget.attestTarget(targetAddr);
        } catch (AttestationServiceHookupException e) {
                log.error("ACDaemon couldn't be contacted. Assuming bad integrity.");
        }
.
.
```

Figure 6.1: Dataconnector-mip snip shot

The attestation request for HD IdP's machine platform is processed in the same manner as that of HD client's machine platform by the data connector. The CS receives the attestation request for a particular HD target machine platform (i.e., client or IdP) and generates a random number called nonce to be part of the attestation request that the CS sent to the target machine platform. The following code generates the nonce.

```
private String generateNonce() {
                char[] hexChar = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9',
                                'A', 'B', 'C', 'D', 'E', 'F' };

                // geneate a new random Challenge
                String nonce = "";
                for (int i = 0; i < 20 * 2; i++) {
                        // compute a nible each time and append to nonce
                        Double rnd = Math.random() * 15;
                        nonce += hexChar[rnd.intValue()];
                }
                // System.out.println("New nonce: " + nonce);
                return nonce;
```

The CS sends the nonce within the attestation request to the HD target machine platform, where ACAgent-Daemon is running to extracts the trust tokens (i.e., PCR10 quoted value and SML) and returns these values back to the CS for validation phase.

To realize the above, two sub classes of the ACAgent -Daemon class is developed namely AttestClientPCR class and AttestClientSML class respectively. The AttestClientPCR extracts the PCR10 value from the TPM and initiates the TPM to sign the PCR10 value along with the nonce received from the CAgent. Similarly, the AttestClientSML reads the /sys/kernel/security/ascii_runtime_measurements file and extract the SML. These two values are returned back to the CS for validation phase.

6.2.5.2 Integrity Validation Phase

The CS receives the trust tokens and performs validation of the nonce, PCR10 value and the SML that it receives as the trust tokens. For the purpose of validation three different sub classes for trust tokens validation is developed, namely, VRN, VRPCR and VRSML. The VRN class is used to validate the nonce received from the HD target machine platform against the nonce that CS sent to the HD target machine platform. For this a nonce matching function is created which performs the nonce matching. If both the nonce matches, it assures that the attestation response from the HD target machine platform is a fresh one and the target machine is not resending some older trust tokens.

The VRPCR is created to validate the PCR10 value. IMA uses PCR10 to store the aggregate hash of all the executables and the hashes are stored in the SML. It means that the CS can calculate its own PCR aggregate value from the SML by performing the same PCR-Extend function that is performed by the TPM to store the aggregate hash in PCR10. The VRPCR first extract the SML list from the trust tokens that is passed to it and re-calculate an aggregate value from the SML. Once the PCR aggregate is calculated it then checks the calculated value against the PCR10 value. If both the values matched, it assured that PCR10 values reflect the SML list and the SML or the PCR10 value is not tampered with any malware or changed by an eavesdropper. Therefore, in the proposed architecture, if an attacker somehow get access to the SML or PCR10 value and modifies it, the changes made to these trust tokens can be easily identified during the VRPCR process. In addition to that in VRCertificate phase the CS performs the certificate validity check to confirm that the attestation response that the CAgent has received is from a genuine TPM and is not a fake one.

Finally, the VRSML is responsible to check each and every measurement in the SML against the validation database that contains known-good hashes about the machine platforms. The VRSML process the SML list and extracts the template-hash from the SML which is a combined hash of the file-hash and file-name calculated for each executable by the IMA. It then checks each template-hash against the validation database and validates the measurements. If all of the SML is successfully validated against the validation database, it is assured that the machine platforms are in a trustworthy state.

The result of all these validation is sent to the Client class (the same class is used for HD client and IdP machine platforms), which combines the results and return the final attestation result to the MIPDC as true or false, meaning HD target machine platform’s integrity is validated successfully or unsuccessfully. The DataConnector checks if the HD client’s machine platform integrity is validated then it sends another attestation request to the CS for validating integrity of HD IdP’s machine platform. The same procedure is followed for IdP attestation as mentioned above. The MIPDC receives both the attestation results and it combines the results into the MPI attribute that is sent to the FD SP for decision making process.

6.3 Experiment Outcome

To validate this work different experiment were carried out: (1) Trustworthy mutual attestation protocol proof of concept and (2) practicable unified security, trust and privacy framework test-bed prototype implementation. The hardware and software components used in (1) and (2) are given in Table 6.1 and Table 6.2 respectively. The various results collected from TMAP proof of concept for native true Web SSO and PUSTPF test-bed are discussed.

6.3.1 Emergent Unified STP Framework Result Analysis

The TMAP proof of concept for native true web SSO was implemented using java. The hardware and software used in TMAP proof of concept implementation are given in Table 6.1. The jTSS is an implementation of TCG software stack for the java platform. The jTSS covers greate part of specification TSS1.1 and substantial parts of specification TSS 1.2. The jTSS featurs are: TSS Device Driver Library (TDDL), TSS Core Service (TCS) and TSS Service Provider (TSP). The jTPMTools are collection of command line tools. The jTPMTools are used for interaction with the TPM and the jTSS. The basic tools include in jTPMTools: tacking and removing ownership, PCRs readind and extending, creating keys (e.g., AIK creation), AIK certification and data binding. The latest kernel version 2.6.35.11 (Ubuntu OS) was installed and configured with the IMA on client and server machine. The client and server machines motherboard contain TPM hardware chip that complies with the TCG TPM specification 1.2.

Table 6.1: TMAP proof of concept experiment setup

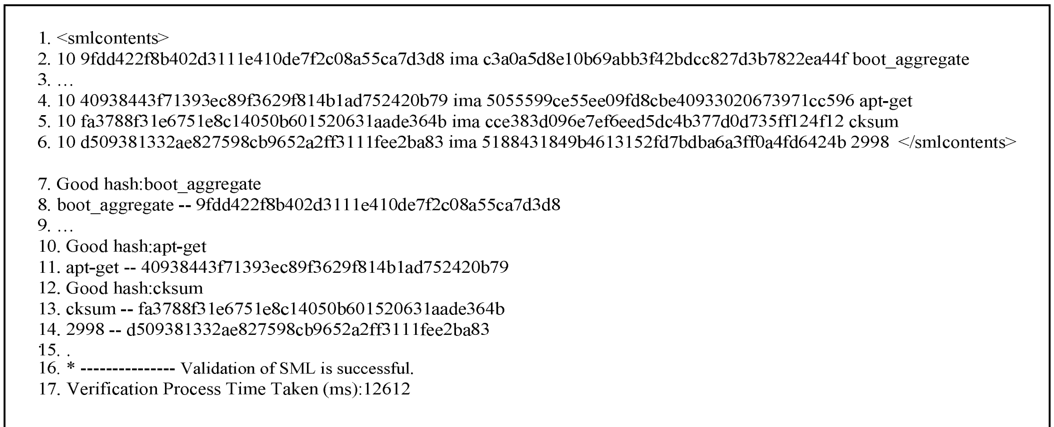
Client	Server
Ubuntu	
RAM 1GB	RAM2.5GB
2.00GHz Genuine Intel (R) CPU	Dual 1.83GHz Intel Core (TM) 2 CPU
MySQL DB (contains good hashes of executables)	
jTSS (used for communication with the TPM) Used to perform the quote operation over the PCR and nonce value, and extract this value for us to report to the challenging party.	
jTpmTools (we use jTSS to communicate with the TPM) Create the AIK using the *AIK_TPM* key *jTpmTools* and jTSS libraries.	
TPM 1.2 (Complies with the TCG TPM specification v1.2)	
Kernel version 2.6.35.11 (Kernel configured to work with the IMA)	

6.3.1.1 Platform Trustworthiness Result

To assess the trustworthiness of the attested machines platform for native true WSSO scheme a rootkit is purposely inserted to the client and server machine to demonstrate "how, the TMAP will reacts if a malevolent program such as a rootkit is detected". The trustworthiness of machine platforms means a machine components are protected and all hashes in the SML (in attestation request and response) were received and successfully compared with the good known hashes (in the DB) to show no rootkit or any other malevolent program is running.

The experiment was performed to test that the TMAP would fulfill the goal of the scheme (i.e., its defense against a particular threat). Figure 6.2 and Figure 6.3 shows that how the precise goal of the protocol is precisely achieved (i.e., against a particular threat such as rootkit (lrk5)).

Figure 6.2 shows the log of a secured and trustworthy machine. It clearly indicates that no malevolent program is running.

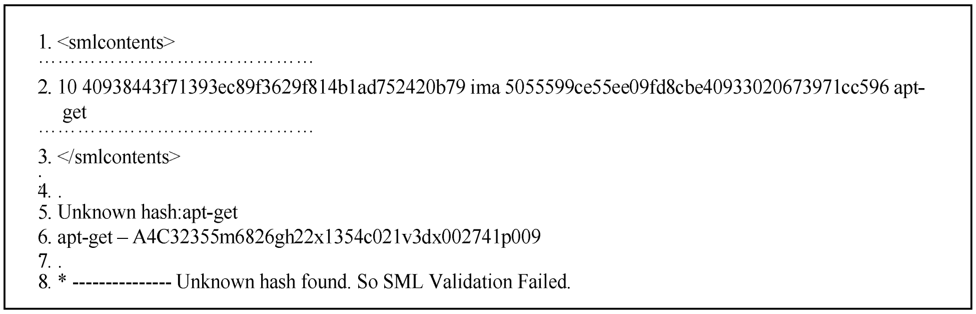


```
1. <smlcontents>
2. 10 9fdd422f8b402d3111e410de7f2c08a55ca7d3d8 ima c3a0a5d8e10b69abb3f42bdcc827d3b7822ea44f boot_aggregate
3. …
4. 10 40938443f71393ec89f3629f814b1ad752420b79 ima 5055599ce55ee09fd8cbe40933020673971cc596 apt-get
5. 10 fa3788f31e6751e8c14050b601520631aade364b ima cce383d096e7ef6eed5dc4b377d0d735ff124f12 cksum
6. 10 d509381332ae827598cb9652a2ff3111fee2ba83 ima 5188431849b4613152fd7bdba6a3ff0a4fd6424b 2998  </smlcontents>

7. Good hash:boot_aggregate
8. boot_aggregate -- 9fdd422f8b402d3111e410de7f2c08a55ca7d3d8
9. …
10. Good hash:apt-get
11. apt-get -- 40938443f71393ec89f3629f814b1ad752420b79
12. Good hash:cksum
13. cksum -- fa3788f31e6751e8c14050b601520631aade364b
14. 2998 -- d509381332ae827598cb9652a2ff3111fee2ba83
15. .
16. * --------------- Validation of SML is successful.
17. Verification Process Time Taken (ms):12612
```

Figure 6.2: Trustworthy and secured system log

Figure 6.3 shows the log of a compromised machine platform to a rootkit attack (the signature of "apt-get" is altered by the attacker).



```
1. <smlcontents>
…………………………………………
2. 10 40938443f71393ec89f3629f814b1ad752420b79 ima 5055599ce55ee09fd8cbe40933020673971cc596 apt-
   get
…………………………………………
3. </smlcontents>
.
4. .
5. Unknown hash:apt-get
6. apt-get – A4C32355m6826gh22x1354c021v3dx002741p009
7. .
8. * --------------- Unknown hash found. So SML Validation Failed.
```

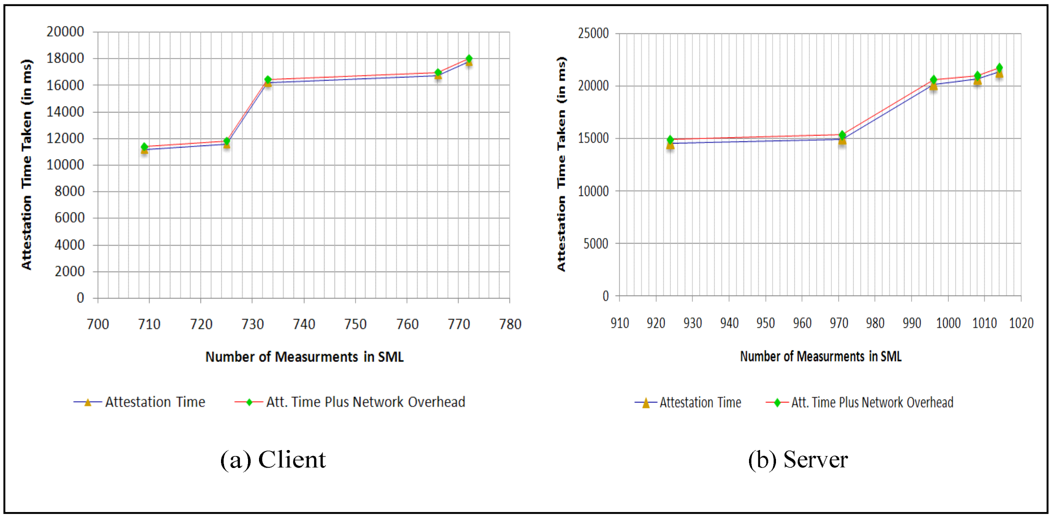
Figure 6.3: Infected or compromised system log

6.3.1.2 Mutual Attestation Performance Measurement Result

a) Client and Server Platform Attestation Time Taken

Figure 6.4 shows (a) the client and (b) the server machine platforms attestation time Vs the number of measurements in SML respectively. The client and server (i.e. SP) machines attestation is performed individually and collected the attestation data for client and server so that it can be graphically presented.

Figure 6.4 shows the client (a) and server (b) attestation data results. The x-axis stand for the independent variable SML (i.e., number of measurement in SML) and y-axis signifies the dependent variable Time (i.e., attestation time taken (in ms)). The graph data (Figure 6.4) obtained for client and server machine platforms attestation ((a) and (b)) shows the number of measurement in SML affect on the attestation time. Simply, when the number of measurements in the SML increments the corresponding number of measurement in the SML attestation time (in ms) also increases. The graph also shows the attestation time plus the network overhead (i.e., attestation request and quote operation). The platforms attestation data analysis shows the increments in the number of executables files on client and server machine distress the corresponding attestation time.



(a) Client

(b) Server

Figure 6.4: Client and server platform attestation graphs

b) Client and Server Platforms Mutual Attestation Round Trip Time

Figure 6.5 shows the round trip bidirectional (mutual) attestation time for the client and server (i.e. SP) machine platforms. Theattestation includes sending and receiving of the attestation request, response and validation operations. The attestation data for this graph is acquired by combining the data of client and server side machine platforms attestation. In Figure 6.5.the x-axis stand for the independent variable "SML" (i.e., number of measurement in SML) and y-axis signifies the dependent variable "Time" (i.e., attestation time taken (in ms)).

Figure 6.5 also reveals the relationship between the number of measurement in the SML and attestation time (i.e., the increase in the number of measurements in the SML raise the corresponding attestation time taken. The same is true for the round-tripe attestation time (in ms) plus the network overhead.

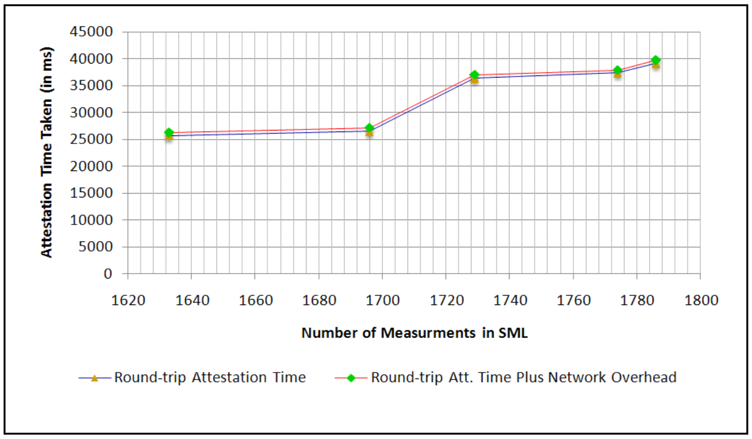


Figure 6.5: Client and server platforms mutual attestation round-trip

6.3.2 Practicable Framework Test-bed Result Analysis

The hardware/software requirements used in PUSTPF test-bed prototype implementation are given in Table 6.2. Ubuntu 9.10 was installed on HD entities (i.e., client and IdP) and CentOS on SP machine. Static IP addresses are assigned to client, IdP and SP machine. The executables good hashes DataBase (MySQL) is created and maintained at the HD IdP machine. The configuration of IMA was perforemed on the HD client and IdP machine. The HD client and IdP machine motherboard contain building TPM 1.2 that complies with the TCG TPM1.2 specification. The jTTS and jTPMTools (section 6.3.1) are set on the HD client and IdP machine.

Table 6.2: Practicable unified STP framework experiment setup

H/W/S/W Specification	HD Entities		FD Entity
	Client	IdP	SP
OS	Ubuntu 9.10		CentOS 5.6
Machines IP Add.	192.168.0.2	192.168.0.3	192.168.0.1
Switch & Cable	5-port 10/100Mbps + category.5e UTP		
RAM	1GB	2.5GB	1GB
CPU	2.00GHz	1.83GHz DC	1.86GHz DC
Web Server		Web server	Web server
Database		MySQL DB which holds good hashes of executables	
Shibboleth		IdP Installation & Configuration	SP Installation & Configuration
IMA	Configuration for Mutual Attestation		
jTTS	jTSS (used for communication with the TPM) Used to perform the quote operation over the PCR and nonce value, and extract this value for us to report to the challenging party.		
jTPM Tools	*jTpmTools* (we use jTSS to communicate with the TPM) Create the AIK using the *AIK_TPM* key *jTpmTools* and jTSS libraries.		
TPM Chip Type	TPM 1.2 (Complies with the TCG TPM specification v1.2)		
Kernel	Version 2.6.35 (Kernel was configured to work with the IMA)		

6.3.2.1 Integrated Authentication Result

The integrated AuthN (Figure 6.6 to Figure 6.9) result is the combination of: (1) Home domain user AuthN process and (2) Home domain machines mutual attestation process.

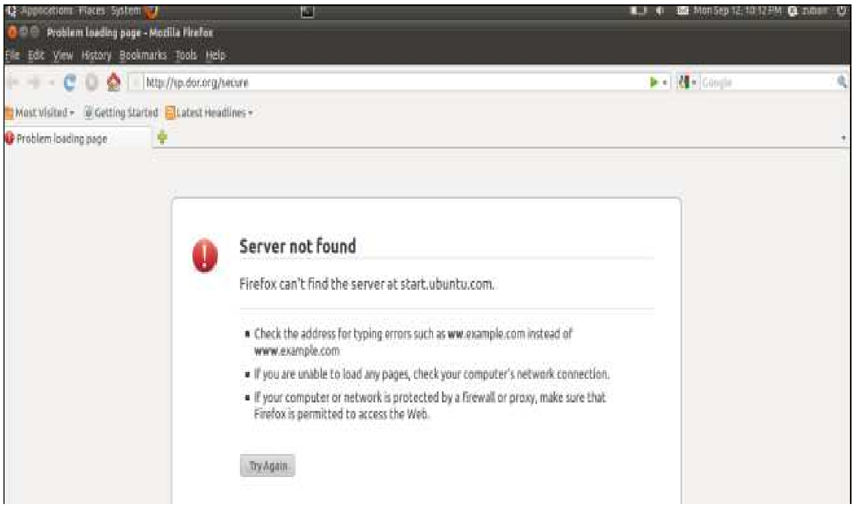
For user AuthN data, LDAP was used to store the user credentials. Whereas, the MA data is derived from the HD clients and IdP's machines platform attestation processed by the CS (part of the HD IdP). For successful user AuthN and MA (i.e. OUTPUT =1) the values of all INPUTS must be "1" (Table 6.3).

Table 6.3: The HD user AuthN and MA abstract level truth table

Scenarios	**Input Process**			**Output**
(section 6.3.2.3)	HD User AuthN Data	*MA* Data		
		HD Client (target)	HD IdP (challenger)	
Scenario - 1 (a)	1	Null	-	0
Scenario - 2 (b)	1	0	-	0
Scenario - 3 (c)	1	1	0	0
Scenario - 4 (d)	1	1	1	1

So, if any value in the INPUT combination is "Null" or "_" or "0" or "1" then the OUTPUT will be "0". The output will be zero (False) in the case when inputs are "Null" or "_" and "0". This means that to get "OUTPUT=1" (AuthN or mutual attestation successful) all INPUT combination must be "1" (Table 6.3). The complete

HD IdPs (www.dod.org) user AuthN and AuthR at the FD SPs (www.dor.org) is given in (Figure 6.6 to Figure 6.9).

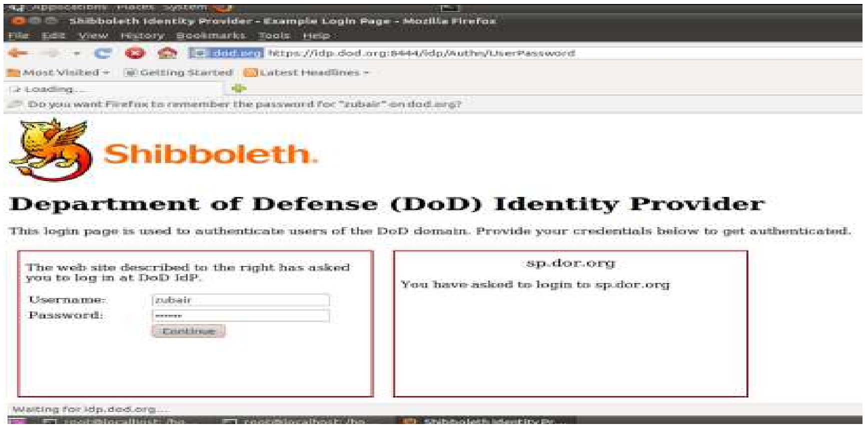

Applications Places System
Mon Sep 12, 10:12 PM zubair
Problem loading page - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://sp.dor.org/secure
Google
Most Visited Getting Started Latest Headlines
Problem loading page
Try Again

Server not found

Firefox can't find the server at start.ubuntu.com.

- Check the address for typing errors such as ww.example.com instead of www.example.com
- If you are unable to load any pages, check your computer's network connection.
- If your computer or network is protected by a firewall or proxy, make sure that Firefox is permitted to access the Web.

Figure 6.6: Protected resource request (http://sp.dor.org/secure)


Applications Places System
Shibboleth Identity Provider - Example Login Page - Mozilla Firefox
File Edit View History Bookmarks Tools Help
dod.org https://idp.dod.org:8444/idp/Authn/UserPassword
Most Visited Getting Started Latest Headlines
Loading...
Shibboleth.
Continue
Waiting for idp.dod.org...

Do you want Firefox to remember the password for "zubair" on dod.org?

Department of Defense (DoD) Identity Provider

This login page is used to authenticate users of the DoD domain. Provide your credentials below to get authenticated.

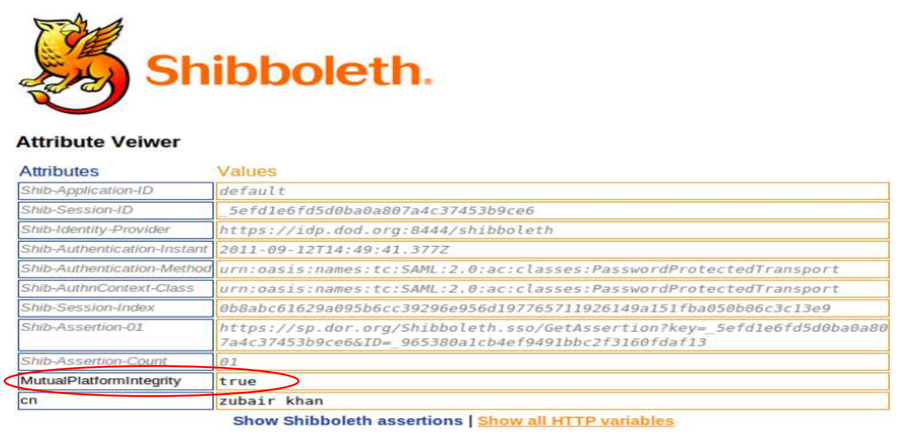
The web site described to the right has asked you to log in at DoD IdP.

Username: zubair
Password: *******

sp.dor.org

You have asked to login to sp.dor.org

Figure 6.7: Protected resource request (http://sp.dor.org/secure)

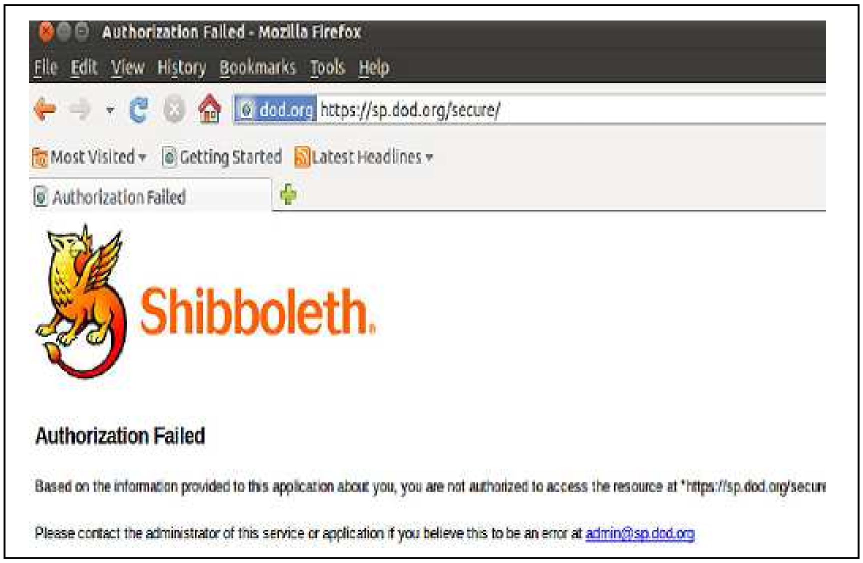


Attribute Veiwer

Attributes	Values
Shib-Application-ID	default
Shib-Session-ID	_5efd1e6fd5d0ba0a807a4c37453b9ce6
Shib-Identity-Provider	https://idp.dod.org:8444/shibboleth
Shib-Authentication-Instant	2011-09-12T14:49:41.377Z
Shib-Authentication-Method	urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
Shib-AuthnContext-Class	urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
Shib-Session-Index	0b8abc61629a095b6cc39296e956d197765711926149a151fba050b06c3c13e9
Shib-Assertion-01	https://sp.dor.org/Shibboleth.sso/GetAssertion?key=_5efd1e6fd5d0ba0a807a4c37453b9ce6&ID=_965380a1cb4ef9491bbc2f3160fdaf13
Shib-Assertion-Count	01
MutualPlatformIntegrity	true
cn	zubair khan

Show Shibboleth assertions | Show all HTTP variables

Figure 6.8: HD user AuthN and machines mutual attestation attribute veiwer


Authorization Failed - Mozilla Firefox
File Edit View History Bookmarks Tools Help
dod.org https://sp.dod.org/secure/
Most Visited
Getting Started
Latest Headlines
Authorization Failed
Shibboleth.

Authorization Failed

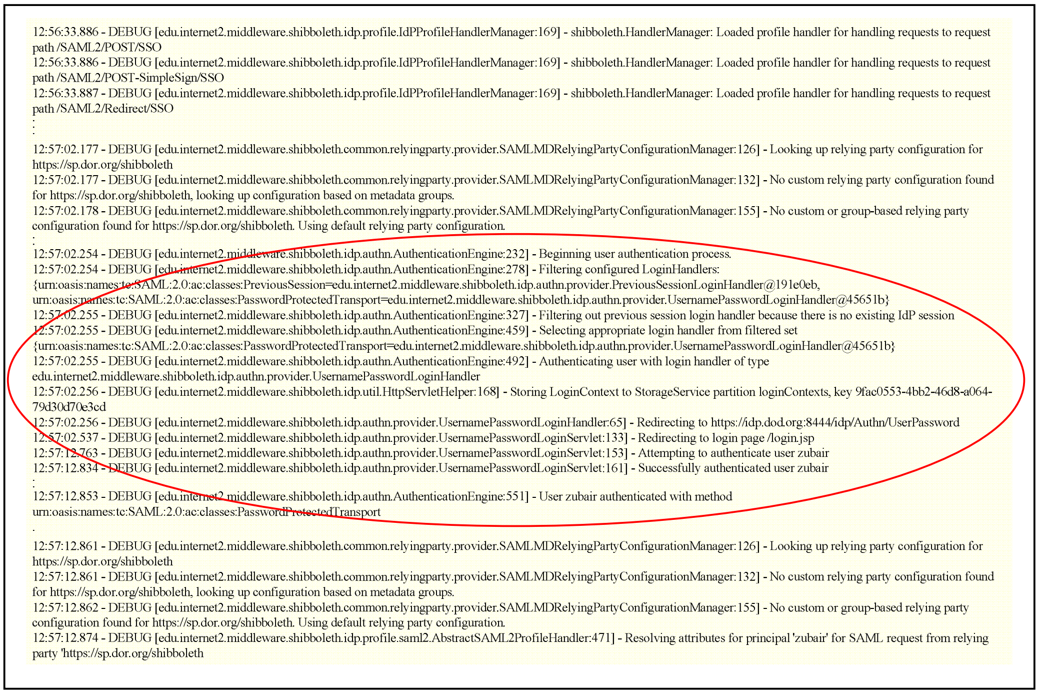
Based on the information provided to this application about you, you are not authorized to access the resource at "https://sp.dod.org/secure

Please contact the administrator of this service or application if you believe this to be an error at admin@sp.dod.org

Figure 6.9: HD machines mutual attestation failure

a) Home Domain User Successful Authentication Process Log Data

The HD user successful AuthN process log data is shown in Figure 6.10. The process log data includes resource request sited at the FD SP (www.dor.org), user redirection to the HD IdP (www.dod.org), user AuthN with his basic AuthN security credentials (UN/PWD) to the HD IdP machine.

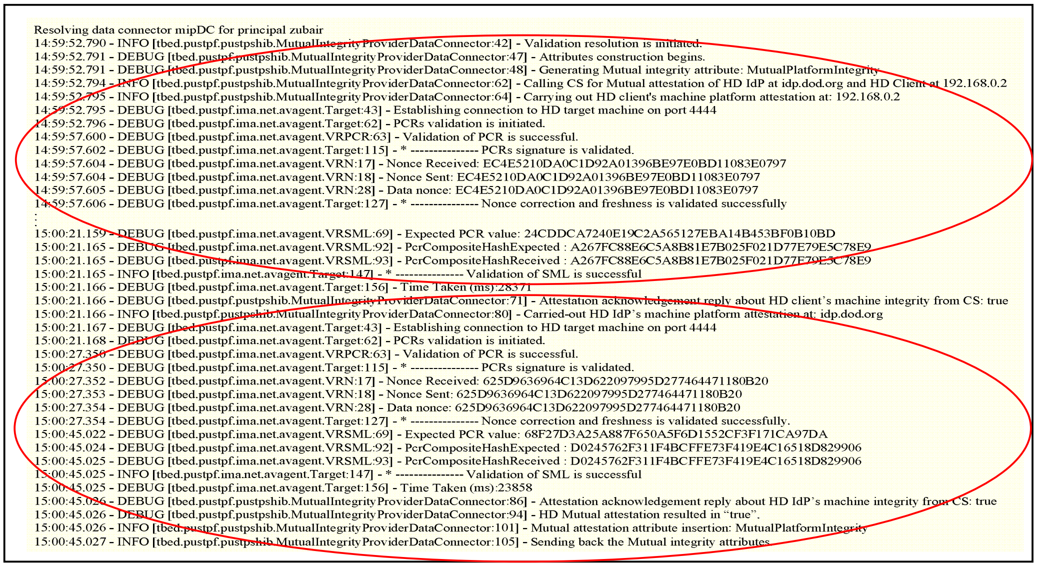


```
12:56:33.886 - DEBUG [edu.internet2.middleware.shibboleth.idp.profile.IdPProfileHandlerManager:169] - shibboleth.HandlerManager: Loaded profile handler for handling requests to request
path /SAML2/POST/SSO
12:56:33.886 - DEBUG [edu.internet2.middleware.shibboleth.idp.profile.IdPProfileHandlerManager:169] - shibboleth.HandlerManager: Loaded profile handler for handling requests to request
path /SAML2/POST-SimpleSign/SSO
12:56:33.887 - DEBUG [edu.internet2.middleware.shibboleth.idp.profile.IdPProfileHandlerManager:169] - shibboleth.HandlerManager: Loaded profile handler for handling requests to request
path /SAML2/Redirect/SSO
:
:
12:57:02.177 - DEBUG [edu.internet2.middleware.shibboleth.common.relyingparty.provider.SAMLMDRelyingPartyConfigurationManager:126] - Looking up relying party configuration for
https://sp.dor.org/shibboleth
12:57:02.177 - DEBUG [edu.internet2.middleware.shibboleth.common.relyingparty.provider.SAMLMDRelyingPartyConfigurationManager:132] - No custom relying party configuration found
for https://sp.dor.org/shibboleth, looking up configuration based on metadata groups.
12:57:02.178 - DEBUG [edu.internet2.middleware.shibboleth.common.relyingparty.provider.SAMLMDRelyingPartyConfigurationManager:155] - No custom or group-based relying party
configuration found for https://sp.dor.org/shibboleth. Using default relying party configuration.
:
12:57:02.254 - DEBUG [edu.internet2.middleware.shibboleth.idp.authn.AuthenticationEngine:232] - Beginning user authentication process.
12:57:02.254 - DEBUG [edu.internet2.middleware.shibboleth.idp.authn.AuthenticationEngine:278] - Filtering configured LoginHandlers:
{urn:oasis:names:tc:SAML:2.0:ac:classes:PreviousSession=edu.internet2.middleware.shibboleth.idp.authn.provider.PreviousSessionLoginHandler@191e0eb,
urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport=edu.internet2.middleware.shibboleth.idp.authn.provider.UsernamePasswordLoginHandler@45651b}
12:57:02.255 - DEBUG [edu.internet2.middleware.shibboleth.idp.authn.AuthenticationEngine:327] - Filtering out previous session login handler because there is no existing IdP session
12:57:02.255 - DEBUG [edu.internet2.middleware.shibboleth.idp.authn.AuthenticationEngine:459] - Selecting appropriate login handler from filtered set
{urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport=edu.internet2.middleware.shibboleth.idp.authn.provider.UsernamePasswordLoginHandler@45651b}
12:57:02.255 - DEBUG [edu.internet2.middleware.shibboleth.idp.authn.AuthenticationEngine:492] - Authenticating user with login handler of type
edu.internet2.middleware.shibboleth.idp.authn.provider.UsernamePasswordLoginHandler
12:57:02.256 - DEBUG [edu.internet2.middleware.shibboleth.idp.util.HttpServletHelper:168] - Storing LoginContext to StorageService partition loginContexts, key 9fac0553-4bb2-46d8-a064-
79d30d70e3cd
12:57:02.256 - DEBUG [edu.internet2.middleware.shibboleth.idp.authn.provider.UsernamePasswordLoginHandler:65] - Redirecting to https://idp.dod.org:8444/idp/Authn/UserPassword
12:57:02.537 - DEBUG [edu.internet2.middleware.shibboleth.idp.authn.provider.UsernamePasswordLoginServlet:133] - Redirecting to login page /login.jsp
12:57:12.763 - DEBUG [edu.internet2.middleware.shibboleth.idp.authn.provider.UsernamePasswordLoginServlet:153] - Attempting to authenticate user zubair
12:57:12.834 - DEBUG [edu.internet2.middleware.shibboleth.idp.authn.provider.UsernamePasswordLoginServlet:161] - Successfully authenticated user zubair
:
12:57:12.853 - DEBUG [edu.internet2.middleware.shibboleth.idp.authn.AuthenticationEngine:551] - User zubair authenticated with method
urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
.
12:57:12.861 - DEBUG [edu.internet2.middleware.shibboleth.common.relyingparty.provider.SAMLMDRelyingPartyConfigurationManager:126] - Looking up relying party configuration for
https://sp.dor.org/shibboleth
12:57:12.861 - DEBUG [edu.internet2.middleware.shibboleth.common.relyingparty.provider.SAMLMDRelyingPartyConfigurationManager:132] - No custom relying party configuration found
for https://sp.dor.org/shibboleth, looking up configuration based on metadata groups.
12:57:12.862 - DEBUG [edu.internet2.middleware.shibboleth.common.relyingparty.provider.SAMLMDRelyingPartyConfigurationManager:155] - No custom or group-based relying party
configuration found for https://sp.dor.org/shibboleth. Using default relying party configuration.
12:57:12.874 - DEBUG [edu.internet2.middleware.shibboleth.idp.profile.saml2.AbstractSAML2ProfileHandler:471] - Resolving attributes for principal 'zubair' for SAML request from relying
party 'https://sp.dor.org/shibboleth
```

Figure 6.10: HD user successful AuthN process log data

b) Home Domain Machines Platform Successful MA Process log Data

The HD IdP and client machines mutual attestation (Figure 6.11) will occur if the HD user is successfully authenticated using his/her basic AuthN credential.



```
Resolving data connector mipDC for principal zubair
14:59:52.790 - INFO [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:42] - Validation resolution is initiated.
14:59:52.791 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:47] - Attributes construction begins.
14:59:52.791 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:48] - Generating Mutual integrity attribute: MutualPlatformIntegrity
14:59:52.794 - INFO [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:62] - Calling CS for Mutual attestation of HD IdP at idp.dod.org and HD Client at 192.168.0.2
14:59:52.795 - INFO [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:64] - Carrying out HD client's machine platform attestation at: 192.168.0.2
14:59:52.795 - DEBUG [tbed.pustpf.ima.net.avagent.Target:43] - Establishing connection to HD target machine on port 4444
14:59:52.796 - DEBUG [tbed.pustpf.ima.net.avagent.Target:62] - PCRs validation is initiated.
14:59:57.600 - DEBUG [tbed.pustpf.ima.net.avagent.VRPCR:63] - Validation of PCR is successful.
14:59:57.602 - DEBUG [tbed.pustpf.ima.net.avagent.Target:115] - * --------------- PCRs signature is validated.
14:59:57.604 - DEBUG [tbed.pustpf.ima.net.avagent.VRN:17] - Nonce Received: EC4E5210DA0C1D92A01396BE97E0BD11083E0797
14:59:57.604 - DEBUG [tbed.pustpf.ima.net.avagent.VRN:18] - Nonce Sent: EC4E5210DA0C1D92A01396BE97E0BD11083E0797
14:59:57.605 - DEBUG [tbed.pustpf.ima.net.avagent.VRN:28] - Data nonce: EC4E5210DA0C1D92A01396BE97E0BD11083E0797
14:59:57.606 - DEBUG [tbed.pustpf.ima.net.avagent.Target:127] - * --------------- Nonce correction and freshness is validated successfully
:
:
15:00:21.159 - DEBUG [tbed.pustpf.ima.net.avagent.VRSML:69] - Expected PCR value: 24CDDCA7240E19C2A565127EBA14B453BF0B10BD
15:00:21.165 - DEBUG [tbed.pustpf.ima.net.avagent.VRSML:92] - PcrCompositeHashExpected : A267FC88E6C5A8B81E7B025F021D77E79E5C78E9
15:00:21.165 - DEBUG [tbed.pustpf.ima.net.avagent.VRSML:93] - PcrCompositeHashReceived : A267FC88E6C5A8B81E7B025F021D77E79E5C78E9
15:00:21.165 - INFO [tbed.pustpf.ima.net.avagent.Target:147] - * --------------- Validation of SML is successful
15:00:21.166 - DEBUG [tbed.pustpf.ima.net.avagent.Target:156] - Time Taken (ms):28371
15:00:21.166 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:71] - Attestation acknowledgement reply about HD client's machine integrity from CS: true
15:00:21.166 - INFO [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:80] - Carried-out HD IdP's machine platform attestation at: idp.dod.org
15:00:21.167 - DEBUG [tbed.pustpf.ima.net.avagent.Target:43] - Establishing connection to HD target machine on port 4444
15:00:21.168 - DEBUG [tbed.pustpf.ima.net.avagent.Target:62] - PCRs validation is initiated.
15:00:27.350 - DEBUG [tbed.pustpf.ima.net.avagent.VRPCR:63] - Validation of PCR is successful.
15:00:27.350 - DEBUG [tbed.pustpf.ima.net.avagent.Target:115] - * --------------- PCRs signature is validated.
15:00:27.352 - DEBUG [tbed.pustpf.ima.net.avagent.VRN:17] - Nonce Received: 625D9636964C13D622097995D277464471180B20
15:00:27.353 - DEBUG [tbed.pustpf.ima.net.avagent.VRN:18] - Nonce Sent: 625D9636964C13D622097995D277464471180B20
15:00:27.354 - DEBUG [tbed.pustpf.ima.net.avagent.VRN:28] - Data nonce: 625D9636964C13D622097995D277464471180B20
15:00:27.354 - DEBUG [tbed.pustpf.ima.net.avagent.Target:127] - * --------------- Nonce correction and freshness is validated successfully.
15:00:45.022 - DEBUG [tbed.pustpf.ima.net.avagent.VRSML:69] - Expected PCR value: 68F27D3A25A887F650A5F6D1552CF3F171CA97DA
15:00:45.024 - DEBUG [tbed.pustpf.ima.net.avagent.VRSML:92] - PcrCompositeHashExpected : D0245762F311F4BCFFE73F419E4C16518D829906
15:00:45.025 - DEBUG [tbed.pustpf.ima.net.avagent.VRSML:93] - PcrCompositeHashReceived : D0245762F311F4BCFFE73F419E4C16518D829906
15:00:45.025 - INFO [tbed.pustpf.ima.net.avagent.Target:147] - * --------------- Validation of SML is successful
15:00:45.025 - DEBUG [tbed.pustpf.ima.net.avagent.Target:156] - Time Taken (ms):23858
15:00:45.026 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:86] - Attestation acknowledgement reply about HD IdP's machine integrity from CS: true
15:00:45.026 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:94] - HD Mutual attestation resulted in "true".
15:00:45.026 - INFO [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:101] - Mutual attestation attribute insertion: MutualPlatformIntegrity
15:00:45.027 - INFO [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:105] - Sending back the Mutual integrity attributes
```

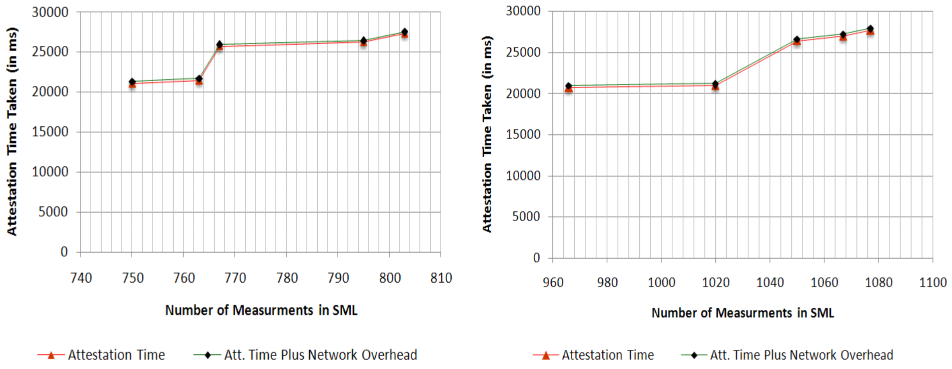
Figure 6.11: HD client and IdP successful MA process log data

Therefore, to access a protected resource sited at the FD the user AuthN and the HD client and IdP machines mutual attestation outcome must be true. The HD user AuthN or machines mutual attestation failure means the AuthR process will not carried-out. The HD client and IdP machines platforms successful mutual attestation process log data outcome is given in Figure 6.11.

6.3.2.2 HD Machine Platforms Performance Measurement Result

a) HD Client and Identity Provider Platform Attestation Time Taken

Figure 6.12 shows the HD client and IdP machines platform attestation time Vs the number of measurements in SML. Figure 6.12 shows the HD client (a) and IdP (b) machine platforms attestation data results. The *x-axis* stand for the independent variable "*SML*" (i.e., number of measurement in SML) and *y-axis* signifies the dependent variable "*Time*" (i.e., attestation time taken (in ms)). The graph data (Figure 6.12) obtained for HD client and IdP machine platforms attestation ((a) and (b)) shows the *number of measurement in SML* affect the *attestation time taken*. This means when the number of measurements in the SML increments the attestation time taken (in ms) of corresponding number of measurement in the SML also increases.



(a) HD client machine platform attestation

(b) HD identity provider platform attestation

Figure 6.12: HD client and idp platforms attestation result

The attestation of the HD client and IdP machine platforms is performed individually for number of times and collected the attestation data so that it can be graphically presented (Figure 6.12). The attestation data (Figure 6.12) reveals the attestation time facts such as when the number of measurements in the SML increments the attestation time (in ms) also increases accordingly. This means the IMA attestation scheme is depended on the number of measurement in the SML list. So if the number of measurement in the SML list increases then corresponding attestation time of measurement in the SML list also increases or vice versa.

b) HD Client and Identity Provider Platforms Attestation Round-trip Time

Figure 6.13 shows the round trip mutual attestation time for the HD client and IdP machines which includes sending and receiving of the attestation request, response and its validation. The *x-axis* stand for the independent variable “*SML*” (i.e., number of measurement in SML) and *y-axis* signifies the dependent variable “*Time*” (i.e., attestation time taken (in ms)). The graph data (Figure 6.13) obtained for HD client and IdP machine platforms mutual attestation attestation. The graph (Figure 6.13) shows the *number of measurement in SML* affect the *attestation time taken*. This means when the number of measurements in the SML increments the attestation time taken (in ms) of corresponding number of measurement in the SML also increases.

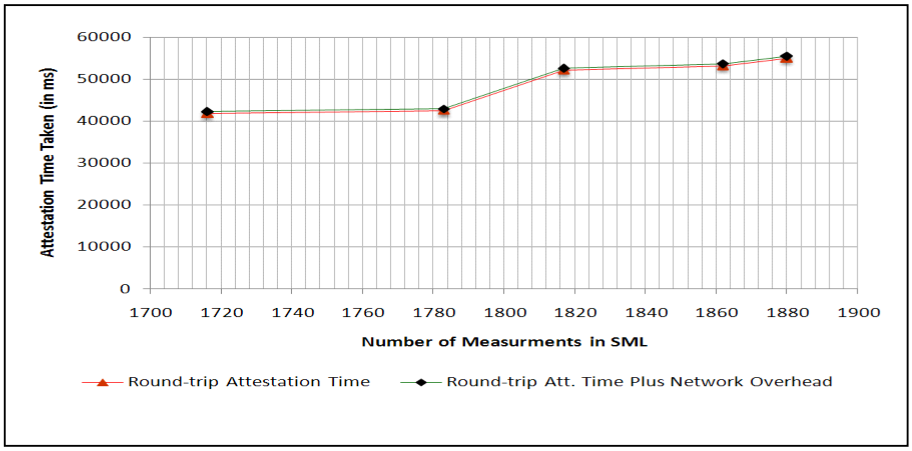


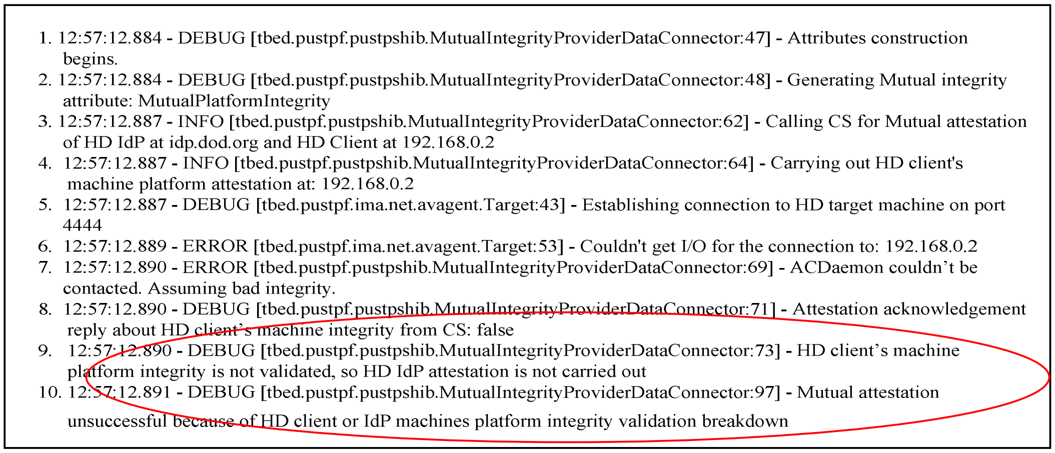
Figure 6.13: HD client and idp round-trip attestation result

6.3.2.3 HD Machines Platform Trustworthiness Result

The HD machines platform trustworthiness is tested through four different scenarios.

a) Scenario-1

When the attestationcollector-daemon at the HD client machine (Figure 6.14) is not running, client machine is not equipped with the TPM or not configured for mutual integrity validation.



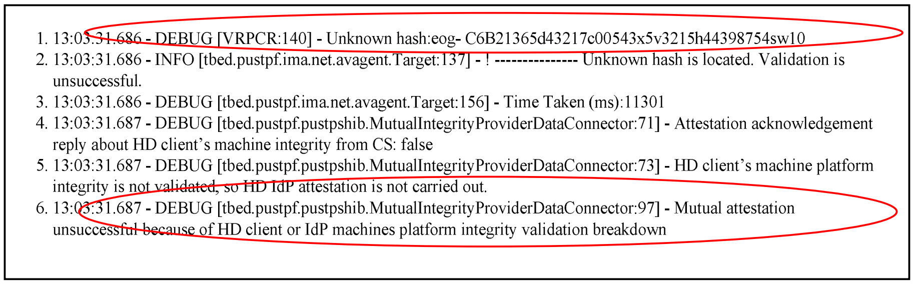
```
1. 12:57:12.884 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:47] - Attributes construction
   begins.
2. 12:57:12.884 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:48] - Generating Mutual integrity
   attribute: MutualPlatformIntegrity
3. 12:57:12.887 - INFO [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:62] - Calling CS for Mutual attestation
   of HD IdP at idp.dod.org and HD Client at 192.168.0.2
4.  12:57:12.887 - INFO [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:64] - Carrying out HD client's
    machine platform attestation at: 192.168.0.2
5.  12:57:12.887 - DEBUG [tbed.pustpf.ima.net.avagent.Target:43] - Establishing connection to HD target machine on port
   4444
6.  12:57:12.889 - ERROR [tbed.pustpf.ima.net.avagent.Target:53] - Couldn't get I/O for the connection to: 192.168.0.2
7.  12:57:12.890 - ERROR [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:69] - ACDaemon couldn’t be
   contacted. Assuming bad integrity.
8.  12:57:12.890 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:71] - Attestation acknowledgement
    reply about HD client’s machine integrity from CS: false
9.   12:57:12.890 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:73] - HD client’s machine
    platform integrity is not validated, so HD IdP attestation is not carried out
10. 12:57:12.891 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:97] - Mutual attestation
    unsuccessful because of HD client or IdP machines platform integrity validation breakdown
```

Figure 6.14: HD client machine not configured with IMA/ no TPM

Figure 6.14 shows when the attestationcollector-deamon could not contact the CS (Figure 5.1) then assume that the HD client’s machine platform integrity is false. In this scenario the HD client’s machine platform integrity failure demonstrates that the client’s machine is not a trustworthy HD organization (i.e., DoD), Figure 6.14, machine assigned to a particular user.

b) Scenario-2

In the second scenario when HD client's machine platform integrity has failed then the CS will not perform the HD IdP machine platform attestation and hence the mutual attestation has failed. In this scenario the HD client’s machine platform integrity failure demonstrates that an unknown “*hash:eog*” was found (Figure 6.15). The unknown hash entry shows that the signature of the “*eog*” is already changed and indicates that a malevolent action had replaced the original “*eog*” with a malicious version. The unknown hash detection given in Figure 6.15 demonstrates that the HD client’s machine is not trustworthy or already compromised to malevolent activity.

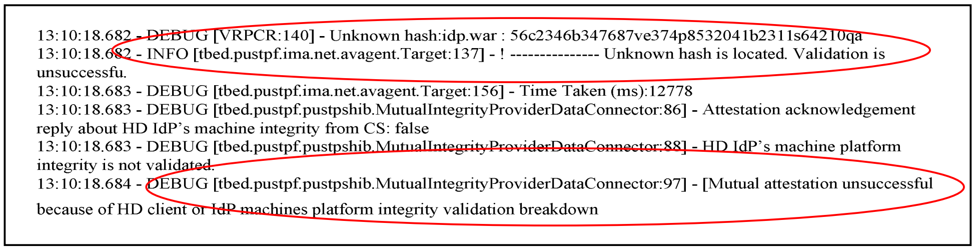


```
1. 13:03:31.686 - DEBUG [VRPCR:140] - Unknown hash:eog- C6B21365d43217c00543x5v3215h44398754sw10
2. 13:03:31.686 - INFO [tbed.pustpf.ima.net.avagent.Target:137] - ! --------------- Unknown hash is located. Validation is
   unsuccessful.
3. 13:03:31.686 - DEBUG [tbed.pustpf.ima.net.avagent.Target:156] - Time Taken (ms):11301
4. 13:03:31.687 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:71] - Attestation acknowledgement
   reply about HD client’s machine integrity from CS: false
5. 13:03:31.687 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:73] - HD client’s machine platform
   integrity is not validated, so HD IdP attestation is not carried out.
6. 13:03:31.687 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:97] - Mutual attestation
   unsuccessful because of HD client or IdP machines platform integrity validation breakdown
```

Figure 6.15: HD client machine platform attestation failure result

c) Scenario-3

In the third scenario, when the HD client's machine integrity is verified, i.e., attestation resulted in true but the HD IdP's machine platform integrity is not verified (Figure 6.16), i.e., attestation resulted in false state. In this case, the mutual attestation has failed.



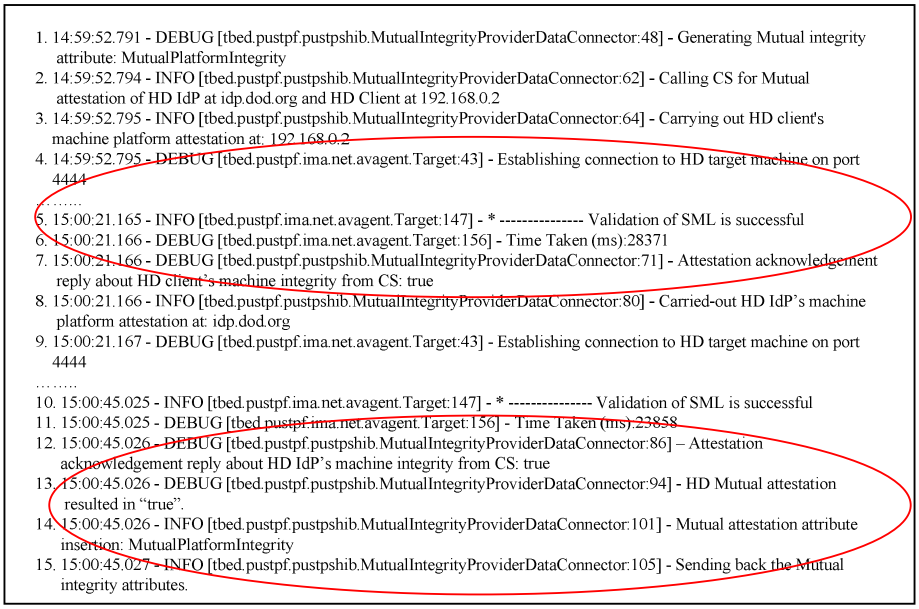
```
13:10:18.682 - DEBUG [VRPCR:140] - Unknown hash:idp.war : 56c2346b347687ve374p8532041b2311s64210qa
13:10:18.682 - INFO [tbed.pustpf.ima.net.avagent.Target:137] - ! --------------- Unknown hash is located. Validation is
unsuccessfu.
13:10:18.683 - DEBUG [tbed.pustpf.ima.net.avagent.Target:156] - Time Taken (ms):12778
13:10:18.683 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:86] - Attestation acknowledgement
reply about HD IdP’s machine integrity from CS: false
13:10:18.683 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:88] - HD IdP’s machine platform
integrity is not validated.
13:10:18.684 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:97] - [Mutual attestation unsuccessful
because of HD client or IdP machines platform integrity validation breakdown
```

Figure 6.16: HD idp machine platform attestation failure result

In third scenario the HD IdP’s machine platform integrity failed and demonstrates that an unknown “*hash: idp.war*” was detected. The unknown hash entry shows that the signature of the “*idp.war*” is changed indicating that a malevolent action had replaced the original “*idp.war*” with a malicious version. The unknown hash detection given in Figure 6.16 demonstrates that the HD IdP’s machine platform is not in a trustworthy state or already compromised to a malevolent activity.

d) Scenario-4

In fourth scenario, (given in Figure 6.17), when the HD client’s and IdP’s machine platform attestation is successfully validated, i.e., mutual platform attestation resulted in true. The HD client’s and IdP’s machines platforms are mutually attested successfully, it shows that both machines platforms are in a trustworthy state or not infected to any malevolent activity.

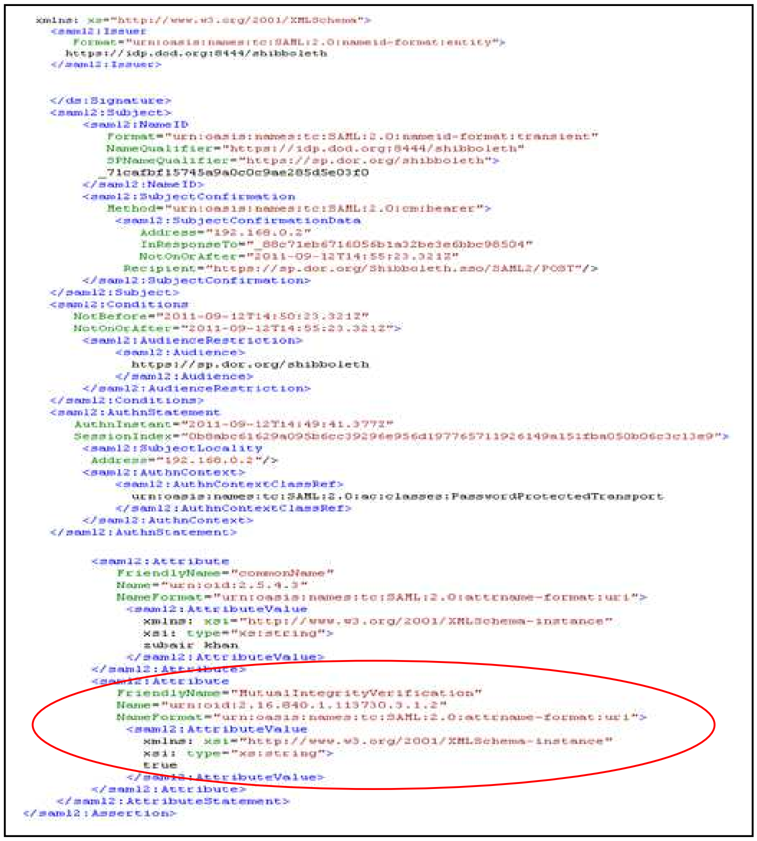


```
1. 14:59:52.791 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:48] - Generating Mutual integrity
    attribute: MutualPlatformIntegrity
2. 14:59:52.794 - INFO [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:62] - Calling CS for Mutual
    attestation of HD IdP at idp.dod.org and HD Client at 192.168.0.2
3. 14:59:52.795 - INFO [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:64] - Carrying out HD client's
   machine platform attestation at: 192.168.0.2
4. 14:59:52.795 - DEBUG [tbed.pustpf.ima.net.avagent.Target:43] - Establishing connection to HD target machine on port
   4444
……..
5. 15:00:21.165 - INFO [tbed.pustpf.ima.net.avagent.Target:147] - * -------------- Validation of SML is successful
6. 15:00:21.166 - DEBUG [tbed.pustpf.ima.net.avagent.Target:156] - Time Taken (ms):28371
7. 15:00:21.166 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:71] - Attestation acknowledgement
    reply about HD client’s machine integrity from CS: true
8. 15:00:21.166 - INFO [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:80] - Carried-out HD IdP’s machine
    platform attestation at: idp.dod.org
9. 15:00:21.167 - DEBUG [tbed.pustpf.ima.net.avagent.Target:43] - Establishing connection to HD target machine on port
   4444
……..
10. 15:00:45.025 - INFO [tbed.pustpf.ima.net.avagent.Target:147] - * -------------- Validation of SML is successful
11. 15:00:45.025 - DEBUG [tbed.pustpf.ima.net.avagent.Target:156] - Time Taken (ms):23858
12. 15:00:45.026 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:86] – Attestation
     acknowledgement reply about HD IdP’s machine integrity from CS: true
13. 15:00:45.026 - DEBUG [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:94] - HD Mutual attestation
      resulted in “true”.
14. 15:00:45.026 - INFO [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:101] - Mutual attestation attribute
     insertion: MutualPlatformIntegrity
15. 15:00:45.027 - INFO [tbed.pustpf.pustpshib.MutualIntegrityProviderDataConnector:105] - Sending back the Mutual
     integrity attributes.
```

Figure 6.17: HD client and idp platforms successful MA result

6.3.2.4 HD Attested Machines Privacy Conservation Proof

Figure 6.18 shows how the HD machine (i.e., client and IdP), DoD, platforms privacy is protected in the FD SP (DoR). The HD IdP machine attest the HD client machine platforms as well as perform its own machine platforms (IdP) attestation. In this way the HD client and IdP machine platforms security credentials are protected because the platforms security credentials are not shared with the foreign domain entity (SP). The “*MutualIntegrityVerification*” value “*true*”, Figure 6.18, is released to the FD SP (DoR).



```
xmlns: xs="http://www.w3.org/2001/XMLSchema">
  <saml2:Issuer
     Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">
    https://idp.dod.org:8444/shibboleth
  </saml2:Issuer>


  </ds:Signature>
  <saml2:Subject>
      <saml2:NameID
         Format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient"
         NameQualifier="https://idp.dod.org:8444/shibboleth"
         SPNameQualifier="https://sp.dor.org/shibboleth">
        _71cafbf15745a9a0c0c9ae285d5e03f0
      </saml2:NameID>
      <saml2:SubjectConfirmation
         Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
          <saml2:SubjectConfirmationData
             Address="192.168.0.2"
             InResponseTo="_88c71eb6716056b1a32be3e6bbc98504"
             NotOnOrAfter="2011-09-12T14:55:23.321Z"
           Recipient="https://sp.dor.org/Shibboleth.sso/SAML2/POST"/>
      </saml2:SubjectConfirmation>
  </saml2:Subject>
  <saml2:Conditions
     NotBefore="2011-09-12T14:50:23.321Z"
     NotOnOrAfter="2011-09-12T14:55:23.321Z">
      <saml2:AudienceRestriction>
          <saml2:Audience>
            https://sp.dor.org/shibboleth
          </saml2:Audience>
      </saml2:AudienceRestriction>
  </saml2:Conditions>
  <saml2:AuthnStatement
     AuthnInstant="2011-09-12T14:49:41.377Z"
     SessionIndex="0b8abc61629a095b6cc39296e956d197765711926149a151fba050b06c3c13e9">
      <saml2:SubjectLocality
       Address="192.168.0.2"/>
      <saml2:AuthnContext>
          <saml2:AuthnContextClassRef>
            urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
          </saml2:AuthnContextClassRef>
      </saml2:AuthnContext>
  </saml2:AuthnStatement>

       <saml2:Attribute
          FriendlyName="commonName"
          Name="urn:oid:2.5.4.3"
          NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri">
           <saml2:AttributeValue
             xmlns: xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi: type="xs:string">
             zubair khan
           </saml2:AttributeValue>
       </saml2:Attribute>
       <saml2:Attribute
          FriendlyName="MutualIntegrityVerification"
          Name="urn:oid:2.16.840.1.113730.3.1.2"
          NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri">
           <saml2:AttributeValue
             xmlns: xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi: type="xs:string">
             true
           </saml2:AttributeValue>
       </saml2:Attribute>
   </saml2:AttributeStatement>
</saml2:Assertion>
```

Figure 6.18: HD attested platforms privacy conservation proof

6.3.3 Comparison between Practicable and Emergent Schemes

Table 6.4 shows the comparative analysis of STP aspects of EUSTPF (section 4.3.1.1) and PUSTPF (section 4.3.1.2).

6.3.3.1 EUSTPF vs. PUSTPF

The comparative analysis of the two schemes is given in Table 6.4. In EUSTPF user identity privacy (anonymity and unlinkability) is protected via Blind Token Generating Service (BTGS). However, sharing of platforms security credentials (measurement) in the EUSTPF may raise network privacy concerns. The other network concerns in EUSTPF scheme are: good hashes management, performance deceleration, trusted privacyCA establishment in the interdomain scenario and expansion to the new domains. The EUSTPF scheme partially or fully provide protection against security threats: (a) rootkits, (b) Trojans that leading to the phishing attack, (c) man-in-the middle attack, (d) relay attack, (e) replay attack and (f) client and server (IdP) legitimacy. The mutual platform trust in EUSTPS is established by validating the integrity of client and server platforms.

In PUSTPF scheme the machine platform security is harden by integrating the Shibboleth user basic AuthN method with the HD client and IdP machine platforms mutual attestation. The trust in the HD client and IdP machine platforms is established by platforms mutual integrity validation technique. The platforms integrity validation ensures that HD client and IdP machine platforms are not compromised to security threat (section 6.3.5.5). In PUSTPF scheme the sharing of platforms security credentials privacy concerns in intra-domain scenario is neglegiable as compared to the inter-doamin scenario. This is mainly due to the trust relationship beucase in intra-domain scenario the entities (client and server) are in strong trust in as compared to the inter-doamin scenario.

Table 6.4: EUSTPF vs. PUSTPF

	Aspects	Security	Trust	Privacy
EUSTPF	Solutions	Binding user AuthN. with the *MA* protocol	Client & server machines platforms mutual integrity validation	BTGS attains the user privacy (i.e. anonymity and unlinkability)
	Advantages	Partial or full protection against threats a, b, c, d, e & f (section 6.3.5.5)		Blind AuthN token attain user privacy
	Network Concerns	External network: (1) Good hashes management, (2) Performance deceleration, (3) Trusted PrivacyCA establishment particularly in inter-domain scenario (4) Expansion to the new domains		platforms measurement privacy concerns (i.e. Threat 7)
PUSTPF	Solutions	HD user AuthN. fused with the HD client and IdP machines mutual attestation	HD client and IdP machines mutual integrity validation	HD client & IdP machines privacy is protected at the FD SP
	Advantages	1. Partial or full protection against threats a, b, c, d, e, f & g (section 6.3.5.5) 2. Mutual attestation scheme use in HD network turnover the IMA concerns into the pros.		1. Attested HD client and IdP machines privacy is protected at the FD SP 2. Trusted Attribute "*MPI*" is exchanged only with the FD to access a resource
	Network Concerns	Using of IMA in inter-domain scenario to strengthen the security and trust between the HD and FD may leads to the privacy concerns		

6.3.4 Performance Measurement and Benchmarking

Table 6.5 illustrates the comparison of the machines platform mutual attestation in EUSTPF and PUSTPF. The attestation data given in Table 6.5 is taken by combining the client side and server side platforms attestation data. In Table 6.5 the *Round-Trip (R-trip)* data means attestation of client (and server) machine (*ATc+s*) in column "*Client and Server*". The column "*HD Client and IdP"* shows the *Round-Trip* (*R-trip*) data for client (and HD IdP) machine platforms (*ATc+idp*). The *SMLc+s* and *SMLc+idp* represents the *number of measurmenet in SML* list for the HD entities (client and IdP). The "*R-trip ATc+s +N/W Overhead c+s*" signifies roundtrip attestation time for client and server plus network overhead for client and server machine.

Table 6.5 shows in column "*Client and Server*" that when number of measurement in the SML list increments the corresponding attestation time round trip and network overhead also increases. .Similarly, in column "*HD Client and IdP*" shows increase in the round trip and overhead attestation time due to increase in the number of measurement in the SML list at HD client and IdP. Another important element can also observed (Table 6.5) that the SML entries in HD organization (i.e. SMLc+idp) is quite large than that of simple client and server attestation. The is mainly due to the Shibboleth large number of configuration and liberies files.

Table 6.5: Machines platform mutual attestation comparison

Machines Mutual Attestation					
Client and Server			HD Client and IdP		
SMLc+s	R-trip ATc+s	R-trip ATc+s + N/W Overhead c+s	SMLc+idp	R-trip ATc+idp	R-trip ATc+idp +N/W Overhead c+idp
1633	25668	26309	1716	41742	42262
1696	26474	27202	1783	42422	42890
1729	36336	37063	1817	52122	52586
1774	37406	37943	1862	53202	53672
1786	39142	39758	1880	55022	55476

The Table 6.6 illustrates the performance comparison for the machines platforms attestation [2], [95], [99], [115]. The Table 6.6 also shows that all of the schemes make use of the IMA. Therefore, the assumption is established on the basis of the SML role in IMA because in IMA the SML plays a major role and strongly affects the attestation time. This can be observed from the client and server machines data

(Figure 6.3 and Figure 6.4) and HD client and IdP machines data (Figure 6.11 and Figure 6.12) that when the no.'s of SML entries increases so the respective attestation time also incrementing. The incrments in attestation time distress the performance of the scheme.

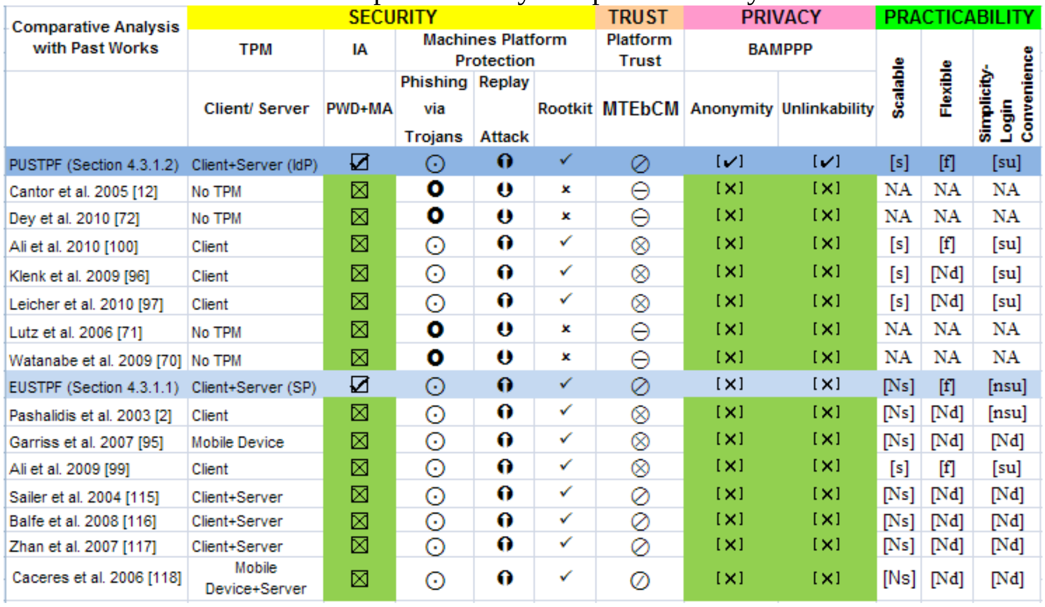
Table 6.6: Comparison of IMA based attestation schemes

Related work	Attestation Type	Attestation Mech.	No. of SML	Attestation Time
TMAP proof of concept (Section 6.2.1.2)	MA	IMA	SML ↑ (e.g. 709)	↑ (e.g. 11146ms)
Pashalidis et al. [2]	RA	IMA	SML ↑ (e.g. 500)	↑ (e.g. 10022ms)
Garriss et al. [95]	RA	IMA	SML ↑ (e.g. 580)	↑ (e.g. 11020ms)
Ali et al. [99]	RA	IMA	SML ↑ (e.g. 630)	↑ (e.g. 11100ms)
Sailer et al. [115]	MA	IMA	SML ↑ (e.g. 20000)	↑ (e.g. 23200ms)

6.3.5 Comparison of Proposed Solution with Existing Works

In this work a PUSTPF is proposed for federated identity and resource access system by combining the security functionalities of the trusted computing with a FIAM Shibboleth system. Shibboleth is used to supports the privacy conservation features that can enhance the HD machines mutual security and the trust and privacy conservation at the FD SPs in our architecture. The anonymity [25] in this work means that "a user may use a resource or service without disclosing the HD client's and IdP machine's measurements". Each column of Table 6.7 discussed in section 6.3.5.1 to 6.3.5.5.

Table 6.7: PUSTPF STP and practicability comparative analysis with other works



Comparative Analysis with Past Works	SECURITY					TRUST	PRIVACY		PRACTICABILITY		
	TPM	IA	Machines Platform Protection			Platform Trust	BAMPPP		Scalable	Flexible	Simplicity-Login Convenience
	Client/ Server	PWD+MA	Phishing via Trojans	Replay Attack	Rootkit	MTEbCM	Anonymity	Unlinkability			
PUSTPF (Section 4.3.1.2)	Client+Server (IdP)	☑	⊙	⬆	✓	⊘	[✓]	[✓]	[s]	[f]	[su]
Cantor et al. 2005 [12]	No TPM	☒	O	⬇	×	⊖	[×]	[×]	NA	NA	NA
Dey et al. 2010 [72]	No TPM	☒	O	⬇	×	⊖	[×]	[×]	NA	NA	NA
Ali et al. 2010 [100]	Client	☒	⊙	⬆	✓	⊗	[×]	[×]	[s]	[f]	[su]
Klenk et al. 2009 [96]	Client	☒	⊙	⬆	✓	⊗	[×]	[×]	[s]	[Nd]	[su]
Leicher et al. 2010 [97]	Client	☒	⊙	⬆	✓	⊗	[×]	[×]	[s]	[Nd]	[su]
Lutz et al. 2006 [71]	No TPM	☒	O	⬇	×	⊖	[×]	[×]	NA	NA	NA
Watanabe et al. 2009 [70]	No TPM	☒	O	⬇	×	⊖	[×]	[×]	NA	NA	NA
EUSTPF (Section 4.3.1.1)	Client+Server (SP)	☑	⊙	⬆	✓	⊘	[×]	[×]	[Ns]	[f]	[nsu]
Pashalidis et al. 2003 [2]	Client	☒	⊙	⬆	✓	⊗	[×]	[×]	[Ns]	[Nd]	[nsu]
Garriss et al. 2007 [95]	Mobile Device	☒	⊙	⬆	✓	⊗	[×]	[×]	[Ns]	[Nd]	[Nd]
Ali et al. 2009 [99]	Client	☒	⊙	⬆	✓	⊗	[×]	[×]	[s]	[f]	[su]
Sailer et al. 2004 [115]	Client+Server	☒	⊙	⬆	✓	⊘	[×]	[×]	[Ns]	[Nd]	[Nd]
Balfe et al. 2008 [116]	Client+Server	☒	⊙	⬆	✓	⊘	[×]	[×]	[Ns]	[Nd]	[Nd]
Zhan et al. 2007 [117]	Client+Server	☒	⊙	⬆	✓	⊘	[×]	[×]	[Ns]	[Nd]	[Nd]
Caceres et al. 2006 [118]	Mobile Device+Server	☒	⊙	⬆	✓	⊘	[×]	[×]	[Ns]	[Nd]	[Nd]

Keys:

BAMPPP = Bidirectional Attested Machines Platform Privacy Protection
MTEbCM = Mutual Trust Establishment between Communicating Machines
MA = Mutual Attestation
IA = Integrated Authentication
PWD = Password
NA= Not Applicable

[f] = flexible
[Nd] = Not discussed
[s] = sclable
[Ns] = Not scalable
[su] = simple to use
[nsu] = Not simple to use

Table 6.8: Security, trust, privacy and practicability keys

SECURITY		**TRUST**		**PRIVACY**		**PRACTICABILITY**	
Keys:		Keys:		Keys:		Keys:	
IA	☑ - Achieved ☒ - Absent	MTEbCM	⊘ - Accomplished ⊗ - Not Accomplished ⊖- Not Applicable	BAMPPP	[✔] - Fully Conserved [✕] − Not Applicable	Scalable	[s] [Ns]
Phishing via Trojans	⊙ - Satisfied ● - Not Satisfied					Flexible Simplicity	[f] [Nd] [su] [nsu]
Replay Attack	⇧ - Shielded ⇩ - Not Shielded					Scalable	[s] [Ns]
Root Kit	✓ - Protected ✘ - Not Protected						

Table 6.7 shows the existing works are mainly focused on the integration of (1) security with the privacy or (2) security with the trust and not too much of the existing works focus on the STP aspects unification in a federated environment. Table 6.8 signifies the STP and practicability keys used in Table 6.7. The comprehensive comparative analysis of security, trust and privacy in PUSTPF are discusses in section 6.3.5.1 to 6.3.5.5.

6.3.5.1 Practicability

Practicability, noun of practicable, signifies something (e.g., a plan or an action) that is flexible, scalable and simple to used, implementable in a real environment within the available technologies [78] and fully compatible with the existing FIAM system infrastructure such as Shibboleth [74].

PUSTPF is practicable because it is well-matched with the existing FIAM infrastructure without modifications at the FD SP side. The main modification is at the HD, in which the changes are easier and feasible to carry out because all of the entities such as the users, clients and IdP are part of the HD organization. The practicability of the PUSTPF and other works is analyzed (Table 6.7) on the basis of scalability, flexibility and simplicity (e.g., login convenience) of the approaches.

a) Scalability

In Shibboleth, an open source and standard FIAM mode, user Auth N handler and other attribute production is delegated to the HD entity IdP. Therefore, due to the segregation of the jobs the FD SP doesn't bind to posses the HD user's identities and client machine records which certainly reduces the identity management redundancy and enhance the scalability if additional FD SPs are added afterward.

In bi-directional attestation the mutually integrity measurement of server and client machine platforms in federated environment leads to the privacy and scalability concerns. In PUSTPF, choosing the HD IdP machine (i.e., server) as a challenger entity guarantees that TMAP conserves: (i) To preserve the HD client and IdP machine platforms privacy in the FD, and also the HD IdP machine platforms privacy at the HD client machine, (ii) Scalable. Therefore, in PUSTPF adding new client and revoking of the current client machine platforms trusted security credentials is an easy task because all client machines are registered with the HD organization. Only the HD organization (i.e., IdP) is responsible for its client machine platform's security credential validation and revocation, updating and management of the HD machine platforms good hashes data base entries.

The schemes such as PUSTPF (section 4.3.1.2), [96], [97], [100] is scalable "[s]" because all of the approaches are compatible with the corresponding infrastructure. Whereas, the schemes (e.g., EUSTPF (section 4.3.1.1), [2], [95], [115], [116], [117], and [118]) are not scalable "[Ns]" because they are not fully comparable with the corresponding infrastructure. The rest of the works (e.g., [12], [70], [71], and [72]) are not making use of a remote or mutual attestation protocol.

b) Flexibility

PUSTPF is flexible because the framework may accommodate any other mutual attestation mechanisms. The only changes may require at the HD client and IdP machines are the agents (e.g., ACAgent and AVAs) according to the newly chosen mutual attestation mechanism.

The "[f]" in the column "flexible" under practicability shows that the attestation collector and verification operation in PUSTPF (section 4.3.1.2), EUSTPF (section

4.3,1.1) and [100] is flexible which means these schemes in the future can accommodate any type of attestation mechanism. The schemes such as [12], [70], [71], [72] are not making use of attestation mechanism. The rest of the works (e.g., [2], [95], [96], [97], [115], [116], [117], [118]) do not discuss “[Nd]” the flexibility of the schemes.

c) Simplicity – Login Convnience

PUSTPF is simple to use in real environment because the HD end user only needs to recall his/her basic AuthN credential (i.e., UN/PWD) and not to memorize the HD IdP and client machines platform security credentials. The HD IdP and client machines TPM must be enabled, activated and owned. The enabling, activation and owning TPM is the pre-requisites and one time process. Later in the integrity reporting phase the HD IdP and client machine TPM, work similar to the smart card, report the collected hashes aggregate plus the nonce signed by the respective TPM to the challenging entity. Therefore, after HD user successful login with basic AuthN credential the HD client and IdP mutual integrity attestation process will execute in an automated way.

The schemes (e.g., PUSTPF (section 4.3.1.2), [96], [97], [99], [100]) are simple to use “[su]”. The schemes (e.g., EUSTPF (section 4.2.1.1) and [2]) are not “[nsu]” because of the complex infrastructure. The rest of the schemes in [95], [115], [116], [117], [118] are not discussed “[Nd]” their schemes use in real environment, whereas simplicity of schemes is not applicable “[NA]” to [12], [70], [71], [72] because these schemes do not make use of attestation schemes.

6.3.5.2 Privacy - Machines Platform Privacy Conservation

Shibboleth, used in this work, short-term random IDs certainly preserve the HD user's privacy (i.e., anonymity) in the FD [98]. While unlinkability [25] means the user may use multiple resources (services) without others being able to link together the mutually attested machine platform measurement. In many works, the pseudonym or pseudonymity has been used to ensure that a user may use a resource or service without disclosing his/her identity [25] to the resource (service) provider. Therefore, the user transaction unlinkability at the SP/RP is achievable by using transaction pseudonyms; the transactions which are independent with respect to the user [119], [120]. In the similar fashion HD mutually attested machines anonymity and unlinkability of the transaction at the FD SPs is conserved.

The machines platform privacy conservation (Table 6.7) is discussed below:

- Sharing the HD IdP machine platform security credentials with the client's machine is not suitable because of two reasons: (1) The HD IdP serving many clients so transferring its platform security credential list of the clients may downgrade the IdP performance and (2) May also raise the privacy concerns because the clients and SPs should not need to know the big boss IdP’s machine's platform credentials. Therefore, in PUSTPF the client’s and IdP’s privacy (i.e. anonymity and unlinkability) are conserved at the FD by constructing the new data connector-MIPDC. The HD IdP’s machine in PUSTPF perform its own platforms attestation instead of transferring the platforms credentials to the HD client’s machine.

- Under Bidirectional Attested Machines Platform Privacy Protection (BAMPPP) in “Privacy” column “[✔]”shows that the HD client and IdP machines platform anonymity and unlinkability is fully conserved at the FD Sp in PUSTPF discussed in section 6.2.2. Whereas, “[✕]”indicates that the rest of the work ([12], [72], [100], [96], [97], [71], [70], [2], [95], [99], [115], [116], [117], and [118]) are not fully conserving the mutually attested platform privacy or mutual platform attestation is not included in these architecture.

6.3.5.3 Trust - Mutual Trustworthiness Establishment

In PUSTPF, both the HD client and IdP machines mutually validate each others’ platforms integrity to confirm that both machines are not infected by a malicious attack such as rootkits. Therefore, uncompromised machine platforms mean they are trustworthy and secure.

Table 6.7 illustrates, that the works ([2], [100], [96], [97], [95], and [99]) utilizes the trusted computing based remote machine platform attestation technique. Whereas others ([115], [116], [117], and [118]) make use of the mutual attestation scheme in different areas. While in this work the focus is federated identity and resource (service) access system.

Beneath Mutual Attestation Establishment between Communicating Machines (MTEbCM) under the "Trust" column "⊘"specifies that PUSTPF (section 4.3.1.2), EUSTPF (section 4.3.1.1), ([115], [116], [117], and [118]) all using the mutual attestation technique to establish mutual trust in the machine's platform. Under the MTEbCM "⊗"against [96], [97], [100], [2], [95], [99] shows all of the works missing mutual platform attestation scheme. Whereas, the "⊖"beneath MTEbCM against [12], [72], [70], [71] shows that the TPM is not included in their architecture.

In this work validation and comparison of IMA based attestation schemes (e.g., the effect of SML on the attestation time) are provided. However, the issue such as an increase in the attestation time for the purpose of a user convenience relates to the usability of the attestation scheme which is out of the scope of this work. In this work using SSO feature the mutual attestation of HD client and HD IdP will carried only once the user AuthN with basic credential is successful. Accessing successive resources (e.g., 2nd, 3rd, 4th,…, Nth) the scheme will reuse the "MPI" attribute which represents the HD IdP and client machines platform trustworthiness. Therefore, in this way until the session is active the user can access multiple resources making use of the "MPI" attribute which certainly brings convenience to the end user.

6.3.5.4 Security - Integration of Basic Authentication with MA

In this work Shibboleth basic user AuthN mechanism integration with the trusted computing based mutual attestation scheme, to strengthen the AuthN mechanism, is deomonstrated. The "☑" under Integrated AuhtN (IA) in the security column indicates that the PUSTPF and EUSTPF completely assure combining of UN/PWD mechanism with the mutual attestation mechanism. However, the rest of the works under IA in the security column "☒" indicates that these approaches are not fulfilling the IA (i.e., UN/PWD and MA) According to existing knowledge this is the first work proposed an IA (i.e., UN/PWD and MA) for federated identity and access management mode.

Under “TPM” in security column reveal that some existing works: (i) Make use of the TPM on “Client + Server” / “Mobile Device + Server” side, whereas the other (2) With “No TPM” or only make use of it on “client or mobile devices”.

6.3.5.5 Security - Threats Analysis

How proposed approach (i.e., in section 6.2.2) may protect the HD machine platform against different kind of threat is discussed in this section. The server is referring to the HD IdP machine. In the proposed schemes the integrity and confidentiality protection of the measurement is critical. Therefore, to guarantee the integrity and confidentiality against some threats (such as replay attacks, tampering and masquerading) the mutual attestation mechanism may employ on top of a secure (e.g., SSL) communication link. The analysis of the scheme against different threats is given in Table 6.7.

The “⊙”, “⬆”and “✓” against PUSTPF (section 6.2.2), EUSTPF (section 6.1.1), [115], [116], [117], [118] shows that both “Client + Server” and “Mobile Device + Server” embedded with a TPM to provide platform protection against Phishing via Trojans, replay attack and rootkits Whereas, “⭘”, “⬇” and “×”” against [12], [72], [70], [71] indicates missing of TPM at either client or server side which shows the vulnerability of the schemes against Phishing via Trojans, replay and rocket attacks. In other works (e.g., [100], [96], [97], [2], [95], [99]) “Yes” denotes only client or mobile device embedded with the TPM to protect the platform against Phishing via Trojans, replay and rootkits threats. The analysis of PUSTPF against different threats is comprehensively discussed in the following:

a) Rootkits

Rootkits are malwares that hide themselves in client’s or server’s machines. The hidden property of rootkit boosts its effective lifetime (or existence) and prevents detection. Therefore, the presence of a rootkit on a client or server machines would allow the adversary to take charge of administrative level control of a rootkit infected client or server machines. The client or server machine platforms mutual integrity checking mitigates this issue because the rootkits cannot hijack the aggregated SHA-1 hash values guarded by a client or server TPM.

b) Trojans that lead Phishing Attack

In Trojans attack that may lead to phishing attack, the adversary may steal the user's credentials, using key-logger, from the infected client or server's machines system registry. In such attacks the attackers, first, lure the users to download the Phishing enabler Trojans from a bogus website and then let it to installed on a client machine. Later, whenever a user opens encrypted link (https) to their bogus websites (that mimic a bank or government sites) it then records a user keystrokes, and/or capture screen shots of a user login operation to capture a user login credentials. To counter this attack, in the proposed scheme the TPM based security mechanism is used to measure machines platform integrity and hash them using SHA-1 in an aggregated format whenever the machine is rebooted. Therefore, the existence of Trojan on the attesting machine can be detected in the attestation validation process by checking the SML list, calculated aggregate against the TPM PCR quoted aggregate.

c) Man-in-the-Middle Attack

The adversary may impersonate a legitimate user (client) to acquire a user login credential via a user agent. The adversary then may pretend (by hijacking the communication) to be the legitimate user and responding to the server. To counter this attack, SSL is mandatory to protect the confidentiality of the platforms security credentials. The attacker may not read the hash (e.g., what it is stand for). However, the attacker may use these hashes to prove the server that I am the client machine.

d) Relay Attack

The adversary can mislead both the client and the IdP machines by merely relaying the messages between a client and IdP and vice versa. The adversary may be either a human or system. In relay attack the adversary can read, replace, and alter the message. In the proposed architecture, let assume that the adversary intercepts the SML and PCR quote which contain the aggregated platform measurement of all components loaded in boot process. However, in practice, due to the strength of a tamper proof hardware with TPM based security chip, the reading, alteration and replacement of the PCR quote can not be possible.

e) Replay Attack

In replay attack either client or server machine could be the victim. In normal operation, the CS part of the server is responsible to generate the machine integrity query and forwards to the target machine. The integrity query consists of a nonce. The addition of the nonce in an attestation response enables the client or server that the integrity query and response is not altered. This check is to confirm that the client and the server machines platforms are not infected to any replay attack.

f) Client and Server Machines Legitimacy

The signing of quote over PCR-10, by a client or server machines, gives assurance of PCR quote validation that it is signed by a legitimate (or genuine) TPM. The signing of the quote by the TPM is the legitimacy criterion for a client's and the server's machines are equipped with a legitimate TPM hardware chip.

6.4 Summary

This chapter explained the experimental desgin, experiment outcomes from the trustworthy mutual attestation protocol proof of concept and the practicable cohesive security, trust and privacy framework test-bed prototype. The findings from this research experiments are: (i) Using a mutual attestation scheme instead of a remote attestation in a federated scenario certainly enhances the mutual trust and security between the communicating machine platforms, (ii) The proposed cohesive security, trust and privacy notion may also be applicable in other scenarios such as e-government, Cloud and Grid computing, (iii) The proposed unified security, trust and privacy framework is flexible enough that it can be incorporated in other relevant attestation schemes. The trusted computing is a new security and trust enabler technology for a computing platform which may take time to evolve.

CHAPTER 7

CONCLUSION AND FUTURE WORKS

7.1 Introduction

This chapter presents the conclusion, limitations and future works for this research work. The conclusion of the research work is described in the second section. The third section describes the research limitations and future works.

7.2 Conclusion

Security, trust and privacy unified solution for FIAM systems are urgently needed to curb the STP concerns in these systems. The existing solutions for federated identity and resource (service) access are mostly concentrating on security and trust, or security and privacy but not all three (i.e., STP) in one.

In existing federated identity and access management systems right of entry to the FD secured resources are conceded by: (i) The HD user AuthN mechanisim, and (ii) The HD IdP returns signed SAMLassertion (i.e., the user identifier) to the FD SP, and (iii) SSL/TLS is use to establish secure communicationin link between the HD user browser and HD IdP machine and (iv) SSL/TLS use for the message encryption between the HD IdP and FD SP. In simple words Shibboleth employs uses cryptographanic techniques to protect the SAML assertions and messages. The SSL/TLS and XML digital signatures are used to protect the SAML profiles. In FIAM (e.g., Shibboleth): (i) HD user trust HD for AuthN and (ii) FD SP trust HD IdP for authenticated HD user identity and attribute assertions. The HD IdP trust engine plays a major role in trust relationships (i) and (ii). The HD IdP’s has two (2) trust engines: (i) Signature trust rngine and (ii) Credentials trust engine. The signature trust engine is utilized to validate a digital signature. The purpose of such valdation to ensure that the credentials used are trusted. The credential trust engine gurantees that the credentials used the FD SPs are valid and trust. .However, current federated identity and access management security and trust solutions are based on the software based cryptography solution. In exsiting FIAM systems, the absence of a trusted computing based mutual security and trust formation mechanism may raise several security concerns. The reason is that the infection of an HD IdP or client machine platforms by a malevolent activity may lead to the concerns: (i) HD user’s credential theft at the HD IdP and client machine, (ii). Absence of tust in HD IdP and client machine platforms and (iii) Resource AuthR in the FD is not linked to trusted machine platforms attribute. Such security and trust concerns can bring heavy loss to the HD organization in the event of an infection by malevolent activity.

In addition to the security and trust issues, sharing of private and confidential information may riase the privacy concerns. Trusted computing binary attestation scheme is used to strengthen the computer security and trust in the computing platforms. However, sharing private and confidential target machine platforms security credentials with the challenger may raise the privacy concerns.

In this work PUSTPF is proposed for the FIAM systems that integrates some of the selected STP in a unified manner. For instance, the chosen STP such as: (i) The integration of Shibboleth user basic AuthN mechanism with the HD machine (IdP and client) platforms mutual attestation, (ii) The HD machine platforms mutual security and trust formation, (iii) The privacy protection of the HD machine platforms measurement at the FD and (iv) The resource AuthR in the FD domain by trusted machine platforms attribute.The performance measurement and benchmarking of the attestation scheme is performed. The security, trust and privacy of PUSTPF is validated by the test-bed prototype implementation. The PUSTPF STP, practicability, scalability, flexiability and simplicity comparative analysis is carried out with the existing works

7.3 Research Limitations and Future Works

The identified limitations and the future work of this research are discussed in the following:

7.3.1 Limitations

- The designing of the proposed TMAP established on the IMA [15] approach which will probably lead to the clients' machine measurement privacy issues in financial transaction scenarios. In such scenarios it may not be acceptable for the end users to share their machine platform security credentials with the HD in making a financial transaction.

- In the PUSTPF, the HD IdP does not currently fully support user privacy conservation because in the AuthN process the user is not completely anonymous and there are possibly links to the user transactions. In this work, the main objective is to protect the HD client and IdP privacy in the FD instead of in the internal network because both the HD entities in the presented scenario are in a strong trust association.

- The transferring of complete machine platform security credentials (i.e., measurement list) may introduce performance overhead.

- Other security aspects such as availability (e.g., service disruption due to power outage, hardware failures, system upgrades and Denial of Service (DoS)) and accountability (e.g., access information audit trail).

- Different types and versions of Operating Systems (OSs), executable and application execution.

- In this work IBM machines are used. These machines motherboard embedded with the TPM hardware chip (i.e., infineon brand). This research not covers the Advanced Micro Devices (AMD) processors which has TPM embedded in the chip. The AMD processors extend the courage to the embedded sytems and mobile device to protect the confidential and sensitive information across a wide range of embeded and mobile applications.

- Platform measurement (hashes) tracking which can be of different varieties. The home domain IdP machine performs measurements for its own attestation and registered clients' machine attestation. This will reduce the complexity of the management of too many good hashes because both entities are from the same internal network.

- Trust measurement - this research focuses mainly on mutual trust formation in between communicating machines (i.e., client and server) on the basis of loaded executable health.

- A physical or hardware attack - this research focuses only on a software attack. However, the proposed architecture may also mitigate some physical attacks: (1) No one can extract the Platform Configuration Registers (PCRs) value from the Trusted Platform Module (TPM), if the TPM is attacked physically and (2) Somehow if the attacker detaches the TPM from the IdP or client machine even then they still cannot extract the data stored in the TPM by using another machine/system.

- Network bandwidth and Usability related limitations.

7.3.2 Future Works

- The use of the PBA [93] (or the attestation approaches based on the PBA) in the proposed PUSTPF, particularly in a mutual attestation protocol, for federated identity and resource (service) access management mode may overcome the issue of the machine platform measurement privacy.

- A separate study is also needed on the use of the blind signature scheme [36] in the proposed PUSTPF which hopefully shall improve the user identity privacy protection at the HD IdP.

- The use of the PBA [93] (or maybe the attestation approaches based on the PBA) may help to overcome the issue of performance overhead in the mutual attestation protocol.

- In this work, the TMAP made use of the IMA approach between the HD IdP and client machines in a web environment based FIAM system. In future the TMAP may accommodate the PBA or any other MA mechanism, such as the Policy Reduced Integrity Measurement Architecture (PRIMA) [112] and the Model based Behavioral Attestation (MBA) [69] etc., among the HD clients and IdPs in the PSTPF (or between the HD client and the FD SP in an EUSTPF).

REFERENCES

[1] V. Samar, "Single sign-on using cookies for Web applications," in *Proc. IEEE 8th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, 1999, pp. 158-163.

[2] A. Pashalidis and C. Mitchell, "Single sign-on using trusted platforms," in *Proc. 6th Int. Conf. on Information Security*, Bristol, UK, 2003, pp. 54-68.

[3] A. Pashalidis and C. Mitchell, "A taxonomy of single sign-on systems," in *Proc. 8th Australian Conf. on Information Security and Privacy*, 2003, pp. 249-264.

[4] G. Ahn and J. Lam, "Managing privacy preferences for federated identity management," in *Proc. of the ACM Workshop on Digital Identity Management*, New York, USA, 2005, pp. 28-36.

[5] B. Atkinson, G. Della-Libera, S. Hada, M. Hondo, P. Hallam-Baker, J. Klein and B. LaMacchia, (April 2002, 05). *Web Services Security (WS-Security) Version 1.0* [Online]. Available: http://www.ibm.com/developerworks/library/ws-secure/

[6] S. Yasin, K. Haseeb and R. J. Qureshi, "Cryptography based e-commerce security: A review. *Int. J. Comp. Sci. Issues*, vol. 9 (2), pp. 132-137, 2012.

[7] R. Dhamija and L. Dusseault. (2008, April). The seven laws of identity management: usability and security challenges. *IEEE Security and Privacy 6 (2)*, 24-29. Available: http://doi.ieeecomputersociety.org/10.1109/MSP.2008.49

[8] J. Jensen, “Federated identity management challenges,” in *Proc. of 7th Int. Conf. on Availability, Reliability and Security*, Trondheim, Norway, 2012, pp. 230-235.

[9] D. Smith, "The Challenge of Federated Identity," *J. Network Security,* vol. 4, pp. 7-9, 2008.

[10] D. Shin, G-J. Ahn and P. Shenoy, "Ensuring information assurance in federated identity management," in *Proc. of IEEE Int. Conf. on Performance, Computing, and Communications,* Phoenix, Arizona, USA, 2004.

[11] A. Pandey and J. R. Sani, “An investigation of challenges to online federated identity management systems,” *Int. J. Eng. Innovation and Research*, pp. 50-54, 2012.

[12] S. Cantor, "Shibboleth architecture, protocols and profiles," Tech. Rep. internet2-mace-shibboleth-arch-protocols-200509, Sept. 2005.

[13] R.L. Morgan, S. Cantor, S. Carmody, W. Hoehn and K. Klingenstein, "Federated security: The shibboleth approach," *J. Educause Quarterly,* vol. 27, p. 6, 2004.

[14] *Trusted Computing Group (TCG)* [Online]. Available: http://www.trustedcomputinggroup.org

[15] R. Sailer, X. Zhang, T. Jaeger and L. van Doorn, "Design and implementation of a TCG-based integrity measurement architecture," in *Proc. of the 13th Conf. on USENIX Security Symp.*, Berkeley, USA, 2004, pp. 223-238.

[16] S. Bajikar, "Trusted platform module based security on notebooks PCs," white paper, Mobile Platforms Group Intel Corp., 2002.

[17] P. Hallam-Baker, "Security assertion markup language," Tech. Rep. ver. 1.0, 2001.

[18] R. Philpott, "Security Assertion Markup Language (SAML) v1.1," Tech. Rep., 2004.

[19] T. Smedinghoff, "Federated identity management: Balancing privacy rights, liability risks, and the duty to authenticate," Tech. Rep.2009.

[20] K. Tracy. (2008, Nov./Dec.). Identity management systems. *IEEE Potentials* [Online]. 34-37.

[21] A. Josang, J. Fabre, B. Hay, J. Dalziel and S. Pope, "Trust requirements in identity management," in *Proc. of Australian Workshop on Grid Computing and E-research*, 2005, pp. 99-108.

[22] D. Platt, *Introducing Microsoft. Net*, 2nd ed. Redmond, WA: Microsoft Press, 2001.

[23] A. Josang and S. Pope, "User centric identity management," in *Proc. of Australia Computer Emergency Response Team Conf.*, Brisbane, Australia, 2005.

[24] U. Fragoso-Rodriguz, M. Laurent-Maknavicius and J. Incera-Dieguez, "Federated identity architectures," in *Proc. of 1st Mexican Conf. on Informatics Security*, 2006.

[25] A. Pfitzmann and M. Köhntopp, "Anonymity, unobservability, and pseudonymity -a proposal for terminology," in Proc. of Int. Workshop on Designing Privacy Enhancing Technologies: Design Issues in anonymity and unobservability, 2001, pp. 1-9.

[26] S. Steinbrecher and S. Köpsell, "Modelling unlinkability," in *Proc. of 3rd Int. Workshop on Privacy Enhancing Technologies*, 2003, pp. 32-47.

[27] F. Hirsch, R. Philpott, E. Maler, C. P. Cahill, J. Hughes and H. Lockhart, *et al.*, "Security and privacy considerations for the OASIS security assertion markup language v2.0," Oasis Standard, Tech. Rep. saml-sec-consider-2.0-os, 2005.

[28] S. Cantor, J. Kemp, R. Philpott, E. Maler, C. P. Cahill and J. Hughes, *et al.*, "Assertions and protocols for the OASIS security assertion markup language," OASIS Standard, Tech. Rep. saml-core-2.0-os, Mar. 2005.

[29] S. Cantor, J. Moreh, R. Philpott, E. Maler, C. P. Cahill and J. Hughes, *et al.*, "Metadata for the OASIS security assertion markup language v2.0," OASIS Standard, Tech. Rep. saml-metadata-2.0-os, Mar. 2005.

[30] C. Cantor, F. Hirsch, J. Kemp, R. Philpott, E. Maler and C. P. Cahill, *et al.*, "Bindings for the OASIS security assertion markup language v2.0," OASIS Standard, Tech. Rep. saml-bindings-2.0-os, Mar. 2005.

[31] E. Maler, P. Mishra, R. Philpott, S. Farrell, I. Reid and H. Lockhart, *et al.*, "Assertions and protocol for the OASIS security assertion markup language v1.1," OASIS Standard, Tech. Rep. oasis-sstc-saml-core-1.1, Sept. 2003.

[32] J. Hughes, S. Cantor, J. Hodges, F. Hirsch, P. Mishra and R. Philpott, *et al.*, "Profiles for the OASIS security assertion markup language v2.0," OASIS Standard, Tech. Rep. saml-profiles-2.0-os, Mar. 2005.

[33] E. Maler, P. Mishra, R. Philpott, I. Reid, K. Sankar and J. Hughes, *et al.*, "Bindings and profiles for the OSAIS Security Assertion Markup Language (SAML) v1.1," OASIS, Tech. Rep.2003.

[34] K. C. Boydstun, S. W. Grimm and S. R. Hentzen, "Identity management system and method," U.S. Patent 7571473, Aug 4, 2009.

[35] S. Clauß and M. Köhntopp, "Identity management and its support of multilateral security," *J. Comput. Networks*, vol. 37, pp. 205-219, 2001.

[36] D. Chaum, "Blind signature for untraceable payments," in *Proc. of Advances in Cryptography*, August 1982, pp. 199-203.

[37] R. L. Rivest, A. Shamir and L. Adleman. (1978, Feb.). A method for obtaining digital signatures and public-key cryptosystems. *Commun. of ACM* [Online]. pp. 20-126.

[38] S. Pearson and B. Balacheff, *Trusted Computing Platforms: Tcpa Technology in Context*. Upper Saddle River: Prentice Hall, 2003.

[39] D. Eastlake and P. Jones, "US Secure Hash Algorithm 1 (SHA)," RFC 3174. IETF Network Working Group. 2001.

[40] "Trusted computing for the Java(tm) platform, Java Trusted Software Stack (jTSS)," Graz University of Technology (TUG), EU OpenTC project (Ref. Nr. 027635).

[41] A. Hermann, "An investigation of Microsoft&#39; Passport protocol and issues regarding its security, privacy," SANS Institute InfoSec Reading Room, 2001.

[42] M. Slemko. (2001). *Microsoft Passport to Trouble [*Online]. Available: http://www.znep.com/~marcs/passport

[43] D. W. Chadwick, "Federated Identity Management," in *Foundations of Security Analysis and Design V*, vol. 5705, ed: Springer-Verlag, 2009, pp. 96-120.

[44] M. Hansen and H. Krasemann, "Privacy and Identity Management for Europe (PRIME)," PRIME consortium, PRIME-White Paper, 2005.

[45] D. Recordon and B. Fitzpatrick, "OpenID Authentication 1.1," Tech. Rep. 2006.

[46] H.-K. Oh and S.-H. Jin, "The security limitation of SSO in OpenID," in *Proc. of 10th Int. Conf. on Advanced Communication Technology*, Gangwon-Do, 2008, pp. 1608-9445.

[47] S. James, "Web single sign-on systems," Survey Rep. on Recent Advances in Network Security, 2007.

[48] A. Nanda and M. Jones, "Identity selector interoperability profile v1.5," Tech. Rep.2008

[49] D. Chappell. (2006, April). *Introducing Windows CardSpace* [Online]. Available: http://msdn.microsoft.com/en-us/library/aa480189.aspx

[50] OECD. (2002). *Organization for Economic Co-operation and Development (OECD) Guidelines on the Protection of Privacy and Transborder Flows of Personal Data*. Available: http://www.oecdbookshop.org/oecd/display.asp?sf1=identifiers&st1=9789264196391.

[51] R. Sherwood, R. Sherwood and B. Bhattacharjee, "Cooperative peer groups in NICE," *J. Comput. Networks*, vol. 50, pp. 523-544, 2006.

[52] A. Abdul-Rahman and S. Hailes, "A distributed trust model," in *Proc. of the ACM Workshop on New Security Paradigm*, New York, USA, 1997, pp. 48-60.

[53] A. Ansper, A. Buldas, M. Roos and J. Willemson, "Efficient long-term validation of digital signatures," in *Proc. of the 4th Int. ACM Workshop on Practice and Theory in Public Key Cryptography: Public Key Cryptography*, 2001, pp. 402-415.

[54] J. D. Lewis and A. J. Weigert, "Social atomism, holism, and trust," *The Sociological Quarterly*, vol. 2l6, pp. 455-471, 1985.

[55] J. K. Rempel, J. G. Holmes and M. P. Zanna, "Trust in close relationships," *J. Personality and Social Psychology*, vol. 49, pp. 95-112, 1985.

[56] M. A. Eayrs, "Time, trust and hazard: Hairdressers' symbolic roles," *J. Symbolic Interaction*, vol. 16, pp. 19-37, 1993.

[57] J. L. Bradach and R. G. Eccles. (1989, August). Price, authority, and trust: From ideal types to plural forms. *Int. J. Annu. Review of Socio.* [Online]. *15*, 97-118. Available: http://www.annualreviews.org/doi/abs/10.1146/annurev.so.15.080189.000525

[58] P. B. Gove’s, "Webster’s 3rd International Dictionary” in *Springfield, Mass*, ed. Merriam-Webster, 1981, p. 198.

[59] H. McKnight and N. L. Chervany, "The meanings of trust," Manage. Inform. Systems Research Center, University of Minnesota, Tech. Rep. MISRC 96-04, 1996.

[60] A. Jøsang and S. L. Presti, "Analysing the relationship between risk and trust," in *Proc. of the 2nd Int. Conf. on Trust Management*, Oxford, UK, 2004, pp. 135-145

[61] S. Cantor, J. Hodges, J. Kemp, P. Thompson and T. Watson, "Liberty identity Federation Framework (ID-FF) architecture overview V1.2," Tech. Rep. errata-v1.0, 2005.

[62] K. Helenius, "OpenID and identity management in consumer services on the Internet," presented at the Current Internet Trends Seminar on Internetworking, Finland, 2009.

[63] F. Culloch, "OpenID and SAML," Stockholm, Tech. Rep. EuroCAMP, 2008.

[64] P. Madsen, Y. Koga and K. Takahashi, "Federated identity management for protecting users from ID iheft," in *Proc. of the ACM Workshop on Digital Identity Management*, New York, USA, 2005, pp. 77-83.

[65] W. D. Yu, S. Nargundkar and N. Tiruthani, "A phishing vulnerability analysis of web based systems," presented at the IEEE Symp. on Computers and Communications, Marrakech, 2008.

[66] D. Jevans, "Microsoft Live ID phishing illustrates the dangerous of federated identity," Privacy and Identity Theft. 2009.

[67] E. Mills, “ID fraud malware infecting PCs at increasing rates," Security - CNET, 2009.

[68] "Trusted Computing Group (TCG) specification architecture overview ver. 1.4," TCG, Tech. Rep., pp. 11-22, 2007.

[69] M. Alam, X. Zhang, M. Nauman, T. Ali and J-P. Seifert, "Model-based behavioral attestation," in *Proc. of the 13th ACM symposium on Access control models and technologies,* New York, USA, 2008, pp. 175-184.

[70] R. Watanabe and T. Tanaka, "Federated authentication mechanism using cellular phone - collaboration with OpenID," in *Proc. of 6th Int.l Conf. on Information Technology: New Generations*, Las Vegas, USA, 2009, pp. 435-442.

[71] D. J. Lutz and R. Campo, "Bridging the gap between privacy and security in multi-domain federations with identity tokens," in *Proc. of 3rd Annu. Int. Conf. on Mobile and Ubiquitous Systems: Networking and Services*, 2006, pp. 1-3.

[72] A. Dey and S. Weis, "PseudoID: Enhancing privacy for federated Login," in *Proc. of 3rd Symp. on Hot Topics in Privacy Enhancing Technologies* Berlin, Germany, 2010, pp. 95-107.

[73] S. Singh and S. Bawa, "A privacy, trust and policy based authorization framework for services in distributed environments,"*J. Computer Science*, vol. 2, pp. 85-92, 2007.

[74] S. Kungpisdan, B. Srinivasan and P. D. Le, "A Practical Framework for MobileSET Payment", in *Proc. of Int. ESociety Conf.*, 2003b, pp. 321-328.

[75] A. Mondal and M. Kitsuregawa, "Privacy, security and trust in P2P environments: A Perspective," in *Proc. of 17th Int. Workshop on Database and Expert Systems Applicat.*, 2006, Krakow, Poland, 4-8 Sept. 2006, pp. 682-686.

[76] S. Karnouskos, A. Hondroudaki, A. Vilmos and B. Csik, "Security, Trust and Privacy in the SEcure MObile Payment Service," in *Proc. of 3rd Int. Conf. on Mobile Business*, New York, U.S.A, 12-13 July 2004.

[77] D. van Rooy and J. Bus, "Trust and privacy in the future internet - a research persepective," *J Identity in the Information Society*, vol. 3, pp. 397-404, 2010.

[78] What is Practicable? http://oald8.oxfordlearnersdictionaries.com/dictionary/practicable

[79] U. Maurer, "Modeling a public key infrastructure," in *Proc. of 4th European Symp. on Research in Computer Security*, Rome, Italy, 1996, pp. 325-350.

[80] J. Camenisch and A. Lysyanskaya, "An efficient system for non-transferable anonymous credentials with optional anonymity revocation," in *Proc. of the Int. Conf. on the Theory and Applicat. of Cryptographic Techniques: Advances in Cryptology*, London, UK, 2001, pp. 93-118.

[81] Camenisch. J and L. A, "Signature schemes and anonymous credentials from bilinear maps," in *Proc. of 24th Int. Conf. on Cryptography*, California, USA, 2004, pp. 56-72.

[82] L. Chen, "Access with pseudonyms," in *Proc. of Int. Conf. on Cryptography: Policy and Algorithms*, Queensland, Australia, 1996, pp. 232-243.

[83] A. Lysyankaya, R. L. Rivest, A. Sahai and S. Wolf, "Pseudonym systems: Extended abstract," in *Proc. of 6th Annu. Int. Workshop on Selected Areas in Cryptography*, Ontario, Canada, 2000, pp. 184-199.

[84] I. B. Damgård, "Payment systems and credential mechanisms with provable security against abuse by individuals," in *Proc. of Advances in Cryptology (Extended Abstract)*, New York, USA, 1990, pp. 328-335.

[85] S. A. Brand's, *Rethinking Public Key Infrastructure and Digital Certificates: Building in Privacy*. Cambridg: MIT Press, 2000.

[86] J. Camenisch and T. Groß, "Efficient attributes for anonymous credentials," in *Proc. of the 15th ACM Conf. on Comput. and Commun. Security*, New York, USA, 2008.

[87] J. Camenisch, S. Hohenberger and A. Lysyanskaya, "Compact e-cash," in *Proc. of 24th Int. Conf. on the Theory and Applicat. of Cryptographic Techniques*, Aarhus, Denmark, 2005, pp. 302-321.

[88] P. Giuseppe and I. Visconti, "An efficient and usable multi-show non-transferable anonymous credential system," in *Proc. of 8th Int. Conf. on Financial Cryptography*, 2004, pp. 196-211.

[89] E. R. Verheul, "Self-blindable credential certificates from the weil pairing," in *Proc. of the 7th Int. Conf. on the Theory and Application of Cryptology and Information Security: Advances in Cryptology*, 2001, pp. 533-551.

[90] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Trans. Inform. Theory,* vol.22, pp. 644-654, 1976.

[91] E. Brickell, J. Camenisch and L. Chen, "Direct Anonymous Attestation (DAA)," in *Proc. of the 11th ACM Conf. on Comput. and Commun. Security*, New York, USA, 2004, pp. 132-145.

[92] A. Leung, L. Chen and C. J. Mitchell, "On a possible privacy flaw in Direct Anonymous Attestation (DAA)," Royal Holloway University, Tech. Rep. RHUL-MA-2007-10, 2007.

[93] A. R. Sadeghi and C. Stüble, "Property-based attestation for computing platforms: Caring about properties, not mechanisms," in *Proc. of the Workshop on New Security Paradigms*, New York, USA, 2004, pp. 67-77.

[94] A. Nagarajan, V. Varadharajan, M. Hitchens and E. Gallery, "Property based attestation and trusted computing: Analysis and challenges," in *Proc. of 3rd Int. Conf. on Network and System Security*, Gold Coast, Queensland, Australia, 2009, pp. 278-285.

[95] S. Garriss, R. Cáceres, S. Berger, R. Sailer, L. van Doorn and X. Zhang, "Towards trustworthy kiosk computing," in *Proc. of 8th IEEE Workshop on Mobile Computing Systems and Applications*, 2007, pp. 41-45.

[96] A. Klenk, H. Kinkelin, C. Eunicke and G. Carle, "Preventing identity theft with electronic identity cards and the trusted platform module," in *Proc. of the 2nd Workshop on System Security,* New York, USA, 2009, pp. 44-51.

[97] A. Leicher, A. U. Schmidt, Y. Shah and I. Cha, "Trusted computing enhanced OpenID," in *Proc. of Int. Conf. on Internet Technology and Secured Transaction*, London, UK, 2010, pp. 1-8.

[98] A. Pashalidis and C. Mitchell, "Privacy in identity and access management," in *Digital Identity and Access Management: Technologies and Frameworks*, ed: IGI Global, 2011, pp. 316-328.

[99] T. Ali and M. Numan, "Incorporating remote attestation for end-to-end protection in web communication paradigm," in *Proc. of the 3rd International Conf. on Internet Technologies and Applicat.*, Wrexham, North East Wales, UK, 2009.

[100] T. Ali, M. Nauman, M. Amin and M. Alam, "Scalable, privacy-preserving remote attestation in and through federated identity management frameworks," in *Proc. of Int. Conf. on Information Sci. and Applicat.*, Seoul, South Korea, 2010, pp. 1-8.

[101] H. A. Simon, *The Science of the Artificial*, 3rd ed. Cambridge: MIT Press, 1996.

[102] D. Chuan, Y. Lin, M. Linru and C. Yuan, "Towards a practicable and scalable trusted software dissemination system," *Int. J. Convergences*, vol. 2, pp. 53-60, 2011.

[103] J. Camenisch, "Better privacy for trusted computing platforms," in *Proc. of the 9th European Symp. on Research in Comput. Security*, Sophia Antipolis, France, 2004, pp. 73-88.

[104] A. Pashalidis and C. Mitchell, "Single sign-on using TCG- conformant platforms," in *Trusted Computing*, ed: Institution of Engineering and Technology, 2005.

[105] L. Trussell, "Essential software development methodology," presented at the IEEE Power Eng. Society Winter Meeting, 1999.

[106] P. Kruchten, *The Rational Unified Process: An Introduction*, 3rd ed. Boston: Addison-Wesley Professional, 2004.

[107] A. Jøsang, "RE: Extending Model Request," Personal e-mail to: Z. A. Khattak, 2010.

[108] F. Warwick and M. S. Baum, *Secure Electronic Commerce: Building the Infrastructure for Digital Signatures and Encryption*, 2nd ed. Upper Saddle River, New Jersey, USA: Prentice Hall, 2000.

[109] H. Abelson and L. Lessig, "Digital identity in cyberspace," white paper, submitted for 6.805/Law of Cyberspace: Social Protocols,1998.

[110] S. W. Ambler, "A managers introduction to the rational unified process," white paper, 2005.

[111] T. Bray. (2000). Extensible markup language (XML) 1.0 (2nd edition). Available: http://ukpmc.ac.uk/abstract/CIT/372776

[112] T. Jaeger, R. Sailer and U. Shankar, "PRIMA: policy-reduced integrity measurement architecture," in *Proc. of the 11th ACM symp. on Access control models and technologies*, Lake Tahoe, California, USA, 2006, pp. 19-28.

[113] *IAIK: Institute for Applied Information Processing and Communications, Graz University of Technology [*Online]. Available: http://www.iaik.tugraz.at/

[114] J. Hughes and E. Maler, "Security Assertion Markup Language (SAML) V2. 0," Tech. Overview, OASIS SSTC Working Draft v-08.

[115] R. Sailer, L. van Doorn and J. P. Ward, "The role of TPM in enterprise security. ," IBM Research, Technical Report RC23363 (W0410-029), 2004.

[116] S. Balfe, E. Gallery, C. J. Mitchell and K. G. Paterson, "Crimeware and trusted computing," in *Crimeware Understanding New Attacks and Defenses*, ed: Addison-Wesley/ Symantec Press, 2008.

[117] J. Zhan, H. Zhang and F. Yan, "Building trusted sub-domain for the grid with trusted computing," in *3rd SKLOIS Conf. on Information Security and Cryptography*, Xining, China, 2007, pp. 463-471.

[118] R. Cáceres and R. Sailer, "Trusted mobile computing," in *Proc. of IFIP Networking Workshop on Security and Privacy in Mobile and Wireless Networking*, Combia, Portugal, 2006.

[119] G. Müller and S. Wohlgemuth. (2007). *Study on privacy in business processes by identity management* [Online]. Available: http://www.fidis.net/fileadmin/fidis/deliverables/fidis-wp14-del14.2-study_on_privacy_in_business_processes_by_identity_management.pdf

[120] M. Gilliot, V. Matyas and S. Wohlgemuth, "Privacy and Identity," in *The Future of Identity in the Information Society*, K. Rannenberg, *et al.*, Eds., ed: Springer Berlin Heidelberg, 2009, pp. 351-390.

[121] S. Lahlou, M. Langheinrich and C. Röcker, "Privacy and trust issues with invisible computers," *J. Commun. ACM*, vol. 48, pp. 59-60, 2005.

[122] A. Emigh, "The Crimeware landscape: Malware, phishing, identity theft and beyond," *J. Digital Forensic Practice*, vol. 1, pp. 245-260, 2006.

[123] OECD. (1980). *Organization for Economic Co-operation and Development (OECD) Guidelines on the Protection of Privacy and Transborder Flows of Personal Data*.

[124] M. Priestley and M. H. Utt, "A unified process for software and documentation development," in *Proc. of IEEE Professional Communication Society Int. Professional Communication Conf. and Proc. of the 18th Annu. ACM Int. Conf. on Computer Documentation: Technology \& Teamwork*, Cambridge, Massachusetts, 2000, pp. 221-238.

[125] S. Pearson, "Trusted computing platforms, the next security solution," Trusted Systems HP Laboratories, Bristol, UK, 2002.

[126] D. Kallath, "Trust in trusted computing – the end of security as we know it," *J. Comput. Fraud and Security,* vol. 2005, pp. 4-7, 2005

[127] C. Mitchell, *Trusted Computing*. London, UK: IEE Press, 2005.

[128] P. Windley, *Digital Identity*: O'Reilly Media, Inc., 2005.

[129] S. S. Y. Shim, G. Bhalla and V. Pendyala. (Dec. 2005) Federated identity management. *IEEE Computer* [Online]. pp. 120-122.

[130] H. Li and M. Singhal. (2007, Feb). Trust management in distributed systems. *IEEE Computer* [Online]. pp. 45-53.

[131] D. Gollmann. (2006, 20 May). Why Trust is Bad for Security. *Electronic Notes in Theoretical Comput. Science* [Online]. *157(3),* 3-9. Available: http://dx.doi.org/10.1016/j.entcs.2005.09.044

[132] R. K. Sullivan, "The case for federated identity," *J. Network Security,* vol. 2005, pp. 15-19, 2005.

[133] M. Hansen, A. Schwartz and A. Cooper. (2008, April). Privacy and Identity Management. *IEEE Security and Privacy*. 38-45. Available: http://doi.ieeecomputersociety.org/10.1109/MSP.2008.41

[134] I. Novakov, "Web single sign-on systems," CESNET Tech. Rep. 21/2006, 2006.

[135] M. L. Garandcolas, F. Law, A. Doshi, M. Williams, Y. Jang, T. Merschen and J. Pan, "Method and System for Single Sign-on user access to multiple web servers," USA Patent US 7,137,006 B1, 2006.

[136] K.-K. R. Choo, "Issue report on business adoption of Microsoft Passport," *J. Inform. Manage. & Comput. Security,* vol. 14 pp. 218-234, 2006.

[137] P. Gosling, "Trojans &amp; spyware: an electronic achilles," *J. Network Security,* vol. 2005, pp. 17-18, 2005.

[138] A. M. Marshall and B. C. Tompsett, "Identity theft in an online world," *J. Comput. Law & Security Review*, vol. 21, pp. 128-137, 2005.

[139] K. J. Biba, "Integrity Considerations for Secure Computer Systems," Tech. Rep. MTR-3153, Mitre Corporation, Mitre Corp, Bedford MA,June 1975.

[140] D. D. Clark and D. R. Wilson, "A Comparison of Commercial and Military Computer Security Policies," in *IEEE Symposium on Security and Privacy*, 1987, pp. 184-194.

[141] S. W. Smith, "Outbound Authentication for Programmable Secure Coprocessors," in *Proceedings of the 7th European Symposium on Research in Computer Security*, 2002, pp. 72-89.

[142] M. Urueña and C. Busquiel, "Analysis of a Privacy Vulnerability in the OpenID Authentication Protocol," presented at the EEE Multimedia Communications, Services and Security, Best Paper Award,, Cracovia, 2010.

[143] B. van Delft and M. Oostdijk, "A Security Analysis of OpenID Policies and Research in Identity Management," in *Policies and Research in Identity Management*. vol. 343, E. de Leeuw, *et al.*, Eds., ed: Springer Boston, 2010, pp. 73-84.

[144] O. Hyun-Kyung and J. Seung-Hun, "The Security Limitations of SSO in OpenID," in *10th International Conference on Advanced Communication Technology*, 2008, pp. 1608-1611.

[145] S. Boeyen, G. Ellison, N. Karhuluoma, W. MacGregor, P. Madsen and S. Sendodan, *et al.*, "Liberty Trust Models Guidelines ver. 1.0," Tech. Rep. liberty-trust-models-guidelines-v1.0.pdf, 2003.

[146] M. Needleman, "The Shibboleth Authentication/Authorization System," *J. Serials Review*, vol. 30, pp. 252-253, 2004.

[147] D. P. Kormann and A. D. Rubin, "Risks of the Passport single signon protocol," *J. Computer Networks*, vol. 33, pp. 51-58, 2000.

[148] R. Oppliger. (2003) Microsoft .NET passport: A security analysis. *IEEE Computer Society*. pp. 29-35.

[149] R. Oppliger, "Microsoft .NET Passport and identity management," *Information Security Technical Report*, vol. 9, pp. 26-34, 2004.

[150] M. Koch and K. M. Möslein, "Identities Management for E-Commerce and Collaboration Applications," *Int. J. Electronic Commerce*, vol. 9, pp. 11-29, 2005.

[151] D. Artz and Y. Gil, "A survey of trust in computer science and the Semantic Web," *J. Web Semantic*, vol. 5, pp. 58-71, 2007.

[152] J. Golbeck, *Trust on the World Wide Web: A Survey*: Now Publishers Inc., 2008.

[153] G. Zhao, D. Zheng and K. Chen, "Design of single sign-on," in *IEEE Int. Conf. on E-Commerce Technology for Dynamic E-Business*, Beijing, 2004, pp. 253-256.

[154] M. Vlad, “Rootkits and malicious code injection,” *J. Mobile, Embeded and Distributed Systems*, vol. 3 (2), pp. 91-99, 2011.

[155] M.I. Ladan, “Web Services: Security Challenges,” in *Proc. of World Congress on Internet Security (WorldCIS)*, Beruit, Lebanon, pp. 160-163, 2011.

[156] E. Bertino, L. D. Martino, F. Paci, A. C. Squicciarini, “Standards for web services security,” in Security for Web Services and Service Oriented Architecture, Springer, pp. 45-77, 2010.

[157] M. Speltens and P. Patterson, “Federated ID management – tackling risk and credentialing users, in ISSE/ SECURE 2007 Securing Electronic Business Probess, pp. 130-135, 2007.

[158] X. Chen, “Identity federation in federated trust healthcare network,” Master Thesis, 2004.

[159] Hacket and K. Hawkey, “Security, privacy and usability requirements for federated identity,” in *IEEE Symp on Security and Privacy*, San Fransisco, USA, May 24-25, 2012.

[160] J. Camenisch, E. Brickell and L. Chen, “Direct Anonymous Attestation: Achieving privacy in remote attestation,” 2004.

PUBLICATIONS

1. Z. A. Khattak, J-L Ab Manan, and S. Sulaiman, "A study on threat model for federated identities in federated identity management system," in *Proc. of Int. Symp. on Inform. Technology*, Kuala Lumpur, Malaysia, 2010, pp. 618-623. http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=05561611

2. Z. A. Khattak, J-L Ab Manan, and S. Sulaiman, "User requirement model for federated identities threats," in *Proc. of 3rd Int. Conf. on Advanced Computer Theory and Engineering*, vol. 6, China, pp. 317–321, 2010. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&zrnumber=5579819

3. Z. A. Khattak, J-L Ab Manan, and S. Sulaiman, "Trusted computing based open environment user authentication model," in *Proc. of 3rd Int. Conf. on Advanced Computer Theory and Engineering*, China, vol. 6, pp. 487–491, 2010. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&zrnumber=5579171

4. Z. A. Khattak, J-L Ab Manan, and S. Sulaiman, "Finding new solutions for services in federated open systems interconnection," in *Proc. of the Ist Int. Conf. on Advances in Computing and Communications*, India, vol. 193, pp. 250-259, Springer-Verlag Berlin Heidelberg, 2011. http://www.springerlink.com/content/tt645h7862240j37/

5. Z. A. Khattak, J-L Ab Manan, and S. Sulaiman, "Proof of concept implementation of trustworthy mutual attestation architecture for true single sign-on," in *Proc. of 10th Int. Conf. on Security and Management*, Las Vegas, Nevada, USA, 2011.

6. Z. A. Khattak, J-L Ab Manan, and S. Sulaiman, "Security, Trust and Privacy (STP) framework for federated single sign-on environment," in *Proc. of 5th Int. conf. on Information Technology and Multimedia*, Kuala Lumpur, Malaysia, 14-16 November 2011.

7. J-L Ab Manan, M. F. Mubarak, M. A Mat Isa, and Z. A. Khattak, "Security, trust and privacy – A new direction for pervasive computing," in *Proc. of WSEAS Int. Conf. on Communication and Information Technology*, Greece, 2011.

8. J-L Ab Manan, Z. A. Khattak, and S. Sulaiman, “Practicable unified security, trust and privacy framework for federated access management,” in Proc. of the 11th Int. Conf. on Trust, Security and Privacy in Computing and Communications, pp. 1411-1416, Liver Pool, UK, 2012.

9. Z. A. Khattak, J-L Ab Manan, and S. Sulaiman, "Analysis of open environment sign-in schemes: privacy-enhanced and trustworthy approach," *J. Advances in Inform. Technology*, vol. 2, pp. 109-121, 2011.

10. Z. A. Khattak, J-L Ab Manan, and S. Sulaiman, “Trustworthy Mutual Attestation Protocol for Local True Single Sign-On System: Proof of Concept and Performance Evaluation,” *J. Transaction of Internet and Information Systems,* vol. 6, pp. 2405-2423, 2012.

11. Z. A. Khattak, J-L Ab Manan, and S. Sulaiman, "Evalouation of Unified Security, Trust and Privacy (STP) Framework (UnifiedSTPF) for Federated Identity and Access Management (FIAM) Mode," *J. Computer Application*, vol. 54, pp. 12-19, 2012.

APPENDIX A

RELATED SOURCE CODE SNIPS

Home Domain Client MODIFICATION

1. Home Domain Client Attestation Collector Package

(i) CLASS: INTERFACE ABSTRACT COLLECTOR

```
    package pustpf.ima.net.client.attestationcollector;
    import org.w3c.dom.Document;
    public interface AbstractCollector {
    public Document process(byte[] nonce);
}
```

(ii) CLASS: ATTEST CLIENT PCR

```
    import iaik.tc.tss.api.tspi.TcIRsaKey;
    import iaik.tc.tss.api.tspi.TcITpm;
    import iaik.tc.tss.impl.csp.TcCrypto;
    import iaik.tc.utils.logging.Log;
    import java.io.File;
    import java.io.FileInputStream;
    import java.security.Signature;
    import java.security.interfaces.RSAPublicKey;
    import javax.crypto.Cipher;
    import javax.xml.parsers.DocumentBuilderFactory;
    import org.bouncycastle.jce.provider.BouncyCastleProvider;
    import org.w3c.dom.Document;
    import org.w3c.dom.Node;
    import org.w3c.dom.traversal.NodeIterator;
    import com.sun.org.apache.xpath.internal.CachedXPathAPI;
    import pustpf.ima.net.service.TssService;
    import pustpf.ima.net.service.XmlService;
    public class AttestClientPCR implements AbstractCollector {




    }
                return pcrs;
    }
    private Node getPCRContainer(Document src) {
        try {
            String xpath = "//pcrs[1]";
            CachedXPathAPI path = new CachedXPathAPI();
            NodeIterator nl = path.selectNodeIterator(src, xpath);
            Node n;
            if ((n = nl.nextNode()) != null) {
                return n;
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        return null;
    }
    public void verify(byte[] signedPcrVal, RSAPublicKey pubKey, byte[] pcrVal) {
        try {
            long valPcrStart = System.currentTimeMillis();
            Signature sig = Signature.getInstance("SHA1withRSA");
            sig.initVerify(pubKey);
            sig.update(pcrVal);
            boolean verifies = sig.verify(signedPcrVal);
            long valPcrEnd = System.currentTimeMillis();
            System.out.println("Pcr Validation Time:"
                    + (valPcrEnd - valPcrStart));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

(iii) CLASS: ATTEST CLIENT SML

```
  package pustpf.ima.net.client.attestationcollector;
  import iaik.tc.tss.api.structs.common.TcBlobData;
  import iaik.tc.tss.api.tspi.TcIContext;
  import iaik.tc.tss.api.tspi.TcIPcrComposite;
  import iaik.tc.utils.logging.Log;
  import iaik.tc.tss.api.structs.tpm.TcTpmPcrComposite;
  import iaik.tc.tss.impl.java.tsp.TcPcrCompositeInfoLong;
  import java.io.File;
  import java.util.StringTokenizer;
  import javax.xml.parsers.DocumentBuilderFactory;
  import org.w3c.dom.Document;
  import pustpf.ima.net.service.TssService;
  import pustpf.ima.net.service.XmlService;
  public class AttestClientSML implements AbstractCollector {
    private String IMAPATH =
"/sys/kernel/security/ima/ascii_runtime_measurements";
    @Override
    public Document process(byte nonce[]) {
            Document sml = null;
            try {
                    sml =
DocumentBuilderFactory.newInstance().newDocumentBuilder()
                                    .newDocument();
                    sml.appendChild(sml.createElement("sml"));
            }
            catch (Exception e) {
                    Log.err(e.getMessage());
                    e.printStackTrace();
            }
            String IMAPATH =
"/sys/kernel/security/ima/ascii_runtime_measurements";
            File f = new File(IMAPATH);
            String contents = "";
            if (f.exists())




    XmlService.appendElement(sml, sml.getDocumentElement(), "smlcontents",
contents);
            System.out.println();
            return sml;
    }
}
```

(iv) CLASS: DAEMON ATTESTATION COLLECTOR

```
    package pustpf.ima.net.client;
    import iaik.tc.utils.logging.Log;
    import javax.xml.parsers.DocumentBuilderFactory;
    import javax.xml.transform.*;
    import javax.xml.transform.dom.DOMSource;
    import javax.xml.transform.stream.StreamResult;
    import org.w3c.dom.*;
    import org.w3c.dom.traversal.NodeIterator;
    import org.xml.sax.InputSource;
    import com.sun.org.apache.xpath.internal.CachedXPathAPI;
    import pustpf.ima.net.client.attestationcollector.*;
    import pustpf.ima.net.service.*;
    public class DaemonAttestationCollectorAgent {
        public Document doAttestation(Document request) {
                Document returnedDoc = null;
                try {
                        returnedDoc = DocumentBuilderFactory.newInstance()
                                        .newDocumentBuilder().parse(
                                                "/home/zubair/responseSkel.xml");
                        String nonce = getNonce(request);
                        byte nonceArray[] = XmlService.toBinArray(nonce);
                        XmlService.appendElement(returnedDoc,
                                        getResponseMainNode(returnedDoc),
"Challenge", nonce);
                        Document pcrs = (new
AttestClientPCR()).process(nonceArray);
                        XmlService.importName(pcrs, returnedDoc ,
                                        getResponseMainNode(returnedDoc));
                        Document sml = (new
AttestClientSML()).process(nonceArray);
                        XmlUtil.importName(sml,returnedDoc ,




    }
    private Node getResponseMainNode(Document src) {
                try {
                        String xpath = "//RequestedSecurityToken[1]";
                        CachedXPathAPI path = new CachedXPathAPI();
                        NodeIterator nl = path.selectNodeIterator(src, xpath);
                        Node n;
                        if ((n = nl.nextNode()) != null) {
                                return n;
                        }
                } catch (Exception e) {
                        e.printStackTrace();
                }
                return null;
        }
}
```

2. Home Domain Client XML and TSS Service Package

(v) CLASS: XML-SERVICE

```
package pustpf.ima.net.service;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import java.math.BigInteger;
public class XmlService {
    static char[] hexChar = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9',
                    'A', 'B', 'C', 'D', 'E', 'F' };
    public static void appendElement(Document doc, Node to, String name,
                    String val) {
            Element newEl = doc.createElement(name);
            Node newElVal = doc.createTextNode(val);
    public static byte[] toBinArray(String hexStr) {
                byte bArray[] = new byte[hexStr.length() / 2];
                for (int i = 0; i < (hexStr.length() / 2); i++) {
                        byte firstNibble = Byte.parseByte(hexStr
                                        .substring(2 * i, 2 * i + 1), 16);
                        byte secondNibble = Byte.parseByte(hexStr.substring(2 * i +
        1,
                                        2 * i + 2), 16);
                        int finalByte = (secondNibble) | (firstNibble << 4);
                        bArray[i] = (byte) finalByte;
                }
                return bArray;
        }
}
```

(vi) CLASS: TSS-SERVICE

```
    package pustpf.ima.net.service;
    import iaik.tc.tss.api.tspi.TcTssAbstractFactory;
    import iaik.tc.tss.api.exceptions.common.TcTssException;
    import iaik.tc.tss.api.tspi.TcIContext;
    import iaik.tc.tss.impl.java.tsp.TcTssLocalCallFactory;
    import iaik.tc.utils.logging.Log;
    public class TssService {
        private static TcTssAbstractFactory cachedFactory = null;
        public static TcTssAbstractFactory getTssFactory() {
                if (cachedFactory == null) {
                        try {
                                TcTssAbstractFactory        factory        =        new
TcTssLocalCallFactory();
                                TcIContext context = factory.newContextObject();
                                context.connect();



    return cachedFactory;
        }
}
```

HOME DOMAIN IDENTITY PROVIDER MODIFICATION

1. Home Domain Identity Provider Attestation Collector Package

(i) CLASS: INTERFACE ABSTRACT COLLECTOR

```
    package pustpf.ima.net.identityprovider.attestationcollector;
    import org.w3c.dom.Document;
    public interface AbstractCollector {
        public Document process(byte[] nonce);
```

```
}
```

(ii) CLASS: ATTEST CLIENT PCR

```
    package pustpf.ima.net.identityprovider.attestationcollector;
    import iaik.tc.tss.api.constants.tsp.TcTssConstants;
    import iaik.tc.tss.api.exceptions.common.TcTssException;
    import iaik.tc.tss.api.structs.common.TcBlobData;
    import iaik.tc.tss.api.structs.tpm.TcTpmPubkey;
    import iaik.tc.tss.api.structs.tsp.TcTssValidation;
    import iaik.tc.tss.api.structs.tsp.TcUuidFactory;
    import iaik.tc.tss.api.tspi.TcIContext;
    import iaik.tc.tss.api.tspi.TcIPcrComposite;
    import iaik.tc.tss.api.tspi.TcIPolicy;
    import iaik.tc.tss.api.tspi.TcIRsaKey;
    import iaik.tc.tss.api.tspi.TcITpm;
    import iaik.tc.tss.impl.csp.TcCrypto;
    import iaik.tc.utils.logging.Log;
    import java.io.File;
    import java.io.FileInputStream;
    import java.security.Signature;
    import java.security.interfaces.RSAPublicKey;
    import javax.crypto.Cipher;
    import javax.xml.parsers.DocumentBuilderFactory;
    import org.bouncycastle.jce.provider.BouncyCastleProvider;
    import org.w3c.dom.Document;
    import org.w3c.dom.Node;
    import org.w3c.dom.traversal.NodeIterator;
    import com.sun.org.apache.xpath.internal.CachedXPathAPI;
    import pustpf.ima.net.service.TssService;
    import pustpf.ima.net.service.XmlService;
public class AttestIdPPCR implements AbstractCollector {



    private Node getPCRContainer(Document src) {
                try {
                        String xpath = "//pcrs[1]";
                        CachedXPathAPI path = new CachedXPathAPI();
                        NodeIterator nl = path.selectNodeIterator(src, xpath);
                        Node n;
                        if ((n = nl.nextNode()) != null) {
                                return n;
                        }
                } catch (Exception e) {
                        e.printStackTrace();
                }
                return null;
        }
    public void verify(byte[] signedPcrVal, RSAPublicKey pubKey, byte[] pcrVal) {
        try {
            long valPcrStart = System.currentTimeMillis();
            Signature sig = Signature.getInstance("SHA1withRSA");
            sig.initVerify(pubKey);
            sig.update(pcrVal);
            boolean verifies = sig.verify(signedPcrVal);
            long valPcrEnd = System.currentTimeMillis();
            System.out.println("Pcr Validation Time:"
                    + (valPcrEnd - valPcrStart));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

(iii) CLASS: ATTEST CLIENT SML

```
  package pustpf.ima.net.identityprovider.attestationcollector;
  import iaik.tc.tss.api.structs.common.TcBlobData;
  import iaik.tc.tss.api.tspi.TcIContext;
  import iaik.tc.tss.api.tspi.TcIPcrComposite;
  import iaik.tc.utils.logging.Log;
  import iaik.tc.tss.api.structs.tpm.TcTpmPcrComposite;
  import iaik.tc.tss.impl.java.tsp.TcPcrCompositeInfoLong;
  import java.io.File;
  import java.util.StringTokenizer;
  import javax.xml.parsers.DocumentBuilderFactory;
  import org.w3c.dom.Document;
  import pustpf.ima.net.service.TssService;
  import pustpf.ima.net.service.XmlService;
  public class AttestIdPSML implements AbstractCollector {
    private          String          IMAPATH          =
"/sys/kernel/security/ima/ascii_runtime_measurements";



  IMAPATH));
            } catch (Exception e) {
                e.printStackTrace();
            }
            try {
                String line = inFile.readLine();
                while (line != null) {
                    contents += line;
                    contents += " \n ";
                    line = inFile.readLine();
                }
                inFile.close();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
        XmlService.appendElement(sml,           sml.getDocumentElement(),
"smlcontents", contents);
        System.out.println();
        return sml;
    }
}
```

(iv) CLASS: DAEMON ATTESTATION COLLECTOR

```
  package pustpf.ima.net.identityprovider;
  import iaik.tc.utils.logging.Log;
  import javax.xml.parsers.DocumentBuilderFactory;
  import javax.xml.transform.*;
  import javax.xml.transform.dom.DOMSource;
  import javax.xml.transform.stream.StreamResult;
  import org.w3c.dom.*;
  import org.w3c.dom.traversal.NodeIterator;
  import org.xml.sax.InputSource;
  import com.sun.org.apache.xpath.internal.CachedXPathAPI;
  import pustpf.ima.net.identityprovider.attestationcollector.*;
  import pustpf.ima.net.service.*;
  public class DaemonAttestationCollectorAgent {
    public Document doAttestation(Document request) {



  String nonce = getNonce(request);
            byte nonceArray[] = XmlService.toBinArray(nonce);
            XmlService.appendElement(returnedDoc,
                    getResponseMainNode(returnedDoc),
"Challenge", nonce);
            Document pcrs = (new AttestIdPPCR()).process(nonceArray);
            XmlService.importName(pcrs, returnedDoc ,
                    getResponseMainNode(returnedDoc));
            Document sml = (new AttestIdPSML()).process(nonceArray);
            XmlService.importName(sml,returnedDoc ,
                    getResponseMainNode(returnedDoc));
        } catch (Exception e) {



  private Node getResponseMainNode(Document src) {
        try {
            String xpath = "//RequestedSecurityToken[1]";
            CachedXPathAPI path = new CachedXPathAPI();
            NodeIterator nl = path.selectNodeIterator(src, xpath);
            Node n;
            if ((n = nl.nextNode()) != null) {
                return n;
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        return null;
    }
}
```

2. Home Domain Identity Provider XML and TSS Service Package

(v) CLASS: XML-SERVICE

```
  package pustpf.ima.net.service;
  import org.w3c.dom.Document;
  import org.w3c.dom.Element;
  import org.w3c.dom.Node;
  import java.math.BigInteger;
  public class XmlService {
    static char[] hexChar = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9',
            'A', 'B', 'C', 'D', 'E', 'F' };
    public static void appendElement(Document doc, Node to, String name,
            String val) {
        Element newEl = doc.createElement(name);
        Node newElVal = doc.createTextNode(val);
        newEl.appendChild(newElVal);
        to.appendChild(newEl);




  }
    public static byte[] toBinArray(String hexStr) {
        byte bArray[] = new byte[hexStr.length() / 2];
        for (int i = 0; i < (hexStr.length() / 2); i++) {
            byte firstNibble = Byte.parseByte(hexStr
                    .substring(2 * i, 2 * i + 1), 16);
            byte secondNibble = Byte.parseByte(hexStr.substring(2 * i +
1,
                    2 * i + 2), 16);
            int finalByte = (secondNibble) | (firstNibble << 4);
            bArray[i] = (byte) finalByte;
        }
        return bArray;
    }
}
```

(v) CLASS: TSS-SERVICE

```
    package pustpf.ima.net.service;
    import iaik.tc.tss.api.tspi.TcTssAbstractFactory;
    import iaik.tc.tss.api.exceptions.common.TcTssException;
    import iaik.tc.tss.api.tspi.TcIContext;
    import iaik.tc.tss.impl.java.tsp.TcTssLocalCallFactory;
    import iaik.tc.utils.logging.Log;
    public class TssService {
        private static TcTssAbstractFactory cachedFactory = null;
        public static TcTssAbstractFactory getTssFactory() {
                if (cachedFactory == null) {
                        try {
                                TcTssAbstractFactory factory = new
TcTssLocalCallFactory();
                                TcIContext context = factory.newContextObject();




    Log.err(tse.getMessage());
                        }
                }
                return cachedFactory;
        }
}
```

(v) CLASS: Target (Client or Identity Provider) Attestation Validation

```
     package tbed.pustpf.ima.net.avagent;

    import java.io.*;
    import java.net.*;
    import javax.xml.transform.Transformer;
    import javax.xml.transform.OutputKeys;
    import javax.xml.transform.TransformerFactory;
    import javax.xml.transform.dom.DOMSource;
    import javax.xml.transform.stream.StreamResult;
    import javax.xml.parsers.DocumentBuilderFactory;
    import org.slf4j.Logger;
    import org.slf4j.LoggerFactory;
    import org.w3c.dom.Document;
    import org.w3c.dom.Node;
    import org.xml.sax.InputSource;
    import tbed.pustpf.ima.net. avagent.services.*;
    public class Target {

        private final Logger log = LoggerFactory.getLogger(Target.class);
        /**
         * @param args
         */
        public static void main(String[] args) {
                }
        public int remotePort = 4444;
        public boolean attestTarget(String targetAddr)
                        throws AttestationServiceHookupException {
                boolean AttestationResult = false;
                long timeStart = System.currentTimeMillis();

                Socket kkSocket = null;
                PrintWriter out = null;
                BufferedReader in = null;

                try {
                        log.debug("Establishing connection to HD target machine on
port {}", remotePort);
                        kkSocket = new Socket(targetAddr, remotePort);
                        out = new PrintWriter(kkSocket.getOutputStream(), true);
                        in = new BufferedReader(new InputStreamReader(kkSocket
                                        .getInputStream()));

                } catch (UnknownHostException e) {
                        log.error("Don't know about host.");
                        throw new AttestationServiceHookupException ();
                } catch (IOException e) {
                        log.error("Couldn't get I/O for the connection to: " +
targetAddr);
                        throw new AttestationServiceHookupException ();
    VRPCR pv = new VRPCR();
                        pv.setRSAPubKey(targetAddr);
                        OutcomeMode pcrResult = pv.verify(response);
                        if (pcrResult == OutcomeMode.
NO_PCRS_VALIDATION_ABORTIVE)
                                log.debug("! --------------- No PCRS found in
response.");
                        else if (pcrResult == OutcomeMode.
PCR_DISSIMILAR_VALIDATION_ABORTIVE)
                                log.debug("! --------------- PCRs Validation failed. ");
                        if (pcrResult == OutcomeMode. VALIDATION_POSITIVE) {
                                log.debug("* --------------- PCRs signature is
validated.");
                                AttestationResult = true;
                        }

                        if (AttestationResult) {
                                VRN nv = new VRN();
                                OutcomeMode nonceResult = nv.verify (response,
nonce);
```

```
                if (nonceResult == OutcomeMode. DISSIMILAR_
NONCE) {
                    AttestationResult = false;
                    log.debug("! --------------- Nonce is stale.
Validation failed. ");
                } else if (nonceResult ==
OutcomeMode.NONCE_SIMILARITY_POSITIVE) {
                    AttestationResult = true;
                    log.debug("*   ---------------   Nonce   correction
and freshness is validated successfully.");




  }

    }

    private String generateNonce() {
        char[] hexChar = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9',
                'A', 'B', 'C', 'D', 'E', 'F' };


        String nonce = "";
        for (int i = 0; i < 20 * 2; i++) {

            Double rnd = Math.random() * 15;
            nonce += hexChar[rnd.intValue()];
        }

        return nonce;
    }
}
```

3. Mutual Integrity Provider-DATACONNECTOR

(v) <u>MUTUAL INTEGRITY PROVIDER-DATACONNECTOR</u>

```
package tbed.pustpf.pustpshib;

import java.util.HashMap;
import java.util.Map;

import org.opensaml.ws.transport.http.HTTPInTransport;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import com.sun.org.apache.xalan.internal.xsltc.runtime.Attributes;
```

```
    import edu.internet2.middleware.shibboleth.common.attribute.BaseAttribute;
    import
edu.internet2.middleware.shibboleth.common.attribute.provider.BasicAttribute;
    import
edu.internet2.middleware.shibboleth.common.attribute.resolver.AttributeResolutionExc
eption;
    import
edu.internet2.middleware.shibboleth.common.attribute.resolver.provider.ShibbolethRe
solutionContext;
    import
edu.internet2.middleware.shibboleth.common.attribute.resolver.provider.dataConnecto
r.BaseDataConnector;

    import tbed.pustpf.ima.net.avagent.AttestationServiceHookupException;
    import tbed.pustpf.ima.net.avagent.Target;

    public class MutualIntegrityProviderDataConnector extends BaseDataConnector {

        private String vsUrl;
        private ATTESTATION_TYPE attestationType;
        public static enum ATTESTATION_TYPE {
                IMA
        };
        private     static     final     String     ATTESTATION_ATTRIBUTE     =
"MutualPlatformIntegrity";

        private final Logger log = LoggerFactory
                        .getLogger(MutualIntegrityProviderDataConnector.class);

        public MutualIntegrityProviderDataConnector(String url) {
                this.setVsUrl(url);
        }

        @Override
        public Map<String, BaseAttribute> resolve(
                        ShibbolethResolutionContext resolutionContext)
                        throws AttributeResolutionException {

                log.info("Validation resolution is initiated.");




    Target attTarget = new Target();
                        HTTPInTransport req = (HTTPInTransport) resolutionContext

        .getAttributeRequestContext().getInboundMessageTransport();
                        targetAddr = req.getPeerAddress();;
                        log.info("Calling CS for Mutual attestation of HD IdP at
idp.dod.org and HD Client at {}", targetAddr);

                        log.info("Carrying out HD client's machine platform attestation
at: {}", targetAddr);

                        attResultClient = attTarget.attestTarget(targetAddr);
                } catch (AttestationServiceHookupException e) {
                        log.error("ACDaemon couldn’t be contacted. Assuming bad
integrity.");
                }
                log.debug("Attestation acknowledgement reply about HD client’s
machine integrity from CS: {}", attResultClient);
                if (attResultClient == false) {
                        log.debug("HD client’s machine platform integrity is not
validated, so HD IdP attestation is not carried out");
                        }



    log.info("Sending back the Mutual integrity attributes.");
                return result;

        }

        @Override
        public void validate() throws AttributeResolutionException {
        }
        public void setAttestationType(ATTESTATION_TYPE attestationType) {
                this.attestationType = attestationType;
        }
        public ATTESTATION_TYPE getAttestationType() {
                return attestationType;
        }
        public void setVsUrl(String vsUrl) {
                this.vsUrl = vsUrl;
        }
        public String getVsUrl() {
                return vsUrl;
        }
}
```

APPENDIX B

COMMAND LINE INSTRUCTIONS AND CODE

1) HOME DOMAIN SHIBBOLETH IDENTITY PROVIDER INSTALLATION

(i) Steps for installing JAVA 1.6 on the Ubuntu OS

- $ curl -L -O 'http <://> www <.> download <.> oracle <.> com/otn <-> pub </> java </> jdk </> 6u25 <-> b06 </> jdk <-> 6u25 <-> linux <-> i586.bin'
- $ chmod + x jdk <–> 6u25 <–> linux <–> i586 <.> bin
- $ <./> jdk <–> 6u25 <–> linux <–> i586 <.> bin (This will install the Java)
- $ java <–> version (This will show java edition "1.6.0")

(ii) Steps for downloading, installing and configuring Apache Tomcat

- Download Apache Tomcat & extract it to the home directory:
  - $ curl -L -O 'http <://> www <.> apache <.> tradebit <.> com </> pub </> tomcat </> tomcat <–> 6 </> v6.0.32 </> bin </> apache <–> tomcat <–> 6.0.32 <.> tar <.> gz'
  - $ tar <–> xvzf apache <–> tomcat <–> 6.0.32 <.> tar <.> gz
  - Rename it to an easy name: $ mv apache – tomcat – 6.0.32 tomcat6
- Set the following variables:
  - $ <space> export <space> JAVA <_> HOME <=/> usr/java6 <–> source </> jdk1.6.0_25
  - $ <space> export <space> PATH <=$> JAVA_HOME </> bin <:$> PATH
  - $ <space> export <space> CATALINA_HOME <=/> home </> zubair </> tomcat6</>
  - $ <space> export <space> CATALINA_BASE <=/> home </> zubair </> tomcat6 </>

- Start tomcat:
  - $ </> home </> zubair </> tomcat6 </> bin </> startup <.> sh
  - Check if Tomcat is working properly by typing the following in the browser: http://localhost:8080 (It will show Tomcat home page)

(iii) Download the latest Identity Provider software package

- $ curl –O <SPACE> ‘http <://> www <.> shibboleth <.> net </> downloads </> identity <-> provider/2.3.0/shibboleth-identityprovider-2.3.0-bin.zip’
- Unzip the archive as:
  - $ <SPACE> jar <-xf> shibboleth <-> identityprovider <-> 2.3.0 <-> bin <.> zip
- Move to the unzipped directory and install the IdP as:
  - $ cd shibboleth-identityprovider-2.3.0
- $ sudo -i (since the idp install command wants to be execute as root main user to make the current user as a root user)
- Set the environment variables for this session:
  - # <SPACE> export <SPACE> JAVA_HOME <=/> usr </> java6 <-> source </> jdk1.6.0_25
  - # <SPACE> export <SPACE> PATH <=$> JAVA_HOME </> bin <:$> PATH
  - # <SPACE> /install.sh

(iv) Copy Xerces and Xalan to the TOMCAT_HOME/endorsed folder

- # cp -r endorsed/ /home/zubair/tomcat6
- # export JAVA_ENDORSED_DIRS=/home/zubair/tomcat6/endorsed/

(v) Setting the JVM memory options

- JAVA <_> OPTS <="-> Djava.awt.headless <=> true <-> Xmx512M <-> XX <:> MaxPermSize <=>128M -Dcom.sun.security.enableCRLDP=true"

(vi) Setting SOAP endpoints for the HD IdP

- Downloaded required jar files such as tomcat6.-.dta. -. ssl. - 1.0.0. .. . jar in to TOMCAT_HOME/lib/.
- Curl – o ‘http <://> shibboleth <.> internet2 <.> edu </> downloads </>maven2 </> edu </> internet2 </> middleware </> security </> tomcat6 </> tomcat6 <-> dta <-> ssl </> 1.0.0 </> tomcat6 <-> dta <-> ssl <-> 1.0.0 <.> jar’
- Afterwards, need to add the code given below to TOMCAT_HOME </> conf </> server <.> xml file.

```
<ConnectorSPACEport="8443"
        Protocol="org.apache.coyote.http11.Http11Protocol"
        SSLImplementation=
"edu[.]internet2.middleware.security.tomcat6.DelegateTo
ApplicationJSSEImplementation"
        Scheme="https"
        SSLEnabled="true"
        clientAuth="true"
        keystoreFile="/opt/shibboleth-idp/credentials/idp.jks"
        keystorePass="keystorePassword" />
```

(vii) Setting the Tomcat’s own variables required for initializing the IdP

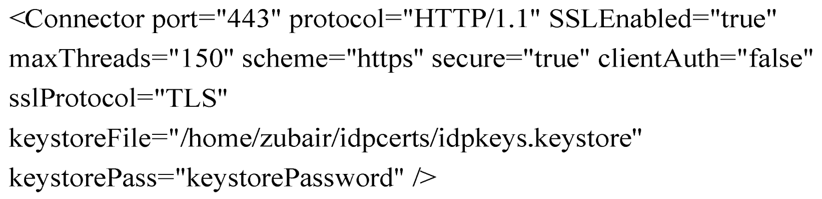
- Create a file by the name idp.xml and place it in TOMCAT_HOME/conf/Catalina/localhost/ folder then copy XML segment of code given below into the file.

```
<Context docBase="/opt/shibboleth-idp/war/idp.war"
          Privileged="true"
          AntiResourceLocking="false"
          antiJARLocking="false"
          unpackWAR="false"
          swallowOutput="true" />
```

(viii) The generation of RSA keystore for the IdP

- sudo mkdir /home/zubair/idpcerts/
- $ sudo keytool -genkey -alias idpkeys -keyalg RSA –keystore /home/zubair/idpcerts/idpkeys.keystore

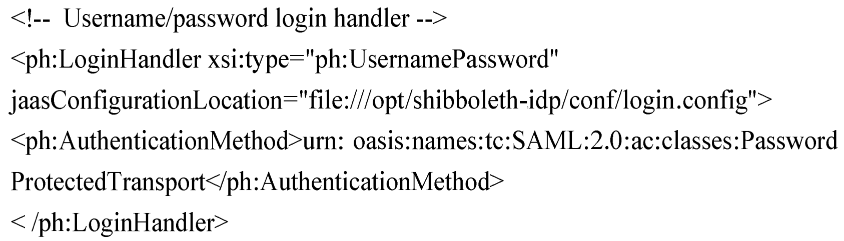
(ix) Adding the XML code to the server.xml in TOMCAT_HOME/conf



```
<Connector port="443" protocol="HTTP/1.1" SSLEnabled="true"
maxThreads="150" scheme="https" secure="true" clientAuth="false"
sslProtocol="TLS"
keystoreFile="/home/zubair/idpcerts/idpkeys.keystore"
keystorePass="keystorePassword" />
```

(x) Configuring LoginHandler in the HD organization IdP

- The following code is added to the IDP_HOME/conf folder



```
<!--  Username/password login handler -->
<ph:LoginHandler xsi:type="ph:UsernamePassword"
jaasConfigurationLocation="file:///opt/shibboleth-idp/conf/login.config">
<ph:AuthenticationMethod>urn: oasis:names:tc:SAML:2.0:ac:classes:Password
ProtectedTransport</ph:AuthenticationMethod>
< /ph:LoginHandler>
```

(xi) ApacheDS installation in the realization of the prototype implementation

- Download and install the ApacheDS as:
  - $ wget http <://> www <.> trieuvan <.> com </> apache <//> directory </>apacheds < /> unstable </> 1.5/1.5.7 </> apacheds <-> 1.5.7-i386.bin
  - $ chmod a <+> x apacheds <-> 1.5.2-i386 <.>bin
  - <./> apacheds <-> 1.5.2-i386 <.> bin (this will install the ApacheDS)
- Now run it by using the command: /etc/init.d/apacheds-1.5.2 start

(xii) The steps to install the Apache Directory Studio

- $ wget http <://> apache.osuosl.org </ > directory </> studio </> stable </> 1.5.3.x.x/
- ApacheDirectoryStudio <-> linux <-> x86 <-> 1.5.3.x.x <.> tar.gz
- $ <SPACE> tar –xvzf <SPACE> ApacheDirectoryStudio <-> linux <-> x86 <-> 1.5.3.x.x <.> tar.gz
- $ <SPACE> mv <SPACE> ApacheDirectoryStudio <-> linux <-> x86 <-> 1.5.3.x.x /opt/ApacheDirectoryStudio
- $ cd /opt/ApacheDirectoryStudio
- $ ./ApacheDirectoryStudio (for initializing the installer)

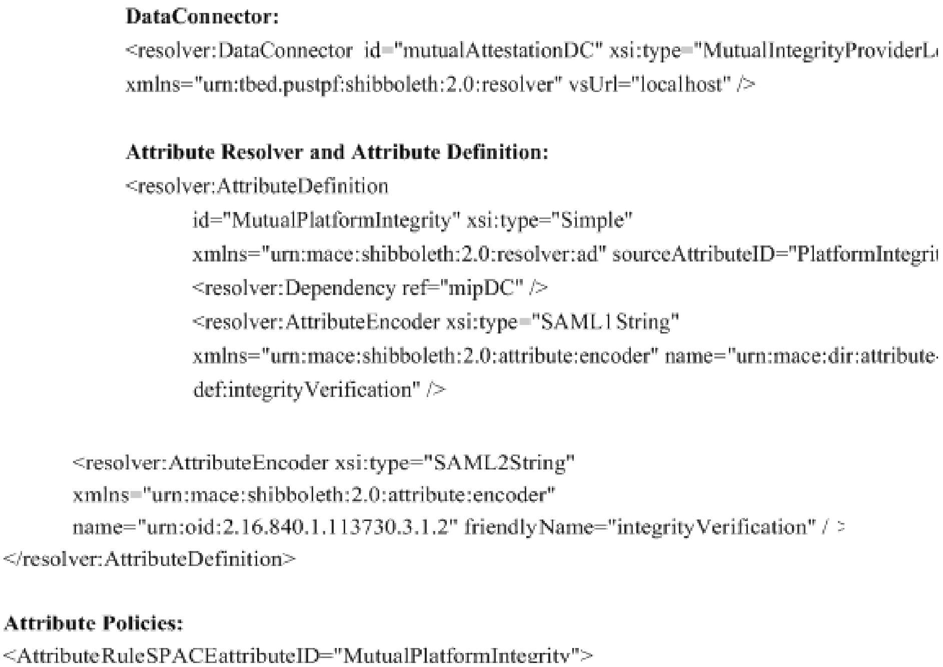
(xiii) Adding below code to the login.config file at the IDP_HOME/conf folder

```
Shib UserPassAuth {
  Edu. vt .middleware.ldap.jaas.LdapLoginModule is required
    ldapUrl= "ldap:// localhost: 10389
    baseDn="ou=users,ou=system"
    tls="true"
    userFilter="uid={0}";
};
```

(xiv) The code given below illustrates the LDAP data connector

```
<!-- LDAP Connector for LDAP data store -->
<resolver:DataConnector xsi:type="dc:LDAPDirectory"
ldapURL="ldap://localhost:10389"
baseDN="ou=users,ou=system" principal = "uid = admin,  ou = system"
principal Credential = "ldapDefaultPassword">
<dc:Filter Template>
<![CDATA[ (uid = $request Context.  principal Name) ]]>
</dc:Filter Template>
</ resolver:  Data Connector >
```

(xv) The dataconnector, attribute resolver, definition, polices



DataConnector:

```
<resolver:DataConnector  id="mutualAttestationDC" xsi:type="MutualIntegrityProviderL
xmlns="urn:tbed.pustpf:shibboleth:2.0:resolver" vsUrl="localhost" />
```

Attribute Resolver and Attribute Definition:

```
<resolver:AttributeDefinition
        id="MutualPlatformIntegrity" xsi:type="Simple"
        xmlns="urn:mace:shibboleth:2.0:resolver:ad" sourceAttributeID="PlatformIntegri
        <resolver:Dependency ref="mipDC" />
        <resolver:AttributeEncoder xsi:type="SAML1String"
        xmlns="urn:mace:shibboleth:2.0:attribute:encoder" name="urn:mace:dir:attribute-
        def:integrityVerification" />


    <resolver:AttributeEncoder xsi:type="SAML2String"
    xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
    name="urn:oid:2.16.840.1.113730.3.1.2" friendlyName="integrityVerification" / >
</resolver:AttributeDefinition>
```

Attribute Policies:

```
<AttributeRuleSPACEattributeID="MutualPlatformIntegrity">
```

2) FOREIGN DOMAIN SHIBBOLETH SERVICE PROVIDER INSTALLATION

(i) Installation of C and C++ compilers using the CentOS installer feature.

- # yum install gcc
- # yum install gcc-c++

(ii) log4shib installation

- # Extract the downloaded package & move to the extracted directory:
  - $ tar -xvzf log4shib-1. 0. 4. Tar <.> gz
  - $ <SPACE> cd log4shib-1. 0. 4
- Configure log4shibd with the following options:
  - $ <SPACE> <./> configure <--> disable-static <-->disable <-> doxygen <--> prefix=/opt
  - $ <SPACE> make
  - $$ <SPACE> make$ <SPACE> install

(iii) Xerces-C Installation and Configuration

- Extract the downloaded package & move to the extracted directory:
  - $ tar -xvzf xcercez.tar.gz
  - $ cd xcers-XX
  - $ ./configure --prefix=/opt/shibboleth-sp --disable-netaccessor-libcurl
  - $ make
  - $ make install

(iv) Installation of OpenSSL and XML-Security-C

- # yum install openssl
- # yum install openssl-devel

Now configure and install XML-Security-C as:

- $./configure --without-xalan --disable-static --with-xerces=/opt/shibboleth-sp --prefix=/opt/shibboleth-sp
- $ make
- $ make install

(v) Installation of XML-Tooling-C package

- # yum install curl-devel
- $ ./configure --with-log4shib=/opt/shibboleth-sp --prefix=/opt/shibboleth-sp -C
- $ make
- $ make install

(vi) The OpenSAML-C package installation

- $ ./configure --with-log4shib=/opt/shibboleth-sp --prefix=/opt/shibboleth-sp -C
- $ make
- $ make install

(vii) Shibboleth SP downloading and installation

- Additional Requirement:
    - Apache 2 (httpd) web server needs to be installed first.
    - # yum install httpd
    - # yum install httpd-devel
- $./configure--with-log4shib=/opt/shibboleth-sp--enable-apache-13--with-apxs=/usr/bin/apxs--enable-apache-20--with-apxs2=/usr/bin/apxs--with-apr=/usr/bin-apr-1-config--with-apu=/usr/bin/apu-1-config--prefix=/opt/shibboleth-sp
- $ make

$ make install

(viii) Apache web server (or httpd) modification

- The configuration file in Apache is located at /etc/httpd/conf/ folder and is known as httpd.conf. The following modifications needs to be performed:
    - The ServerName directive has been changed to reflect own created SP server as sp.dor.org.
    - During the execution of the SP software certain runtime errors related to resource mapping appears. In order to alleviate these errors at

runtime a directive naming UseCanonicalName must be set to On. This directive could be found in the same configuration file listed above.

- Restart the Apache by issuing the command given below:
  - #<SPACE>/etc< / >init< . >d< / >httpd restart <PRESS ENTER>
- The Shibboleth SP daemon (shibd) running on the SP must have to be started independently. This daemon is responsible for handling requests coming to the FD organization SP. Restart the daemon by issuing the following command:
  - # <SPACE> /etc </> init <.>d </> shibd start <PRESS ENTER>

(ix) Restart the daemon by issuing the command

- # <SPACE> / etc </> init <.> d </> shibd start <PRESS ENTER>

(x) Openssl installation

- # yum install mod_ssl openssl

(xi) Own RSA key generation to create the certificate for enabling the SSL

- An RSA key pair by the name spserver.key for the SP server side is created as:
  - # openssl genrsa -out spserver.key 1024
- Using the above key pair a certificate have been created for the FD organization SP by the name spserver.crt as:
  - # openssl req -new -x509 -days 3650 -key spserver.key -out spserver.crt
- Both the key pair and the certificate need to be placed on the FD organization SP machine platform from where it can be accessible by the Apache web server for the establishment of secure connection. Therefore, these keys have been placed in the following location at the FD SP machine:
  - # cp spserver.crt /etc/pki/tls/certs/
  - # cp spserver.key /etc/pki/tls/private/
- The Apache web server also needs to be informed about the location of the keys and certificate that was created and placed in the previous step. This

is done by editing the ssl.conf file residing at the FD organization SP at location /etc/httpd/conf.d/ folder. Modify the value for the SSLCertificateFile option in the conf file to reflect the location of the certificate file. Similarly, the SSLCertificateKeyFile value is modified to reflect the location of the key file.

(xii) Metadata files configuration for idp and sp

relying-party.xml file:

```
<!-- Location for the SP's Metadata.  -->
<metadata:MetadataProvider xsi:type="FilesystemMetadataProvider"
Xmlns="urn:mace:shibboleth:2.0:metadata" id="DOR-SP-METADATA"
metadataFile="/opt/shibboleth-idp/metadata/sp-metadata.xml" />
```

shibboleth2.xml file:

```
<! -- IdP’s metadata maintained locally by the SP -->
<MetadataProvider file="idp-metadata.xml"/>
```

(xiii) Protected and Unprotected Resource Access Policy Definition

```
<Location /secure>
 AuthType shibboleth
 ShibRequestSetting requireSession 1
 ShibExportAssertion On
 require MutualPlatformIntegrity true
</Location>
```

(a) Protected resource access policy code

```
<Location /unsecure>
 AuthType shibboleth
 ShibRequireSession Off
 require shibboleth
</Location>
```

(b) Unprotected resource access policy code

(xiv) WSSO MetaData Configuration at HD IdP

```
<IDPSSODescriptor protocolSupportEnumeration="urn:mace:shibboleth:1.0  ------------------------------ (1)
purn:oasis:names:tc:SAML:1.1:rotocol urn:oasis:names:tc:SAML:2.0:protocol">

<ArtifactResolutionService Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"  ----------(2)
Location="https://idp.example.org:8443/idp/profile/SAML1/SOAP/ArtifactResolution"
index="1"/>
<ArtifactResolutionService Binding="urn:oasis:names:tc:SAML:2.0:bindings:SOAP"
Location="https://idp.example.org:8443/idp/profile/SAML2/SOAP/ArtifactResolution"
index="2"/>

<SingleSignOnService Binding="urn:mace:shibboleth:1.0:profiles:AuthnRequest"  -----------------------(3)
                Location="https://idp.example.org/idp/profile/Shibboleth/SSO" />
<SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
                 Location="https://idp.example.org/idp/profile/SAML2/POST/SSO" />
<SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST-SimpleSign"
           Location="https://idp.example.org/idp/profile/SAML2/POST-SimpleSign/SSO" />
 <SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect"
```

(xv) WSSO MetaData Configuration at FD SP

```
<SPSSODescriptor
        protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol
urn:oasis:names:tc:SAML:1.1:protocol" xmlns="urn:oasis:names:tc:SAML:2.0:metadata">
        <Extensions xmlns="urn:oasis:names:tc:SAML:2.0:metadata">
           <!-- Extension to permit the SP to receive IdP discovery responses. -->
           <idpdisc:DiscoveryResponse
              Binding="urn:oasis:names:tc:SAML:profiles:SSO:idp-discovery-protocol"
              Location="https://sp.dod.org/Shibboleth.sso/Login"
              index="1" xmlns:idpdisc="urn:oasis:names:tc:SAML:profiles:SSO:idp-discovery-protocol"/>
           <idpdisc:DiscoveryResponse
              Binding="urn:oasis:names:tc:SAML:profiles:SSO:idp-discovery-protocol"
              Location="https://sp.dod.org/Shibboleth.sso/Login"
              index="2" xmlns:idpdisc="urn:oasis:names:tc:SAML:profiles:SSO:idp-discovery-protocol"/>
        </Extensions>
<AssertionConsumerService
           Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
           Location="https://sp.dod.org/Shibboleth.sso/SAML2/POST"
           index="1" isDefault="true" xmlns="urn:oasis:names:tc:SAML:2.0:metadata"/>
```

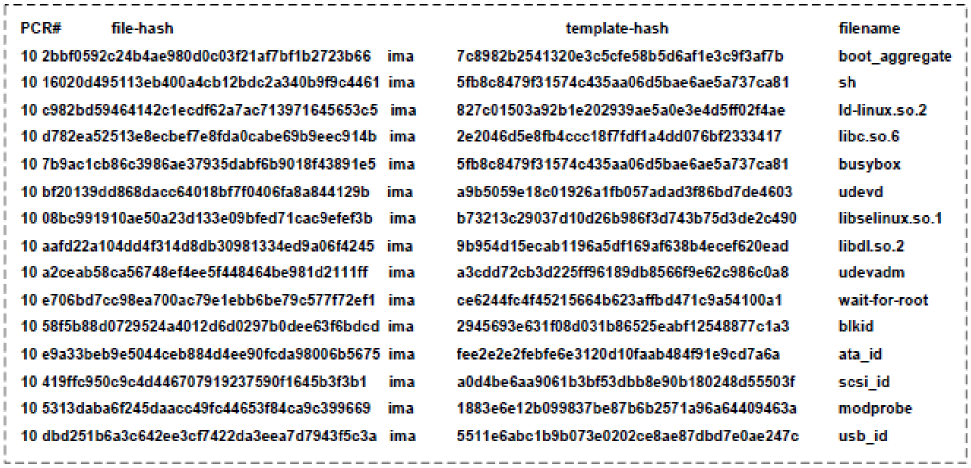
3) STEPS FOR IMA INSTALLATION AND CONFIGURATION

(i) Linux Kernel Complilation for Incorporating IMA for HD IdP and Client

- A new kernel have downloaded for compilation from the kernel download page:
    - $ curl –c<SPACE>http< :// >www< . >kernel< . >org< / >pub< / >linux< / >kernel< / >v2.6< / >longterm< / >v2.6.35< / >linux< ->2.6.35.x.tar.bz2
- Then extracted the kernel to the location /usr/src/kernel:
    - $ sudo<space>tar< –>C< / >usr< / >src/kernel –jvxf linux< ->2.6.35.x.tar.bz2
- To enter to the extracted directory:
    - $ cd /usr/src/kernel/linux-2.6.35.x
- In order to configure the drivers and modules for the new kernel, a configuration file is required that can be used as a starting point for the configurations. for this purpose an existing configuration file is copied by copying it from the /boot/ folder to the newly extracted kernel folder by using the command:
    - $ cp /boot/config-2.6.x-x-generic /usr/src/kernel/linux-2.6.35.x

- Rename the configuration file to .config file as:
  - $ mv config-2.6.x-x-generic .config
- Before starting the kernel configuration some prerequisites are required to be install for the configuration process.
- $ sudo apt-get install libncurses5-dev
- Then, the configuration of the new kernel is carried out in order to include all the device drivers and kernel modules by the following command. After executing the command a configuration process will start that will prompt for each configuration options that need to be included in the kernel for compilation.
- $ sudo make oldconfig
  - When the selection portion completes all the changes are saved to the .config file and edit this file to incorporate the TPM device drivers and IMA option as:
- $ sudo make menuconfig
- The above command will open a GUI of kernel configuration options the TPM device drivers are selected and the IMA to be included in the new kernel.
- The TPM device drivers option can be selected from: Device Drivers -- > Character Devices -- > TPM. Select the * for the TPM device drivers to include all the drivers.
- The IMA option can be selected from: Security Tab -- > Integrity Measurement Architecture. Select it and save the .config file.
- The new kernel is now ready for compilation, the compilation is done as:
  - $ sudo make
- It will take some time to compile the kernel, afterwards the modules are compiled for the kernel as:
  - $ sudo make modules
- After the modules are compiled the modules are installed first as:
  - $ sudo make modules_install
- Finally, the new kernel is installed as:
  - $ sudo make install

- The new kernel is now installed but to use this kernel instead of the older kernel the following steps are carried-out:
    - $ cd /boot
    - $ sudo mkinitramfs -k -o initrd.img-2.6.35< . >x 2.6.35< . >x
- In order to update the grub entries for the new kernel the following command is used:
    - $ sudo update-grub2
- Now, restart the system (client or IdP platform) and the boot options will show the newly compiled kernel with TPM and IMA enabled. These steps are performed before installing and configuring the Shibboleth IdP but for the sake of clarity the steps are describe here in the IMA implementation section.
- To check whether IMA is working properly or not, upon restart execute the following command in a terminal to view the measurement list calculated by the IMA for each executable after loading the executables:
- $ sudo cat /sys/kernel/security/ima/ascii_runtime_measurements
- The command will result in the Stored Measurement Log (SML) (example of stored measurement log is given below).



```
PCR#     file-hash                                             template-hash                          filename
10 2bbf0592c24b4ae980d0c03f21af7bf1b2723b66  ima   7c8982b2541320e3c5cfe58b5d6af1e3c9f3af7b   boot_aggregate
10 16020d495113eb400a4cb12bdc2a340b9f9c4461  ima   5fb8c8479f31574c435aa06d5bae6ae5a737ca81   sh
10 c982bd59464142c1ecdf62a7ac713971645653c5  ima   827c01503a92b1e202939ae5a0e3e4d5ff02f4ae   ld-linux.so.2
10 d782ea52513e8ecbef7e8fda0cabe69b9eec914b  ima   2e2046d5e8fb4ccc18f7fdf1a4dd076bf2333417   libc.so.6
10 7b9ac1cb86c3986ae37935dabf6b9018f43891e5  ima   5fb8c8479f31574c435aa06d5bae6ae5a737ca81   busybox
10 bf20139dd868dacc64018bf7f0406fa8a844129b  ima   a9b5059e18c01926a1fb057adad3f86bd7de4603   udevd
10 08bc991910ae50a23d133e09bfed71cac9efef3b  ima   b73213c29037d10d26b986f3d743b75d3de2c490   libselinux.so.1
10 aafd22a104dd4f314d8db30981334ed9a06f4245  ima   9b954d15ecab1196a5df169af638b4ecef620ead   libdl.so.2
10 a2ceab58ca56748ef4ee5f448464be981d2111ff  ima   a3cdd72cb3d225ff96189db8566f9e62c986c0a8   udevadm
10 e706bd7cc98ea700ac79e1ebb6be79c577f72ef1  ima   ce6244fc4f45215664b623affbd471c9a54100a1   wait-for-root
10 58f5b88d0729524a4012d6d0297b0dee63f6bdcd  ima   2945693e631f08d031b86525eabf12548877c1a3   blkid
10 e9a33beb9e5044ceb884d4ee90fcda98006b5675  ima   fee2e2e2febfe6e3120d10faab484f91e9cd7a6a   ata_id
10 419ffc950c9c4d446707919237590f1645b3f3b1  ima   a0d4be6aa9061b3bf53dbb8e90b180248d55503f   scsi_id
10 5313daba6f245daacc49fc44653f84ca9c399669  ima   1883e6e12b099837be87b6b2571a96a64409463a   modprobe
10 dbd251b6a3c642ee3cf7422da3eea7d7943f5c3a  ima   5511e6abc1b9b073e0202ce8ae87dbd7e0ae247c   usb_id
```

(ii) Implementation of the jTSS

- jTSS requires Java to be installed as prerequisite. The Java installation steps are covered in the section of HD organization IdP installation.
- To place the jTSS, its libraries and other tools required for communication with TPM a directory is created for this purpose by the name tpm-tools as:

  - $ mkdir /zubair/zubair-project/tpm-tools
- The jTSS software and its related packages are downloaded as:
  - $ cd /zubair/zubair-project/tpm-tools
- $<SPACE> wget<SPACE>–C http:// nchc .dl .sourceforge .net/ sourceforge/trustedjava/jTSS_0.5.tar. bz2
- Extract the downloaded package as:
  - $ tar –xvjf jTSS_0.5.tar.bz2
- Enter into the extracted folder as:
  - $ cd jTSS_0.5

(iii) jTPMTools Downloading and Extraction

- $ wget http ://nchc. dl.sourceforge .net/ sourceforge/ trustedjava/jTpmTools_0.5.tar. bz2
- $ tar jxf jTpmTools_0.5.tar.bz2
- $ cd jtpmtools

(iv) Command to take TPM Ownersjip

  - $ cd /zubair/zubair-project/tpm-tools/jTpmTools
  - $ ./jtt.sh take_owner –o ownersecret (the ownersecret could be any password/secret word)

(v) Home Domain IdP and Client Machine AIK Creation

  - ./jtt.sh aik_create –a aiksecret –l aiklabel –o ownersecret (aiksecret could be any secret word, it is used during IMA for extracting trust tokens from the TPM, aiklabel is used for identifying different aiks, and the ownersecret is the TPM owner secret that is provided in the previous step).

APPENDIX C

DEFINATION OF TERMINOLOGIES

Attacker: Attacker is an entity which may install an unauthorized program on a client or server machine without the knowledge of a machine user or administrator. The objective of an attacker maybe user credentials theft, wealth and personal or industrial sabotage.

Basic User Authentication Credential: It is referred to the basic user AuthN credential such as username/password issued by the HD organization.

Federation: When pool of organizations or service providers that come together to form “Circle of Trust (CoT)” for the purpose of distributing user’s identities information wit each other.

Federated Identity: Federated identity signifies subject’s e-ID and attributes linkage which stocked across various separated identity administration systems.

Foreign Domain Organization: It is an external network entity which may provide resources or services to HD organization users.

Home Domain Client Machine: It is a user desktop/laptop machine assigned by the HD organization to the new user or staff.

Home Domain User: The HD user is a registered user of HD organization.

Home Domain Organization: It is a local or internal network which consists of entities such as a user, client and IdP. In this work the HD IdP also registers the HD client and its own machine platforms measurement in a good hashes repository. The IdP is the one who responsible to release a user AuthN and attributes assertion in response of the AuthN and user attributes request received it from the FD organization SP.

Integrated Authentication: Integrated AuthN is different than two factors AuthN which normally combines one or more user AuthN techniques (e.g., username/password, biometric and smartcard etc.) to authenticate the user. But integrated AuthN in this thesis means integration of user AuthN mechanism (e.g., username/password) with the client and server machines platform mutual attestation technique.

Machine Platforms Security Credentials: It refers to the IMA configured and TPM enabled and activated machine platform boot-time time measurement (i.e., binary values). One copy of the boot-time measurement list is recorded in the SML and second list of these measurements are aggregated in TPM to protect the integrity of the boot-time measurement list. These measurements are type of security credentials and reports to the challenger as an integrity response for integrity validation purpose.

Malevolent: A malicious activity such as Trojan, Rootkit and key-Loger etc. which is installed and executed on a client or server machine for the purpose of stealing user credentials or IDtheft.

Practicable: It means a notion, project, or scheme that may competent of being completed by means of available tools and in particular conditions as they are feasible, executable, viable and workable.

Resource or Service: Resource or Service such as utilizing library portal, air-line reservation system, or secret document which may provide by the FD organization to the HD users. The resource or service may be protected or unprotected.

Threat: Threat is a risk to a machine platform by installing a malevolent program on computer machine to get control (or unauthorized access) of an infected machine.

Trustworthy: Trustworthy in this work refers to a machine platform that may operate in secured, reliable and privacy protected manner while performing an online transaction. Secured means no malevolent or unauthorized program installed or active whereas reliable means a dependency of a security device (e.g., TPM) which must always work accordingly.

Trusted Token: Trusted token in this thesis refers to a trusted machine platform security credentials measurement list image (i.e., target machine) which is trusted by the challenger machine and represents the trustworthiness or honesty of a target machine.

Web Single Sign-On (WSSO): Using SSO facility user can access multiple resources or services in web environment through a single credential and one-time log-in operation. Whenever user tries to access a WSSO-enabled Web resource or service, the WSSO forwards the user's agent to an authentication service provider to let the user log-in. The user agent then sends rear to the resource or service provider and the user may access the requested resource or a service.