TCG Trusted Network Communications TNC Architecture for Interoperability

Specification Version 1.5
Revision 4
7 May 2012
Published

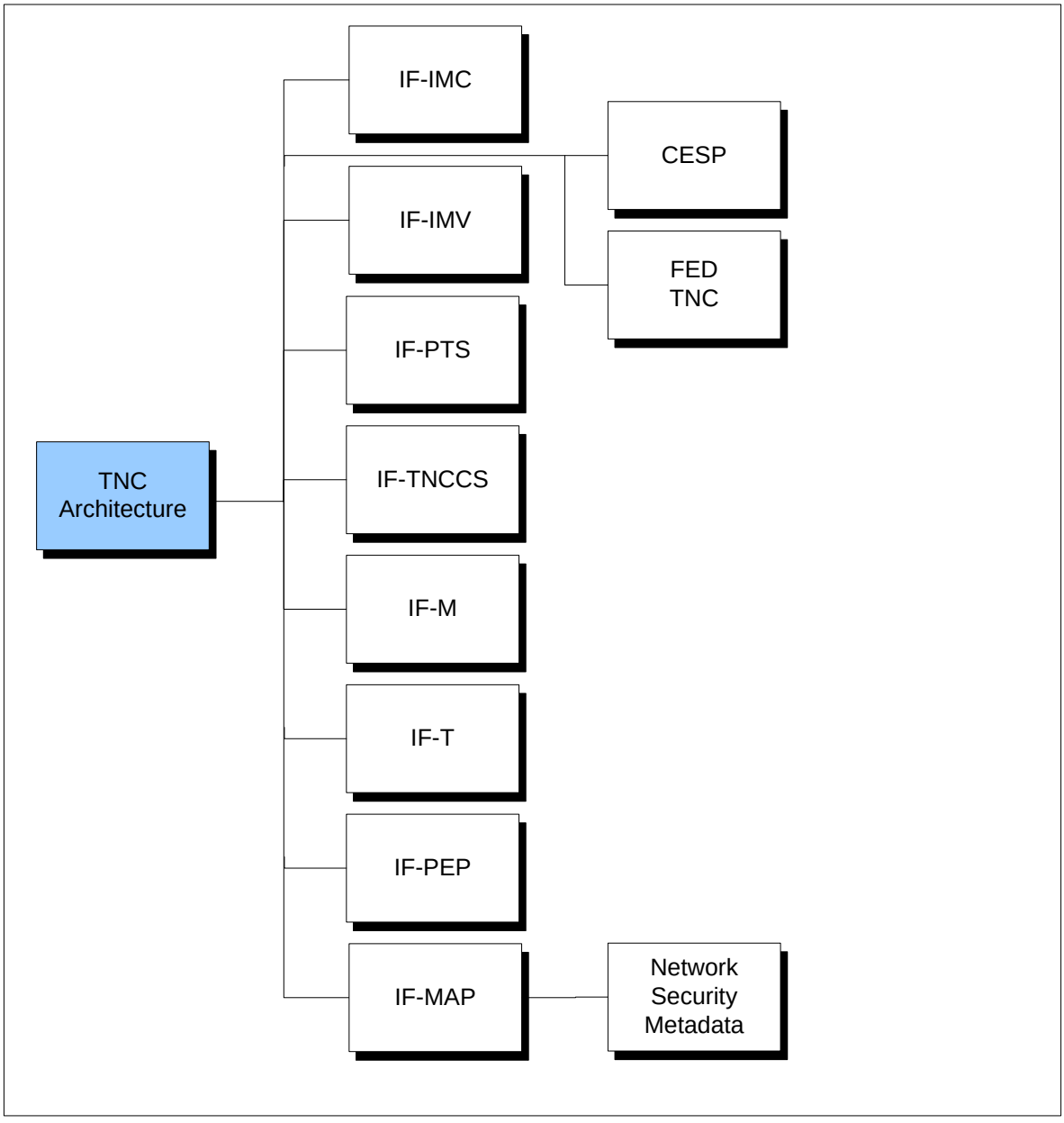
Contact:
admin@trustedcomputinggroup.org

TCG

TCG PUBLISHED

Disclaimer

IWG TNC Document Roadmap



Acknowledgement

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

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

Scott Kelly	**Aruba Networks**
Amit Agarwal	**Avaya**
Mahalingam Mani	**Avaya**
Jeffery Dion	**Boeing**
Steven Venema	**Boeing**
Peter Wrobel	**CESG**
Mark Townsend	**Enterasys**
Michael McDaniels	**Extreme Networks**
Hidenobu Ito	**Fujitsu Limited**
Seigo Kotani	**Fujitsu Limited**
Houcheng Lee	**Fujitsu Limited**
Sung Lee	**Fujitsu Limited**
Graeme Proudler	**Hewlett-Packard**
Mauricio Sanchez	**Hewlett-Packard**
Ren Lanfang	**Huawei Technologies**
Jiwei Wei	**Huawei Technologies**
Han Yin	**Huawei Technologies**
Diana Arroyo	**IBM**
Guha Prasad Venkataraman	**IBM**
Sean Convery	**Identity Engines**
Chris Hessing	**Identity Engines**
Morteza Ansari	**Infoblox**
Stuart Bailey	**Infoblox**
Ivan Pulleyn	**Infoblox**
Ravi Sahita	**Intel Corporation**
Ned Smith	**Intel Corporation**
Josh Howlett	**JANET (UK)**
Yan Avlasov	**Juniper Networks**
Roger Chickering	**Juniper Networks**
Charles Goldberg	**Juniper Networks**
Steve Hanna (Editor, TNC co-chair)	**Juniper Networks**
PJ Kirner	**Juniper Networks**
Lisa Lorenzin (Editor)	**Juniper Networks**
John Jerrim	**Lancope**
Tom Price	**Lumeta**
Matt Webster	**Lumeta**
Ryan Hurst	**Microsoft**
Sandilya Garimella	**Motorola**
Meenakshi Kaushik	**Nortel**
Paul Sangster (TNC co-chair)	**Symantec Corporation**
Brad Upson	**UNH InterOperability Lab**
Lauren Giroux	**US National Security Agency**
Chris Salter	**US National Security Agency**

Thomas Hardjono (Editor)	**Wave Systems**
Greg Kazmierczak	**Wave Systems**

Table of Contents

1 Scope and Audience

The Trusted Network Communications Work Group (TNC-WG) 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 platforms may find this document helpful in understanding the architecture defined by the TNC-WG. The TNC approach enables more transparency into the trust decision made by a trusted platform as it allows inspection within the systems boundary.

The document is intended to be a guide and orienting document with respect to the body of TNC specifications and is not intended to provide normative requirements.

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 Communications (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 [19] 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 information pertaining to the Platform-Authentication exchange [2].

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 (or “Platform Credential Authentication”), while the second aspect is the *integrity verification* (or “Integrity Check Handshake”) 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 a TPM there are an unlimited number of identities that can be deployed to effect privacy of the user on the platform. Note that claimed identity in a platform may or may not be related to the user or any actions performed by the user (see [3]).

In the remainder of this document, the term “Platform-Authentication” therefore should generally be understood as consisting of both aspects, namely establishing proof of identity (e.g. via AIK-certificates) 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 in addition to the usual user authentication. Machine level Platform-Authentication is crucial for the security and authorization of network-access requests at both layer-2 and layer-3. Furthermore, due to the large number of attacks from malware (worms, viruses, spyware) and alike against higher layers of the network stack, network operators need the ability to evaluate the security posture (defensive measures) against such threats prior to allowing access.

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 others 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, elements 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 architecture 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 (Platform Credential Verification) and the integrity verification (Integrity Check Handshake) 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 software version of mandated applications, completeness of virus-signature databases, 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 the sense that endpoint integrity checking is used as input to the authorization decision for gaining access to the network.
- *Access Policy*: ensuring that the endpoint machine and/or its user authenticates and discloses their security posture before connecting to the network, leveraging a number of existing and emerging standards, products, or techniques.
- *Assessment, Isolation and Remediation*: ensuring that systems requesting network access that do not meet the security policy requirements for endpoint compliance (or security posture) 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 database 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 security posture of the platform and may include services such as anti-virus scanners, 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 primary roles in the TNC architecture are the Access Requestor (AR), the Policy Enforcement Point (PEP), the Policy Decision Point (PDP), the Metadata Access Point (MAP), and the MAP Client. The AR requests access to a protected network. The PDP compares the AR's credentials (e.g. user certificates, password, etc) and information about its security posture against certain network access policies, and then decides whether network access should be granted to the AR. If a PEP is present, the PDP then communicates its decision to the PEP, which actually grants or denies access (i.e. enforces access control). Optionally, MAP Clients, which might not be directly involved with the decision to grant network access, may coordinate with both the PDP and the PEP in monitoring and enforcing network security policy compliance by receiving and sharing information through the MAP.

Although this concept is common today in many networks, the TCG seeks to add *Platform Credential Authentication* (using the TPM-related certificates) and *Integrity Verification Handshake* (using the registers within the TPM) to network access decision by the PDP. Together, we refer to these two aspects as Platform-Authentication.

In this section we describe how the TNC Architecture relates to the broader IWG Architecture [2] and the IETF AAA Architecture [14][18]. This is followed by a detailed discussion of the TNC Architecture, the roles in the architecture, the functions within those roles, 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.

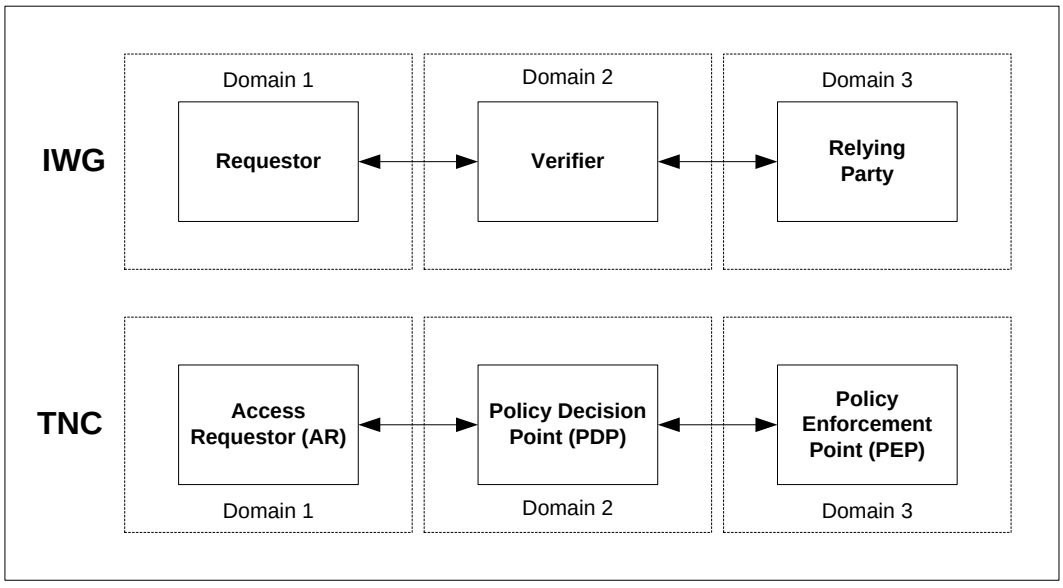


Figure 1: Basic Model underlying the IWG and TNC Architectures

In the IWG architecture when responding to a request from a *Requestor* element, 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) [19].

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 element that is seeking network connectivity to a given network. The Verifier is referred to as the *Policy Decision Point* (PDP), since that element 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 (customized for the network environment) and based on input from various sources of integrity information. The element 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) [18].

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 hardware-rooted feature that distinguishes the IWG and TNC architectures from other architectures.

3.2 Relationship with the AAA Architecture in the IETF

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 Requestor'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 level, Anti-Virus signatures, etc).

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 endpoint integrity 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 [16][17] and a AAA Authorization Framework [14] in the AAAARCH-RG Research Group [13], efforts in the AAA Working Group focusing on RADIUS, Diameter, the NAI and Network Access [15], as well as efforts in the Policy Framework Working Group.

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 roles, functions, and interfaces which are discussed below.

The five columns in this figure depict the five *roles* in the TNC architecture: the Access Requestor (AR), the Policy Enforcement Point (PEP), the Policy Decision Point (PDP), the Metadata Access

Point (MAP), and the MAP Client. Within each role (column), the boxes depict the *functions* within those roles. Three horizontal shaded *layers* are depicted grouping the functions, while the *interfaces* that will be standardized by the TNC are depicted by named lines. These layers, roles, functions, and interfaces are described below.

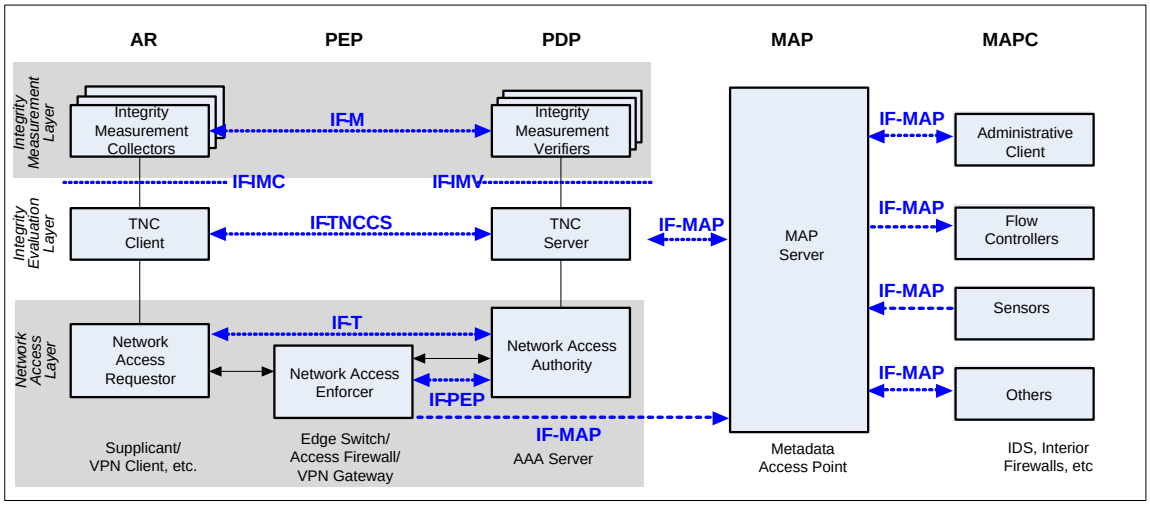


Figure 2: The TNC Architecture

It is important to note that Figure 2 shows the functions (of each role) that pertain to integrity verification and network security 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 an additional 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 functions of each role in the TNC Architecture are not the only components implementing security and network connection management.

Additionally, a single physical element in a network environment may play more than one role in the TNC Architecture. For example, a switch or wireless access point configured to authenticate endpoints with 802.1X supplicants via 802.1X, but assign a default access policy (e.g. guest VLAN) to non-supplicant endpoints, fulfills the role of both the PEP and PDP; such a network device is referred to as a combined PEP/PDP. Another example is a policy server that both provisions access control policy to a PEP and subscribes to information from a MAP; that policy server is both a PDP and a MAPC.

3.4 Roles

The required roles within the TNC Architecture are the Access Requestor (AR) and the Policy Decision Point (PDP). The optional roles are the Policy Enforcement Point (PEP), the Metadata Access Point (MAP), and the MAP Client (MAPC).

All roles and functions in the architecture are logical ones, not physical ones. The element performing a particular role or component providing a particular function 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.

3.4.1 Required Roles

- *Access Requestor* (AR): The role of the AR is to seek access to a protected network in order to conduct activities on the network.
- *Policy Decision Point* (PDP): The role of the PDP is to perform the decision-making regarding the AR's network access request, in light of the access policies.

3.4.2 Optional Roles

- *Policy Enforcement Point* (PEP): The PEP is the element which is connected to the AR; the role of the PEP is to enforce the decisions of the PDP regarding network access. Use cases which do not require the PEP include those which conduct network compliance monitoring, suggest remediation recommendations, and exclude direct enforcement.
- *Metadata Access Point* (MAP): The role of the MAP is to store and provide state information about ARs which may be useful to policy decision making and enforcement. This information includes, but is not limited to, device bindings, user bindings, registered address bindings, authentication status, endpoint policy compliance status, endpoint behavior, and authorization status.
- *MAP Client (MAPC)*: The role of the MAP Client is to publish to, or consume from, the MAP state information about ARs. A MAP Client may both publish and consume state information, and might not be directly connected to the AR.

3.5 Layers

Three (3) abstract layers of the architecture are identified, grouping components providing similar functions:

- *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 functions found in this layer are the Network Access Requestor (NAR), the Network Access Enforcer (NAE) and the Network Access Authority (NAA).
- *The integrity evaluation layer*: The function of the components in this layer is to evaluate the overall integrity of the Access Requestor with respect to certain access policies, with input from the functions at the integrity measurement layer. The functions found in this layer are the TNC Client (TNCC) and the TNC Server (TNCS).
- *The integrity measurement layer*: This layer contains plug-in components whose function is to collect and verify integrity-related information for a variety of security applications on the Access Requestor. The functions found in this layer are the Integrity Measurement Collectors (IMCs) and the Integrity Measurement Verifiers (IMVs).

3.6 Functions

Referring to Figure 2, the functions making up the roles are as follows.

3.6.1 Access Requestor

The Access Requestor consists of the following functions:

- *Network Access Requestor (NAR)*: The NAR is the function 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 function is a software component that runs on an AR, aggregating integrity measurements from IMCs and orchestrating the reporting of local platform and IMC measurements (Integrity Check Handshake). Here, Integrity Check Handshake could be an example of a TCG attestation protocol in the context of the TNC.
- *Integrity Measurement Collector (IMC)*: The IMC function 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 is designed for multiple IMCs to interact with a single (or multiple) TNC Client/TNC Server, thereby allowing customers to deploy complex integrity policies involving a range of vendors products.

3.6.2 Policy Enforcement Point

The PEP consists of the following function:

- *Network Access Enforcer (NAE)*: The NAE function controls access to a protected network. The NAE consults an NAA to determine whether this access should be granted. One example of the NAE 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 functions:

- *Network Access Authority (NAA)*: The NAA function decides whether an Access Requestor (AR) should be granted access. The NAA consults a TNC Server to determine whether the AR's integrity measurements comply with the PDP's security policy. In many cases, an NAA will be included within a AAA Server but this is not required.
- *TNC Server (TNCS)*: The TNCS function 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 function verifies a particular aspect of the AR’s integrity, based on measurements received from IMCs and/or other data.

3.6.4 Metadata Access Point

The MAP consists of the following function:

- *Metadata Access Point Server (MAPS)*: The MAPS function is a component to which other TNC components may publish, subscribe, and search data which reflects the state of TNC elements and aids in decision making and policy enforcement. The MAPS allows components which are not involved with the initial network access process, like Flow Controllers, to enforce policies based on relationships to endpoints, users, capabilities, roles, device activities and postures as well as other run time data. The MAPS allows elements which are not directly connected to an AR, like Sensors, to publish information about network activities which may be of interest to PEPs, PDP, and other MAP Clients.

3.6.5 MAP Client

MAP Clients consist of the following functions:

- *Administrative Client*: The Administrative Client function enables administrative operations such as monitoring, investigation, and provisioning by utilizing information from the MAP and publishing information to the MAP via IF-MAP. Examples of Administrative Clients include data visualizers, configuration management databases (CMDBs), Policy Information Points (PIPs), and provisioning servers. Examples of operational activities enabled include data exploration, asset management, and certificate lifecycle management.

- *Flow Controller*: The Flow Controller function makes and enforces decisions about network activities utilizing information from the MAP. Flow Controllers take action (e.g. block) on network flows (i.e. network traffic associated with a particular AR, device, user, etc.) based on data obtained via IF-MAP. Examples of Flow Controllers include internal firewalls, inline intrusion prevention systems (IPSs), rate limiters, and proxies. Examples of network activities being controlled include accessing particular services in a network, accessing particular geographies in a network, and restricting the amount of bandwidth allowed.
- *Sensor*: The Sensor function monitors network activities and publishes information to the MAP via IF-MAP. Examples of Sensors include intrusion detection devices, network virus detection devices, layer 3 traffic monitors, and application traffic scanners. Examples of network activities being monitored include accessing particular services in a network, authentication activity, broadcast requests for various services (e.g. DHCP), and advertising of services.

3.7 TNC Interfaces

There are a number of interfaces shown in **Figure 2** which define relationships between functions and the protocols and messages exchanged between functions. 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 [10].

Software, firmware and hardware components are expected to report status information to the TNC Client on the AR platform. 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 collects 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 AR is a Trusted Platform with a TPM, the integrity-measurements are also deposited in the AR's Stored Measurement Log [1]. How the measurements were collected on the platform (e.g. 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 deliver 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 [11].

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 batches of integrity measurements)

(b) Messages from IMVs to IMCs (such as requests for additional integrity measurements, or remediation instructions)

(c) Session management messages, as they pertain 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 TNCC or TNCS.

Several alternate protocols for IF-TNCCS have been released: IF-TNCCS-SOH and the previously published XML version of IF-TNCCS. The different feature sets of these protocols are the reason to have these alternate protocols. The reader is directed to [6], [7], and [30] for more details.

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 Communications, but to the larger TCG requirements around platform management. TNC provides IF-M protocol bindings for TLV[31].

3.7.5 Network Authorization Transport Protocol (IF-T)

IF-T pertains to the transportation of messages between the AR element and the PDP element. The functions 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 reader is directed to [7] for more information. TNC provides IF-T protocol bindings for Tunneled EAP Methods[8] and TLS[28].

3.7.6 Platform Trust Services Interface (IF-PTS)

Although not shown in **Figure 2**, there is an additional interface that is under development in the TCG. This is the IF-PTS interface. IF-PTS provides platform trust services to ensure that TNC components are trustworthy. See Section 6 and refer to the IF-PTS Specification[25] for additional details.

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. For more details about IF-PEP, refer to the IF-PEP Specification [9].

3.7.8 Metadata Access Point Interface (IF-MAP)

IF-MAP allows elements in the TNC architecture to share and correlate stateful runtime metadata such as relationships of TNC components to endpoints, users, capabilities, roles, and attributes. IF-MAP provides publish, subscribe, and search interfaces between MAP Clients and the MAP. The data published and available via IF-MAP *augments* other sources of data for security related decision making. Searches and subscriptions using IF-MAP return data which *approximately* reflects recent metadata values and relationships as reported by MAP Clients. For more details about IF-MAP, refer to the IF-MAP Specification [24] and IF-MAP Metadata for Network Security Specification[32].

3.8 TNC Support Profiles

The TNC family of specifications includes support profiles for aspects of network access control which are related to, but do not fall directly under, the TNC Architecture.

3.8.1 Clientless Endpoint Support Profile

In today's environments, many endpoints exist that do not - or cannot - run a TNC Client, and therefore cannot provide integrity information, yet still require access to a protected network. In the TNC approach, an endpoint without a TNC Client is defined as a Clientless Endpoint (CE).

Clientless Endpoints are addressed by the Clientless Endpoint Support Profile (CESP), which outlines an approach and enforcement mechanisms to ensure interoperability and enforce compliance in environments where some endpoints lack a TNC Client. There should be no expectation that the CESP will provide the same level of security provided for endpoints with clients; the goal is to increase the ability of network operators to provide security for environments that contain clientless endpoints. For more details, refer to the CESP Specification[26].

3.9 Federated TNC

TNC standards specify how to assess the security posture of an AR as it connects to the network. This assessment is performed by a TNCS belonging to the same security domain as the AR; there exists a direct trust relationship between the AR and the TNCS.

This trust relationship is sufficient provided that the AR only accesses services within its own security domain. Federated TNC[27] addresses how the endpoint's posture should be assessed by a service within other security domains. This specification defines how an endpoint's posture can be queried and supplied such that a security domain, other than the endpoint's own, can make authorization decisions controlling that endpoint's access to its networks and applications.

3.10 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. Note also that certain transports may impose limits on the number of round trips that may be used. For instance, IF-TNCCS-SOH 1.0 only permits a single round trip.)

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 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).

G5 *Ability to share information*: allow the TNC elements to share information observed on the network so it can be factored into various security decisions.

G6 *Common security metadata schema*: Since the coordination of distributed network security components is a consideration in the TNC design, the meaningful metadata communicated

between the TNC components will be structured according to a common TNC Metadata Scheme.

A1 *Protection and reliability of message transport*: Since the integrity measurements data communicated between a TNC Client and TNC Server and metadata communicated between TNC components are not self-protecting, it is assumed that an underlying mechanism will provide for the protection of the data as it is delivered.

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.

3.11 Basic Message Flows across Interfaces for Network Access

There are a number of fundamental message types that are exchanged between components in the architecture, across the various interfaces defined above. The basic messages required for granting access to the network are summarized in Figure 3 and are described in the following. Note that in the following illustration, several levels of 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 Credential Authentication and Integrity Check Handshake. Note, however, that in other situations this may not necessarily be the order of processing. Detailed examples of metadata sharing messages which may augment the process below and provide other coordination between TNC components can be found in [24].

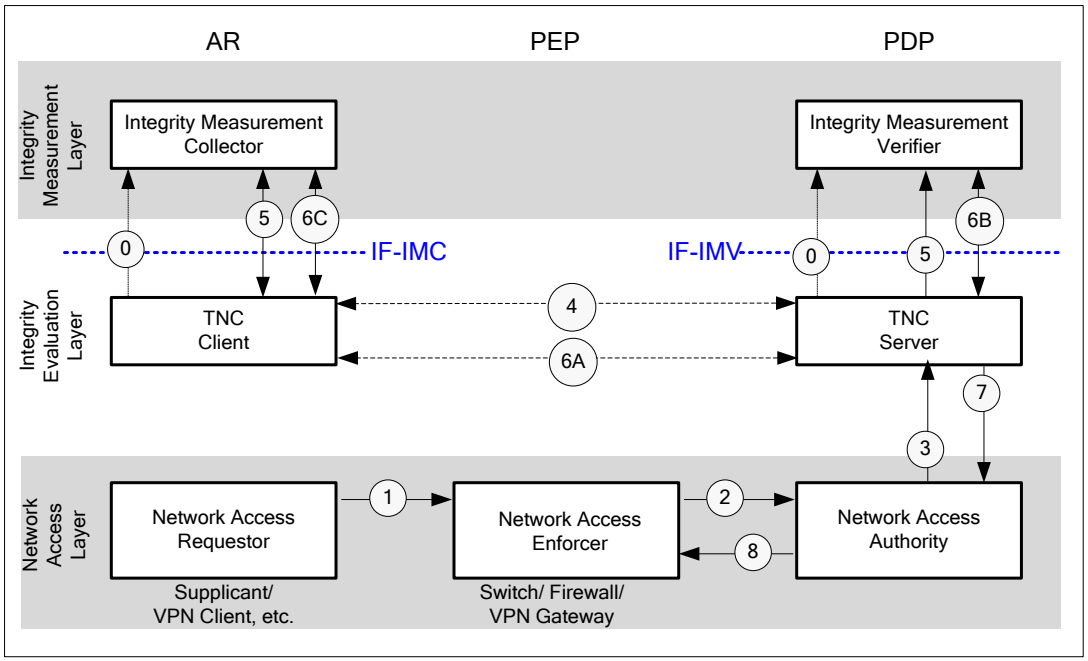


Figure 3: Message Flow between access granting components in the TNC-Architecture

- **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 operations). 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 link and network layers.
- **Flow 2:** Upon receiving a network connection request (from the NAR), the NAE sends a network access decision request to the NAA. Here, the NAA is assumed to have been configured to perform User Authentication, Platform Credential Authentication and Integrity Check Handshake.

  User authentication can occur between the NAA and the AR. Platform Credential Authentication and Integrity Check Handshake may have occurred 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 Credential Authentication and Integrity Check Handshake 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 Credential Authentication with the TNCC, verifying, for example, that valid (un-revoked) AIK-credentials are used by both endpoints.
- **Flow 5:** Assuming that Platform Credential 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, NAE 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 NAE to enforce. The NAA must also indicate its final decision to the TNCS which will be sent to the TNCC. Typically, the NAE indicates its execution of the decision (e.g. Port open in 802.1X) to the NAR.

The above represents the basic behavior of elements 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 topology 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 decisions 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 functions. Similarly, this architecture does not preclude a solution whereby a vendor provides a single product that encompasses the TNCS and IMV functions, 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 functions (e.g. IMV) to be implemented as separate back-end components.

The remainder of this document will primarily focus on the network access process separate from considerations of metadata exchange and coordinated network security enabled by IF-MAP. Design, architecture, and security considerations which result from IF-MAP can be found in [24]. Future versions of this document are expected to have more detail regarding the network security enabled by IF-MAP.

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 functions 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, the TNCC and the TNCS each generate a unique connection ID that is made available to the local 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 are 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 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 (IF-IMV) which define the communications and interactions between the TNCC (TNCS) and IMC (IMV). These are summarized in the following, with further detailed information found in [10] and [11].

- *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 status reporting), 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 is 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.

Normally a single TNCC initiates the TNCCS session to the TNCS. The NAR knows ahead of time which TNCC should receive response messages from the TNCS on that session. However the NAA would need to be able to route the inbound messages to the appropriate TNCS should multiple be available. Currently the TNC messages do not include an indication of the desired end TNCS, so the NAA needs a mechanism for making this message delivery decision when multiple TNCS exist. NAA implementations SHOULD allow for policy to be specified at run-time to define which TNCS should receive particular message types. This policy might be done via a set of configuration setting or be dynamic as TNCC startup and register with the NAR.

In the reverse direction, the TNCS currently doesn't initiate new sessions to the TNCC. However this likely will become necessary in order for the TNCS to re-verify the TNC endpoint is still in compliance with policy (e.g. when the network policy changes.) TNCC/NAR implementers are encouraged to support run-time policy enabling the NAR to make decisions about the destination TNCC when an inbound connection request from the NAA occurs.

- *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 in general (e.g. 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 elements implementing the TNC Architecture, it may not be possible for a TNCC or TNCS to restart an Integrity Check 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.

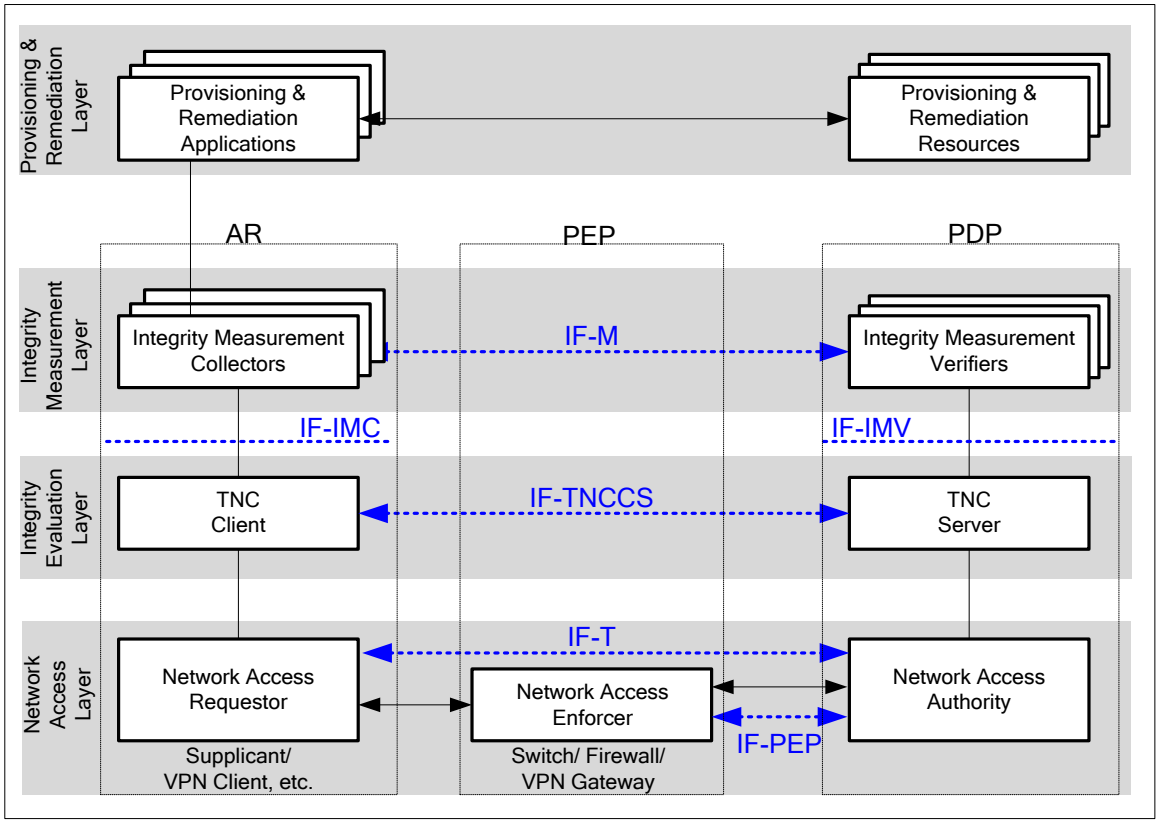


Figure 4: The Provisioning and Remediation Layer in the TNC Architecture

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 if necessary delivers 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 perform 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.

At this point, it is important to note 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 to a separate network – referred to here as the Isolation Network – in order to provide remediation services to the AR. 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. 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 [21].

(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 (see RFC2865 and RFC3580) [21].

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, elements that 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 platform owner's 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 elements 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). 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* (Platform Credential Authentication) and *platform integrity information* (Integrity Check Handshake) as part of an authentication event to an authentication server (PDP or Verifier). 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 (e.g. unavailable to malware code running on the CPU) to operate on sensitive data. The TPM implements protected capabilities and shielded locations. Among others, it is used to protect and report aggregations of integrity measurements that are stored inside the TPM's *Platform Configuration Registers* (PCRs). The TPM also stores cryptographic keys used to authenticate reported measurements. Depending on the platform and its implementation, TPM protected capabilities can include additional security functionality such as cryptographic key management, random number generation, sealing data to system state, and monotonic counters.
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. stored measurement log). The report is signed using the private key held (e.g. AIK-certificate) located in shielded locations in the TPM. Integrity measurement is a trusted function which "measures" (e.g. computes hash) of components of the platform that are measurable (e.g. software, configurations etc.). The result is placed into an integrity measurement log. A digest of the measurements is then added to PCRs in the TPM so any tampering with the log can be detected. Reporting involves sending portions of the integrity measurement log to other parties (e.g. Verifier) along with a signed set of PCRs which the other party can use to validate the logs contents prior to making trust decisions.
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 element.

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 TCG approach can provide:

- *Evaluation and Decision Making*: Following the TCG authentication model in [1], when a requestor platform issues a request (e.g. to resources) to a relying party, that relying party needs to make a trust decision about the requesting system's platform. The TCG model allows the relying party to evaluate the integrity measurements discussed above during this decision. Some relying parties may wish to delegate this evaluation to a 3rd party and merely review the results when making the decision. 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.
- *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.

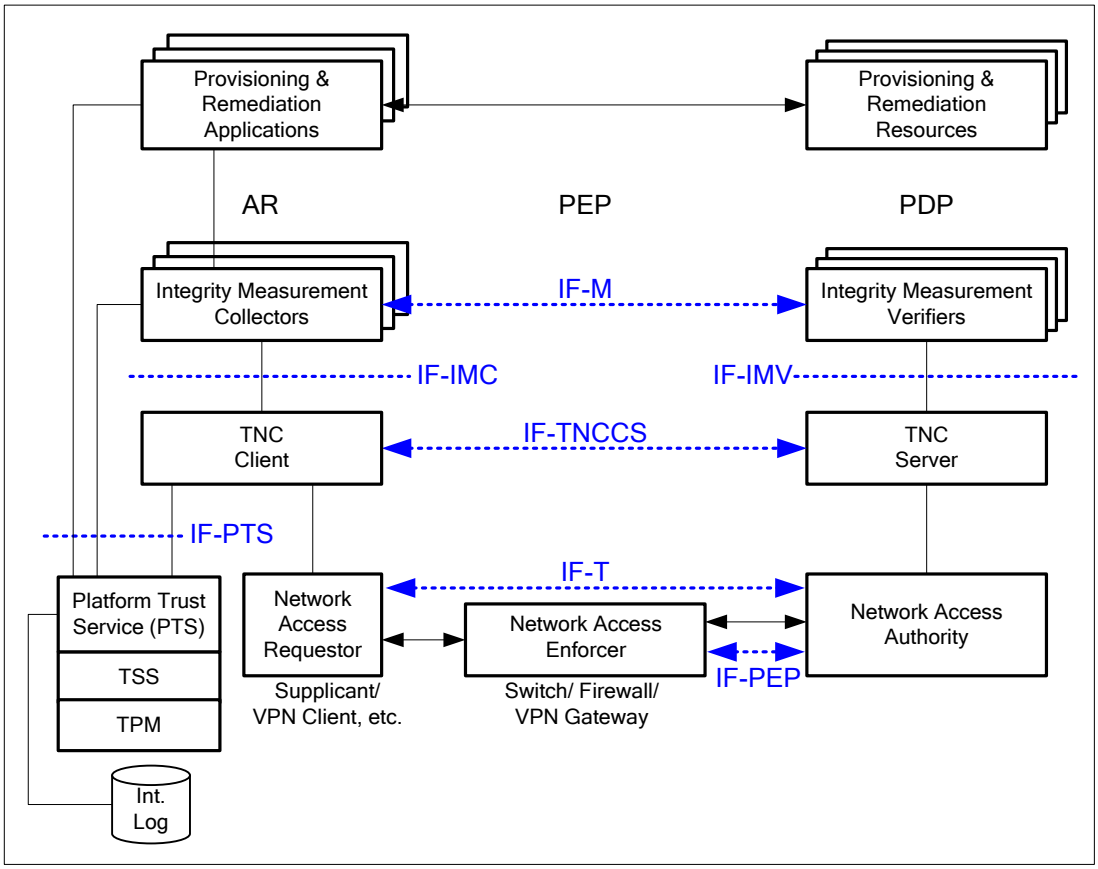


Figure 5: The TNC Architecture with the Trusted Platform Module (TPM)

6.2 Roles

The roles in TNC Architecture do not change with the introduction of trusted platforms (Figure 5). However, the concept of platform ownership and the *owner* role should be considered. The TNC architecture identifies the three roles of 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 the AR and PEP/PDP owners are different, Platform-Authentication and remote attestation requires both parties to trust a common element called the Privacy Certificate Authority (Privacy-CA OR Platform-CA) who issues AIK-certificates to trusted platforms. In particular, the Privacy-CA is the element that seeks the AR's EK-certificate and in-turn issues an AIK-certificate for the AR's trusted platform. As such, when the AR uses the AIK-certificate within a Platform-Authentication event, the PDP as the verifier needs to trust the same Privacy-CA and accept the AIK-certificate issued by that Privacy-CA. The components, protocols and interfaces described below support interactions between these elements.

6.3 Functions

In addition to previously described TNC functions, the TNC architecture includes additional functions when a TCG trusted platform makes up the host environment. The additional functions 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 [5] is a middleware stack that enables applications to use higher level interfaces for communication with the 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 hardware component implements protected capabilities, shielded locations, and other functions as described above [4].

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 measurement logs and ensures the logs accurately reflect the Platform Configuration Register (PCR) state. In addition to pre-boot and OS integrity state, the PTS 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. Therefore TNC components should use the PTS for reporting entries in the Integrity Management Log.

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. Updates to the Integrity Measurement Log (IML) files are 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.

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 PTS is designed in such a way that it can be suitably deployed with tunneling protocols (e.g. within EAP), making use of 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 Integrity 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 the 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 certain classes of MITM attacks (where a local redirector is involved).

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 use of the TPM PCRs to validate the Integrity measurement Log prevents a system from lying about what the platform is running so others can determine if the endpoint has the desirable integrity. 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 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.

6.5 TNC and the TCG Integrity Management Model

The current TNC architecture accommodates platforms that possess a TPM and makes extensive use of the TPM as the hardware root of trust. Among others, this allows a PDP to gain some assurance that information regarding the AR platform-state reported to by the PTS (on the AR) is rooted in trust that is based on cryptographic information that is bound to the TPM hardware.

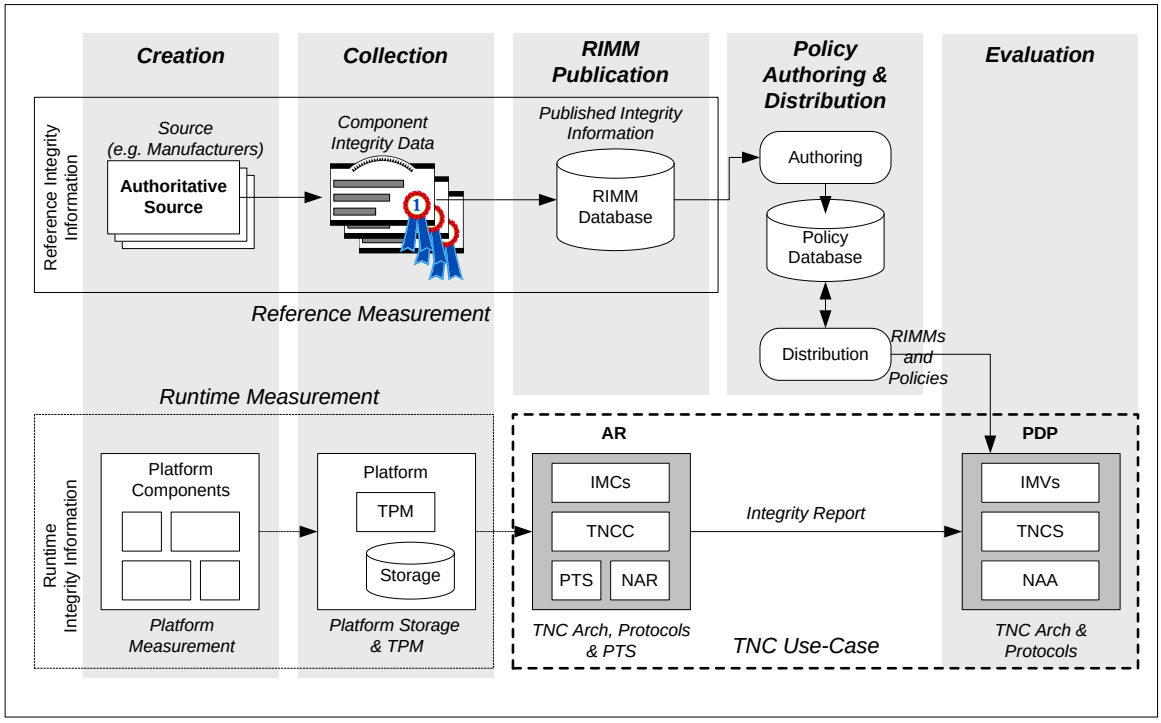


Figure 6: The TNC Architecture within the TCG Integrity Management Model

Although the TPM hardware itself provides a strong anchor of trust, another important dimension of trusted computing concerns the platform-state information that is being reported by the PTS in the AR to the PDP. That is, there is the aspect of *how* the platform-state information is being reported to (i.e. protocols, methods) and there is the aspect of *what* platform-state information is being reported.

To that extent the TCG has developed an *Integrity Management Model* (IMM). Among others the purpose of the IMM is to define the lifecycle of platform-related information (e.g. component manufacturer, model, etc) and define how this information affects the levels of trust accorded to components within a platform and thus to the platform as a whole.

The relationship of the TNC architecture and protocols within the TCG Integrity Management Model (IMM) is shown in Figure 6. Here, integrity management consists of five broad phases that are divided across two kinds of activities. The first set of activities – Reference Measurement – refers to the collection of (static) integrity information and data pertaining to components that make-up a platform. These measurements are likely to come from the manufacturer or developer

of the component so their customers can recognize a valid instance of the component at run-time. This set of activities can be considered to be “out-of-band” from the perspective of a given use-case, such as trusted network connections since they occur prior to the actual usage scenario that makes use of the integrity information.

The second set of activities – Runtime Measurement – pertains to the actual use of the integrity information within a given Platform-Authentication event, in which the integrity of the components of the platform (e.g. AR) are measured and stored inside the Integrity Measurement Log (IML) of the platform and later used within a Platform-Authentication exchange, namely the TNC Integrity Check Handshake.

The TNC architecture and protocols play a crucial role in IMM as it represents a Platform-Authentication use-case (in the context of network access control) which makes use or consumes the integrity information collected and processed by the various phases of the IMM. More specifically, in Figure 6 the result of the runtime measurement of the AR platform is communicated to the PDP (as the Evaluator) as part of the network access request of the AR. The specific term used in this case is *integrity report* which represents the set of component integrity information about an AR which is communicated by the AR to the PDP within a Platform-Authentication event.

The PDP itself uses the policies inside the Policy Database (see Figure 6) pertaining to the AR as part of its decision-making regarding the AR. It is important to note that besides traditional information within the Policy Database (e.g. user ID, ACL, etc.) the Policy Database contains additional information pertaining to the components of the AR platform. More specifically, the Policy Database contains *Reference Integrity Measurement Manifest* (RIMM) records which denote the expected (good) reference value for each component of the AR platform. Using the RIMM information, the PDP is thus able to compare the reported component integrity information (in the Integrity Report communicated from the AR) against a good benchmark or reference value as found in the Policy Database.

The RIMM information represents the end-product of the Reference Measurement phase. Among others, the RIMM contains integrity information from the manufacturer or vendor of the component which is source-authentic and which has been canonicalized according the TCG Core Integrity Scheme standard. The evaluator of a RIMM records will thus be able to verify the creator of the RIMM.

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 [19] 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

Today, remote access based on VPNs have 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 [22] 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 [23] 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 SSL 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 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) [20] – 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.

In the context of the TNC Architecture and specifications for interfaces among the architecture's components, the use of HTTP and HTTPS will be situation dependent. Since the messaging semantics of HTTP and HTTPS is limited, implementers of the TNC Architecture and components may need to add additional logic beyond those inherent within HTTP and HTTPS.

Note that the TCG notions of Platform-Authentication itself is network-independent in the sense that an integrity report can be delivered from a requestor to a verifier across various transport mechanisms. Thus, the TLS protocol can be extended (i.e. using its extensions feature) to carry integrity report (unidirectional) as well a multi-message Integrity Check Handshake.

This approach may be of interest to a number of web-services providers, application-layer vendors 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 PDP 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 [21] 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 AR (i.e. client, EAP-Peer or Supplicant) and PDP (Authentication Server), extensions have thus been defined in RFC3579[29] for RADIUS itself to support EAP. The aim of the extensions is to use RADIUS to shuttle RADIUS-encapsulated EAP packets between the AR (or PEP in the TNC Architecture) and the PDP. 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. It also encompasses an optional MAP, which can interface the TNC Server with Sensors and Flow Controllers. There are a number of security aspects pertaining to the architecture as a whole that 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 NAE 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 elements. This need is particularly acute in the case of platforms without a TPM.
- *Secure Channel between MAP and MAP Clients*: Metadata communicated between MAP Clients and the MAP is security sensitive. The confidentiality and integrity of this data must be preserved. Therefore, communication between the MAP and the MAP Clients is required to be secure.
- *Self-integrity of MAP and MAP Clients*: Compromise of the MAP or MAP Clients could lead to corruption of the MAP database. This could lead to network access being denied or allowed improperly. Therefore, the integrity of the MAP and MAP Clients must be protected. The MAP and MAP Clients should also be checked to ensure their ongoing health (e.g. using TNC integrity checks and/or a TPM).
- *Security of Remediation Solutions*: In the event that remediation of an 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 implementations 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 elements, 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).

This section is only a brief summary of security considerations related to the TNC architecture. Each TNC interface specification includes an in-depth Security Considerations section that analyzes the security issues relevant to that interface and makes recommendations for appropriate countermeasures. Each interface specification also contains normative requirements for countermeasures relevant to that interface. All parties are urged to review these sections in detail to understand and properly implement these countermeasures.

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.
- *Anonymity of published information:* MAP Clients may publish information about endpoint health, network access, events (which may include information about what services an endpoint is accessing), roles and capabilities, and the identity of the end user operating the endpoint. Any of this published information may be queried by other MAP Clients and could potentially be used to correlate network activity to a particular end user. Care should be taken by deployers of IF-MAP to ensure that the information published by MAP Clients does not violate agreements with end users or local laws and regulations.

These measures ensure that privacy can be properly protected in the TNC architecture.

10 References

[1] Trusted Computing Group, *TCG Specification Architecture Overview*, Revision 1.4, August 2007.

[2] Trusted Computing Group, IWG *Reference Architecture for Interoperability (Part 1)*, Specification Version 1.0, June 2005.

[3] Trusted Computing Group, *TCG Credential Profile,* Specification Version 1.1, May 2007.

[4] Trusted Computing Group, *TPM Specifications v1.2*, March 2011.

[5] Trusted Computing Group, *TSS Specifications v1.2*, January 2006.

[6] Trusted Computing Group, *TNC IF-TNCCS Specification v1.2*, May 2009.

[7] Trusted Computing Group, *TNC IF-TNCCS-SOH Specification v1.0*, May 2007.

[8] Trusted Computing Group, *TNC IF-T: Protocol Bindings for Tunneled EAP Methods Specification v1.1*, May 2007.

[9] Trusted Computing Group, *TNC IF-PEP: Protocol Bindings for RADIUS Specification v1.1*, February 2007.

[10] Trusted Computing Group, *TNC IF-IMC Specification v1.2*, February 2007.

[11] Trusted Computing Group, *TNC IF-IMV Specification v1.2*, February 2007.

[12] Trusted Computing Group, *TCG Glossary*. See https://www.trustedcomputinggroup.org/groups/glossary/

[13] C. de Laat, G. Gross, L. Gommans, J. Vollbrecht, D. Spence, *Generic AAA Architecture*, RFC 2903, Experimental, August 2000, IETF.

[14] J. Vollbrecht, et al., *AAA Authorization Framework*, RFC 2904, Informational, August 2000, IETF.

[15] J. Vollbrecht et al., *AAA Authorization Application Examples*, RFC 2905, Informational, August 2000, IETF.

[16] S. Farrell et al, *AAA Authorization Requirements*, RFC 2906, Informational, August 2000.

[17] B. Aboba et al., *Criteria for Evaluating AAA Protocols for Network Access*, RFC 2989, Informational, November 2000, IETF.

[18] R. Yavatkar, D. Pendarakis, R. Guerin, *A Framework for Policy-based Admission Control*, RFC 2753, January 2000, IETF.

[19] IEEE802, *Port-Based Network Access Control*, IEEE Std 802.1X-2004, December 2004, Institute for Electrical and Electronics Engineers (IEEE).

[20] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, H. Levkowetz, *Extensible Authentication Protocol* (EAP), RFC3748, Standards Track, June 2004, IETF.

[21] C. Rigney, S. Willens, A. Rubens, W. Simpson , *Remote Authentication Dial In User Service* (RADIUS), RFC2865, Standards Track, June 2000.

[22] D. Harkins, D. Carrel, *The Internet Key Exchange* (IKE), RFC 2409, Standards Track, November 1998, IETF.

[23] C. Kaufman, *Internet Key Exchange (IKEv2) Protocol*, RFC4306, Standards Track, December 2005, IETF.

[24] Trusted Computing Group, *TNC IF-MAP Binding for SOAP Specification v2.1*, May 2012.

[25] Trusted Computing Group, *IWG IF-PTS Specification v1.0,* November 2006.

[26] Trusted Computing Group, *TNC CESP Specification v1.0,* May 2009.

[27] Trusted Computing Group, *TNC Federated TNC Specification v1.0*, May 2009

[28] Trusted Computing Group, *TNC IF-T Binding to TLS v1.0*, May 2009

[29] B. Aboba, P. Calhoun, *RADIUS (Remote Authentication Dial In User Service) Support For Extensible Authentication Protocol (EAP)*, RFC3579, September 2003, IETF.

[30] Trusted Computing Group, *TNC IF-TNCCS: TLV Binding v2.0*, January 2010.

[31] Trusted Computing Group, *TNC IF-M: TLV Binding*, March 2010.

[32] Trusted Computing Group, *TNC IF-MAP Metadata for Network Security v1.1*, May 2012.

11 TNC Glossary

When used in TNC documents, the following terms are defined as below. Please also see [12] for broader TCG related terminology.

Term	Definition
Access Requestor (AR)	**Within the TNC framework for endpoint integrity, the Access Requestor is the role of the element seeking connectivity to a network that implements the TNC Architecture. The AR consists of three main functions: the NAR, the TNCC and the IMC. See glossary for the definition of these components.**
Administrative Client	**An optional function in the TNC framework which may not be directly involved with initial network access decisions nor directly connected to the AR. Administrative Clients may share information with other TNC components through IF-MAP and may enable administrative operations such as monitoring, investigation, and provisioning.**
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.**
Flow Controller	**An optional function in the TNC framework which may not be directly involved with initial network access decisions nor directly connected to the AR. Flow Controllers may share information with other TNC components through IF-MAP and may aid in on-going and granular enforcement of network security policy compliance.**
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 Information	**The set of platform specific information that makes up a Trusted Platform. This ranges 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.**
Integrity Measurement Collector (IMC)	**An IMC is a function of 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 function of the PDP that verifies a particular aspect of the AR's integrity, based on measurements received from an IMC and/or other data. Multiple IMVs may reside on a single PDP.**
Isolation	**The action of separating an Access Requestor onto a separate network – virtual or physical – possibly, though not necessarily, for the purposes of performing Remediation on that AR.**
Metadata Access Point (MAP)	**The role in the TNC framework of a broker/server to which metadata may be published and from which metadata may be searched and subscribed to using the IF-MAP protocol.**
Metadata Access Point Client (MAPC)	**The role in the TNC framework of an element which publishes metadata to or searches / subscribes to metadata from a MAP.**
Metadata Access Point Server (MAPS)	**The component of the MAP providing the function that allows other TNC components to publish, subscribe to, and search metadata.**
Network Access Decision	**Refers to the decision sent from the NAA to the NAE via IF-PEP to control the endpoint's network access. This decision may be a simple binary value (allow or deny network access) or it may include information (such as a VLAN ID) for purposes such as Isolation. Alternatively, it may include information (such as a VLAN ID) for purposes such as Isolation.**
Network Access Authority (NAA)	**The Network Access Authority (NAA) is the network layer function of the PDP that decides whether a Network Access Requestor (NAR) should be granted access to a network.**
Network Access Enforcer (NAE)	**The Network Access Enforcer (NAE) is the network layer function of the PEP that consumes and enforces access control policies from a Network Access Authority (NAA).**
Network Access Requestor (NAR)	**The Network Access Requestor (NAR) is the function 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 the role of an element 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 the role of an element 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 element seeking network connectivity (that fails integrity check) with the necessary software, firmware and integrity-related parameters updates.**
Sensor	**An optional function of an element in the TNC framework which may not be directly involved with initial network access decisions nor directly connected to the AR. Sensors may monitor network and AR activity and the share information with other TNC components through IF-MAP.**
TNC Client (TNCC)	**The TNCC function 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.**
TNC Server (TNCS)	**The TNCS is the function 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.**
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).**