

TCG Trusted Network Connect TNC Architecture for Interoperability

**Specification Version 1.0
Revision 4
3 May 2005
Published**

Contact: admin@trustedcomputinggroup.org

TCG PUBLISHED

Copyright © TCG 2004-2005

TCG

Copyright © 2005 Trusted Computing Group, Incorporated.

Disclaimer

THIS SPECIFICATION IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE. Without limitation, TCG disclaims all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification and to the implementation of this specification, and TCG disclaims all liability for cost of procurement of substitute goods or services, lost profits, loss of use, loss of data or any incidental, consequential, direct, indirect, or special damages, whether under contract, tort, warranty or otherwise, arising in any way out of use or reliance upon this specification or any information herein.

No license, express or implied, by estoppel or otherwise, to any TCG or TCG member intellectual property rights is granted herein.

Except that a license is hereby granted by TCG to copy and reproduce this specification for internal use only.

Contact the Trusted Computing Group at www.trustedcomputinggroup.org for information on specification licensing through membership agreements.

Any marks and brands contained herein are the property of their respective owners.

Acknowledgement

The TCG wishes to thank all those who contributed to this specification. This document builds on numerous work done in the various working groups in the TCG.

Special thanks to the members of the TNC contributing to this document:

Mark Beadles	Endforce, Inc.
Kazuaki Nimura	Fujitsu Limited
Steve Hanna	Funk Software, Inc.
Paul Crandell	Hewlett-Packard
Boris Balacheff	Hewlett-Packard
Diana Arroyo	IBM
Tina Bird	InfoExpress, Inc.
Ned Smith	Intel Corporation
Ravi Sahita	Intel Corporation
Barbara Nelson	iPass
John Vollbrecht	Meetinghouse Data Communications
Paul Sangster	Sun Microsystems, Inc.
Babak Salimi (TNC co-chair)	Sygate Technologies, Inc.
Bryan Kingsford (TNC co-chair)	Symantec
Thomas Hardjono (Editor)	VeriSign, Inc.

Revision History

	Document started by Thomas Hardjono	5/21/2004
Rev 1.0_r1	Version 1.0_r1 submitted for 60-day internal TCG review.	2/11/2005
Rev 1.0_r2	Multiple corrections from various reviewers.	4/14/2005
Rev 1.0_r3	Final corrections.	4/18/2005

Table of Contents

1	Scope and Audience	8
2	Introduction	9
2.1	Endpoint Integrity: Background	9
2.2	Aim and Purposes	10
3	The TNC Architecture	11
3.1	Relationship with the IWG Architecture	11
3.2	Relationship with the AAA Architecture in the IETF	12
3.3	TNC Architecture	12
3.4	Layers	13
3.5	Entities	13
3.6	Components	14
3.6.1	Access Requestor	14
3.6.2	Policy Enforcement Point	14
3.6.3	Policy Decision Point	14
3.7	TNC Interfaces	15
3.7.1	Integrity Measurement Collector Interface (IF-IMC)	15
3.7.2	Integrity Measurement Verifier Interface (IF-IMV)	15
3.7.3	TNC Client-Server Interface (IF-TNCCS)	15
3.7.4	Vendor-Specific IMC-IMV Messages (IF-M)	16
3.7.5	Network Authorization Transport Protocol (IF-T)	16
3.7.6	Platform Trust Services Interface (IF-PTS)	16
3.7.7	Policy Enforcement Point Interface (IF-PEP)	16
3.8	Goals and Assumptions	16
3.9	Basic Message Flows Across Interfaces	17
4	Design Aspects of the TNC Architecture	20
4.1	Aspects of TNC Client and TNC Server Interaction	20
4.2	Aspects of TNCC-IMC Interaction and TNCS-IMV Interaction	21
5	Assessment, Isolation and Remediation	23
5.1	Phases in Network Access Control	23
5.2	Assessment Phase	24
5.3	Isolation Phase	24
5.4	Remediation Phase	24
5.5	Remediation in the TNC Architecture	25
6	TNC Architecture with the Trusted Platform Module	26
6.1	Features of a Platform with a TPM	26
6.2	Entities	27
6.3	Components	28
6.3.1	Platform Trust Services	28
6.4	Interface IF-PTS	29
7	Technologies Supporting the TNC Architecture	31
7.1	Network access technologies	31
7.1.1	802.1X	31
7.1.2	VPNs	31
7.1.3	PPP	32
7.2	Message transport technologies	32
7.2.1	Protected EAP Methods	32
7.2.2	TLS and HTTPS	32
7.3	Authentication Server (AS) technologies	32
7.3.1	Radius	33
7.3.2	Diameter	33
8	Security Considerations	34
9	Privacy Considerations	35
10	References	36

11 TNC Glossary 37

1 Scope and Audience

The Trusted Network Connect Sub Group (TNC-SG) is working to define and promote an open solution architecture that enables network operators to enforce policies regarding the security state of endpoints in order to determine whether to grant access to a requested network infrastructure. Endpoint integrity policies may involve integrity parameters spanning a range of system components (hardware, firmware, software and application settings), and may or may not include evidence of a Trusted Platform Module (TPM). This security assessment of each endpoint is performed using a set of asserted integrity measurements covering aspects of the operational environment of the endpoint.

Architects, designers, developers and technologists who are interested in the development, deployment and interoperation of trusted systems may find this document helpful in understanding the architecture defined by the TNC-SG.

2 Introduction

The TNC architecture focuses on interoperability of network access control solutions and on the use of trusted computing as the basis for enhancing security of those solutions. Integrity measurements are used as evidence of the security posture of the endpoint so access control solutions can evaluate the endpoint's suitability for being given access to the network.

The purpose of the current document is to define the *Trusted Network Connect* (TNC) architecture for interoperable network access control and authorization. The TNC architecture will leverage and integrate with existing network access control mechanisms such as 802.1X [18] or others. The TNC specifications will also define interoperability interfaces to allow for the exchange of new types of attributes in the context of network access control solutions. Those attributes will include endpoint compliance information, software state attestation, as well as *Platform-Authentication* information. The software state attestation and platform authentication mechanisms will be based on the principles and features of Trusted Computing, as defined by the Trusted Computing Group (TCG) [1].

Note that in the remainder of this document, the term “platform-authentication” carries the specific TCG meaning of performing verification of the integrity status of a platform using the features of *Trusted Platforms* [1]. These features represent the core functionality of trusted computing as defined and specified by the TCG.¹ The term “platform-authentication” as used in the context of TNC pertains to two related aspects of authentication. The first aspect is the *proof of identity* of the platform, while the second aspect is the *integrity-status* of the platform. In the specific context of the TCG, proving the identity of a platform is performed using any non-migratable key (e.g., an AIK). Since there are an unlimited number of non-migratable keys associated with the TPM there are an unlimited number of identities that can be authenticated. Note that claimed identity in a platform may or may not be related to the user or any actions performed by the user.

In the remainder of this document, the term “Platform Authentication” (or “Platform Integrity”) therefore should generally be understood as consisting of both aspects, namely establishing proof of identity (e.g. via AIKs) and platform integrity verification.

2.1 Endpoint Integrity: Background

The growth of the Internet IP infrastructure in the last few years has introduced new technologies and new security challenges. One of these security challenges concerns the increasing need for machine-to-machine identification and authentication, and network access authorization at the IP layer, in addition to the usual user authentication. Machine level platform-authentication are crucial for the security and authorization of network-access requests at both layer-2 and layer-3. Furthermore, due to the increase attacks at the higher layers (e.g. virus and Trojans), a deeper problem that needs to be addressed is that of achieving *endpoint integrity*.

The problem of endpoint integrity concerns the *trustworthiness* of two communicating endpoints (e.g. Client and Server) from the perspective of the integrity conditions of the two endpoints, including their identities. By the term *integrity* we mean the relative purity of the endpoints from software (and hardware) that are considered harmful to the endpoint itself and other with whom it interacts. This problem of harmful software is best exemplified by the growing number of virus and Trojan attacks on corporate networks. Many employees today connect their mobile devices (e.g. laptops, PDAs) at home to the open Internet, often resulting in malware being inadvertently downloaded onto the device. When connected to the corporate network, the device becomes a distributor of the malware to other devices on the Enterprise network.

¹ Since the term *Platform Authentication* carries a distinct TCG meaning, the two words are hyphenated (“platform-authentication”) in the current document to differentiate it from the more general meaning of authentication/authorization of a general computing platform.

The goal of Trusted Computing as defined by the Trusted Computing Group (TCG) is to improve trustworthy behavior of platforms and to permit trustworthy verification. Verifiers have the ability to decide when it is safe to extend the enterprise boundary to a connecting platform based on the *integrity information* reported by the platform and by the *proof-of-identity* supplied by the platform. Through trusted network connection protocols and trusted platform mechanisms, entities seeking connectivity can be platform-authenticated and authorized (against some network policy) before being given full network connectivity. More specifically, in the context of endpoint authentication and authorization the aim is to ascertain the security state of a given platform or device. A strong hardware-protected root-of-trust is needed to ensure malware and improperly configured software cannot report an erroneous status.

One important goal of the TNC approach is to use the TCG platform-authentication approach as a critical part of achieving true trusted network connections. The model adopted is a 3-party model in which an Access Requester requests network access to a Policy Decision Point which in-turn provides its validation outcome (access granted/denied) to a Policy Enforcement Point (e.g. switch, 802.11 AP). The term “policy” in the current document refers to network-access control policies or rules, which in the case of the TNC should include rules concerning both the integrity aspects of the platform as well as the identity aspects of the platform.

2.2 Aim and Purposes

The aim of the TNC architecture is to provide a framework within which consistent and useful specifications can be developed to achieve a multi-vendor network standard that provides the following features:

- *Platform Authentication*: the verification of a network access requestor’s proof of identity of their platform and the integrity-status of that platform.
- *Endpoint Policy Compliance (Authorization)*: establishing a level of ‘trust’ in the state of an endpoint, such as ensuring the presence, status, and upgrade level of mandated applications, revisions of signature libraries for anti-virus and intrusion detection and prevention system applications, and the patch level of the endpoint’s operating system and applications. Note that policy compliance can also be viewed as *authorization*, in which an endpoint compliance to a given policy set result in the endpoint being authorized to gain access to the network.
- *Access Policy*: ensuring that the endpoint machine and/or its user authenticates and establishes their level of trust before connecting to the network, leveraging a number of existing and emerging standards, products, or techniques.
- *Assessment, Isolation and Remediation*: ensuring that endpoint machines not meeting the security policy requirements for ‘trust’ can be isolated or quarantined from the rest of the network, and if possible an appropriate remediation applied, such as upgrading software or virus signature libraries to enable the endpoint to comply with security policy and become eligible for connection to the rest of the network.

The TNC architecture and specifications will ensure that hosts are interoperable. That is, hosts that implement the protocols, software and/or hardware will be able to connect to a network while ensuring a minimum level of compliance to organizational policies controlling network access. Policies may apply to the operational state of the platform and may include services such as anti-virus checkers, personal firewalls, intrusion detection systems operating system configuration and application patch levels. The goal of enforcing these policies is to prevent compromise of the host, the network, or other network resources.

3 The TNC Architecture

The entities in the TNC architecture are the Access Requestor (AR), the Policy Decision Point (PDP), and the Policy Enforcement Point (PEP). The AR requests access to a protected network. The PDP compares the AR's credentials to certain network access policies and thereby decides whether network access should be granted. The PDP then communicates its decision to the PEP, which actually grants or denies access.

Although this concept is common today in many networks, the TCG seeks to add *integrity measurement/reporting* and *platform-authentication* to the credentials evaluated by the PDP.

In this section we describe how the TNC Architecture relates to the broader IWG Architecture [2] and the IETF AAA Architecture [13][17]. This is followed by a detailed discussion of the TNC Architecture, the entities in the architecture, the components within those entities, and the interfaces to be defined by the TNC.

3.1 Relationship with the IWG Architecture

The TNC Architecture is derived from the broader IWG Architecture. Therefore, the platform-authentication model underlying the IWG Architecture also underlies the TNC Architecture. This is shown in Figure 1, with mappings to the TNC Architecture.

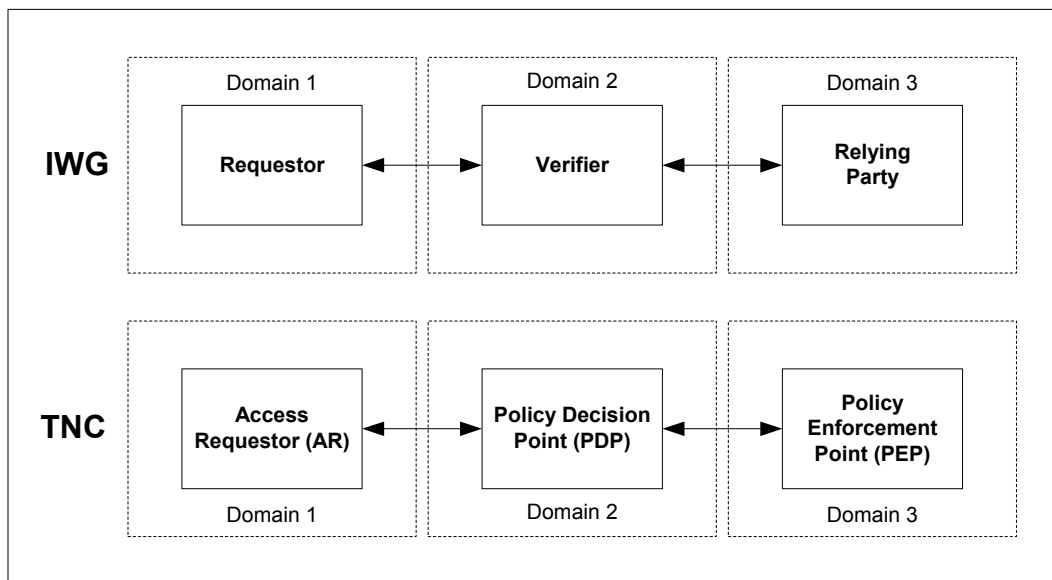


Figure 1: Basic Model underlying the IWG and TNC Architectures

In the IWG architecture when responding to a request from a *Requestor* entity, a *Relying Party* is dependent on the decision outcome of a *Verifier*. This basic behavior maps quite readily to the basic network connection request behavior, in which a network capable device (e.g. a client or 802.1X Supplicant) seeks network connectivity and access to resources available on the network, through another device (e.g. 802.1X Authenticator, switch) relying on the permissions decision of a third device (e.g. AAA Server) [18].

In the TNC architecture, the AR acts as an IWG Requestor, the TNC PDP acts as an IWG Verifier, and the TNC PEP acts as an IWG Relying Party.

In the context of the TNC architecture, the Requestor is more specifically referred to as the *Access Requestor (AR)*, which is the entity that is seeking network connectivity from a given network. The Verifier is referred to as the *Policy Decision Point (PDP)*, since the entity makes the

actual decision (access granted fully, access granted partially, or access denied) with respect to the corresponding request. The PDP performs this decision-making based on a set of policies (customize for the network environment) and based on input from various sources of integrity information. The entity that actually carries out the PDP's decision (e.g. open/close port in 802.1X) is referred to as the *Policy Enforcement Point* (PEP) [17].

Though not visible in Figure 1, another important aspect shared between the two architectures is the use of the trusted computing feature of *integrity measurement* and *integrity verification* to establish a decision regarding a network access request. It is this feature that distinguishes the IWG and TNC architectures from other architectures.

3.2 Relationship with the AAA Architecture in the IETF

The TNC Architecture seeks to enhance AAA-related architectures and protocols developed in the IETF with increased security functions that are provided by Trusted Platforms. As such, the TNC Architecture does not exist in a vacuum, but rather relies on other established technologies that have been standardized in the IETF in the area of AAA. The broad aim of the TNC efforts is the same as and builds upon those of the AAA-related efforts in the IETF, namely to provide network access to endpoints that have been successfully authenticated and meet network-access authorization policies.

The work in the IETF in the area of AAA has proceeded for a number of years now, focusing on various aspects of AAA. These include efforts related to the architecture of a AAA system [15][16] and a AAA Authorization Framework[13] in the AAAARch Research Group, efforts in the AAA Working Group focusing on Radius, Diameter, the NAI and Network Access, as well as efforts in the Policy Framework Working Group.

What the TNC Architecture adds to the field of AAA is the ability to measure and report on the security state of the endpoint platform as part of an authentication and authorization process. This measurement involves capturing the security-relevant operational state of the endpoint as integrity information that can be sent to a AAA Server. In communicating a client's integrity information to a AAA Server, the TNC Architecture uses and extends existing protocols defined within the IETF so that it does not impact AAA architectures that are being deployed in the field today. Here, the TNC Architecture seeks to provide a richer set of security attributes for use in authorization policies. Thus, a Requestor can be given or denied network access based on a set of finer grain rules that peer deeper into the Client's system state. In this way, a AAA Server can provide authorization to a Client not only on the basis of the Client's network-related attributes (e.g. IP address, domain) and user-related attributes (e.g. user password, user certificate), but also on the Client platform integrity state (e.g. hardware configuration, BIOS, Kernel versions, OS patch, Anti-Virus signatures, etc).

3.3 TNC Architecture

The TNC Architecture is shown in Figure 2. The architecture is intentionally general due to the need for it to encompass a variety of network devices, topologies and implementation configurations. The architecture incorporates several entities, components, and interfaces which are discussed below.

The three columns in this figure depict the three *entities* in the TNC architecture: the Access Requestor (AR), the Policy Enforcement Point (PEP), and the Policy Decision Point (PDP). Within each entity (column), the boxes depict the *components* within those entities. Three horizontal shaded *layers* are depicted grouping the components. And the *interfaces* that will be standardized by the TNC are depicted by named lines. These layers, entities, components, and interfaces are described below.

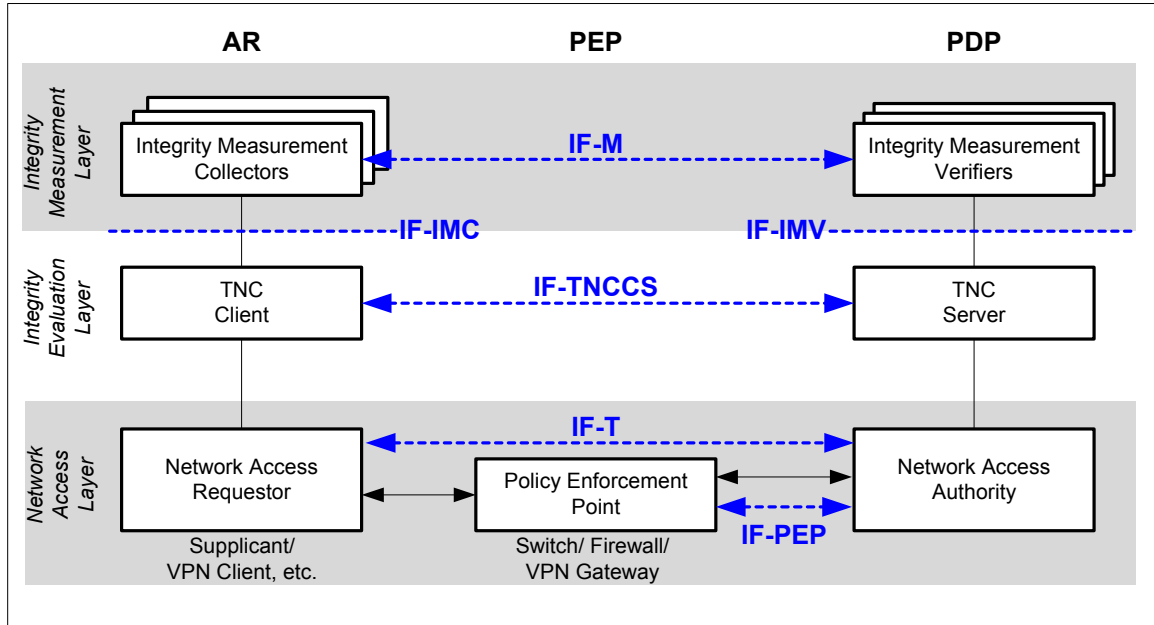


Figure 2: The TNC Architecture

It is important to note that Figure 2 shows the components (of each entity) that pertain to integrity verification established through interfaces defined as part of the work-scope of the TNC. The TNC Architecture does not preclude other components that implement other functions pertaining to network access control, and networking and security in general. For example, the Network Access Authority (NAA) could be implemented as just one component within a Radius Server within a given 802.1X usage, with the Radius Server also obtaining other policy-related information from other sources (e.g. other servers). As such, it is important for the reader to understand that the components of each entity in the TNC Architecture are not the only components implementing security and network connection management.

3.4 Layers

Three (3) abstract layers of the architecture are identified, grouping entities possessing similar functions or roles:

- *The network access layer:* These are the components whose main function pertains to traditional network connectivity and security. They may support a variety of networking technologies (e.g. VPN, 802.1X). The components found in this layer are the Network Access Requestor (NAR), the Policy Enforcement Point (PEP) and the Network Access Authority (NAA).
- *The integrity evaluation layer:* The components in this layer are responsible for evaluating the overall integrity of the Access Requestor with respect to certain access policies.
- *The integrity measurement layer:* This layer contains plug-in components that collect and verify integrity-related information for a variety of security applications on the Access Requestor.

3.5 Entities

The Entities within the Architecture are the Access Requestor (AR), the Policy Enforcement Point (PEP) and the Policy Decision Point (PDP):

- *Access Requestor (AR)*: The AR is the entity seeking access to a protected network.
- *Policy Decision Point (PDP)*: The PDP is the entity performing the decision-making regarding the AR's request, in light of the access policies.
- *Policy Enforcement Point (PEP)*: The PEP is the entity that enforces the decisions of the PDP regarding network access.

All entities and components in the architecture are logical ones, not physical ones. An entity or component may be a single software program, a hardware machine, or a redundant and replicated set of machines spread across a network, as appropriate for its function and for the deployment's needs.

3.6 Components

Referring to Figure 2, the components making up the entities are as follows.

3.6.1 Access Requestor

The Access Requestor consists of the following components:

- *Network Access Requestor (NAR)*: The NAR is the component responsible for establishing network access. The NAR can be implemented as a software component that runs on an AR, negotiating its connection to a network. There may be several NARs on a single AR to handle connections to different networks. One example of a NAR is the Supplicant in 802.1X, which is often implemented as software on a client system.
- *TNC Client (TNCC)*: The TNCC is a software component that runs on an AR, aggregating integrity measurements from IMCs and assisting with the management of the *Integrity Check Handshake* and the measurement and reporting of platform and IMC integrity. Here, Integrity Check Handshake is an example of a TCG attestation protocol in the context of the TNC.
- *Integrity Measurement Collector (IMC)*: The IMC is a software component that runs on an AR, measuring security aspects of the AR's integrity. Examples include the Anti-Virus parameters on the Access Requestor, Personal Firewall status, software versions, and other security aspects of the AR. Note that the TNC Architecture accommodates implementation situations where multiple IMCs reside on a single AR, catering for corresponding different applications.

3.6.2 Policy Enforcement Point

The PEP consists of the following components:

- *Policy Enforcement Point (PEP)*: The PEP is a component that controls access to a protected network. The PEP consults a PDP to determine whether this access should be granted. One example of the PEP is the Authenticator in 802.1X, which is often implemented within the 802.11 Access Point.

3.6.3 Policy Decision Point

The PDP consists of the following components:

- *Network Access Authority (NAA)*: The NAA is a component that decides whether an Access Requestor (AR) should be granted access. The NAA may consult a TNC Server to determine whether the AR's integrity measurements comply with the NAA's security policy. In many cases, an NAA will be an AAA Server but this is not required.
- *TNC Server (TNCS)*: The TNCS is a component that manages the flow of messages between IMVs and IMCs, gathers IMV Action Recommendations from IMVs, and combines those recommendations (based on policy) into an overall TNCS Action-Recommendation to the NAA.

- *Integrity Measurement Verifier (IMV)*: The IMV is a component that verifies a particular aspect of the AR's integrity, based on measurements received from IMCs and/or other data.

3.7 TNC Interfaces

There are a number of interfaces shown in Figure 2, which define relationships between components, and the protocols and messages exchanged between components. These interfaces are briefly discussed here.

3.7.1 Integrity Measurement Collector Interface (IF-IMC)

IF-IMC is the interface between Integrity Measurement Collectors (IMCs) and a TNC Client (TNCC). IF-IMC is primarily used to gather integrity measurements from IMCs so they can be communicated to Integrity Measurement Verifiers (IMVs) and to enable message exchanges between the IMCs and the IMVs. It also allows IMCs to coordinate with the TNC Client as needed. For more details about IF-IMC, refer to the IF-IMC Specification [3].

Software, firmware and hardware components are expected to report status information to the TNC Client on the AR platform. The components that must report information are potentially previously identified during platform integration or configuration. The TNC Client supports an API to allow these components to communicate with it locally to report component specific status information. The TNC Client acts as a conduit for the IMC that collect information from possibly multiple software, firmware and hardware components, and delivers the integrity measurements to the peer IMV through the TNC Server. In the case where the Client is a Trusted Platform with a TPM, the integrity-measurements are also deposited in the Client's Stored Measurement Log. How the measurements were handled on the platform (whether a TPM was used or not) must also be conveyed to the TNC Server.

3.7.2 Integrity Measurement Verifier Interface (IF-IMV)

IF-IMV is the interface between IMVs and a TNC Server (TNCS). IF-IMV is primarily used to receive integrity measurements sent from client-side IMCs to corresponding IMVs, to enable message exchanges between the IMCs and the IMVs, and to allow IMVs to supply their recommendations to the TNCS. For more details about IF-IMV, refer to the IF-IMV Specification [4].

3.7.3 TNC Client-Server Interface (IF-TNCCS)

IF-TNCCS relates to interaction between the TNC Client and the TNC Server as it pertains to the exchange of integrity measurement data. More specifically, this interface defines a protocol that conveys:

- (a) Messages from IMCs to IMVs (such as integrity measurements)
- (b) Messages from IMVs to IMCs (such as requests for additional integrity measurements, or remediation instructions)
- (c) Session management messages as it pertains to (a) and (b) above, and other session synchronization information between the TNC Client and TNC Server.

Note that the contents of the messages being passed between the IMCs and IMVs ((a) and (b) above), are opaque to the IF-TNCCS layer. IF-TNCCS relies on the underlying network authorization transport protocol (IF-T) to provide a secure authenticated channel to protect the messages in transit between the TNC client and the TNC server, and ensure they are delivered to the correct TNC-C or TNC-S.

The TNC expects to standardize this protocol in the future, preferably based on existing protocols with TNC-specific modifications to carry integrity related information. An example of such a protocol is the *TLS-Attestations Extensions* protocol [8], which can be run over several underlying

types of transport sessions. One common approach adopted by many transport protocols and authentication protocols is to use the *Type-Length-Value* (TLV) structure that allows for maximum flexibility in conveying information within other enveloping message structures. Examples of protocols using TLVs include PEAP and TTLS in the context of the 802.1X authentication framework.

3.7.4 Vendor-Specific IMC-IMV Messages (IF-M)

IF-M pertains to vendor specific information exchange that may occur between IMCs and IMVs. These messages are identified by a message type with an allocation system designed to avoid accidental reuse of types. In practice these messages are carried over the IF-TNCCS interface. The TNC expects to standardize certain widely useful IF-M messages.

Note that both IF-TNCCS and IF-M are relevant not only to Trusted Network Connections, but to the larger TCG requirements around platform management.

3.7.5 Network Authorization Transport Protocol (IF-T)

IF-T pertains to the transportation of messages between the AR as an entity and the PDP as an entity. The components that deal with message transport in this case are the Network Access Requestor (in the AR) and the Network Access Authority (within the PDP).

The TNC does not expect to standardize this protocol in the future but will provide certain recommended bindings describing how this protocol can be implemented. For example, the IF-T protocol could be implemented using EAP within 802.1X. Another example might run over plain IP (with suitable security layered on).

3.7.6 Platform Trust Services Interface (IF-PTS)

IF-PTS provides platform trust services that ensure that TNC components are trustworthy. The TNC expects to standardize this interface in the future. See Section 6 for additional detail.

3.7.7 Policy Enforcement Point Interface (IF-PEP)

IF-PEP allows the PDP to communicate with the PEP, especially allowing the PDP to instruct the PEP to isolate the AR during remediation and later grant it full network access once remediation is complete. The TNC expects to document this interface in the future, although it may well be based on standards from other bodies. See Section 6.

3.8 Goals and Assumptions

There are a number of requirements and assumptions with regards to the interfaces and messages of the TNC Architecture in Figure 2 from the perspective of security and message transport:

- G1** *Common Integrity Schema*: The integrity measurements communicated between the TNC Client and the TNC Server will be structured according to a common TNC Integrity Scheme.
- G2** *Independence of Integrity Schema definition*: The Integrity Scheme definition will be defined independent from the underlying transport protocol mechanism between the AR and the PDP.
- G3** *Number of messages*: There is no limit to the number of messages exchanged between an IMC and an IMV within a given platform-authentication event. (Note, however, that in practice there is a limited time for completing the authentication, and thus IMV/IMC

implementers are encouraged to minimize the data exchanged, and the number of roundtrips required to complete their assessment.)

- G4** *Endpoint integrity checking as part of endpoint authentication and authorization:* Since the verification of security compliance is core to the TNC design, the TNC Architecture requires that integrity checking be supported either as part of (during) an overall authentication/authorization event (e.g. user authentication, AIK-certificate validation, etc), or as a separate event after (following) other forms of authentication have been performed. This decoupling allows re-verification of integrity information to be done independent of other authentication events (e.g. periodic checking of AV-status every few minutes vs. user-authentication at network logon time).
- A1** *Protection and reliability of message transport:* Since the integrity measurements data communicated between a TNC Client and TNC Server is not self-protecting, it is assumed that an underlying mechanism will provide for the protection of the data as it is delivered between the two endpoints.
- A2** *Platform-authentication invocation:* For an Integrity Check Handshake, the IMC will always initiate by sending the first message in an authentication dialog between the IMC and the IMV.

3.9 Basic Message Flows Across Interfaces

There are a number of fundamental message types that are exchanged between components in the architecture, across the various interfaces defined above. These are summarized in Figure 3 and are described in the following. Note that in the following illustration, several levels authentication and authorization are assumed to have been configured to occur before a connection request can be completely fulfilled. In this example, these consist of the following order: User Authentication, Platform-Authentication and Integrity Check. Note, however, that in other situations this may not necessarily be the order of processing..

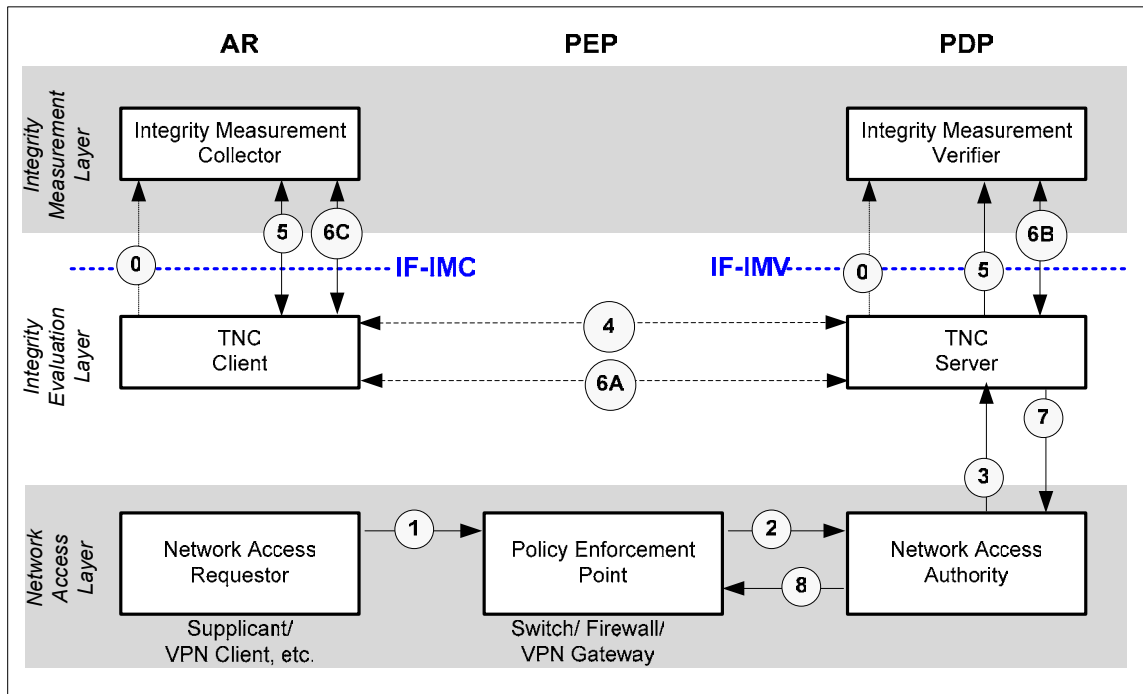


Figure 3: Message Flow between components in the TNC-Architecture

- **Flow 0:** Prior to beginning a network connection and integrity-check handshake attempt, the TNCC must discover and load each relevant IMC using the platform-specific binding. The TNCC must then initialize the IMC, which includes defining the necessary connection IDs and IMC IDs, and ensuring that the TNCC has a valid connection state with the IMC.

During the load process, the TNCC may check the integrity of the IMCs. This is optional. If a TPM is present, this check will typically involve hashing the IMCs and adding their hashes to a PCR (i.e. performing one or more TPM Extend operation). If no TPM is present, this check may involve checking the signatures on the IMCs. Integrity checks during IMC loading are done completely by the TNCC since there is no TNCS or IMV available. TNCS and IMVs will get a chance to do platform authentication of the endpoint platform later in the sequence of events.

Similarly, the TNCS must discover and load each relevant IMV using the platform-specific binding.

- **Flow 1:** When a network connection attempt is triggered (automatically or by user request), the NAR at the AR (client) initiates a connection request at the network layer.
- **Flow 2:** Upon receiving a network connection request (from the NAR), the PEP sends a network access decision request to the NAA. Here, the NAA is assumed to have been configured to perform User Authentication, Platform Authentication and Integrity Check in that order.

User authentication can occur between the NAA and the AR. Platform Authentication and Integrity Check has to occur between the AR and the TNCS.

Note that since an ordering of authentication has been configured, failure in one authentication will discontinue other forms of authentication and integrity check. That is, if the user fails user authentication with the NAA, then platform authentication and integrity check will not proceed.

- **Flow 3:** Assuming that User Authentication succeeded between the user (on the AR) and the NAA, the NAA then informs the TNCS of the connection request.
- **Flow 4:** The TNCS then performs (mutual) Platform Authentication with the TNCC, verifying, for example, that valid (unrevoked) AIK-certificates are used by both endpoints.
- **Flow 5:** Assuming that Platform Authentication succeeds between the TNCS and TNCC, the TNCS indicates to the IMVs (using interface IF-IMV) that a new connection request has occurred and that an Integrity Check Handshake needs to be carried out by the TNCS. Similarly, the TNCC indicates to the IMCs (using interface IF-IMC) that a new connection request has occurred and that an Integrity Check Handshake needs to be carried out by the TNCC. The IMCs respond by giving a number of IMC-IMV messages to TNCC across IF-IMC.
- **Flow 6A:** In order for an Integrity Check Handshake to occur, the TNCS and TNCC begin the exchange of messages pertaining to the integrity check. These messages will be relayed through the NAR, PEP and NAA, and will continue until the TNCS is satisfied with the integrity status of the AR. Flow 6A shows this as a peer connection between the TNCS and TNCC.
- **Flow 6B:** The TNCS passes each IMC message to the matching IMV or IMVs through IF-IMV (using message types associated with the IMC messages to find the right IMV).

Each IMV analyzes the IMC messages. If an IMV needs to exchange more messages (including remediation instructions) with an IMC, it provides a message to the TNCS through IF-IMV. If an IMV is ready to decide on an IMV Action Recommendation and IMV Evaluation Result, it gives these to the TNCS through IF-IMV.

- **Flow 6C:** Similarly, the TNCC will forward messages from the TNCS to the matching IMC or IMCs through IF-IMC, and send messages from the IMCs to the TNCS.
- **Flow 7:** When the TNCS has completed its Integrity Check Handshake with the TNCC, it then sends its TNCS Action Recommendation to the NAA. Note that the NAA may still have the option of not granting network access if other security policy requirements have not been met by the AR (even though the AR has passed the Integrity Check).
- **Flow 8:** The NAA then sends its network access decision to the PEP to enforce. The NAA must also indicate its final decision to the TNCS which will be sent to the TNCC. Typically, the PEP indicates its execution of the decision (e.g. Port open in 802.1X) to the NAR.

The above represents the basic behavior of entities in the architecture (assuming a successful connection request, without remediation). Each specific deployment of the architecture will have its own unique policy configuration and network topological aspects that will dictate how additional steps may occur.

4 Design Aspects of the TNC Architecture

The current architecture is aware of the reality that there are multiple technologies relevant to achieving endpoint integrity at differing layers of the IP stack, and that therefore a robust and extensible design needs to be achieved in order to accommodate as wide a scenario set as possible.

In the current section we outline and describe some design aspects and design decision behind the TNC Architecture, providing a brief description of these aspects as they relate to the interaction among the various components of the architecture.

Note that this architecture does not preclude a solution whereby a vendor provides a single product that encompasses the TNCC and IMC roles. Similarly, this architecture does not preclude a solution whereby a vendor provides a single product that encompasses the TNCS and IMV roles, where the IMV does not use the IF-IMV API to communicate with the TNCS (i.e. an embedded IMV). In addition, the current architecture allows for certain function (e.g. IMV) to be implemented as a separate back-end components.

4.1 Aspects of TNC Client and TNC Server Interaction

There are a number of design aspects of the TNCC and TNCS interaction that warrant highlighting in order to provide some background information regarding the behavior of an AR and PDP. These are summarized in the following:

- *TNCC-TNCS connection management*: The interaction between a TNCC and TNCS represents rich behavior that covers various phases of endpoint integrity establishment (e.g. checking, remediation, retry/re-connection, etc).

In order to support these various behaviors, the TNC Architecture has designated the support for connection management to be best implemented within the TNCC and TNCS. Among other benefits, this allows the IMCs and IMVs to be designed and implemented without dependence or concern regarding their underlying transport mechanisms. In addition, the TNCC and TNCS are deemed to be the best entities to maintain contextual information regarding a TNCC-TNCS session (over an underlying transport connection) in order to support the notion of reconnections (or session re-establishments).

One construct that the TNCC-TNCS connection management deploys is that of a *network connection ID*, which represents a logical relationship between a TNCC and TNCS. For a given connection between the TNCC and TNCS, the TNCC and its IMCs establish a local connection ID. Similarly, the TNCS and its IMVs also establish a unique local connection ID. When a new connection is initiated, it is the TNCC and the TNCS each generate a unique connection ID that is made available to the IMC and an IMV respectively, in order for them in turn to identify their corresponding relationships. Note that the TNCS-generated connection ID and the TNCC-generated connection ID may be different, for the same connection.

The primary purpose of the connection ID is as a local handle for an IMC or the IMV to maintain state information associated with the connection. In addition, the connection ID may be maintained across multiple Integrity Check Handshakes. This enables IMCs and IMVs to each maintain state information associated with an earlier handshake (e.g. for status refreshes), and also allows for Integrity Check Handshake re-tries to be initiated from an IMC or IMV. Note that the connection ID is a local construct and is not sent or shared between the TNCC and the TNCS. Furthermore, for each network connection (between a TNC-Client and TNC-Server), there is exactly one TNC-Client and one TNC-Server.

- *IMC-to-IMV message delivery*: One key function of the TNCC and TNCS is to provide message delivery transportation between an IMC and an IMV. The TNCC and TNCS is not

required to understand the semantics of the information communicated by the IMC-IMV pair. Each message consists of a body, type and intended recipient type. The TNCC and TNCS use the message type information and recipient type to route and deliver messages to the appropriate destination IMC or IMV.

Note that to the IMC and IMV, messaging is achieved using interface IF-IMC and IF-IMV. As such, an IMC actually performs a function-call (instead of explicitly sending message). The appropriate *SendMessage* and *ReceiveMessage* set of APIs are defined as part of the IF-IMC and IF-IMV specifications. The *RequestHandshakeRetry* API may be used by IMCs or IMVs to originate an Integrity Check Handshake retry, enabling IMC-IMV messaging. The IMC may specify the connectionID (or wildcard for all available connections) identifying IMVs/TNCSs that should receive the message.

TNCS implementers should be aware that the TNCC may gather all the messages that IMCs want to send before sending off the messages for that round. Once all IMCs have finished sending their messages for a round, the TNCC will send those messages to the TNCS and await its response.

- *Number of message rounds and underlying transport:* Since the current TNC Architecture seeks to be applicable to a broad range of network transport mechanisms (e.g. Dial-up PPP, EAP/802, VPNs, etc), the issue of minimizing message rounds used by an IMC-IMV pair becomes an important aspect.

In order to accommodate a broad range of use scenarios, the current architecture defines a number of basic messaging behaviors, realized through a number of message-related functions in interface IF-IMC. These include a function used by a TNCC to indicate to IMCs that an Integrity Check Handshake is beginning, a function for an IMC to respond to this initiation, and a return function when the IMC has sent all the messages. Note that that messages can be delivered in one or more rounds.

Note that the IF-IMC interface is defined to also allow the TNCCs and TNCSs to employ messaging mechanisms that are not based on rounds or flows. However, they must deploy a round-based messaging system over those protocols (the IMCs send messages, then the IMVs send messages, etc.).

- *Error handling:* IF-TNCCS is responsible for communicating error messages between TNCC and TNCS. Error messages / codes are defined to ensure TNCCS protocol state is deterministic. Errors in other components (e.g. IMC/IMV) are to be addressed by the corresponding layer (e.g. IF-M). Therefore, an error in an IMV may result in an IMV/IMC protocol message, while the IF-TNCCS messaging may be successful.

4.2 Aspects of TNCC-IMC Interaction and TNCS-IMV Interaction

There are a number of design aspects of IF-IMC which define the communications and interactions between the TNCC and IMC. These are summarized in the following, with further detailed information found in [3]. Likewise for IF-IMV.

- *Secure channel between the IMC and IMV:* The TNC Client and TNC Server are assumed to provide a secure communications tunnel between the IMCs and the IMVs. This security requirement allows an IMC-IMV pair to focus only on their main function (e.g. AV), while leaving the details of the secure channel implementation (e.g. EAP tunnel, VPN, etc) up to the TNCC and TNCS.
- *Support for multiple TNCC on a single AR:* There are circumstances where a given TNCC/TNCS pair are uniquely designed to support a service relating to the integrity status

reporting and verification. It is difficult – if not impossible – to mandate a single TNCC that would cater for all varieties of TNCS/IMV pairs and underlying network technologies. As such, it is perceived that a platform may in fact possess multiple TNCCs according to the local network security policy requirements. The current TNC Architecture accommodates for such a case of multiple TNCCs per platform (AR).

In this respect, the IF-IMC API defines the support for multiple TNCCs on a single AR. Furthermore, it supports multiple overlapping network connections and Integrity Check Handshakes for a single TNCC.

- *Platform-independence*: Given the increasing heterogeneity of most networks today, the TNC Architecture anticipates the deployment of various devices and services within a network based on various platforms – each of which will require endpoint integrity verifications.
- *Support for connection state across remediation and handshake re-tries*: When a new TNCC-TNCS relationship is established, the TNCC and TNCS independently choose a network connection ID to refer to that relationship. The TNCC and TNCS inform the IMCs and IMVs of the new network connection and update them whenever the state of the network connection changes. When a network connection is complete, the TNCC and TNCS notify the IMCs and IMVs that the network connection ID will be deleted and then does so.

The TNCC and TNCS are not required to maintain the network connection ID across multiple connection attempts, remediation and connection retries. In fact, it will be common for the TNCS to avoid maintaining such state but for the TNCC to maintain the state for some time. There are some benefits to maintaining the network connection ID. When an AR fails to pass all integrity verification requirements as defined by the network policy, the AR is typically redirected to a remediation facility or function (e.g. separate remediation LAN) in order for its IMCs to collect/update their integrity data (e.g. AV-signature data or software updates). If the TNCC connection state is maintained, each IMC can inform the TNCC using the connection ID of the completion of its updates.

One potential benefit of maintaining the network connection ID at the TNCS is the possibility of the TNC Server informing all TNC Clients on the network of an update of the network access policies and updates to the IMVs. This will trigger the TNC Clients to update their own integrity data and to perform an Integrity Check Handshake retry based on their updated integrity data.

Finally, maintaining a network connection ID on a TNCC or TNCS allows an IMC or IMV to request an Integrity Check Handshake retry when the IMC or IMV detects that an attack on the client platform has been attempted since the last Integrity Check Handshake.

Due to the variety of possible underlying network entities implementing the TNC Architecture, it may not be possible for a TNCC or TNCS to restart an integrity handshake when requested. The TNCC and TNCS must support a method for an IMC or IMV to request a handshake retry, but it is acceptable for the TNCC or TNCS to simply return an error code and not retry the handshake.

- *Support for multiple connections*: The IF-IMV API must support multiple overlapping network connections and Integrity Check Handshakes for a single TNCS from multiple TNCCs, and communication between the TNCS and multiple IMVs. Similar requirements hold for IF-IMC.
- *Support for recommending isolation*: IF-IMV must have some mechanism for IMVs to recommend isolation and compliance information to the TNCS, so that isolation can properly be supported on the network. This may stop short of an explicit mechanism for knowing which network to assign for isolation, but there must be a way to pass intelligence from IMVs to the TNCS.

5 Assessment, Isolation and Remediation

Although not visibly evident within the TNC Architecture of Figure 2, one important feature of the architecture is its extensibility and support for the isolation and remediation of ARs which do not succeed in obtaining network access permission due to failures in integrity verification. Figure 4 shows an additional layer addressing remediation and provisioning.

Note that in the current TNC Architecture document, remediation is out of scope and is treated briefly for completeness.

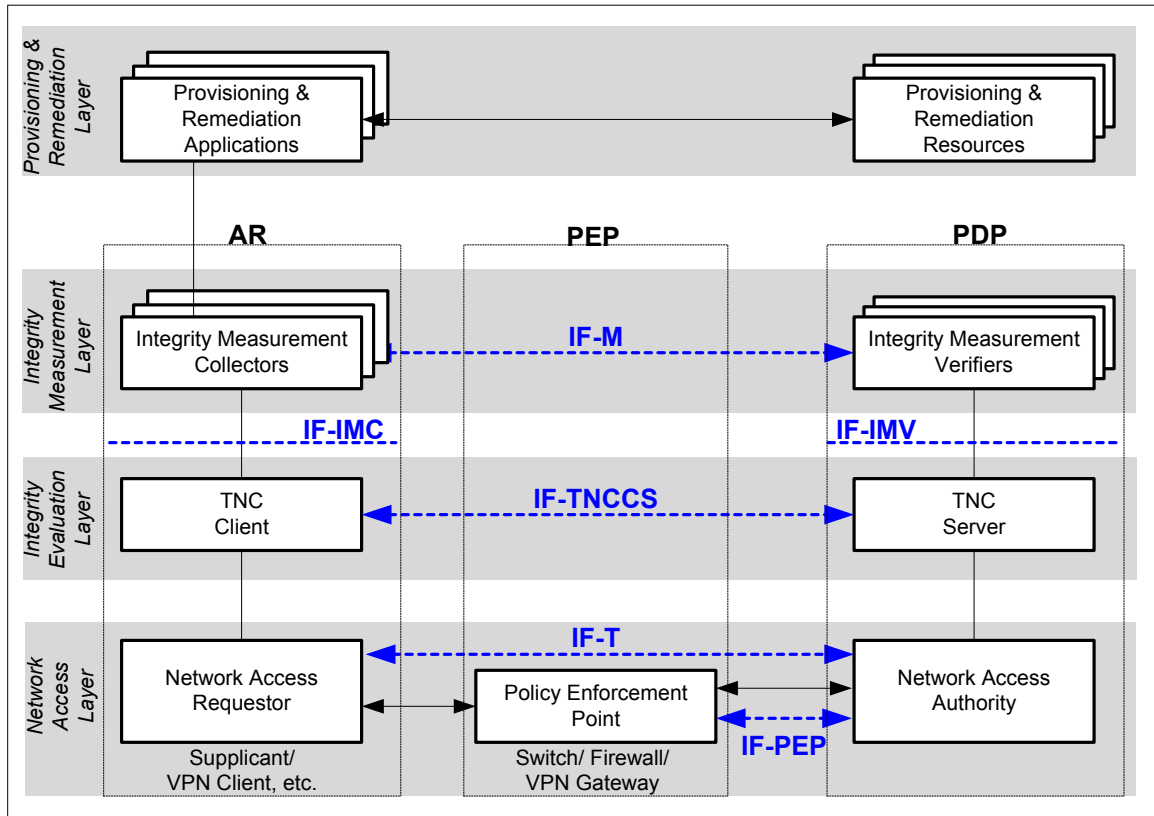


Figure 4: The Provisioning and Remediation Layer in the TNC Architecture

5.1 Phases in Network Access Control

In order to understand the actions needed to remedy ARs that fail integrity verification, it is useful to view network connection requests in three basic phases from the perspective of integrity verification:

- *Assessment:* In this phase, the IMVs perform the verification of the AR following the policies set by the Network Administrator and optionally deliver remediation instructions to the IMCs.
- *Isolation:* If the AR has been authenticated and is recognized to be one that has some privileges on the network but has not passed the integrity-verification by the IMV, the PDP may return instructions to the PEP to redirect the AR to an isolation environment where the AR can obtain integrity-related updates.

- *Remediation*: Remediation is the process of the AR obtaining corrections to its current platform configuration and other policy-specific parameters in order to bring it inline with the PDP's requirements for network-access of the PDP.

5.2 Assessment Phase

In the Assessment Phase, the TNC Client reports its current integrity status to the TNC Server. Upon receiving the client integrity status, the IMVs with the aid of the TNCS performs an assessment of the AR based on the set of policies defined by the network administrator. The IMV can make one of three IMV Action-Recommendations (Allow, Isolate or Block) or it can make no recommendation.

If the platform is a Trusted Platform that deploys a TPM, then certain basic verifications, such as authenticating the platform's AIK-certificates, should be verified first before other more platform-specific verifications are performed.

It is important to note here that the TNCS dialog with the TNCC may consist of several rounds of messages, where in each round the IMVs request more detail. This represents an extension to the basic behavior of the TNCC simply reporting all its integrity information in a single set of messages to the TNCS.

If the IMVs find that remediation is needed, they will typically send remediation instructions to the IMCs in the final message of their dialog. The IMCs may execute these instructions immediately or hold them until some form of network access is available.

5.3 Isolation Phase

An important tool in the effort to remediate ARs that fail integrity verification is the isolation of that AR entity to a separate network – referred to here as the Isolation Network – in order to provide remediation services to the AR entity. This protects the AR from the full network and vice versa, preventing the spread of viruses and worms. There are a number of technical approaches today to achieve network isolation for the AR entity. Two of these are as follows:

- (a) *VLAN Containment*: VLAN containment permits the AR to access the network in a limited fashion. Typically the primary purpose of the limited access is to allow the AR to access on-line sources of remediation data (e.g. virus definition file updates, worm removal software, software patches, etc). In some cases, no remediation is offered and the AR is instead offered access to limited services, in such a fashion as to limit the potential for impact to the network or other attached hosts. Radius provisions VLAN containment using the Tunnel-Private-Group-ID attribute, as specified in [RFC3580].
- (b) *IP Filters*: In the case of IP filters, the PEP is configured with a set of filters which defines network locations reachable by the isolated AR. Packets from the AR destined to other network locations are simply discarded by the PEP. Radius selects filter rules for application to a network access session using the Filter-ID attribute [RFC2865, RFC3580].

5.4 Remediation Phase

The TNC Architecture in Figure 4 accommodates a number of schemes for remediation. The intent of remediation is generally universal, namely that of performing updates to the software and firmware of the AR to help it comply with the current network policy.

The general aim of remediation is to bring the AR up to date in all integrity-related information, as defined by the current policy for authorization. Examples include OS patches, AV updates, firmware upgrades, etc. Section 5.5 below discusses the TNC approach to remediation in further detail.

After remediation has been completed, the IMCs can ask the TNCC to retry the Integrity Check Handshake, which results in another Assessment Phase. This second phase may be shorter than

the first since the IMCs may be able to send only the data that has changed (if supported by the IMVs).

5.5 Remediation in the TNC Architecture

The TNC Architecture supports remediation, both from the trusted network connect (endpoint integrity) perspective, and from the broader TCG platform manageability perspective. In the Architecture, entities take-on a specific role may have additional functions in other contexts beyond endpoint integrity.

The TNC Architecture support for remediation and provisioning is expressed in the corresponding *Provisioning & Remediation layer* in Figure 4. The layer contains applications, services and other resources necessary to establish and maintain a trusted platform according to the owners specifications. It is relevant not only for the remediation needs of trusted network connections – where enterprises can keep their system up to date – but also for the broader needs of Trusted Computing. These may include any of the following:

- Compliance and policy evaluation
- Collection / distribution of baseline measurements
- Provisioning of policies, settings, software and firmware
- Trusted-platform specific operations (see Section 6).

There are two entities relevant to remediation in the TNC Architecture (see Figure 4):

- *Provisioning & Remediation Applications (PRA)*: The Remediation Application can be implemented in several forms. For example, the PRA could be implemented as part of the Access Requestor (AR) entity. Here, the PRA communicates with the IMC and provides it with specific types of integrity information. An example of an embodiment of the PRA would be the Anti-Virus application software that communicates with sources of Anti-Virus parameters (e.g. latest AV signature files). Note that the PRA could be implemented as part of the IMC. As another example, the PRA/IMC could be an agent that updates the TPM and the TSS (part of the PTS), which obtains updates from the TPM Manufacturer.
- *Provisioning & Remediation Resources (PRR)*: The PRR represents the various sources of integrity information needed to update the AR so that it can be successfully verified by the PDP at the next re-attempt of the handshake. Examples of the PRR include Enterprise servers, vendor services (e.g. FTP server), CDs/DVDs containing the update parameters, and others.

6 TNC Architecture with the Trusted Platform Module

The TNC Architecture accommodates both platforms that have a TPM and those that do not. In this section we further delve into the details of the TNC Architecture for platforms that possess a TPM.

6.1 Features of a Platform with a TPM

One of the core value propositions of the TNC approach is that a hardware protected root-of-trust within devices or platforms can help establish the self-integrity of the platform, and to communicate *platform proof-of-identity* and *platform integrity information* as part of an authentication event to an authentication server. Trust in a platform is built bottom-up, starting at the base with Trusted Platform Module (TPM) hardware bound to the platform's motherboard.

An important concept that distinguishes the TCG approach to platform-authentication is the notion of a trusted platform containing a TPM that features *protected capabilities*, *integrity measurement and storage*, *integrity reporting* and *attestations*. All four properties or functions are core to trusted computing. These features are as follows:

1. *Protected Capabilities*: Protected capabilities are a set of commands with exclusive permission to access *Shielded Locations*. Shielded locations are places (memory, register, etc.) where it is safe to operate on sensitive data. The TPM implements protected capabilities and shielded-locations. Among others, it is used to protect and report integrity measurements that are stored inside the TPM's *Platform Configuration Registers* (PCR). The TPM also stores cryptographic keys used to authenticate reported measurements. Depending on the implementation, TPM protected capabilities can include additional security functionality, such as cryptographic key management, random number generation, sealing data to system state, monotonic counters and others.
2. *Integrity Measurement and Storage*: Integrity measurement is the process of obtaining metrics of platform characteristics that affect the integrity (trustworthiness) of a platform; storing those metrics; and putting digests of those metrics in PCRs. An intermediate step between integrity measurement and integrity reporting is *integrity storage*. Integrity storage stores integrity metrics in a log and stores a digest of those metrics in PCRs.
3. *Integrity Reporting*: Integrity reporting is the process of attesting to the contents of integrity storage (i.e. PCRs), protected using private keys held in shielded locations in the TPM. The philosophy of integrity measurement, storage and reporting is that a platform may be permitted to enter any state possible including undesirable or insecure states, but that it may not be permitted to lie about states that it was or was not in. The TPM contains a number of Roots of Trust (RT), in the form of TPM-bound keys that are non-transitory, non-migrateable and inaccessible by processes outside the TPM.
4. *Attestations*: Attestation is the process of vouching for the accuracy of information, such that a relying party can use the attestation to decide whether it trusts the remote platform. A platform can attest to its description of platform characteristics that affect the integrity (trustworthiness) of a platform. Obviously, all forms of attestation require reliable evidence of the attesting entity.

In addition to the fundamental features of Trusted Platforms that are mentioned above, in the context of platform-authentication (see TCG Infrastructure Architecture specifications [2]), there are additional benefits that the trusted platform approach can provide:

- *Evaluation and Decision Making*: When a platform seeks to assess the integrity trustworthiness of a second platform with which it is communicating, it needs to evaluate that second platform based on some policies which are meaningful and actionable by both platforms. The outcome of platform evaluation is not limited to binary results (such

as success/fail), but may include ranges of values (e.g. 1 to 100) indicating the level confidence the evaluating platform has with regards to its assessment. Note that the outcome of an evaluation process by an evaluating platform may be consumable by a third party who must understand the semantics of the evaluation result coming from the evaluating platform.

- **Enforcement and Response:** Depending on the exact configuration of an evaluating platform, the platform may in fact be a *policy enforcement point* (PEP) for a given set of environmental-specific policies. In addition, the platform may return *responses* to another platform, of whom it evaluated.

These features play an important role when an AR seeks to obtain network access by reporting its integrity measurements to the PDP, which perform evaluation and decision-making regarding the access request, and which directs its evaluation results to the PEP for enforcement.

In order for a TNC Client implementation to be able to make use of the TPM and its functionality, a separate layer of services – called the Platform Trust Services – has been introduced. This layer provides some level of abstraction in order for both the TNC Client and the IMC to query their underlying platform trust information within the AR on which they operate.

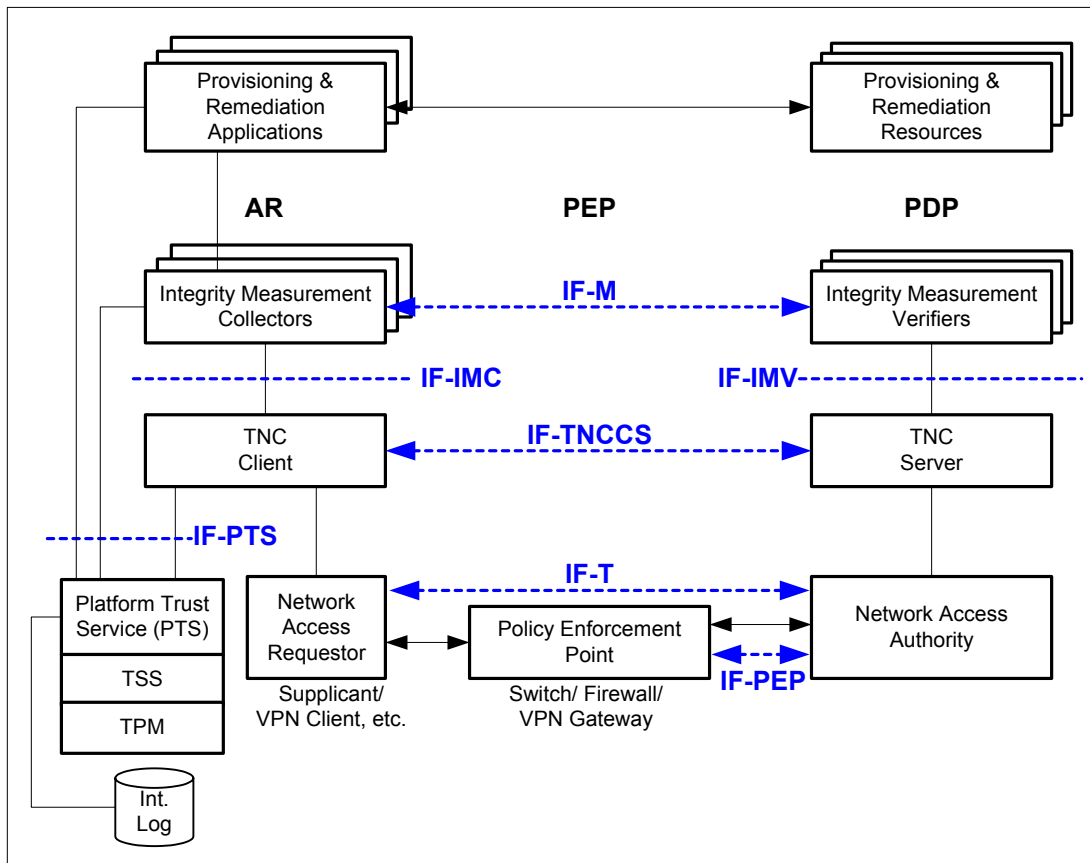


Figure 5: The TNC Architecture with the Trusted Platform Module (TPM)

6.2 Entities

The entities in TNC Architecture do not change with the introduction of trusted platforms (Figure 5). However, the concept of platform ownership and the *owner* entity should be considered. The TNC architecture identifies three entities, AR, PEP and PDP. In most if not all cases, the PEP and PDP have the same *owner*. In other words, they are controlled by the same IT department or

service provider. The AR may also have the same owner as PEP and PDP, but ownership should be re-validated before extending special privileges. Usually this is part of platform-authentication with a PDP.

In the case where AR and PEP/PDP owners differ, platform-authentication and attestation allow the AR and PDP to identify one or more trusted 3rd parties with whom AIK credentials can be established. In the language of Trusted Computing, the trusted 3rd party is referred to as the Platform-CA or Privacy-CA. The trusted 3rd party is another entity that may play a role in establishing platform trust. The components, protocols and interfaces described below support interactions between these entities.

6.3 Components

In addition to previously described TNC components, the TNC architecture includes additional components when a TCG trusted platform makes up the host environment. The additional components are described here:

- *Platform Trust Service (PTS)*: The PTS is a system service that exposes trusted platform capabilities to TNC components. PTS services include protected key storage, asymmetric cryptography, random numbers, platform identity, platform configuration reporting and integrity state tracking.
- *The TCG Software Stack (TSS)*: The TSS [10] is a library that implements TPM support functions. These include unlimited key storage (off-chip protected), key caching and higher-level interface abstraction.
- *The Trusted Platform Module (TPM)*: The TPM provides hardware-based key storage, verifiable platform identity, verifiable configuration reporting, cryptography and security services [9].

6.3.1 Platform Trust Services

PTS architecture can be divided into four classes of functionality, namely; TNC component integrity services, platform-authentication, trust transitivity and support for cryptography. The PTS may possess TPM *owner authorization* privileges as required to perform TPM operations. Some of these functionalities are described below, while others have been described in-depth elsewhere (see [1] and [2]).

6.3.1.1 TNC Component Integrity Services

The PTS maintains system integrity logs and ensures the logs accurately reflect the Platform Configuration Register (PCR) state. In addition to pre-boot and OS integrity state, the TPM can capture application integrity state.

The PTS exposes interfaces for TNC Client (TNCC) and Integrity Measurement Collectors (IMC) software to extend PCRs and write to integrity measurement logs. The PTS converts platform specific integrity log entries into an interoperable format according to TCG integrity schema specifications. All log entries must be in the independent format before being sent over an IF-TNCCS interface.

The PTS manages TPM finite resources including key storage, PCRs, measurement logs and transport sessions. It ensures processes and threads vying for access to these resources are serialized through appropriate process and thread locking mechanisms. Access to integrity log files is controlled such that log entries are synchronized with respect to PCR contents.

An abstract representation of PCRs is exposed over IF-PTS to processes seeking to record and report integrity values. The IWG Integrity Schema [7] specification provides one such abstraction.

6.3.1.2 Application Protocols

The PTS participates in protocols that establish verifiable platform identities, platform-authentication and reporting of platform configuration state. The *inner-application protocols* are supported by the PTS by providing access to TPM protected keys including Attestation Identity Keys (AIK) and key encryption keys (KEK). The PTS may possess privileges necessary to use AIKs and KEKs or to perform other TPM protected operations.

Several protocols are anticipated to be supported by the TNC:

- Platform-authentication using an AIK and other KEKs as needed
- Platform attestation using TPM PCRs and measurement log entries.
- Platform identity registration of AIK using the TPM EK.
- Platform monitoring protocol for reporting the presence of the platform integrity agents

Other application protocols may be supported as determined by TNC requirements.

6.3.1.3 Trust Transitivity

The PTS provides component loading and registration services that can be used to capture integrity state of TNC components before execution threads are passed. The PTS cooperates with platform trust capabilities, including the Roots of Trust for Measurement (RTM) to establish transitive trust linkages.

The PTS may employ any available platform specific anti-spoofing and anti-tampering techniques as necessary to strengthen trust assurances.

6.3.1.4 Security Considerations for Network Connection with TPM

Use of a TPM helps address a man-in-the-middle threat to TNC Client and other components. TPM non-migratable and certified-migratable keys may be used to establish connections to PDP and PEP endpoints. Fixing the communications endpoint to hardware minimizes MITM attacks that divert traffic to another platform.

The TPM platform configuration registers can be used to more reliably capture and report platform configuration information thereby reducing the threat of rogue software on the client platform performing MITM redirection.

The TPM alone is insufficient at preventing all software-based tampering, but is effective at establishing the platform as an endpoint for trusted network connect. To close the vulnerability gap between the TPM and TNC components, a number of platform specific techniques may be employed. While it is not the goal of the TNC architecture to define specific techniques, it is an objective to define interfaces for TNC components to be integrity checked prior to their being relied upon by policy decision points.

6.4 Interface IF-PTS

PTS services and functionality is exposed to host processes through interface IF-PTS. Any of the TNC components may access PTS services through IF-PTS.

As a system service, the PTS must be discovered and the form of inter-process communication (IPC) established. As an element in a transitive trust chain, mechanisms for measuring a TNC component and for transferring execution control must be established.

As an arbiter of finite resources, the PTS must have a way to publish available resources and a way to block access to allocated resources.

The operating status and error condition of the PTS must be available to subscribers. The PTS may start and stop while subscribers remain operational. Individual service requests should be acknowledged by success or failure notifications. In case of no acknowledgement, a timeout or keep-alive mechanism should be employed to ensure deterministic interaction semantics.

7 Technologies Supporting the TNC Architecture

Although integrity measurement and reporting is core to the value proposition of the TNC philosophy and approach, the TNC Architecture acknowledges other networking technologies as providing the infrastructure support surrounding the core elements of the TNC Architecture. Some of the technologies are discussed in this section. Note that the TNC is not standardizing specific protocol bindings for these technologies at this time, though such binding may be required in the future.

7.1 Network access technologies

One of the important propositions regarding the concept of integrity measurement and reporting in the context of endpoint integrity establishment is the applicability of the concept to numerous network access environments and scenarios.

Although network access environments – such as remote access environments (e.g. IPsec VPNs) and on-campus WiFi access – may vary reflecting the richness of types of Internet access today and in the future, endpoint integrity remains a common requirement and pressing need in all these environments. Three of the most common network access environments are 802.1X, VPNs and PPP.

7.1.1 802.1X

The 802.1X standard [18] provides a framework for port based access control (PBAC) that is increasingly becoming accepted for LANs and WLANs. The TNC Architecture itself maps quite readily into an 802.1X framework as indeed the TNC Architecture was designed with great awareness of 802.1X and its increasing deployment today.

Thus, for example, a Supplicant in 802.1X maps quite readily to an AR in this architecture. Here, the Supplicant that wishes port access at an Authenticator (e.g. 802.11 Access Point, Switch) will be authenticated by the Authentication Server (AS) based on the access policies defined in the AS.

Integrity measurement and reporting can enhance an 802.1X deployment by providing the AS with additional data regarding the integrity status of the Supplicant. One possible embodiment would be the addition of TNC Client and a TNC Server to the Supplicant and AS respectively and the addition of the necessary methods to communicate integrity reporting between the two endpoints.

7.1.2 VPNs

Remote access based on VPNs has today become a day-to-day necessity for many enterprises, with many VPNs established using the IKE protocol and the IPsec protocol. Using endpoint integrity reporting, one possible enhancement to IPsec VPNs would be for integrity information to be communicated as part and parcel of mutual authentication and key establishment. Thus, the IKE version-1 [21]key establishment protocol could be extended to include integrity reporting at the end of phase-1 after the IKE peers have authenticated. Recently the IETF has defined an internet draft for the next version of IKE [22] which includes support for carrying EAP-based messages for IKE peer (possibly also user) authentication. The TNC architecture could leverage this EAP transport to improve alignment with other EAP-based approaches described with the TNC architecture.

Another breed of VPNs emerging recently is the SSL VPN, based on the SLL or TLS protocol. A number of vendors are already shipping products supporting SSL VPNs while some access service providers are offering SSL VPN services. Augmenting an SSL VPN offering with endpoint integrity can be achieved by enhancing the basic TLS exchange with the TLS-Attestation Extensions protocol [8] which will deliver integrity measurement information between the SSL Client and Server. Such an enhancement would strengthen endpoint integrity verification, extending beyond the traditional SSL identity certificates and other authentication technologies.

7.1.3 PPP

The Point-to-Point (PPP) protocol is the standard method for transporting multi-protocol datagrams over point-to-point links, and is the basis for dial-up access to the Internet over the public PSTN today.

From the security perspective, typically EAP is used over PPP to transport security-related parameters (see below).

7.2 Message transport technologies

The TNC Architecture also acknowledges the existence of various means of message transport mechanisms and protocols today, and aims to be flexible enough to map to deployment architectures that use those diverse transport mechanisms and protocols.

7.2.1 Protected EAP Methods

The Extensible Authentication Protocol (EAP) [RFC3748] – originally designed to carry authentication information in the context of PPP/Dial-Up – has today gained broader use in the context of 802.1X beyond just authentication. Several EAP methods have been proposed for various functions, making the basic EAP itself appear to be a general transport mechanism to those EAP methods located higher in the EAP stack. Notably, the use of TLVs and AVPs above EAP allows EAP itself to be more agnostic regarding upper layer handshakes and message flows.

A number of EAP methods for authentication lend themselves to carry integrity measurement information for mutual platform-authentication. These include – but are not limited to – the TLS-based EAP methods, such as EAP-TLS, EAP-TTLS, PEAP and EAP-FAST. These EAP methods can be enhanced by adding the TLS-Attestations Extensions protocol which would carry endpoint integrity measurement information as part of the authentication handshake and master key establishment (e.g. TKIP key in 802.11).

7.2.2 TLS and HTTPS

In different areas of Internet technology development, the HTTP protocol is viewed by many as being the lowest common denominator for transport of application-related messages. This is particularly true in the area of web services, where various web services protocols (e.g. SOAP, WS*) presume the existence of HTTP (over TCP) as the basic reliable transport mechanism.

The current TNC Architecture anticipates this view of HTTPS, and reflects this through the inclusion in the Architecture (Figure 2) of the peer logical relationship between IMCs and IMVs, and between the TNC Client and TNC Server. This approach allows an implementer of a TNC Client and/or TNC Server to use HTTPS as the transport mechanism to deliver integrity measurement information between the two, independent of the lower communications layers underlying the HTTPS instance. This approach may be of interest to a number of application-layer vendors (e.g. anti-virus) as well as Internet access providers that use the Web-Login page over HTTPS to authenticate users (e.g. WiFi Hotspot providers and WISPs).

7.3 Authentication Server (AS) technologies

The TNC Architecture does not mandate any particular protocol to be used for communication with and within the PDP. However, two of the most widely-supported protocols suitable for this purpose are Radius and Diameter.

7.3.1 Radius

The Radius protocol [RFC2865] has a long history dating to the early developments of dial-up (over PPP) and in many ISPs today Radius remains to be the primary standardized authentication protocol of choice. The TNC Architecture acknowledges the wide-spread deployment of Radius, and anticipates that rich and interesting combinations of Radius with other technologies (e.g. EAP) will ensure a development path forward for many implementers of Radius today.

With the growing popularity of EAP as a way to allow various authentication methods to be used between the Client (EAP-Peer or Supplicant) and Authentication Server (AS), extensions have thus been defined in RFC3579 for Radius itself to support EAP. The aim of the extensions is to use Radius to shuttle Radius-encapsulated EAP packets between the Authenticator (or PEP in the TNC Architecture) and the AS. Two new attributes that were introduced into Radius in RFC3579 to achieve this are the EAP-Message and Message-Authenticator attributes.

Along these lines, one possible addition to Radius could be a new attribute to directly carry integrity measurement information. This would allow for further flexibility of Radius to address cases where EAP is not deployed above (inside) Radius.

7.3.2 Diameter

One of the aims of the development of the Diameter protocol (RFC3588) is to address some of the deficiencies of the Radius protocol, including transport reliability, capabilities negotiation and roaming support. Diameter employs attribute value pairs (AVPs) to carry various payloads relating to user authentication, service authorization, resource usage, and others. The use of AVPs allows the base protocol to be extended for use in new applications through the addition of new AVPs.

In the context of the TNC Architecture, one possible extension could be AVPs to carry directly integrity measurement information from the AR to the PDP.

8 Security Considerations

The current architecture document focuses on aspects of endpoint integrity between an AR and the PDP, containing a TNC Client and TNC Server respectively. There are a number of security aspects pertaining to the architecture as a whole which need to be highlighted, as these are relevant to implementations that seek to be conforming to the architecture and achieving security at the highest levels. These aspects are discussed in the following:

- *Secure Channel between AR and PDP:* In order to communicate integrity values and parameters between the TNC-Client and the TNC-Server, a secure channel must be established for this exchange. One possible location to establish this secure channel is between the NAR (at the AR) and the NAA (at the PDP). This channel must be end-to-end in the sense that the PEP must not gain access to the contents of this secure channel. The exact implementation of this secure channel is dependent on the area of application and network configuration. An example of this channel would be one established through PEAP or TTLS, running over EAP in the context of the 802.1X configuration. Similarly, an IKE Phase-1 SA could be used to negotiate a special Phase-2 SA that then protects the integrity information transfer in the case of a VPN.
- *Authorization for TNC Client/TNC Server and IMC/IMV:* In general, a TNCC (TNCS) should only communicate with authorized IMCs (IMVs). This requirement comes from the need to prevent bogus IMCs (IMVs) from opening communications with valid TNC Clients, thereby opening the possibility of a Denial-of-Service attack (at the very least) against the TNCC/TNCS.
- *Self-integrity of AR and PDP:* Since the integrity values being communicated between two endpoints are only as good as the self-integrity of these respective endpoints, it is paramount that both the AR and PDP are protected against attacks that illegally modify the system configurations of these entities. This need is particularly acute in the case of platforms without a TPM.
- *Security of Remediation Solutions:* In the event that remediation of a AR requires that AR to communicate with a Remediation Server (RS) and obtain integrity-related updates, it is important to consider the security of the RS. If signed updates with careful versioning are placed on the RS, some protection against RS compromise can be achieved. However, strong protection for the RS should be employed.
- *Protection of Information Assets across Interfaces:* It is important that implementation of the TNC Architecture protect information assets as these traverse the various interfaces defined in the architecture. These information assets include state change notifications (between TNCC and IMV, and between TNCS and IMC), message exchanges between entities, vendor specific messages (exchanged between the IMC and IMV as peers) and remediation results (from TNCC to the IMV).
- *Protection of interfaces from threats:* The ability of a TNCC on a platform to discover the IMCs on that platform has benefits as well as security risks. Thus, a TNCC must have sufficient privileges (set by the Administrator according to policy) in order to access information regarding available IMCs on the same platform. The design and implementation of interfaces must therefore prevent against spoofing (by a rogue IMC/IMV), against denial of service (provided by a legitimate IMC/IMV and against illegal tampering (IMC/IMV parameters modified).

9 Privacy Considerations

Privacy is an important issue in the context of trusted network connections. Some aspects that are pertinent to the TNC are as follows:

- *Anonymous access is supported:* User authentication (of the client system to the server system) is not required in order to perform an integrity measurement handshake. In scenarios which require protection of the user identity, anonymous network access is supported by this architecture.
- *Owner controlled policy:* The architecture allows for negotiation of which measurements may be needed to make access decisions. The platform owner is presumed to have control over the privacy policy and privacy related negotiations. In other words, measurements can be more specific than what is requested and client policies can dictate when it is desirable to abort the connection request in the interest of preserving privacy.
- *Disclosure control mechanisms.* The architecture does not prevent IMCs from implementing a disclosure control mechanism driven by privacy policy. IMC implementers may employ filtering on outbound flows to block, replace, modify or un-sign integrity reports. The IMC interface specification does not specify the content of messages exchanged between IMC and IMV, hence the TNCC does not appear to be an appropriate place to apply privacy controls. However, vendor specific extensions to IMCs appear reasonable.
- *IMC selection.* The user may determine which IMCs can be installed and/or loaded by the TNCC based on an assessment of the IMC ability to protect privacy.
- *Encryption of sensitive information:* If a client does choose to provide specific measurement information in order to gain network access, that information can be encrypted once it leaves the client in order to provide protection of the sensitive measurement data and prevent disclosure to unauthorized parties.

10 References

- [1] Trusted Computing Group, *TCG 1.1b Specification Architecture Overview*, Revision 0.14, March 2004.
- [2] Trusted Computing Group, *IWG Reference Architecture for Interoperability*, Specification Version 1.0, Revision 0.86, April 2005, Work in Progress.
- [3] Trusted Computing Group, *IF-IMC Interface Specification*, v.1.0, May 2005.
- [4] Trusted Computing Group, *IF-IMV Interface Specification*, v.1.0, May 2005.
- [5] Trusted Computing Group, *Infrastructure Use Cases*, Specification Version 1.0, Revision 0.25, February 2004, Work in Progress.
- [6] Trusted Computing Group, *Credential Profile for v1.1b*, Specification Version 1.0, Revision 0.96, April 2005, Work in Progress.
- [7] Trusted Computing Group, *Integrity Information Schema*, Specification Version 1.0, Revision 0.7, April 2005, Work in Progress.
- [8] Trusted Computing Group, *TLS Extensions for Attestation*, Specification Version 1.0, Revision 0.8, July 2004, Work in Progress.
- [9] Trusted Computing Group, *TPM Specifications v1.2*, October 2003.
- [10] Trusted Computing Group, *TSS Specifications v1.1*, August 2003.
- [11] Trusted Computing Group, *TCG Glossary*, June 2004. Work in Progress.
- [12] C. de Laat, G. Gross, L. Gommans, J. Vollbrecht, D. Spence, *Generic AAA Architecture*, RFC 2903, Experimental, August 2000, IETF.
- [13] J. Vollbrecht, et al., *AAA Authorization Framework*, RFC 2904, Informational, August 2000, IETF.
- [14] J. Vollbrecht et al., *AAA Authorization Application Examples*, RFC 2905, Informational, August 2000, IETF.
- [15] S. Farrell et al, *AAA Authorization Requirements*, RFC 2906, Informational, August 2000.
- [16] B. Aboba et al., *Criteria for Evaluating AAA Protocols for Network Access*, RFC 2989, Informational, November 2000, IETF.
- [17] R. Yavatkar, D. Pendarakis, R. Guerin, *A Framework for Policy-based Admission Control*, RFC 2753, January 2000, IETF.
- [18] IEEE802, *Port-Based Network Access Control*, IEEE Std 802.1X-2001, June 2001, Institute of Electrical and Electronics Engineers.
- [19] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, H. Levkowitz, *PPP Extensible Authentication Protocol (EAP)*, RFC3784, Standards Track, June 2004, IETF.
- [20] P. Funk, S. Blake-Wilson, N. Smith and H. Tschofenig, *TLS Inner Application Extension (TLS/IA)*, draft-funk-tls-inner-application-extension-00.txt, October 2004, IETF, Work in Progress.
- [21] D. Harkins, D. Carrel, *The Internet Key Exchange (IKE)*, RFC 2409, November 1998, IETF.
- [22] C. Kaufman, *Internet Key Exchange (IKEv2) Protocol*, Internet Draft <http://www.ietf.org/internet-drafts/draft-ietf-ipsec-ikev2-17.txt>, September 2004, IETF.

11 TNC Glossary

When used in TNC documents, the following terms are defined as below:

Term	Definition
Access Requestor (AR)	Within the TNC framework for endpoint integrity, the Access Requestor is the entity seeking connectivity to network that implements the TNC Architecture. The AR consists of three main components, namely the NAR, TNC Client and the IMC. See glossary for the definition of these components.
Endpoint Integrity Information	This is information provided by IMCs describing the status and configuration of the AR.
Endpoint Policy Compliance	The actions towards establishing a level of 'trust' in the state of an endpoint, such as ensuring the presence, status, and upgrade level of mandated applications, revisions of signature libraries for anti-virus and intrusion detection and prevention system applications, and the patch level of the endpoint's operating system and applications.
IMV Action Recommendation	Refers to the recommendation given by each IMV to the TNCS as to what type of network access or isolation action should be taken based on the IMV's evaluation. Example IMV Action Recommendations include: recommend full (normal) access; recommend isolation (limited or quarantined access); and recommend denial (no access).
IMV Evaluation Result	Refers to the result returned by each IMV to the TNCS regarding the state of the endpoint's compliance, based on the IMV's evaluation. Example IMV Evaluation Results include: endpoint is compliant with policy; endpoint is non-compliant and non-compliance is minor; endpoint is non-compliant and non-compliance is major; compliance is unknown.
Integrity Check Handshake	A handshake between a TNCC and a TNCS during which the integrity of an Access Requestor is checked against policy to determine whether the Access Requestor should be given network access. This is an example of attestation protocol in the context of TNC.
Integrity Measurement Collector (IMC)	An IMC is a software component that runs on an Access Requestor (AR), measuring certain aspects of the AR's integrity, including software versions, patches, Anti-Virus and others. An IMC may use the TCG Platform Trust Service (PTS) to obtain integrity information regarding every component of the platform on which the IMC sits. Multiple IMCs may reside on a single AR.
Integrity Measurement Verifier (IMV)	An Integrity Measurement Verifier (IMV) is the component of the PDP that verifies a particular aspect of the AR's integrity, based on measurements received from an IMCs and/or other data. Multiple IMVs may reside on a single PDP.
Isolation	The action of separating a TNC Client onto a separate network – virtual or physical – possibly, though not necessarily, for the purposes of performing Remediation on that Client.
Network Access Authority (NAA)	The Network Access Authority (NAA) is the network layer component of the PDP that decides whether a Network Access Requestor (NAR) should be granted access to a network.

Network Access Requestor (NAR)	The Network Access Requestor (NAR) is the component of the Access Requestor (AR) responsible for negotiating and establishing network access onto a given network. The NAR is expected to implement network layer protocols, covering security, message transport and others. In the context of 802.1X, the NAR can be identified as the Supplicant.
Platform Authentication	The act of verifying both the proof-of-identity and integrity-status of a given platform.
Platform Trust Services (PTS)	The Platform Trust Services (PTS) is a system service that exposes trusted platform capabilities to TNC components that reside on a Trusted Platform containing a TPM. PTS services include protected key storage, asymmetric cryptography, random numbers, platform identity, platform configuration reporting and integrity state tracking.
Policy Decision Point (PDP)	The PDP is an entity evaluating the status of a TNC Client (seeking network connectivity) and deciding upon some network-related action to be enforced by the PEP. The PDP embodies the security and integrity related policies governing the network.
Policy Enforcement Point (PEP)	The PEP is a component within the TNC Architecture that controls access to a protected network, whose policies are implemented through a Policy Decision Point (PDP). The PEP enforces the decision of the PDP.
Remediation	The action of updating an entity seeking network connectivity (who fails integrity check) with the necessary software, firmware and integrity-related parameters updates.
TNC Server (TNCS)	The TNCS is the component on the PDP that manages the flow of messages between Integrity Measurement Collectors (IMC) and Integrity Measurement Verifiers (IMV), gathers recommendations from IMVs, and combines those recommendations (based on policy) into an overall TNCS Action Recommendation to the NAA.
TNC Client (TNCC)	The TNCC is the software component on the Access Requestor (AR) that aggregates integrity measurements (from IMCs), assists the management of the Integrity Check Handshakes and assists in the measurement and reporting of platform and IMC integrity.
TNCS Action Recommendation	Refers to the final, combined recommendation given by the TNCS to the NAA. Phase I of the specification does not mandate values for this recommendation, however, example action recommendations are expected to include: recommend full (normal) access; recommend isolation (limited or quarantined access); and recommend denial (no access).
Integrity Information	The set of platform specific information that make-up a Trusted Platform. These range from information about a platform's hardware components, TPM information (e.g. versions), PCRs, peripherals, Trusted Building Blocks, OS/Kernel, drivers, Applications, Anti-Virus information and others. Each specific use-case determines which information set will be of interest. As such, it is expected that for a given situation these will be pre-determined or pre-configured by an authorized entity (e.g. IT administrator).
Integrity Measurements	The informational output from applying an Integrity Measurement process.

