ETSI TR 103 308 V1.1.1 (2016-01)

CYBER;
Security baseline regarding LI and RD
for NFV and related platforms

Reference
DTR/CYBER-0006

Keywords
cyber security, Lawful Interception, NFV

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Contents

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://ipr.etsi.org).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Technical Report (TR) has been produced by ETSI Technical Committee Cyber Security (CYBER).

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Introduction

ETSI ISG NFV (and others) are creating an ecosystem whereby traditional network functions that may have been tangible, are now virtualized, potentially onto commercial "off the shelf" hardware. There is a requirement for ISG NFV to utilize features and functions available within the underlying platform for the purposes of ensuring lawful interception (LI) and Retained Data (RD) operations are appropriately protected and delivered - the present document intends to outline those requirements, capabilities and how they could be utilized.

The security principles themselves can include:

- Effective use of TPMs/Roots-of-Trust/Trusted-boot.
- Hardware and Software Integrity for NFV related platforms.
- Validation of hardware components.
- Restriction of interfaces.
- Process isolation.
- Effective and appropriately secure logging/reporting/crash management.
- Control of 'Root' account or equivalents.
- OAM access is authenticated and isolated as appropriate.
- Availability of patching/software update process.
- Management of logical entities in terms of physical and (potentially) legal constraints.

The present document intends to promote the minimum set of security features that telecommunications network equipment subject to LI or RD operations should have, and operators should expect, regardless of whether the vendor wishes to undergo an assurance process.

The establishment of a baseline will also simplify establishing security principles for more specific network equipment. For example, the baseline would be a natural place to start when establishing security principles/requirements for NFV hosts.

1 Scope

The present document treats the Lawful Interception (LI) and, where relevant, Retained Data (RD) capability being virtualized, taking into account the legal and physical challenges of doing so. This initial study is focused on the LI and RD aspects and establishes the fundamental security principles for generic platforms upon which the related groups can build. It includes a minimum set of security principles for those generic telecommunications platforms that are subject to LI that will allow the virtualized network functions to utilize the features necessary to afford them appropriate protection and at the same time to undertake appropriate activities (LI and RD). Establishing such a baseline will help the industry as a whole to be better protected against Cyber threats.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the reference document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

Not applicable.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI TS 101 331: "Lawful Interception (LI); Requirements of Law Enforcement Agencies".

[i.2] ETSI GS NFV-SEC 003: "Network Functions Virtualisation (NFV); NFV Security; Security and Trust Guidance".

[i.3] S. Cadzow: "Secure Cryptographic Mechanisms - entropy and randomness".

NOTE Available at http://www.tvra-tools.eu/blog/technology/cryptography/secure-cryptographic-mechanisms-entropy-and-randomness/.

[i.4] T. Ristenpart and S. Yilek: "When Good Randomness Goes Bad: Virtual Machine Reset Vulnerabilities and Hedging Deployed Cryptography", ISOC, 2010.

[i.5] Z. Gutterman, B. Pinkas and T. Reinman: "Analysis of the Linux Random Number Generator".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

Element Management System (EMS): management function for a VNF

Hosting Provider: entity that owns and/or runs the NFVI and is assumed to provide the interfaces for an operator to manage their VNF

Network Functions Virtualisation Infrastructure (NFVI): totality of all hardware and software components that build up the environment in which VNFs are deployed

NFV Management and Orchestration (MANO): component consisting of the Orchestrator, Virtualized Infrastructure Manager and VNF Manager

NOTE: It may additionally contain other management related systems as necessary (e.g. including but not limited to security orchestration).

Operator: runs the network and manages the VNFs

Orchestrator: component in charge of the management of NFV infrastructure and software resources

Virtualized Infrastructure Manager: allocates resources to the NFV infrastructure (i.e. energy efficiency)

Virtualized Network Function (VNF): software implementation of a network function

VNF Manager: in charge of the lifecycle of an NFV

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

AUC	Authentification Centre
CPU	Central Processor Unit
EMS	Element Management System
GPU	Graphics Processing Unit
GSMA	Global System for Mobile communications (GSM) Association
IN	Intelligent Network
IPC	Inter-Process Communication
ISG	Industry Specification Group
ISO	International Standards Organisation
LEA	Law Enforcement Agency
LEMF	Law Enforcement Monitoring Facility
LI	Lawful Interception
MANO	Management and Orchestration
NFS	Network File System (protocol)
NFV	Network Functions Virtualisation
NFVI	Network Functions Virtualisation Infrastructure
OAM	Operatjion and Maintenance
PRNG	Pseudo Random Number Generator
RD	Retained Data
SDN	Software Defined Networking
SMS	Short Message Service
SSL	Secure Sockets Layer
TC	Technical Committee
TPM	TrustIed Platform Module
VM	Virtual Machine
VNF	Virtualized Network Function
VNFI	Virtualized Network Function Infrastructure

4 Problem statements

4.1 In support of LI and RD for NFV

The following requirements derived from LI and RD work in both ISG NFV and TC LI describe the challenges facing NFV. The following general security issues need to be considered:

- The LI VMs are likely to have to exist in a separate security domain.
- The VM hypervisor administration often needs to enable the LI functionality but once paired to an external LI administration system it needs not to be possible to disable it again without authorization by the LI administration system.
- All VMs in the array may need to use code signing integrity protection to prevent unauthorized and/or undetected VM module code changes.
- The LI administration system needs to be able to dual sign the LI VMs and check the signatures.
- Each LI VM may need to be pairable using separate security keys.
- All events performed by the LI VMs need to generate logged events which can be read by the LI administration system (prevents changes to VMs outside of audit times). Log events which result from events occurring within the LI VMs need to only be visible to the LI administration system and not the hypervisor or other VMs/administrators.

4.2 For LI in NFV

- **Virtual Network Functions can be verified** - the intent behind this principle is that VNFs are signed and that the static component of the VNF can be verified during boot, run-time, suspension and transfer. Additionally, data used by the VNFs is properly structured and hence its integrity can be checked.
- Data used by the VNF is properly structured (according to known schemata), can be integrity checked (e.g.: signing, etc.) and is tamper-proof.
- **The virtualization layer contains a security component** - the intent behind this principle is that security is a central component of the virtualization layer. In addition, the virtualization layer needs to facilitate the integrity scanning of VNFs and the monitoring of network links.
- **Host platforms are updatable and auditable** - the intent behind this principle is that host platforms are limited in functionality and that this functionality has been scrutinized and kept up-to-date.
- **The virtual and physical architecture is managed** - the intent behind this principle is the operator is able to define a secure virtual architecture that will be implemented by the host infrastructure. Furthermore, operators need to be able to place security constraints on the physical architecture that is hosting the virtual network.
- **NFV management needs to incorporate security management** - the intent behind this principle is that the NFV management system is the entity which is able to fully understand the risks to the architecture. As such, it needs to be able to ensure that components of the system are instructed to protect data appropriately and that the monitoring processes are in place to detect any compromise of the system. The target list is critical information, which should be protected such that only the appropriate LI functions are entitled to read or modify this information.
- **The location is verified and evidential** - the location of any LI activity is known to ensure LI activity takes place where it can be protected and is legal to do so. This is critical to LI functionality. An essential requirement is that the LI takes place in the country/jurisdiction, or authorized countries/jurisdictions, that issued the authorization (this means that any LI-VMs plus the network function VMs that are being monitoring need to be in-country).
- **The virtual machine only operates with "known" LI virtual machines** - if an LI VM is instantiated it can only operate with the VM's it is allowed to. This implies the use of techniques described elsewhere in the present document e.g. software signing, attestation, trusted elements.

- **The target list and related signalling is appropriately protected -** the target list needs to be encrypted, as may the output traffic on X1. Any encryption needs to be robust, and not under full control within the VM layer.
- **Timestamps** - Synchronization needs to be built into the fabric of NFV by way of either physical layer functions on the NIC or NID or Operating System support for precise time. NFV could also enable a more tightly synchronized network by making precise time a generic function available everywhere.
- **The NFV system has to provide a source of true random numbers** - the intent behind this is to avoid the problem of weakened cryptography due to inherent problems in pseudo-random number generation in virtual machines.

5 Transposed LI and RD requirements into NFV platforms

5.1 Attestation

- User wants to store an attribute (e.g. hardware ID, location, etc.) and user needs to subsequently trust when that attribute is recalled/requested.
- User needs to verify arbitrary system information (e.g. VNF integrity).
- User needs to verify authentication of arbitrary system information (e.g. is VNF signed by a trusted party).

NOTE: User in this context means management entity or network operator.

5.2 Data at rest encryption

Although this could be implemented higher up the stack, to be done well it may require hardware support, for the encryption of target lists.

Selectors used to identify traffic to forward to the LEMF are sensitive and have to be stored in a secure manner. Access to these selectors should be restricted to those who should have access, and an audit record should be kept of the addition, update, access and removal of the selectors. The system used to store the LI selectors has to support secure distribution of those selectors to the parts of the network that need them to perform LI. The storage mechanism chosen should be resistant to a compromise of the virtualization layer.

5.3 Data in Transit Encryption

Similarly to data at rest encryption, all data in transit should be secured using suitable meanings in an end-to-end manner. If end-to-end security cannot be achieved then any break should be via a trusted entity.

5.4 Verified, Trusted or Measured Boot

Ensures system integrity. A secured boot is a process where the integrity of various components in a boot sequence have been measured and found to be either:

- in accordance with expected values or;
- within tolerable ranges for the required host profile.

See ETSI GS NFV-SEC 003 [i.2], clause 4.4.5.1 for further details.

This is important for Sensitive Application Functions. The VNFI may be to verify integrity of databases, static configuration data and application root key chain (e.g. AUC)

5.5 Tamper Evidence and Resistance

Databases, communication, storage should have resistance to data tampering, and also it should be evident if the data has been tampered with.

5.6 Physical telemetry

Provide user (e.g. MANO) with physical telemetry data (e.g. processor usage, network usage, etc.). This could potentially be used to verify any discrepancies between what the hypervisor and hardware report.

5.7 Secure encryption

A location to store keys in an appropriate manner ensuring they cannot be compromized by e.g. a privileged hypervisor manager or errant process.

5.8 Secure execution

Supporting location attestation by ensuring LI and RD operations can only take place on known hardware. Code execution toolsets can help in this case. Some of these are proprietary to certain vendors, but protected mode architecture to create a privilege separation between operating systems and applications appears a very useful concept in achieving this aim.

Certain LI functions will require that code be executed in a secure execution environment so that the code execution is secure. This may include the decryption of LI target identifiers received from a law enforcement agency or the signing of LI VNFs prior to deployment (so that the MANO layer can assert that they have been deployed in a known-good state and have not been modified in transit). This code should run externally to the virtualization platform as a compromised hypervisor could inspect/manipulate instructions.

5.9 Secure Cryptographic Mechanisms

The flexibility of virtual machines introduces new challenges in ensuring randomness, as highlighted in [i.3]. Operations that were previously not possible with hardware machines, such as snapshotting and cloning, mean that the method by which non-repeated pseudo-random byte-streams were guaranteed (by writing a seed to disk before shutting down), may no longer work [i.4]. Indeed it has been demonstrated that it is possible to predict with some degree of accuracy what the pseudo-random byte-stream will be for a non-running virtual machine, given sufficient knowledge of what sources of entropy are used and the ability to inspect them.

Virtual machines make possible the following situations that can negatively affect the generation of random numbers:

- Re-use of state from template images.
- Re-use of state from a non-persistent virtual machine re-starting in a previous state (the so-called reset vulnerability [i.4]).
- Inspection of the state of a paused/suspended virtual machine.

5.10 Re-use of State from Templated Images

It is expected that telecoms equipment vendors or operators will maintain a library of virtual machine template images for the provisioning of VNFs. Any two (or more) virtual machines created from a single template will initially start in the same entropy-state and thus will all generate the same random byte-stream until additional machine-specific entropy has been gathered.

5.11 Random Number Generation, Seeding and Operating System Devices/Components

Random number generation and seeding have to be guaranteed to be random. For example, a VM might be created for a template with a deliberately "broken" /dev/random, then that VM template may be cryptographically signed and cryptographic hash distributed to various vendors. Then each instantiation of that VM would then effectively be using the same random number seeding and random number generation/device - all trusted too.

5.12 Reset Vulnerability

The virtual machine reset vulnerability is similar to the re-use of state problem above. Virtual machine state can be preserved in a snapshot - this preserves the state of the disk, memory, etc. Although the virtual machine will continue to run after the snapshot, it is possible to revert the state back to the state contained within the snapshot. This may be done for a variety of reasons but the outcome is to revert to a 'known good state'. A side-effect of preserving the state of a virtual machine is that the entropy-state will also be stored meaning that every time the virtual machine starts from a particular snapshot it will generate the same random byte-stream until further entropy has been gathered [i.1].

5.13 Inspection of State

A virtual machine that is paused or suspended can have its state examined by external tools. A knowledge of the random number generation process (which, for open-source operating systems, is available to anyone who can understand the source code) and the machine state means that it is possible to predict with reasonable certainty the next random number (and possibly some more after that) that the pseudo-random number generator will generate.

5.14 Memory Inspection

The general case of inspection of state above relates not just to investigation of the random number system but all data stored within that VM.

Techniques such as randomized memory layout may complicate the investigation and reconstruction of data, however with the ability to see the whole memory of a VM, one can also see the operating system's memory allocation tables thereby compounding this ostensibly, security feature.

The hypervisor environment should therefore provide means to prevent unauthorized suspension and memory inspection if even possible.

5.15 Secure Cryptographic Mechanism Summary

For a period after boot following a clone from a template or roll-back to a snapshot, a virtual machine will generate predictable random numbers. It is not possible to define how long this period is. The criticality (or otherwise) of this behaviour will depend on the application for which the virtual machine is intended, however given that NFV deployments may be highly transitory, it would be unwise to assume that this is a problem that can be ignored. Virtualized network functions should use a source of entropy external to their own environment to source a standardized random number generator (for example those specified in GSMA or ISO standards, as these have been studied and proved to be secure) to mitigate against the attacks above.

6 Trust models

6.1 Secure Cryptographics Mechanisms - Entropy and Randomness

Cryptographic security is non trivial and requires an understanding of a number of concepts. Two of these are entropy and randomness and they are strongly interlinked. In communications security there is often an assumption that any transmitted message will have some variation to all other transmitted messages. However, in some communications systems there may be very strong levels of commonality between transmitted messages that may be exploited. A goal of cryptography is to mask the similarity between messages and commonly this is referred to as maximizing the entropy of a transmitted message. If a message is to be encrypted and it has a low entropy, the cryptographer has to raise the entropy prior to encryption, or as part of the encryption process. There are a number of examples of messages with an inherently low entropy e.g. SMS, Telematics status messages, Call setup messages.

Mathematical sources often discuss message entropy, but at the root is Shannon's "A Mathematical Theory of Communication". Essentially, if the attacker knows or guesses that the message can take a small set of values, the probability of correctly guessing bit n+1 after receiving bit n tends towards 1, whereas for a random binary alphabet the probability of a correct guess should always be 0,5. In a cryptographic context, where Alice is sending a message m to Bob in the form of a binary string, the rule of thumb is that the bigger the entropy of the message m, the more guesses required by an attacker to guess m. After encryption of message m to generate message c, the entropy of c should be as high as possible.

The rule of thumb for randomness is that if an attacker that can get access to all the historic random elements (all n values) this has to give zero information to correctly guess the value of the (n+1)th element. This, the ability for a system to generate random numbers is central to most forms of modern cryptography. Typically cryptographic suites such as OpenSSL rely on operating system provided methods for sourcing random numbers. Examples include /dev/(u)random on unix based platforms, and rand)s() on Windows. None of these methods provide truly random numbers - that would require a physical source of randomness. Instead, they are seeded by sources of entropy within the system and subsequently updated periodically. Typically when an operating system is restarted, a pseudo random seed is written to disk and this seed is used as an additional source of entropy when the operating system starts again. This ensures that the machine will not boot in the same entropy-state that it has booted in previously. Systems providing insufficient randomness have been shown to compromise the integrity of security suits running on them e.g. one vendors implementation of SSL.

Analysing pseudo random number generators is a difficult task. The Linux PRNG for example is part of the kernel, so any modifications to enable introspection require that the kernel be recompiled. This may affect kernel provided entropy sources in such a way that the experimentation no longer represents what would happen on an unmodified kernel. The source code for the Linux random number generator is available, but is poorly documented and not up to date (Gutterman, Pinkas, Reinman [i.5]).

6.2 Basic Model of a Virtualized System

For the purposes of this discussion the present document uses the informal model in figure 1 to define terminology and relationships between elements. This is not meant to be an architecture but rather a framework in which concepts can be defined.

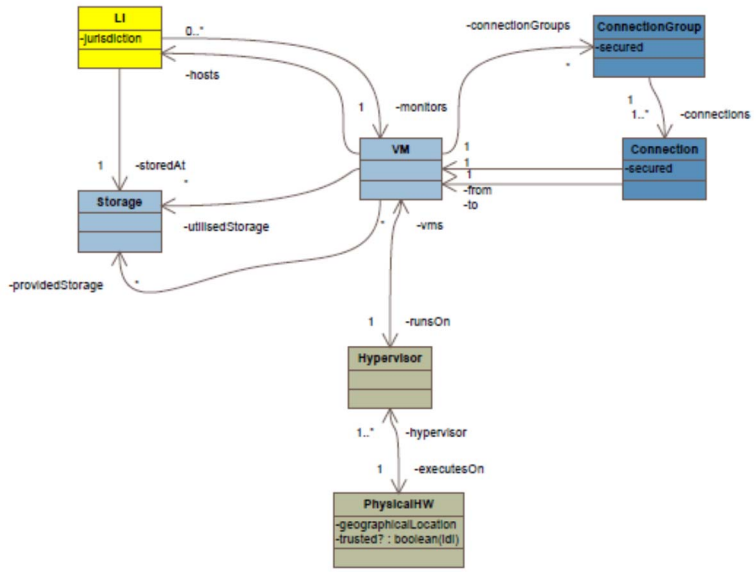

Figure 1: Class Diagram of VM and LI Concepts

It is assumed that a physical hardware element has a geographical location assigned by some means, for example asset tagging and an associated system, or via a trust mechanism such as a TPM. Differentiation between physical hardware that exists within a trusted or untrusted environment in this respect is then clear. Further, additional formal rules cannot be provided but the above diagram is sufficiently self-explanatory for this discussion.

Some points on the conceptualization:

- A piece of physical hardware (CPU, Memory, Physical Disks, etc.) will run at least one hypervisor which in turn will run zero to many Virtual Machines.
- A generic concept of connection is presented which represents all the various forms of connection between any pair of VMs. This could be IPC, network connections, etc. of any sort:
  - A connection group is secure if and only if all individual connections within that group are secure - this can mean either/both hop-by-hop or/and end-to-end security as appropriate. The term "connection group" is used in a generic or abstract sense to capture the communication mechanisms, protocols and infrastructure between any two components.
  - No definition of "secure" is provided other than its common sense meaning.
- The description abstracts away how storage devices are provisioned by and to VMs and just state that they are rooted in some VM. In reality the provisioning could be from a dedicated storage appliance to a virtual disk or NFS mount, etc.
- It is asserted that a VM utilizing storage being provided by another VM implies that a connection exists between the two VMs:
  - This can be generalized further in that a whole connection path exists via intermediate VMs organized as a connection group.
  - A simple direct connection between two VMs implies that the connection group has a single connection.
  - A complex connection between two VMs implies that the connection group has multiple connections forming a path via one or more intermediate VMs. This allows us to consider cases where intermediate VMs process the connection, ie: a notion of session or "connectionfulness". Such intermediate VMs could be firewalls, SDN routers or other kinds of "helping" (or otherwise) functionality.
- The concept of LI here denotes all the necessary functionality for enabling, managing and providing an LI trace:
  - An LEA monitors a VM, meaning some functionality, entity, person contained therein and defined by the scope of the LI itself.
  - The data collected is stored in some storage facility.
  - The LI entity could be some VM, VM functionality, etc. but this is not modeled deliberately to simplify this model.
  - The LI entity is issued within the scope of some jurisdiction which defines a given geographical area.
  - The LI functionality is hosted by some VM, which may or may not be the same VM that is being monitored.

Virtual Machine Mobility

As VMs can be moved between data centers, hypervisors, etc. as load balancing and other requirements dictate, the addition of LI therefore complicates this in that it restricts the mobility. In the coarse grained case mobility is according to an algorithm such as that given below:

```
moveVM( v:VM, h:Hypervisor ):
// v is the VM to be moved, h is the target hypervisor
pre:
    //avoid the nonsensical, already running on this hypervisor case
    v.runsOn != h
post:
    v.runsOn = h
```

In the case of a VM running on trusted hardware, let us assume that it is needed to ensure that the VM is moved to similarly trusted hardware:

```
moveVM( v:VM, h:Hypervisor ):
pre:
        v.runsOn != h
        h.executesOn.trusted = true
then
        v.runsOn = h
```

In the above case, the necessary trust measurements have to be valid. If the VM in question is being moved with respect to a geographical location then the following is the case:

```
moveVM( v:VM, h: Hypervisor );
pre:
        v.runsOn != h
        h.executesOn.trusted = true
        h.executesOn.geographicalLocation IN v.hosts.jurisdiction
then
        v.runsOn = h
```

It can be seen here that the extension pre-condition required is to check that the geographical location of the target physical hardware is still within the jurisdiction of the hosted LI monitoring.

Requirements

The following requirements (non-exhaustive) therefore need to be considered:

1) The physical hardware running the VM (and hypervisor) has to be at a geographical location within the scope of the LI's jurisdiction. This applies to both the VM being monitored and the VM running the LI entity (in case the two VMs are different). For example:

   - If the LI is issued in Finland and the VM being monitored is currently running on hardware at a data center in Helsinki then this would be acceptable, as Helsinki is within Finland.

   - If the LI is issued in Finland ad the VM being monitored is currently running on hardware at a data center in London then this would not be acceptable, as London is not within Finland.

2) The VM that provides the storage that is utilized by the LI has to be within the geographical location within the jurisdiction of the LI:

   - The storage and management of the storage therefore have to be within the geographical area defined by the LI jurisdiction.

3) The VM being utilized to store the data from the LI has to be within the geographical location within the jurisdiction of the LI:

   - This allows the LI to utilize a VM that does not directly host the storage, but mounts it from some other VM (appliance, device, etc.).

4) There has to be a connection path between the VM being monitored and the VM or appliance that is running the LI, if they are not the same VM.

5) It has to exist a connection path between the VM utilizing the storage and the VM providing the storage, if they are not the same VM.

6) If a connection exists between the VM hosting the LI and another VM then this connection has to be secured by some means:

   - The implication of this is that the whole path, ie the connection group is secure.

   - The connection therefore might utilize any number of intermediate VMs.

7) No intermediate VM within the VM's that form of the connection path within a connection group may store information:

   - This might be difficult to achieve if caching is considered, etc. but captures the notion that only the utilized and provided storage of the target VM that the LI stores its collected data at is the only correct place to store said data.

   - Note the differences here between hop-by-hop and end-to-end security.

8) The physical hardware that hosts the VM that itself hosts the LI entity should be trusted using some suitable TPM or similar mechanism:

   - This trusted mechanism has to be able to provide the geographical location.

   - A trusted geographical location could be provided by means such as trusted GPU, administrative means, secure geo-tagging. What form is for discussion.

9) The physical hardware that hosts the provided storage utilized by the LI entity should be trusted using some suitable TPM or similar mechanism:

10) Movement of the physical hardware should invalidate the embedded geographical location in the TPM, and also cause any TPM measurements related with this to fail:

   - This could possibly be combined with asset management.

   - How to achieve this is for discussion.

11) VM migration:

   - A VM under monitoring as well as a VM running the LI entity can be migrated but only to sufficiently trusted places.

   - The geographical location of the target hardware has to be compliant with the jurisdiction of the LI entity.

12) The physical hardware that hosts the VM that itself hosts the LI entity should be trusted using some suitable TPM or similar mechanism:

   a) This trusted mechanism has to be able to provide the geographical location.

   b) A trusted geographical location could be provided by means such as trusted GPU, administrative means, secure geo-tagging.

13) The physical hardware that hosts the provided storage utilized by the LEA should be trusted using some suitable TPM or similar mechanism.

14) Movement of the physical hardware should invalidate the embedded geographical location in the TPM, and also cause any TPM measurements related with this to fail:

   a) This could possibly be combined with asset management.

   b) How to achieve this? For discussion.

15) VM migration:

   a) A VM under monitoring as well as a VM running the LEA can be migrated but only to sufficiently trusted places.

   b) The geographical location of the target hardware has to be compliant with the jurisdiction of the LEA.

7 Security objectives

7.1 Host platform security objectives

Threats relevant to the host platform	
T.1.1	Damage shared physical resource
T.1.2	Gain physical access host infrastructure
T.1.3	Exploit the virtualization layer and host platform
T.1.4	Falsify reporting of the host infrastructure
T.1.5	Spoof the host infrastructure

O.1.1: The host platform needs to prevent damage to the platform from VNFs.

The host platform needs to monitor use by VNFs hosted on the platform and prevent actions which may damage the platform (e.g. over-heating, frequent memory use).

O.1.2: The host platform needs to be resistant to compromise.

The security of the host platforms is essential to the security of the wider system. Hence, host platforms need to be locked down, protected and patched regularly to reduce the risk of compromise.

O.1.3: The host platform needs to limit the VNFs it will run in accordance with the security of the platform.

Host platforms will differ in terms of the physical risk in their installed location and in terms of the protections available. If a host platform is at greater risk, it may not be appropriate for it to run sensitive VNFs (e.g. HSS/HLR). Hence the host platform should prevent these VNFs from running on the platform.

O.1.4: It needs to be possible to record and validate the LI and RD -related actions of a VNF from the host platform.

This functionality is essential to understand how a system may have been compromised and to clean up the system once a compromise has occurred.

O.1.5: The confidentiality of virtualized processing and data needs to be maintained.

The host platform is in control of the security data and processing performed within the platform. This objective may also be partly fulfilled by the virtualization layer.

7.2 Related objectives

O.1.6: Host Platforms need to be uniquely identifiable by the NFV system.

O.1.7: The MANO needs to be able to control the distribution of VNFs to host platforms based on the risks associated with those platforms.

O.1.8: The MANO system needs to verify the origin of messages from VNFs, host infrastructure and operator systems.

O.1.9: The confidentiality of messages between the MANO and VNFs, host infrastructure and operator systems need to be maintained.

History

Document history		
V1.1.1	January 2016	Publication