TCG Trusted Network Connect

TNC IF-TNCCS

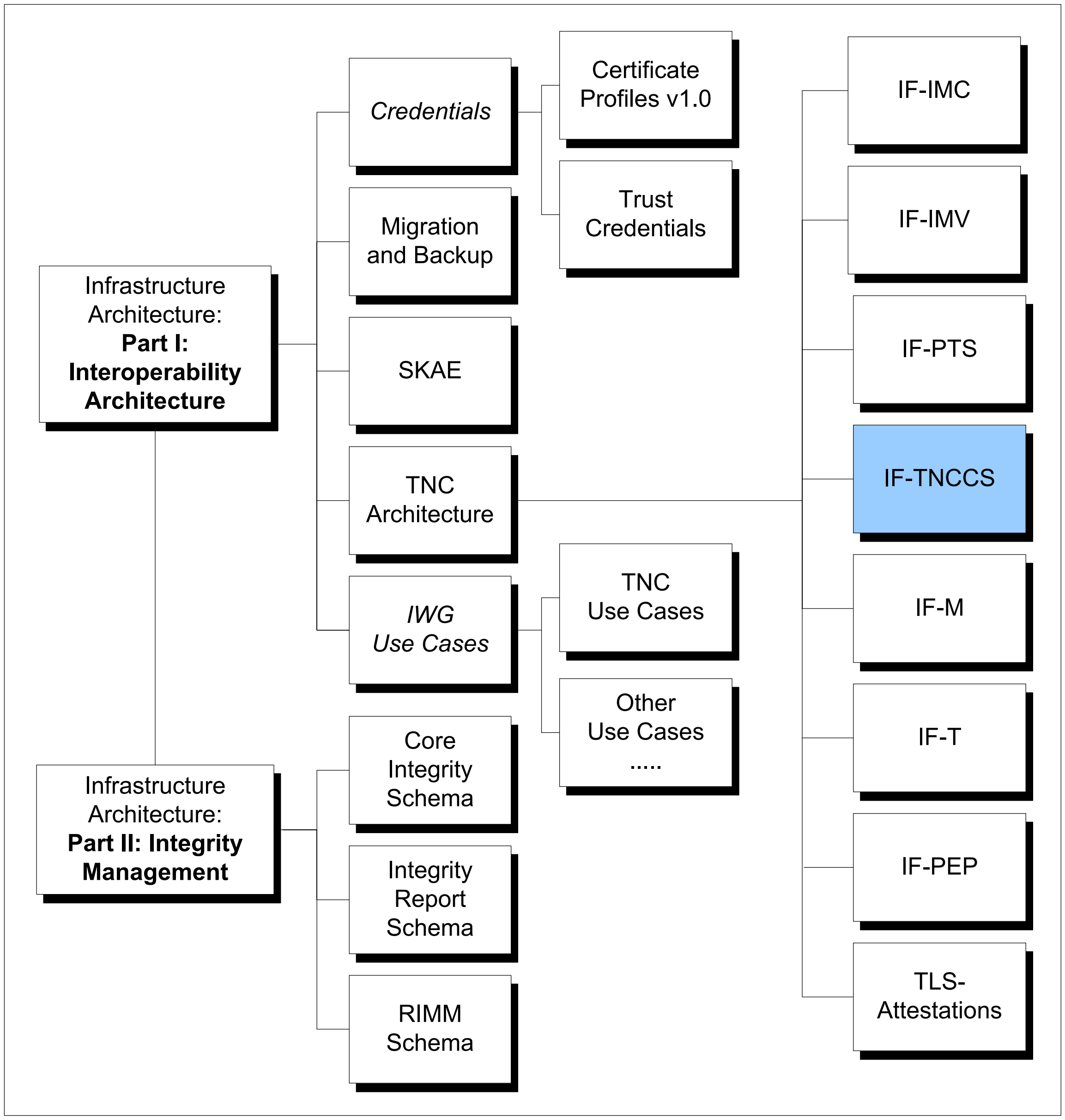
Specification Version 1.0
Revision 2
1 May 2006
Published

Contact: admin@trustedcomputinggroup.org

TCG

TCG PUBLISHED

IWG TNC Document Roadmap



Acknowledgement

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

Special thanks to the active members of the TNC who contributed to the discussions and/or wordings used in this document:

Mahalingam Mani	Avaya
Kazuaki Nimura	Fujitsu Limited
Boris Balacheff	Hewlett-Packard
Mauricio Sanchez	Hewlett-Packard
Diana Arroyo	IBM
Lee Terrell	IBM
Ravi Sahita	Intel Corporation
Ned Smith	Intel Corporation
Barbara Nelson (Editor)	iPass
Steve Hanna (TNC co-chair)	Juniper Networks, Inc.
Gene Chang	Meetinghouse Data Communications
John Vollbrecht	Meetinghouse Data Communications
Joseph Tardo	Nevis Networks
Sandilya Garimella	Motorola
Thomas Hardjono	SignaCert, Inc.
Bryan Kingsford	Symantec Corporation
Paul Sangster (TNC co-chair)	Symantec Corporation
Scott Cochrane	Wave Systems
Greg Kazmierczak	Wave Systems

Table of Contents

1 Introduction

1.1 Scope and Audience

The Trusted Network Connect Sub Group (TNCSG) is defining an open solution architecture that enables network operators to enforce policies regarding endpoint integrity when granting access to a network infrastructure. Integrity measurements are carried between the TNC Client and TNC Server on a protocol called IF-TNCCS (Trusted Network Connect Client-Server), as shown in figure 1 below.

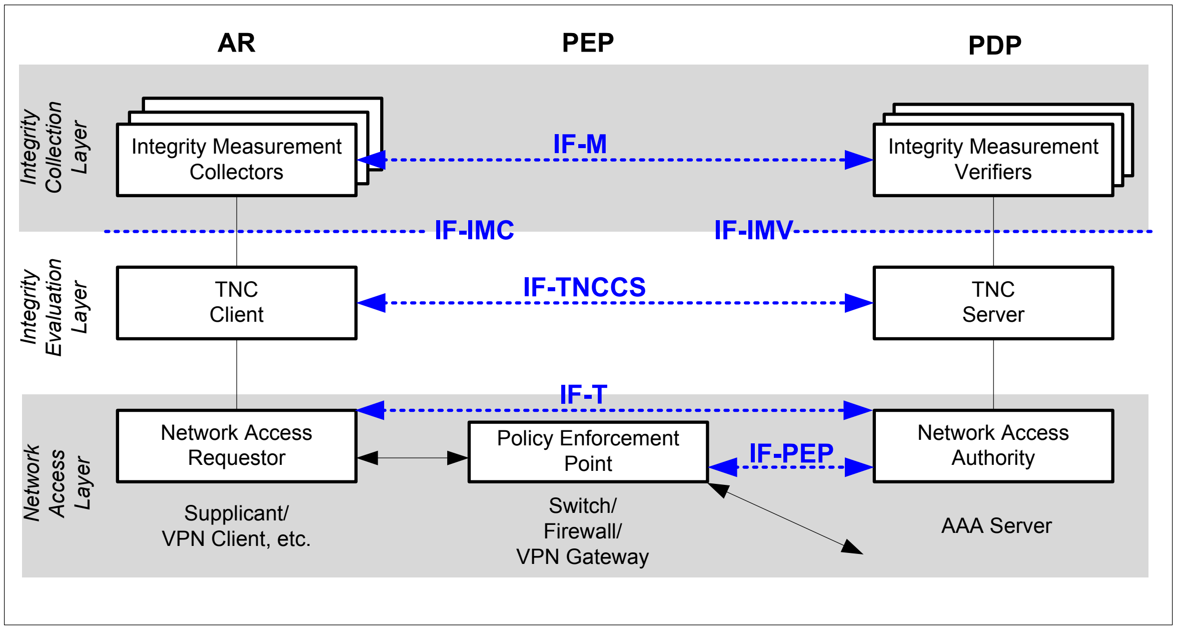

AR
PEP
PDP
Integrity Collection Layer
Integrity Evaluation Layer
Network Access Layer
Integrity Measurement Collectors
IF-M
Integrity Measurement Verifiers
IF-IMC
IF-IMV
TNC Client
IF-TNCCS
TNC Server
IF-T
Network Access Requestor
Policy Enforcement Point
IF-PEP
Network Access Authority
Supplicant/ VPN Client, etc.
Switch/ Firewall/ VPN Gateway
AAA Server

Figure 1 - TNC Architecture

This specification is integral to the TCG Trusted Network Connect Sub-Group's (TNC) reference architecture. Specifically, this specification defines the IF-TNCCS protocol, which is used to communicate integrity measurements between a TNC Client and a TNC Server.

Architects, designers, developers, and technologists interested in the development, deployment, and interoperation of trusted systems will find this document necessary in providing a specific mechanism for communicating integrity information.

Before reading this document any further, the reader should review and understand the TNC architecture as described in [1]. If the reader is building a TNC Client that supports IF-IMC, the reader is encouraged to read [5] prior to reading this document. If the reader is building a TNC Server that supports IF-IMV, the reader is encouraged to read [6] prior to reading this document.

1.2 Keywords

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in RFC 2119 [2]. This specification does not distinguish blocks of informative comments and normative requirements. Therefore, for the sake of clarity, note that lower case instances of must, should, etc. do not indicate normative requirements.

2 Background

2.1 Role of IF-TNCCS

IF-TNCCS describes a standard way for the TNC Client and the TNC Server to exchange messages. More specifically, this interface defines a protocol and format for carrying:

(a) Messages from IMCs to IMVs (such as integrity measurements)

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

(c) Messages from TNCCs to TNCSs (such as control messages)

(d) Messages from TNCSs to TNCCs (such as the TNCCS-Recommendation message)

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.

2.2 Requirements

Here are the requirements that the IF-TNCCS protocol must meet in order to successfully play its role in the TNC architecture.

- Meets the needs of the TNC architecture

  The protocol must support all the functions and use cases described in the TNC architecture as they apply to the relationship between the TNC Client and the TNC Server.

- Efficient

  The TNC architecture delays network access until the endpoint is determined not to pose a security threat to the network based on its asserted integrity information. To minimize user frustration, it is essential to minimize delays and make IMC-IMV communications as rapid and efficient as possible. Efficiency is also important when you consider that some network endpoints are small and low-powered and that some networks have high latency, high cost, or low bandwidth.

- Extensible

  IF-TNCCS will need to expand over time as new features are added to the TNC architecture. The IF-TNCCS protocol must allow new features to be added easily, providing for a smooth transition and allowing newer and older architectural components to continue to work together.

- Easy to use and implement

  The protocol should be easy for TNC Client and TNC Server vendors to use and implement. It should allow them to enhance existing products to support the TNC architecture and integrate legacy code without requiring substantial changes. The protocol should also make things easy for system administrators and end-users. Components of the TNC architecture should plug together automatically without requiring manual configuration.

2.3 Assumptions

Here are the assumptions that the IF-TNCCS protocol makes about other components in the TNC architecture.

- Format of integrity measurements

  The format of the IMC-IMV messages that IF-TNCCS conveys between the TNC Client and the TNC Server is opaque to IF-TNCCS. Each IMC-IMV message is simply represented as a binary piece of data within the IF-TNCCS message schema.

- Chunking

  It is assumed that a complete IF-TNCCS batch of messages can be delivered by IF-T, and that IF-TNCCS is not responsible for segmenting message batches. If the messages must be divided into smaller pieces to map them to some underlying transport protocol, it is the responsibility of IF-T to segment and reassemble the batch.

- Transport

  IF-T is the underlying transport protocol for ALL IF-TNCCS communication. It is assumed that IF-T will provide a reliable transport mechanism, ensuring the timely delivery of IF-TNCCS batches of messages in the same order in which they were sent.

- Security

  IF-TNCCS relies on IF-T for secure transport of messages being exchanged between the TNC Client and the TNC Server. IF-T should indicate what security services it is providing (e.g. encrypted data tunnel). IF-TNCCS does not provide a means for mutual authentication of the TNC Client and the TNC Server. The NAR/NAA should make available some indication of the level of authentication that was performed. Note that IMCs and IMVs may choose to protect their data if they are concerned with exposure to untrusted TNC Servers.

2.4 TNCC and TNCS Features

This section documents the features of TNC Clients and TNC Servers who participate in the IF-TNCCS protocol.

2.4.1 Integrity Check Handshake

One of the primary functions of IF-TNCCS is to facilitate message exchanges between IMCs and IMVs to share security state allowing the IMVs to factor the integrity of the IMC's security software state into the access control decision. These communications always take place within the context of an Integrity Check Handshake. In such a handshake, the IMCs send a batch of messages (typically, integrity measurements) to the IMVs and the IMVs optionally respond with a batch of messages (remediation instructions, queries for more information, etc.). This dialog may go on for some time until the IMVs decide on their IMV Action Recommendations. The TNCS then communicates its TNCS recommendation to the TNCC and to the NAA.

2.4.2 Remediation and Handshake Retry

In several cases, it is useful to retry an Integrity Check Handshake. First, an endpoint may be isolated until remediation is complete. Once remediation is complete, an IMC can inform the TNCC of this fact and suggest that the TNCC retry the Integrity Check Handshake. Second, a TNCS can initiate a retry of an Integrity Check Handshake (if the TNCS or IMV policies change or as a periodic recheck). Third, an IMC or IMV can request a handshake retry in response to a condition detected by the IMC or IMV (suspicious activity, for instance). In any case, it's generally desirable (but not always possible) to reuse state established by the earlier handshake and to avoid disrupting network connectivity during the handshake retry.

Future versions of IF-TNCCS will define mechanisms by which a client or server may trigger a handshake retry without the overhead of disconnecting and reconnecting. Handshake retry can be accomplished with the current IF-TNCCS specification by initiating a completely new handshake.

2.4.3 Messages

One of the critical functions of the TNC architecture is conveying messages between IMCs and IMVs. Each message sent in this way consists of a message body, a message type, and a recipient type. IF-TNCCS supports two flavors of messages: IMC-IMV messages and TNCC-TNCS messages. IMC-IMV messages are generated by an IMC and delivered to zero or more IMVs, or generated by an IMV and delivered to zero or more IMCs. TNCC-TNCS messages are generated by the TNCC and consumed by the TNCS or generated by the TNCS and consumed by the TNCC.

The message body is a sequence of octets (bytes). The TNCC and TNCS SHOULD NOT parse or interpret the message body for IMC-IMV messages. Interpretation of the message body is left to the ultimate recipients of the message. A zero length message is perfectly valid and MUST be properly delivered by the TNCC and TNCS just as any other IMC-IMV message would be.

The message type is a four octet number that uniquely identifies the format and semantics of the message. The method used to ensure the uniqueness of message types while providing for vendor extensions is described in section 2.4.4.

The recipient type is a flag indicating whether the message should be delivered to the TNCS/IMVs or the TNCC/IMCs. Messages sent by the TNCC are delivered to the TNCS and vice versa.

2.4.4 Vendor IDs

The TNC architecture routes messages between IMCs and IMVs based on their message type. Each message has a message type that uniquely identifies the format and semantics of the message. A message type is a 32-bit number.

To ensure the uniqueness of message types while providing for vendor extensions, vendor-specific message types are formed by placing a vendor-chosen message subtype in the least significant 8 bits of the message type and the vendor's vendor ID in the most significant 24 bits of the message type. Message types standardized by the TCG will have the reserved value zero (0) in the most significant 24 bits.

SMI Private Enterprise Numbers are used to provide a separate identifier space for each vendor. IANA provides a registry for SMI Private Enterprise Numbers at http://www.iana.org/assignments/enterprise-numbers. Any organization (including non-profit organizations, governmental bodies, etc.) can obtain one of these numbers at no charge and thousands of organizations have done so. Within this document, SMI Private Enterprise Numbers are known as "vendor IDs". Vendor ID zero (0) is reserved for identifiers defined by the TCG. Vendor ID 16777215 (0xffffff) is reserved for use as a wildcard. A TNCC or TNCS MUST NOT send messages with a vendor ID of 0xffffff.

TNC Clients and TNC Servers MUST properly deliver messages with any message type.

2.4.4.1 Vendor ID Values

These are reserved vendor ID values. Other vendor IDs between 1 and 16777214 (0xfffffe) may be used as described above.

Vendor ID Value	Value	Definition
TNC_VENDORID_TCG	0	Reserved for TCG-defined values
TNC_VENDORID_ANY	0xffffff	Wild card matching any vendor ID

2.4.4.2 Message Subtype Values

This is a reserved message subtype value. Other message subtypes between 0 and 254 may be used as described above.

Message Subtype Value	Value	Definition
TNC_SUBTYPE_ANY	0xff	Wild card matching any message subtype

2.4.5 IMC-IMV Message Delivery

The routing and delivery of IMC-IMV messages is governed by message type and recipient type. Each IMC and IMV indicates which IMC-IMV message types it wants to receive. The TNCC and TNCS are then responsible for ensuring that any IMC-IMV message sent during an Integrity Check Handshake is delivered to all recipients that have a recipient type matching the message's recipient type and that have indicated the wish to receive messages whose message type matches the message's message type. If no recipient has indicated a wish to receive a particular message type, the TNCC and TNCS can handle these messages as they like: ignore, log, etc.

WARNING: The message routing and delivery algorithm just described is not a one-to-one model. A single message may be received by several recipients (for example, two IMVs from a single vendor, two copies of an IMC, or nosy IMVs that monitor all messages).

One message sub-type (TNC_SUBTYPE_ANY) is reserved for special handling. If an IMC or IMV indicates that it wishes to receive messages of type TNC_SUBTYPE_ANY for a given vendor the TNCC or TNCS MUST deliver all IMC-IMV messages that match that vendor to that IMC or IMV. Similarly, if an IMC or IMV indicates that it wishes to receive messages of type TNC_SUBTYPE_ANY for any vendors (TNC_VENDORID_ANY), the TNCC or TNCS MUST deliver all IMC-IMV messages that it receives to that IMC or IMV.

IF-TNCCS allows an IMC or IMV to send and receive messages using this messaging system. Note that this system should not be used to send large amounts of data. The messages will often be sent through PPP or similar protocols that do not include congestion control and are not well suited to bulk data transfer. If an IMC needs to download a patch (for instance), the IMV should indicate this in the remediation instructions. The IMC will process those instructions after network access (perhaps isolated) has been established and can then download the patch via any appropriate protocol.

2.4.6 TNCC-TNCS Message Delivery

In addition to IMC-IMV messages, IF-TNCCS supports the exchange of messages (standard and vendor-specific) between a TNC Client and a TNC Server. These messages may be defined in XML or a base64-encoded blob of data. TNCC-TNCS messages have a message type, constructed in the same way as IMC-IMV message types (i.e. using vendor IDs). If a TNCC or TNCS receives a TNCC-TNCS message that it does not understand, the TNCC or TNCS SHOULD ignore the message and MAY log it.

TNCC-TNCS messages should be used for conveying control information between a TNCC and a TNCS, rather than for conveying integrity measurements that were generated by a TNCC. Integrity measurements should be conveyed in IMC-IMV messages, whether they were generated by a TNCC (which is really acting as an IMC in this instance) or by an IMC. This allows an IMV to register interest in all measurements, whether they were generated by an IMC or by a TNCC.

IF-TNCCS allows a TNCC or TNCS to send and receive messages using this messaging system. Note that this system should not be used to send large amounts of data. The messages will often be sent through PPP or similar protocols that do not include congestion control and are not well suited to bulk data transfer.

2.4.7 Batches

Messages being sent from the IMCs/TNCC to the IMVs/TNCS and vice versa are sent in batches, rather than requiring that each message be sent independently. A single batch of messages may contain zero or more TNCC-TNCS messages and zero or more IMC-IMV messages.

Batches of messages being conveyed between a TNCC and a TNCS will frequently be carried over protocols (like EAP) that require participants to take turns in sending ("half duplex"). To operate well over such protocols, the TNCC sends a batch of messages and the TNCS responds with some messages.

A new Integrity Check Handshake always starts with a batch of zero or more messages being sent from the TNCC to the TNCS. Each IMC-IMV message is delivered to the IMVs that have registered interest in the message's message type. The TNCS consumes any TNCC-TNCS messages in that batch. If the TNCS or any IMVs have messages to send in response, the TNCS sends a batch of messages to the TNCC. The TNCC delivers each IMC-IMV message to the IMCs that have registered interest in the message's message type. The TNCC consumes any TNCC-TNCS messages in that batch. This half-duplex process continues until one side or the other has no more messages to send, or one side or the other decides to complete the handshake.

In an Integrity Check Handshake, each batch of messages has a unique batch id to aid in troubleshooting. The batch id is a sequentially incrementing integer that starts at 1 at the beginning of each Integrity Check Handshake. The first batch (from the TNCC to the TNCS) is batch 1, the second batch (from the TNCS to the TNCC) is batch 2, and so on.

2.4.8 Completing the handshake

If no IMCs want to send a message in a particular batch, the TNCC will proceed to complete the handshake, by sending a TNCCS-batch containing no IMC-IMV messages. This indicates to the TNCS that the TNCC has no more measurements to provide, and would like to complete the handshake.

Similarly, if no IMVs want to send a message in a particular batch, this indicates to the TNCS that the IMVs are ready to make their recommendations. The TNCS gathers the IMV Action Recommendations, and makes an overall TNCS Action Recommendation, which it conveys to the TNCC in a TNCCS-Batch. (There will be no IMC-IMV messages in this batch.)

Note that a TNCC or TNCS MAY cut off IMC-IMV communications at any time for any reason, including limited support for long conversations in underlying protocols, user or administrator intervention, or policy. This can be done by indicating to IF-T that the TNCC or TNCS does not wish to proceed with the connection, or by completing the handshake as outlined above.

2.4.9 Error handling

If the TNC Server receives a malformed batch, or has an internal error, it should discard the batch, and generate a TNCCS-Recommendation. It should include a TNCCS-Error message with the TNCCS-Recommendation message, to provide information to the client on the error. The TNC Server should also log information about the malformed batch to aid in troubleshooting. When the TNC Client receives the TNCCS-Error message, it may log the event.

If the TNC Client receives a malformed batch, or has an internal error, it should discard the batch, and send a TNCCS-Error message to the TNCS. It should not include any IMC-IMV messages in the batch with the TNCCS-Error message. When the TNCS receives the TNCCS-Error message, it should log the event and generate a TNCCS-Recommendation message.

2.4.10 Reliability

For successful enterprise deployments, reliability of TNC Servers is important. To ensure this reliability, organizations may employ redundant TNC Servers. Organizations may also require active failover as well as other features that provide a level of high availability for critical networks.

3 IF-TNCCS Schema

3.1 Schema Namespace

The IF-TNCCS schema's namespace is defined using the `targetNamespace` attribute of the schema's root-level `schema` element:

`targetNamespace="`*`namespace`*`"`

The schema's namespace reflects the schema version, and is currently defined as follows:

http://www.trustedcomputinggroup.org/IWG/TNC/1_0/IF_TNCCS#

The schema XSD file will be available from the following location:

https://www.trustedcomputinggroup.org/XML/SCHEMA/TNCCS_1.0.xsd

3.2 Dependent Schema Definitions

3.2.1 W3C XML Schema Syntax

The IF-TNCCS schema relies upon data structures defined by the World Wide Web Consortium's (W3C) XML-Schema syntax. Consequently, the IF-TNCCS schema imports the W3C's XML schema with the following namespace:

`http://www.w3.org/2001/XMLSchema`

The IF-TNCCS schema associates the above-mentioned schema with the "xs" namespace prefix.

3.3 Schema: Complex Types

The following sections document the complex types defined within the IF-TNCCS schema.

3.3.1 TNCCS-Batch

The `TNCCS-Batch` element represents a batch of TNCCS-messages being carried across the TNCCS interface. The TNCCS-batch contains zero or more messages.

Component	Entity	Type	Description
TNCC-TNCS-Message	Sequence	TNCC-TNCS-Message	Zero or more TNCC-TNCS messages
IMC-IMV-Message	Sequence	IMC-IMV-Message	Zero or more IMC-IMV messages
BatchId	Attribute	xs:nonNegativeInteger	Unique identifier for this batch of messages
Recipient	Attribute	TNCCS-Recipient-Type	Identifies whether this batch should be delivered to TNCS or TNCC.

3.3.2 IMC-IMV-Message

The `IMC-IMV-Message` complex type represents one message being carried across the TNCCS interface from an IMC to zero or more IMVs or vice versa. Each TNCCS-Message consists of a message type (4 octets) and a message body (base64-encoded arbitrary binary data).

Component	Entity	Type	Description
Type	Element	TNCCS-Message-	Unique identifier to identify the type of

		Type	message (4 octets)
Base64	Element	xs:base64Binary	The contents of the message (opaque to TNCC and TNCS)

3.3.3 TNCC-TNCS-Message

The `TNCC-TNCS-Message` complex type represents one message being carried across the TNCCS interface from a TNCC to a TNCS or vice versa. Each TNCCS-Message consists of a message type (4 octets) and a body (XML or base64-encoded arbitrary data).

Component	Entity	Type	Description
Type	Element	TNCCS-Message-Type	Unique identifier to identify the type of message (4 octets)
XML	Element	xs:any	The contents of the message (if XML)
Base64	Element	xs:base64Binary	The contents of the message (if base 64 encoded)

3.3.4 TNCCS-Recipient-Type

The `TNCCS-Recipient-Type` type represents the destination for this message (TNCC or TNCS).

3.3.5 TNCCS-Message-Type

The `TNCCS-Message-Type` type is a 4-octet identifier to identify the type of the message. It is a message type as described in sections 2.4.3 and 2.4.4.

3.3.6 TNCCS-Recommendation

TNCCS-Recommendation is a TNCC-TNCS message sent from the TNC Server to the TNC Client to indicate that the Integrity Check Handshake has been completed, and the TNC Server is ready to provide a TNCS Action Recommendation (allowed, isolated or none) to the NAA. The TNCCS-Recommendation message contains the TNCS Action Recommendation that will be made. (Note that the NAA may choose to ignore the recommendation and grant a different level of access.)

3.3.7 TNCCS-Error

TNCCS-Error is a TNCC-TNCS message sent from the TNC Server to the TNC Client (or vice versa) to indicate that there was a problem with the last batch of messages received. When sent by the TNC Server, it will always be in the same batch as the TNCCS-Recommendation message. When sent by the TNC Client, it will always be in a batch that does not contain any IMC-IMV messages.

3.4 Schema definition

The following section defines the IF-TNCCS schema.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="http://www.trustedcomputinggroup.org/IWG/TNC/1_0/IF_TNCCS#"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.trustedcomputinggroup.org/IWG/TNC/1_0/IF_TNCCS#"
elementFormDefault="qualified" attributeFormDefault="unqualified" version="1.0"
id="TNCCS-Batch">
```

```
<xs:element name="TNCCS-Batch">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="TNCC-TNCS-Message" type="TNCC-TNCS-Message"
    minOccurs="0" maxOccurs="unbounded"/>
   <xs:element name="IMC-IMV-Message" type="IMC-IMV-Message"
    minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="BatchId" type="xs:nonNegativeInteger" use="required">
  </xs:attribute>
  <xs:attribute name="Recipient" type="TNCCS-Recipient-Type" use="required">
  </xs:attribute>
 </xs:complexType>
</xs:element>

<xs:complexType name="TNCC-TNCS-Message">
 <xs:sequence>
  <xs:element name="Type" type="TNCCS-Message-Type"/>
  <xs:choice>
   <xs:element name="Base64" type="xs:base64Binary"/>
   <xs:element name="XML" type="XML"/>
  </xs:choice>
 </xs:sequence>
</xs:complexType>

<xs:complexType name="IMC-IMV-Message">
 <xs:sequence>
  <xs:element name="Type" type="TNCCS-Message-Type"/>
  <xs:element name="Base64" type="xs:base64Binary"/>
 </xs:sequence>
</xs:complexType>

<xs:complexType name="XML">
 <xs:sequence>
  <xs:any namespace="##any" processContents="lax"/>
 </xs:sequence>
</xs:complexType>

<xs:simpleType name="TNCCS-Message-Type">
 <xs:restriction base="xs:hexBinary">
  <xs:minLength value="4"/>
  <xs:maxLength value="4"/>
 </xs:restriction>
</xs:simpleType>

<xs:simpleType name="TNCCS-Recipient-Type">
```

```
  <xs:restriction base="xs:NMTOKENS">
   <xs:enumeration value="TNCS"/>
   <xs:enumeration value="TNCC"/>
  </xs:restriction>
 </xs:simpleType>

 <xs:simpleType name="TNCCS-Recommendations">
  <xs:restriction base="xs:NMTOKENS">
   <xs:enumeration value="allow"/>
   <xs:enumeration value="none"/>
   <xs:enumeration value="isolate"/>
  </xs:restriction>
 </xs:simpleType>

 <xs:simpleType name="TNCCS-Recommendation-Message-Type">
  <xs:restriction base="TNCCS-Message-Type">
   <xs:enumeration value="00000001"/>
  </xs:restriction>
 </xs:simpleType>

 <xs:complexType name="TNCCS-Recommendation">
  <xs:attribute name="type" type="TNCCS-Recommendations"/>
 </xs:complexType>

 <xs:element name="TNCCS-Recommendation" type="TNCCS-Recommendation"/>

 <xs:simpleType name="TNCCS-Errors">
  <xs:restriction base="xs:NMTOKENS">
   <xs:enumeration value="batch-too-long"/>
   <xs:enumeration value="malformed-batch"/>
   <xs:enumeration value="invalid-batch-id"/>
   <xs:enumeration value="invalid-recipient-type"/>
   <xs:enumeration value="internal-error"/>
   <xs:enumeration value="other"/>
  </xs:restriction>
 </xs:simpleType>

 <xs:simpleType name="TNCCS-Error-Message-Type">
  <xs:restriction base="TNCCS-Message-Type">
   <xs:enumeration value="00000002"/>
  </xs:restriction>
 </xs:simpleType>

 <xs:complexType name="TNCCS-Error">
  <xs:attribute name="type" type="TNCCS-Errors"/>
 </xs:complexType>
```

```
  <xs:element name="TNCCS-Error" type="TNCCS-Error"/>

</xs:schema>
```

4 Security Considerations

4.1 Threat Model

In analyzing the TNC threat model for IF-TNCCS, threats can be organized into several categories.

4.1.1 Network Attacker with Passive or Active Capabilities

A network attacker is able to intercept communications between the TNCC and the TNCS. This attacker may only be able to monitor these communications (a passive attack) or may have the ability to change these communications (an active attack).

A successful passive network attack can lead to disclosure of confidential data (such as integrity measurements), potentially leading to privacy violations or attacks on the endpoint based on vulnerabilities disclosed through the integrity measurements.

A successful active network attack can change data in transit without detection. Changing integrity measurements could cause the PDP to make an incorrect decision (such as granting access to a non-compliant endpoint). Changing remediation instructions could cause the endpoint to perform an incorrect remediation, potentially leading to compromise of the endpoint.

4.1.2 Compromise of Endpoint or Server

An attacker may be able to compromise critical components of the endpoint or server (such as the TNCC, NAR, NAA, or TNCS). This would allow the attacker to subvert the normal functioning of these components.

Compromise of endpoint components might allow the attacker to report false integrity measurements, which could cause the PDP to make an incorrect decision (such as granting access to a non-compliant endpoint).

Compromise of server components might allow the attacker to change PDP behavior, resulting in an incorrect decision (such as granting access to a non-compliant endpoint) or in sending incorrect remediation instructions, potentially leading to compromise of the endpoint.

4.2 Countermeasures

4.2.1 IF-T Security

IF-T protocol bindings are required to provide secure transport for IF-TNCCS messages, including mutual authentication, integrity protection, and confidentiality. If properly and successfully implemented, IF-T security protects against active and passive network attacks.

4.2.2 Platform Security

Protection against compromise of endpoint or server components is provided by underlying platform security, optionally supplemented by TPM protections detailed in the next section. The endpoint and server platforms should provide adequate protection (memory protection, authentication of privileged users, physical security, etc.) to prevent compromise of components running on those platforms. To the extent that platform security is not or may not be adequate, TPM protections should be employed.

4.2.3 TPM Protections

To provide additional protection against compromise of endpoint components if endpoint platform security is not or may not be adequate, a TPM may be used to securely measure the integrity of the endpoint platform and convey these measurements to the server where they can be verified. This allows compromise of endpoint components to be detected. Similarly, a TPM on the server platform may be employed to detect compromise of the server platform.

5 Example Message Flows

This section provides an informative (non-binding) walkthrough of a typical exchange of messages between a TNCC and a TNCS.

5.1 New Connection

Figure 2 below illustrates the sequence of events when a client is attempting to connect to a protected network. In this example, one IMC sends a message to its respective IMV. The TNC Client also sends a message to the TNC Server in the same batch. The IMV responds with a message to the IMC, which then sends one more message in reply. After this exchange of messages, the TNCS communicates its recommendation to the TNCC, and the client is granted access.

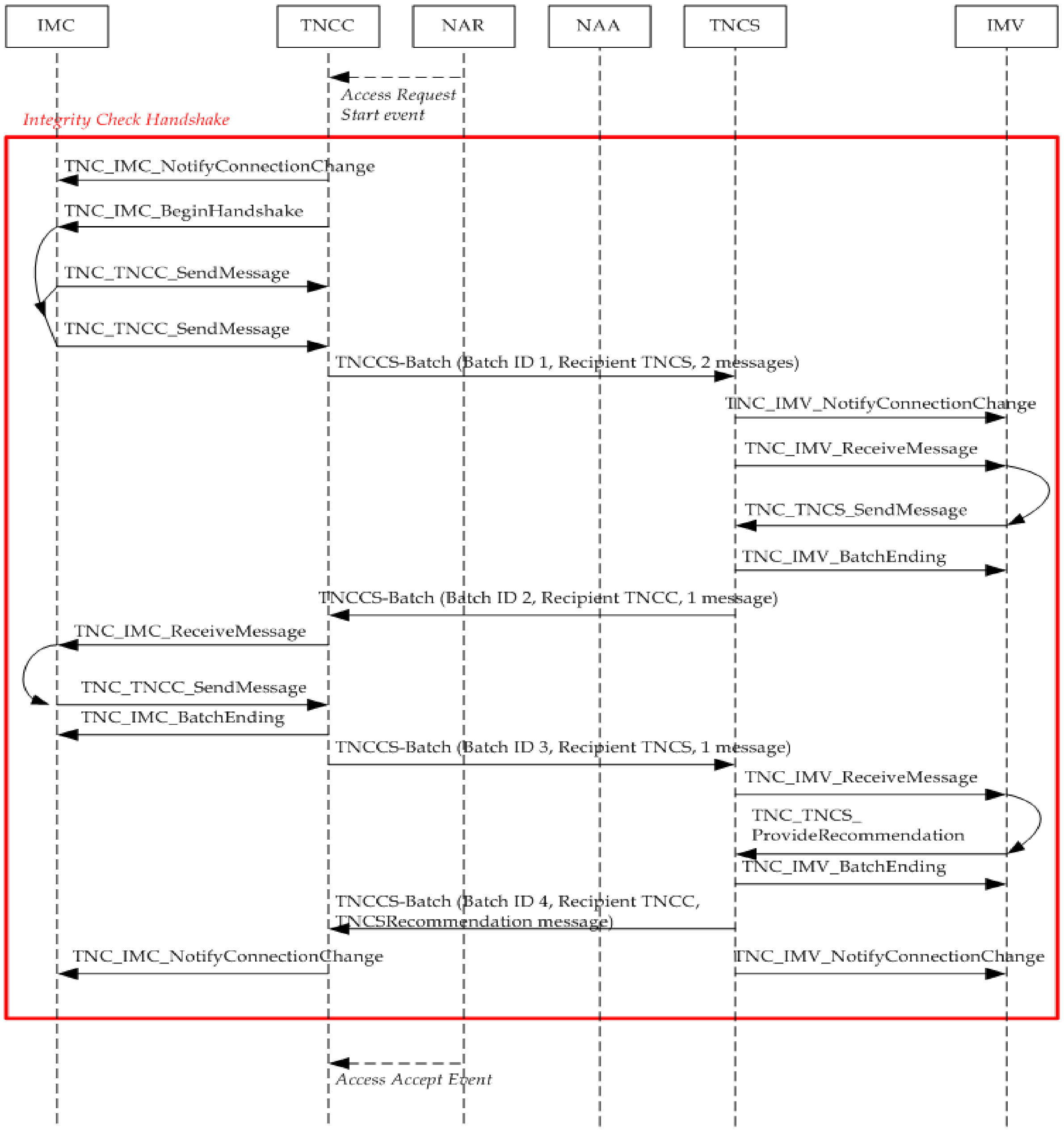

IMC
TNCC
NAR
NAA
TNCS
IMV
Access Request Start event
Integrity Check Handshake
TNC_IMC_NotifyConnectionChange
TNC_IMC_BeginHandshake
TNC_TNCC_SendMessage
TNC_TNCC_SendMessage
TNCCS-Batch (Batch ID 1, Recipient TNCS, 2 messages)
TNC_IMV_NotifyConnectionChange
TNC_IMV_ReceiveMessage
TNC_TNCS_SendMessage
TNC_IMV_BatchEnding
TNCCS-Batch (Batch ID 2, Recipient TNCC, 1 message)
TNC_IMC_ReceiveMessage
TNC_TNCC_SendMessage
TNC_IMC_BatchEnding
TNCCS-Batch (Batch ID 3, Recipient TNCS, 1 message)
TNC_IMV_ReceiveMessage
TNC_TNCS_ ProvideRecommendation
TNC_IMV_BatchEnding
TNCCS-Batch (Batch ID 4, Recipient TNCC, TNCSRecommendation message)
TNC_IMC_NotifyConnectionChange
TNC_IMV_NotifyConnectionChange
Access Accept Event

Figure 2 - Message Flow for New Connection

6 Sample XML

This section provides some simple examples of what the XML instance data might look like.

6.1 Multiple messages

This example is for a batch of two messages being sent from a TNCC to a TNCS (batch 1 from the message flow in section 5.1). The first is an XML-encoded vendor-specific TNCC-TNCS message. The second is a base64-encoded vendor-specific IMC-IMV message.

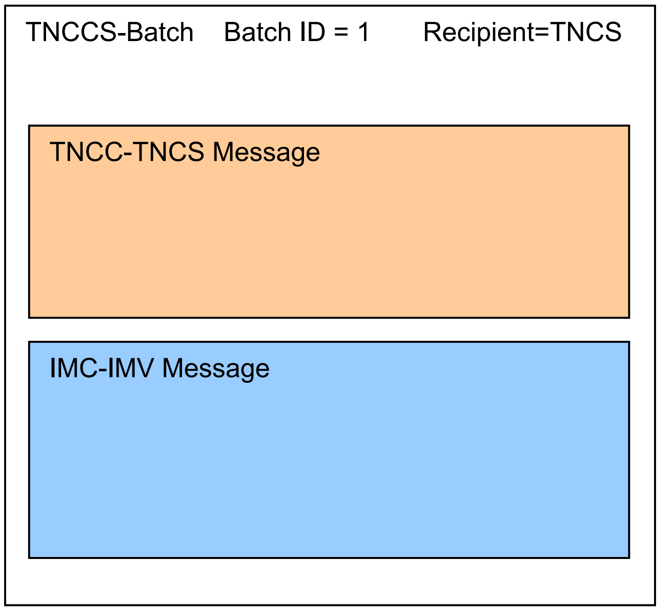


Figure 3 - Batch 1 from Message Flow in Section 5

```
<?xml version="1.0"?>
<TNCCS-Batch BatchId="1" Recipient="TNCS"
xmlns="http://www.trustedcomputinggroup.org/IWG/TNC/1_0/IF_TNCCS#"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation=
"http://www.trustedcomputinggroup.org/IWG/TNC/1_0/IF_TNCCS#
https://www.trustedcomputinggroup.org/XML/SCHEMA/TNCCS_1.0.xsd">
    <TNCC-TNCS-Message>
        <Type>0304BBDD</Type>
        <XML><MyTNCMessage id=99>Some content</MyTNCMessage></XML>
    </TNCC-TNCS-Message>
    <IMC-IMV-Message>
        <Type>AABB0001</Type>
        <Base64>342jkdslfjklsd34</Base64>
    </IMC-IMV-Message>
</TNCCS-Batch>
```

6.2 TNCCS-Recommendation

This example (batch 4 from the message flow in section 5.1) is a batch coming from the TNCS to the TNCC containing a single message, the TNCCS-Recommendation.

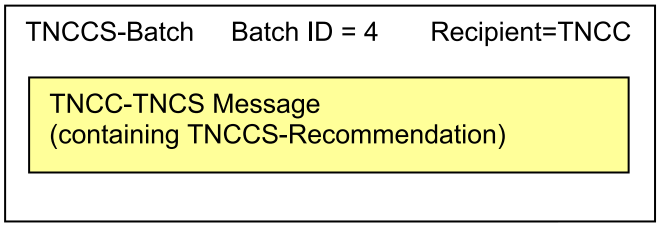


Figure 4 - Batch 4 from Message Flow in Section 5

```
<?xml version="1.0"?>
<TNCCS-Batch BatchId="4" Recipient="TNCC"
xmlns="http://www.trustedcomputinggroup.org/IWG/TNC/1_0/IF_TNCCS#"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation=
"http://www.trustedcomputinggroup.org/IWG/TNC/1_0/IF_TNCCS#
https://www.trustedcomputinggroup.org/XML/SCHEMA/TNCCS_1.0.xsd">
    <TNCC-TNCS-Message>
        <Type>00000001</Type>
        <XML><TNCCS-Recommendation type=”allow”></TNCCS-Recommendation></XML>
    </TNCC-TNCS-Message>
</TNCCS-Batch>
```

7 References

7.1 Normative References

[1] Trusted Computing Group, *TNC Architecture for Interoperability*, Specification Version 1.1, May 2006.

[2] Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels”, Internet Engineering Task Force RFC 2119, March 1997.

[3] W3C Extensible Markup Language (XML) 1.1. http://www.w3.org/TR/xml11/

[4] XML Schema http://www.w3.org/XML/Schema

7.2 Informative References

[5] Trusted Computing Group, *TNC IF-IMC*, Specification Version 1.1, May 2006.

[6] Trusted Computing Group, *TNC IF-IMV*, Specification Version 1.1, May 2006.

[7] Trusted Computing Group, *TNC IF-T*, Specification Version 1.0, May 2006.

[8] Trusted Computing Group, *TNC IF-PEP*, Specification Version 1.0, May 2006.