ETSI GR NFV-SEC 003 V1.2.1 (2016-08)

Network Functions Virtualisation (NFV);
NFV Security;
Security and Trust Guidance

Disclaimer

The present document has been produced and approved by the Network Functions Virtualisation (NFV) ETSI Industry Specification Group (ISG) and represents the views of those members who participated in this ISG.
It does not necessarily represent the views of the entire ETSI membership.

Reference
RGR/NFV-SEC003ed121

Keywords
NFV, security

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 (https://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 Group Report (GR) has been produced by ETSI Industry Specification Group (ISG) Network Functions Virtualisation (NFV).

Modal verbs terminology

In the present document "**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.

1 Scope

The present document has been developed to describe the security and trust guidance that is unique to NFV development, architecture and operation. Guidance consists of items to consider that may be unique to the environment or deployment. Supplied guidance does not consist of prescriptive requirements or specific implementation details, which should be built from the considerations supplied.

Guidance is based on defined use cases, included in the present document, that are derived from the Security Problem Statement and are unique to NFV. Relevant external guidance will be referenced, where available.

2 References

2.1 Normative references

Normative references are not applicable in the present document.

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 GS NFV 001: "Network Functions Virtualisation (NFV); Use Cases".

[i.2] CSA CloudTrust.

[i.3] ETSI GS NFV-SWA 001: "Network Functions Virtualisation (NFV); Virtual Network Functions Architecture".

[i.4] UEFI specification: Unfied Extensible Firmware Interface Forum, Unified Extensible Firmware Interface (UEFI) Specification, 2016.

NOTE: Available at http://www.uefi.org/specifications.

3 Abbreviations

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

ABAC	Attribute-Based Access Control
API	Application Programming Interface
BIOS	Basic Input Output System
CA	Certificate Authority
CDN	Content Distribution Network
CLI	Command Line Interface
CPU	Central Processing Unit
CPUID	CPU Identifier
CSA	Cloud Security Alliance
DDoS	Distributed Denial of Service
DHCP	Dynamic Host Configuration Protocol
DMA	Direct Memory Access
DNA	DeoxyriboNucleic Acid

DNS	Domain Naming Service
DoS	Denial of Service
DPI	Deep Packet Inspection
DRM	Digital Rights Management
EM	Element Manager
EMS	Element Management System
FIPS	Federal Information Processing Standards
GPS	Global Positioning System
GTP-C	GPRS Tunnelling Protocol-Control
GTP-U	GPRS Tunnelling Protocol-User Data Tunneling
GUI	Graphical User Interface
HSM	Hardware Security Module
HSS	Home Subscriber Server
HVM	Hardware Virtual Machine
IAM	Identity and Access Management
IMS	IP Multimedia Subsystem
IMSI	International Mobile Subscriber Identity
IO	Input/Output
IP	Intellectual Property
IT	Information Technology
LI	Lawful Intercept
LUN	Logical Unit Number
MAC	Media Access Control
MANO	Management and Orchestration
MME	Mobile Management Entity
NE	Network Element
NF	Network Function
NFV	Network Function Virtualisation
NFVI	Network Function Virtualisation Infrastructure
NFVO	Network Function Virtualisation Orchestrator
NIC	Network Interface Card
NTP	Network Time Protocol
OA&M	Operations, administration and management
OS	Operating System
PKI	Public Key Infrastructure
RADIUS	RADIUS protocol
RAM	Random Access Memory
RBAC	Rights-Based Access Management
SDN	Software Defined Networking
SIP	Session Initialization Protocol
SSAE	Statement on Standards for Attestation Engagements
SVA	Security Virtual Appliance
SWA	Software Architecture
TBOOT	Trusted Boot
TOR	Top of Rack
TPM	Trusted Platform Module
TXT	Trusted eXecution Technology
UEFI	Unified Extensible Firmware Interface
UUID	Unique Universal IDentifier
VA	Virtual Appliance
VIM	Virtual Infrastructure Manager
VLAN	Virtual LAN (Local Access Network)
VM	Virtual Machine
VMM	Virtual Machine Monitor
VNF	Virtual Network Function
VNFC	Virtual Network Function Component
VNFCI	Virtual Network Function Component Instance
VNFD	Virtual Network Function Descriptor
VNFM	Virtual Network Function Manager
VoLTE	Voice over LTE
VPC	Virtual Private Cloud
vSwitch	virtual Switch

VTPM Virtual Trusted Platform Module

4 Network Function Virtualisation Security

4.1 NFV High-Level Security Goals

Security is Embedded in NFV DNA

Security is defined as the state of being protected (secured) as well as those measures applied to achieve/maintain/validate protection.

The dynamic nature of Network Function Virtualisation demands that security technologies, policies, processes and practices are embedded in the genetic fabric of NFV.

Additional high-level security goals for NFV include:

- Establish a secured baseline of guidance for NFV operation, while highlighting optional measures that enhance security to be commensurate with risks to confidentiality, integrity and availability.
- Define areas of consideration where security technologies, practices and processes have different requirements than non-NFV systems and operations.
- Supply guidance for the operational environment that supports and interfaces with NFV systems and operations, but avoid redefining any security considerations that are not specific to NFV.

NOTE: NFV security considerations are very similar to hypervisor-based virtualisation security considerations in their architecture and interfaces. However, security architects and operations managers are instructed to consider use cases beyond hypervisor-based constructs to include cloud orchestration, virtual appliances and empower future innovations.

4.2 NFV Security Use Case Summaries

The following use cases describe the need for security within the VNF, between VNFs and secured interfaces and interchanges external to the VNF. The use cases are summarized for brevity, highlighting important security functions and considerations unique to NFV.

4.2.1 Intra-VNFSec: Security within Virtual Network Functions

Within the VNF, security measures and processes are required for VNF operations, for contained VNFC operations, and for secured interface with external assets and services. Specifically, this clause describes the use cases that are unique within a VNF.

4.2.1.1 VNFC-Specific Security Use Cases

Sensitive authentication data in workloads

NFV workloads routinely possess sensitive authentication data used for authenticating the workload, its processes and users. This sensitive authentication data can consist of passwords, private keys, cryptographic certificates, tokens and other secrets. This data should be protected during all phases of the NFV security and trust lifecycle and should be considered highly dynamic in nature, with updates likely during instantiation, hibernation/suspension, and VNF retirement. NFV workloads containing sensitive authentication data reside within and may be described as VMs, VAs, VNFs and VNFCs. Guidance for this use case should describe the processes, procedures and technologies unique to NFV that would satisfy the use case, pointing to external best practices where available.

Function and capability authorization control for VNFs

There are many functions and capabilities that will be provided by various parts of a VNF and various different entities within NFV may request that these functions and capabilities are employed. It is not always appropriate to provide authorization for an entity to access these, even when the same entity has previously done so. Authorization for use of these functions and capabilities may be controlled by a number of techniques and across a number of variables, including identity, trust, joint or delegated decision making and API security.

Guidance for this use case should describe the key technologies for use in the context of authorization control for VNFs, and how they may be used within an NFV context.

4.2.1.1.1 VNFC Creation

The creation of a VNFC will require updates to networking, credentialing, encryption, licensing, configuration and other settings unique to the new VNFC that impact security. Creating a VNFC can be accomplished in one of the following ways:

- The instantiation of a newly-defined VNFC.
- The instantiation of a VNFC with pre-configured state the cloning of an existing VNFC.

Guidance for these use cases should describe update and verification processes and procedures, virtual asset tracking and audit records.

4.2.1.1.2 VNFC Deletion

The retirement and deletion of a VNFC and its VNFCIs will require updates to networking, credentialing, encryption, licensing, configuration and other settings unique to VNFC removal that impact security. When requests for secured wipe and verified destruction are made, the actions taken should be forensically sound. When a VNFC is to be made unavailable, for re-use or re-creation, deletion of all possible instances (VNFCIs) should be verified across backups and archives, cloned images and other copies.

Guidance for this use case should describe update and verification processes and procedures, virtual asset tracking and audit records. Asset management should ensure certificate revocation and updates of IP whitelisting/blacklisting.

4.2.1.1.3 VNFC Configuration and Package Management

The updates to a VNFC and associated VNFCIs include patching, updating, new/modified software packages and configuration changes. These changes can include:

- Patching of the operating system, drivers and virtual machine components.
- Adding dynamic updates to the configuration (DNS, DHCP, etc.).
- Management of virtual machines and virtual appliances, including security virtual appliances.
- Updates to event-based configuration guidance, such as whitelists and blacklists.
- New versions of application software, software frameworks (e.g. Java) and software components.

Guidance for this use case should describe update and verification processes and procedures.

4.2.1.1.4 VNFCI Migration

Migrating a VNFCI is desired for maintenance of underlying VNF infrastructure, failover in the event of VNF infrastructure failure and disaster recovery in the event of a site failure condition. Migrations are often performed as "live migrations" that should not incur downtime to the operations of the VNFC when correctly functioning.

Migration concerns include memory reuse, feature parity, configuration compatibility and service availability.

4.2.1.1.5 VNFC Operational State Changes

Operational state changes (planned and unplanned/intentional or unintentional) can significantly affect VNFC security. A partial list of operational state changes includes:

- Hibernation, sleep, resumption, abort, restore, suspension.
- Power-on and power-off (either physical or virtual).
- Instantiation, whether pre-configured or not.
- Patching and maintenance.
- High-availability, recovery and data-in-motion changes during live migration.
- Integrity verification failure, crash and OS compromise.
- Retirement and termination.

Guidance for this use case should describe integrity verification processes and procedures including logging and audit.

4.2.1.1.6 VNFC Topology Changes

Topology changes that affect the security of the VNFC can result in loss of communication, unintended traffic flows, loss of intended traffic flows and other issues including:

- Network IP address and VLAN updates.
- Service chaining.
- Failover and disaster recovery.

Guidance for this use case should describe awareness of topology changes and resiliency.

4.2.1.1.7 VNFC Scale-Up and Scale-Down

The scale-up and scale-down of a VNFC affect sizing and can alter the memory, storage and processing requirements, resulting in differences in class of service, monitoring thresholds, performance thresholds and backups. Scale-up and scale-down are also referred to as vertical scalability.

Guidance for this use case should describe architectural and operational changes associated with increased/decreased requirements for the VNFC due to scale-up/scale-down.

4.2.1.1.8 VNFC Scale-In and Scale-Out

Scale-in and scale-out of a VNFC affects multiple resources (e.g. services and communications) that spread the VNFC workload, resulting in differences in class of service, monitoring thresholds, performance thresholds, networking and backups. Scale-in and scale-out are also referred to as horizontal scalability.

Guidance for this use case should describe architectural and operational changes associated with increased/decreased requirements for the VNFC due to scale-in/scale-out, as well as dependencies between systems utilized for scalability.

4.2.2 Infra-VNFSec: Security between Virtual Network Functions

Virtual Network Functions that communicate directly with each other have special security needs, as network-level security enforcement is often not inherent in the communication path. Characteristics include:

- Secured orchestration for and between VNF domains.
- Flows are often not through firewalls or other network policy enforcement points.
- Virtual Appliances and Security Virtual Appliances need to be configured to be part of the traffic flow.
- Service chaining capabilities need to be enforced, if available.
- Requires strong VNF-VNF security measures and individual VNF resiliency to attack.

4.2.3 Extra-VNFSec: Security external to Virtual Network Functions

The security of VNFs is reliant on the security of the physical infrastructure, environment and external services. The following use cases identify key issues external to the VNF that directly impact VNF and VNFC security.

Regulatory and jurisdictional impact on NFV deployments

NFV deployments will exist, as current telecommunications services do, in a regulatory and jurisdictional environment. The virtualisation of network functions leads to new requirements both on the VNFs themselves and on the management and orchestration components with which they are controlled. Issues include Lawful Intercept, Auditing and Service Level Agreements, and although there are many similarities to existing practise, the advent of NFV brings some changes.

In addition, future NFV deployments may increasingly be spread across borders, leading to multiple sets of requirements being placed on operators. The ability to administer services across borders and to migrate services in real-time between different jurisdictions presents further challenges.

The trust and security document will identify key legal and regulatory issues and address appropriate processes and technologies.

Authentication, Authorization and Accounting for NFV

NFV deployments will be complex, with multiple administrative domains within the same deployment, for example:

- NFVI - comprising:
  - Network(s)
  - Hypervisor
  - Compute
  - Storage
- SDN
- Service network
- VNFM
- Orchestration

The authentication, authorization and accounting requirements across these domains will be different, some having regulatory requirements, for instance. In addition, there will be a mix of human and system entities requiring services.

In some deployments, there will be requirement for external parties - such as other operators - to be able to access and administer parts of the NFV deployment, and this will also include access to authentication, authorization and accounting services.

Although each NFV deployment will be different, there will be some common technologies, features and best practices. The trust and security document will identify and describe these.

4.3 NFV External Operational Environment

These are items of consideration for the external operational environment that are unique to supporting Network Function Virtualisation. Included are physical security, hardware, services, policies and practices.

4.3.1 External Physical Security Guidance

A referenced standard for physical security should be described and documented to support NFV needs. This may include facility (i.e. SOC2, SSAE 16) specialized hardware (i.e. FIPS, TPM) and other considerations that are relied upon for NFV for confidentiality, integrity availability and audit.

4.3.2 External Hardware Guidance

- Discuss Trusted Computing Base.
- Include the use of physical taps as required for Lawful Intercept.
- Describe VNF usages of FIPS and HSM.
- Other hardware advantages? Requirements?

4.3.3 External Service Guidance

4.3.3.1 DNS

Ensure the ability to update newly instantiated, suspended, hibernated, migrated and restarted images with relevant DNS information.

4.3.3.2 IP Addressing, DHCP and Routing

Ensure the ability to update newly instantiated, suspended, hibernated, migrated and restarted images with relevant IP addressing, including routing tables, route health information and whitelists/blacklists. A VNF that is acting as a router or DHCP server should be validated before routes and addressing updates are propagated.

4.3.3.3 Time Services and NTP

Ensure the ability to update newly instantiated, suspended, hibernated, migrated and restarted images with current time and time zone information. Log all changes to time, date and time zone. Log changes to time server source.

4.3.3.4 Geolocation

Ensure the ability to update newly instantiated, suspended, hibernated and restarted images with relevant geolocation information. Log all changes to geolocation along with the mechanisms and sources of location information (i.e. GPS, IP block, and timing).

- Discuss Geolocation sources.

4.3.3.5 Security Visibility and Testing

This clause encompasses all of the facilities outside of NFV used for security monitoring, vulnerability scanning, penetration testing, and NFV security monitoring and reporting.

- External visibility into VNF and VNFC.
- External components of Introspection services Monitoring, Logging, Reporting, Analytics and Auditing.

- List security monitoring use cases.
- Lawful intercept.

4.3.3.6 Certificate Authority

In order to allow trust relationships to be built, maintained and revoked, a Certificate Authority (CA) is considered a key service within any given NFV context. There are a number of issues that should be addressed in the choice of CA and how it is deployed and run:

1) There are a great deal of operational best practices associated with running a Certificate Authority, and industry standards should be consulted. These include physical security, operational processes and Human Resources concerns.

2) Though a single NFV deployment may exist in a variety of geographical locations, different jurisdictional and legislative topologies may necessitate separate CAs in different geographical locations.

3) There are several reasons for utilizing a CA at a level *above* that of a single NFV deployment or even a single carrier's organizational unit. These include:

a) To allow certification of specific VNF components (VM templates, VNFDs, etc.) across various vendors.

b) To allow for certification of hardware components and certificates associated with hardware.

c) To allow hosting of VNFs or VNFCIs on NFV deployments owned and/or operated by organizations other than the carrier to whom those components belong.

d) To allow for management and orchestration of various NFV components on NFV deployments owned and/or operated by organizations other than the carrier to whom those components belong.

4) The use of single and multiple Certificate Authorities should be architected around defined NFV-specific trust objectives as outlined in the trust fabric.

4.3.3.7 Identity and Access Management

The external Identity and Access Management (IAM) systems should be capable of managing credentials in newly instantiated, suspended, hibernated and restarted images, as well as credential management for retired VNFs.

4.3.4 External Policies, Processes and Practices Guidance

This clause describes the unique aspects of policies, process and practices that should exist in the external environment to support and maintain NFV security.

4.3.4.1 Regulatory Compliance Considerations for NFV

Ensure that guidance for regulatory compliance allows for virtualised operation and does not require physical resources.

4.3.4.2 Forensic Considerations for NFV

Ensure that guidance for forensics allows for virtualised operation and does not require physical resources.

4.3.4.3 Legal/Lawful Intercept Considerations for NFV

Ensure that guidance for lawful intercept allows for virtualised operation. Ultimately, this will not require physical resources (taps), however an interim hybrid solution of physical and virtualised means to satisfy LI requirements may exist for years.

4.3.4.4 Considerations for NFV Analytics and Service Level Agreements (SLAs)

Ensure that guidance for analytics and the satisfaction of SLAs allow for virtualised operation and does not require physical resources.

4.4 NFV Security Management Lifecycle

The NFV Security Management Lifecycle represents the security processes spanning from NVF platform guidance, through VNF instantiation, operation and retirement.

This clause is intended to represent a *process* view of NFV security and supply associated process guidance.

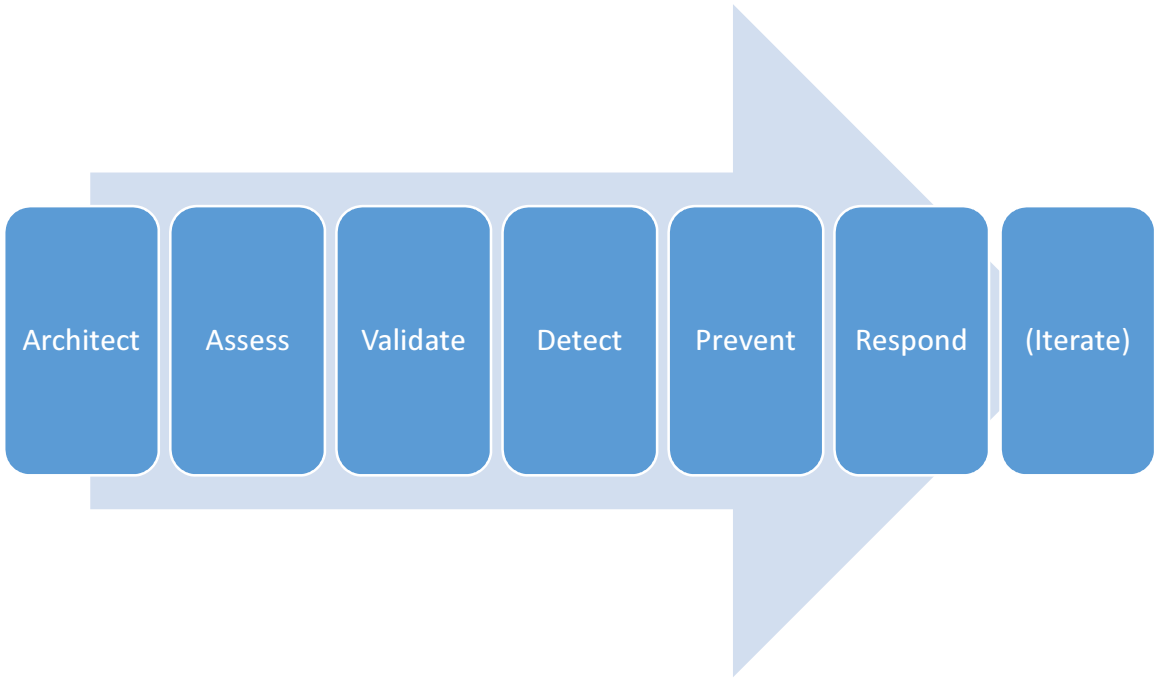

Figure 1: NFV Security Management Processes

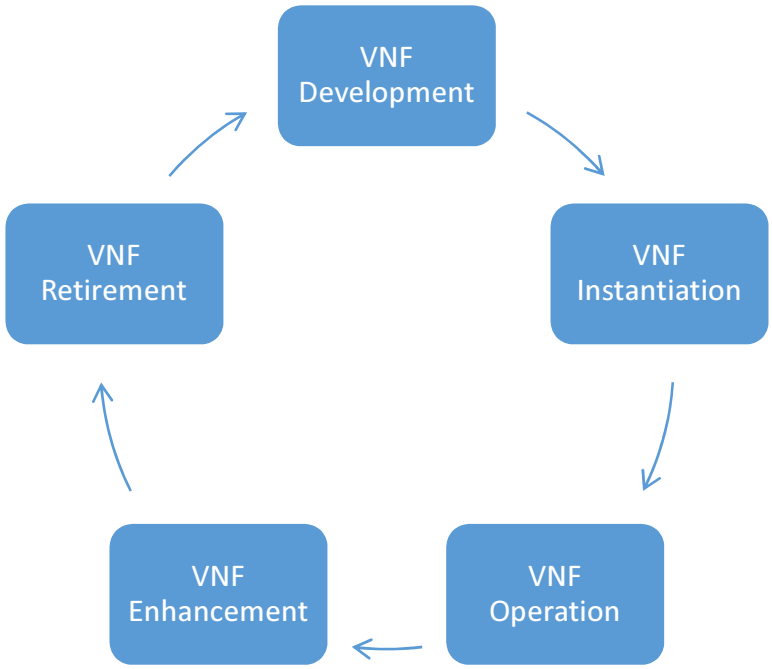

Figure 2: VNF Lifecycle

4.4.1 NFV Threat Landscape

This clause describes the unique threats associated with NFV.

4.4.1.1 Threat Vectors, Monitoring and Detection

This clause describes various threat vectors which may be relevant for components and deployments such as EPC, IMS and other use cases (e.g. RGW, CDN, Virtualised Firewall). Guidance is also provided that describes how these threats can be detected and mitigated in a virtualised environment.

Loss of availability:

- Flooding an EPC interface/Element: Attackers flood an interface/network element resulting in DoS condition in signalling plane and data plane (e.g. multiple authentication failure on s6a, DNS lookup, Malware).
- Crashing EPC network elements: Attackers crash a network element by sending malformed packets, buffer overflow.

Loss of confidentiality:

- Eavesdropping: Attackers eavesdrop on sensitive data on control and bearer plane.
- Data Leakage: Unauthorized access to sensitive data on the server (HSS profile, etc.).

Loss of integrity:

- Traffic modification: Traffic modification via variation of Man-in-the-Middle attack. Attackers modify information during transit (DNS redirection, etc.).
- Data modification: Attacker captures admin credentials which facilitates unauthorized access to EPC network elements and installs Malware.
- Attackers modify data on network element (change the NE configurations).

Loss of control:

- Control the network: Attackers control the network via protocol or implementation flaw.
- Compromise of network element: Attackers compromise of network element via management (OA&M) interface.

Insider attacks:

- Insiders make data modification on network elements; make unauthorized changes to NE configuration, etc.

Theft of Service:

- Attackers exploit a flaw to use services without being charged. For example, attacker exploits a flaw in HSS/PCRF/PCEF to use services without being charged.

4.4.2 NFV Platform Guidance

This clause describes guidance for the hardware, software and service platform that directly supports NFV resources.

4.4.2.1 Platform visibility and validation

Platform visibility and validation describes the mechanisms to view and verify resources and services within the NFV environment. These capabilities are typically utilized to validate running processes, for workloads to have visibility into their operating environment and resources, as well as for introspection into the virtual environment.

4.4.2.1.1 Workload Visibility into Physical and Virtualised Resources

Workloads, including virtual machines, virtual appliances and VNFs need to have carefully prescribed interfaces into physical and virtualised resources to ensure appropriate visibility. In some instances, it is not permissible or desirable for a workload to have any visibility or knowledge as to the operating environment and whether the workload is running virtualised. In other instances, workload visibility and knowledge of select or total environmental aspects - including virtualisation aspects - may be required.

Virtual abstractions, privacy, security and external management are often cited as reasons to limit workload visibility into the physical and virtualised operating environment. The direct use of physical resources such as a TPM or HSM, as well as practices including lawful intercept may require total or limited workload visibility into the physical and virtualised operating environment. Requirements for workload abstraction and/or visibility should be carefully architected into the NFV system, clearly documented in policies and APIs and considered in their implications to security, privacy and lifecycle management.

In 'Formal Requirements for Third Generation Architectures, the authors define three characteristics of a virtualised environment: equivalence, resource control and efficiency. The first of these, equivalence, implies that an application (and Operating System) running within a virtualised environment should have no knowledge of the fact that it is running "virtualised". However, for reasons of efficiency and performance, most virtual machines are run in at least a partly "paravirtualised" environment, meaning that the Operating System has - or can derive - at least some knowledge of the fact that it is running virtualised.

Hypervisors attempt, however, to isolate most of the physical and software properties and of the hardware compute environment on which the virtual machines are operating. These properties include but are not limited to versioning information of physical BIOS and hypervisor, location and arrangement of physically installed components such as CPU, memory, and IO adapters, and various virtualisation artefacts such as host physical to guest physical address translation information. This is to ensure that Operating Systems and applications executing within virtual machines are not able to interfere with:

a) The operation of other virtual machines running on the same hypervisor host.

b) The operation of the software running them (the hypervisor, platform Operating System, software stack, etc.).

c) The hardware hosting the above.

Further, even those virtual machines running in a purely "hardware-assisted virtualisation" mode (known, for example, as "HVM guests" on Xen-derived hypervisor systems) may be able to derive the fact that they are running virtualised via a variety of probes and attacks on their host system.

Within NFV, concealment of the fact that a VNFC Instance (VNFCI) is running virtualised is not, in any case, considered necessary for two key reasons:

1) The very nature of a VNFCI is that it has been designed to run within a virtualised environment, and is likely to have been subject to software optimization techniques taking that in mind at both the VNFCI and VNF levels of abstraction.

2) A VNFCI is controlled by a VNF Manager component (VNFM) which resides in the Management and Orchestration stack (management and orchestration). The VNFM coordinates the operation of the various VNFCs that it controls with other parts of an NFV deployment, including the Virtual Infrastructure Manager (VIM) which has control over various aspects of the NFV Infrastructure (NFVI). The VNFM, therefore, may have knowledge - ranging from fairly abstract to extremely detailed - of the virtualised infrastructure in which the VNFCI is executing.

Attempting to prevent the VNFM from providing this information to a VNFCI is therefore considered neither desirable nor plausible.

Given this, the following recommendations are made that:

- NFV deployments ensure that hypervisors and other elements of the NFVI provide appropriate levels of isolation for the security requirements of that deployment;

- vendors are encouraged to write "well-behaved" applications which do not attempt to use knowledge of the virtualisation environment to subvert such isolation or other techniques to preserve the planned operation of NFV deployments.

These considerations for network traffic inspection and management that duplicate packets (i.e. virtual port mirroring) or are compensatory mechanisms to bypass packet loss are carefully audited and architecture to ensure they are compliant with Lawful Intercept and other stated requirements.

4.4.2.1.2 Introspection

The hypervisor is fully aware of the current state of each guest OS it controls. As such, the hypervisor may have the ability to monitor each guest OS as it is running, which is known as introspection.

Introspection Risk for NFV

Hypervisor introspection, including administrative and process introspection, presents a risk to the confidentiality, integrity and availability of the NFV. Introspection can enable the ability to view, inject, and/or modify operational state information associated with NFV through direct or indirect methods. Access to state information can result in the ability to arbitrarily read and/or write the contents of memory, storage, keystores and other NFV operational aspects. Access can be direct via CLI (command line interfaces), GUI (graphical user interfaces), DMA (direct memory access), API (application programming interfaces) and other methods. Indirect access involves the use of side channels and other means that allow the inference of data and modification via resources with privileged relationships.

Administrative models that enable an admin, root or super user account type that has full access to system resources allows visibility into and modification of private keys, symmetric keys, passwords in memory, networking and system configuration, intellectual property and other resources within the NFV. Introspection based modification of logging, reporting and alerting can disguise malicious access, unintentional modification, vulnerabilities and administrative mistakes. Likewise, similarly privileged processes also possess these capabilities to introspect.

The use of administrative models and process introspection that could allow unfettered access and modification capabilities is an architectural, operational and audit risk to the NFV ecosystem.

4.4.2.2 Access Visibility for Data and Control Packets in Virtualised Environment

In order to provide end-subscriber intelligence one needs to be able to correlate data in both the user plane and control plane. This correlated analytics can produce useful information in the form of subscriber's IP address, IMSI address, end user device, application, location, bandwidth consumed by the application. These analytics can also help the operators to keep track of the network usage, subscriber dynamics, any network anomaly in the network such as malware and DDOS related attacks in the control plane or data plane. In a non-virtualised environment many of the interfaces between the networking components are exposed and hence the traditional active or passive probes can be used to monitor the packets in the data plane and control plane and perform deep packet inspection (DPI) function and correlation. For example, using passive probes, one can monitor control plane messages such as GTP-C, S1-C for EPC and SIP control messages for VoLTE, track end-to-end control plane messages for session call creation and termination, etc. These probes can also capture user plane packets such as GTP-U for protocol analysis and deep packet analysis. With the help of control plane messages and data plane packets these probes can provide network and user experience analytic, KPIs, and help addressing security impacts to the mobile customers, mobile carrier, and the downstream in general public.

However, in an NFV environment, these interfaces are either concealed by consolidated vertical "function silos" or by collapsed stack (e.g. connection by virtual socket and not by IP packets) within the VM or inter-VM within the same hypervisor thus making it difficult to probe the desired data. In some cases vendors try to optimize processing power and reduce the signalling latency by implementing proprietary, non-3GPP standards for some of the interfaces. Thus, there are many issues that need to be looked into in order to obtain the desired data under these circumstances.

There can be different forms of virtual tap (vTap) and "front-end processor(s)" - vFEP deployment scenarios that are based on how the NFVs are deployed in the network. In a broader landscape, we can envision that the vTap, served as selective-filtering/splicer role, will have to be part of the service chaining fully-aware by the NFV Service orchestrator such that when NFV modules move, grow/suspend due to scale-out/scale-in, vTap will be always in place.

We provide few use case scenarios below and describe how vTap can be used in those scenarios.

In one type of deployment scenario one VNF may run on one or more dedicated set of blades similar to an instance of a VPC (Virtual Private Cloud). In that scenario, all inter-VNF communication is through the switch fabric. For example, in the case of virtualised EPC, vMME, vPGW, vSGW, vMSP may all run in different blade. Similarly, in another instance of vEPC deployment, multiple VNFs (e.g. vMME and vSGW) may run on the same blade. The vTap may selective-filter the focused traffic and replicate the traffic to the vFEP running in another VM. vTap does the first level of selection and concentration. It can be using SDN technology, "introspection vSwitch" technology, both are "selective"; or even the basic SPAN/RSPAN feature offered by vSwitch or TOR switch without any "selective-filtering" at all. The vFEP forms the second tier in the data collection hierarchy, which performs the collector/protocol normalizer/pre-processor/aggregator front-end tasks. The vFEP can be a single VM, or be distributed to many VMs located in multiple chassis, as dictated by performance/capacity requirements.

In second type of deployment scenario, VNFs do not use any 3GPP specific protocol for inter communication but uses either vendor proprietary protocols or use shared memory or local database to communicate in order to optimize the inter VNF communication. These systems expose the standard interfaces only when their systems need to interoperate with another vendor component. For example, some of the vEPC vendors use non-standard interfaces for the communication between MME and HSS instead of standard DIAMETER-based S6a interface, but exposes DIAMETER interface while talking to external HSS. In some cases, they use optimized protocols in order to reduce the signalling overhead. In those scenarios, it would be difficult using the normal vTap to gather both control and data packets to provide any analytics or detect any specific anomaly in the network. In these situations, the vendors will need to expose their APIs directly to vFEP to "bring it back to the more normalized form in 3GPP" or via Element Management System (EMS) or by some other means they will need to provide the required Meta data in a timely fashion to feed the vFEP. Push/pull mechanism or subscribe/notify mechanism can be implemented to notify the related control and data information to vFEP.

While deployment scenario 1 has been looked into by many vTap, vFirewall, or SDN vendors, scenario 2 is bit challenging and needs more special vendor-specific work such that information can be obtained in scenarios where the interfaces are not exposed.

4.4.2.3 Validation of Root of Trust and Chain of Trust

This clause describes guidance associated with visibility into trust associations and the ability to verify aspects of trust:

- Hardware-based roots of trust.
- Software-based roots of trust.
- Certificate-based roots of trust.
- Validation guidance for non-repudiation.

4.4.2.4 Services validation

This clause describes guidance associated with visibility into services required by the VNF and the ability to verify the appropriate and secured usage of those services.

4.4.3 Certificate, Credential and Key Management within NFV

4.4.3.1 Certificate management

This clause describes the unique aspects of managing cryptographic certificates within NFV, including newly instantiated, suspended, hibernated and restarted images, as well as certificate management for retired VNFs. Interfaces to and reliance on the environment external to NFV is further described in clause 4.3 of the present document.

4.4.3.2 Credential Management

This clause describes the unique aspects of managing credentials within NFV, including newly instantiated, suspended, hibernated and restarted images, as well as credentials associated with retired VNFs. Interfaces to and reliance on the environment external to NFV is further described in clause 4.3 of the present document.

4.4.3.2.1 Void

4.4.3.2.2 Role of Identity, keys and certificates

There are a variety of different actions and functions that a VNF - and its constituent VNFCs - may need to perform, and which have a security element, of which the most obvious are identity checking and data encryption/decryption.

The standard security tools required to perform them rely on a single primitive - the secure provision of a private asymmetric key and associated public key to a specific architectural component - and the application of a number of processes.

A private asymmetric key (with associated public key), and a trust relationship with a Certificate Authority (CA), can act as the primitives to enable all of other actions and functions. Assuming that a Certificate Authority exists and that there is network access available to it as required, the core mechanism, then, is the provisioning of a public/private key pair to a VNFC. Once such a key pair is provided, the entity can communicate with other parties (see, however, clause 5.4.3.1, which discusses how without an initial root of trust, such communication cannot themselves be trusted) to create or retrieve other cryptographic components such as session keys for communication. However, in order to allow for structured trust relationships with these other entities, they are likely to require that the public key is signed or at least validated by a mutually trusted party such as a certificate authority.

The mechanisms available for initial provisioning of a public/private key pair include:

- Self-generation of key pairs for later validation by an external party:
  - Certificate Authority
  - VNFM
- Provision by trusted party - network
- Provision by trusted party - storage
- Injection by hypervisor

Other techniques using TPMs (trusted platform modules) and/or HSMs (hardware security modules) are not examined here and are subjects for further study.

4.4.3.2.3 Credential Injection by hypervisor

There are three approaches to allowing a hypervisor to inject keys - or other information - into a VNFC-containing VM which it is hosting:

1) Direct injection into RAM - the hypervisor, having access to the VM's RAM, could inject keys into a known address or buffer, allowing the VNFC (which would need to be primed to accept them via some process in the boot sequence of the initial image).

2) Injection into file system - the hypervisor, having access to the VM's filesystem, could inject keys into a known location in local (hypervisor-maintained) storage. This could be performed before the booting up of the VNFC, allowing it to come up with the keys already in place.

3) Injection via agent - a number of hypervisor vendors provide agents which can reside in a VNFC, and are provided with a secured local communication path to the hypervisor host. If a VNFC image were provisioned with one, and told to contact the hypervisor as part of the boot process, keys could be provisioned via this channel.

All of these approaches, of course, require that the hypervisor is considered a trusted party within an NFV deployment, but this is *sine qua non* anyway, since a hypervisor has easy access to any information stored within a VNFC - or which it can access - and therefore should be considered - and configured - to have at least the same level of trust as any VNFCs which is hosts.

4.4.3.3 Key Management

This clause describes the unique aspects of managing cryptographic keys within NFV, including newly instantiated, suspended, hibernated and restarted images, as well as key management issues associated with retired VNFs. Interfaces to and reliance on the environment external to NFV is further described in clause 4.3 of the present document.

4.4.3.3.1 Key Management and security within cloned images

In general, cloned images should not possess cryptographic key pairs utilized by their original image. Propagation of two or more images with the same key pairs immediately cancels out the notion of utilizing key pairs for the purpose of establishing identity.

A further reason is that images or virtual disks may not be signable, but can possess a cryptographic checksum for the purpose of establishing identity. So, a master image should not possess its key pairs prior to generating the checksum. Instead, this should be part of the provisioning process for a new VNFC: see clause 4.4.3.2.2 for a further discussion of this topic.

When creating a certificate based on the key pair and the identity of the VNFC, it is important to ensure that the identity associated with it is both unambiguous and also not based on information which may change. For instance, when a new VNFC is created in a VM, one or more MAC addresses are likely to be associated with it. These MAC addresses are not appropriate identifying characteristics of the VNFC, as they are liable change if a new vNIC is applied or on migration of the underlying VM to a different hypervisor instance.

4.4.3.3.2 Key Management and security within migrated images

When images are migrated, regardless of the vehicle for accomplishing the migration, they should possess the same MAC addresses, CPUID, and other hardware signatures that they possessed prior to the migration. In this event, there is no need to replace or update any internal key pairs associated with identity. There is a need, however, to ensure the integrity and confidentiality of the keys during the migration process.

If any of the key hardware signatures is modified (i.e. the MAC address for the virtual NIC changes), the key management system should have clear policy that describes action to be taken, including the destruction of keys, regeneration of keys and alerting of the event.

4.4.3.3.3 Self-generation of key pairs

A standard approach to provisioning an entity with keys is to allow it to use standard key generation tools as provided by the OS or other applications. The standard tool within Linux® is *ssh-keygen*.

NOTE: "Linux is a registered trademark of Linus Torvalds. This information is given for the convenience of users of the present document and does not constitute an endorsement by ETSI of the product named. Equivalent products may be used if they can be shown to lead to the same results."

4.4.4 Multiparty Administrative domains

4.4.4.1 Rational

In a typical scenario for NFV there will not be a single organization controlling and maintaining a whole NFV system. The organizations may either be departments of the same root organization e.g. a network department and an IT datacentre department or they may be different companies providing different functional blocks of the NFV Architectural Framework. The basic use cases for these scenarios are described in ETSI GS NFV 001 [i.1].

4.4.4.2 Administrative domains

Administrative domains can be mapped to different organizations and therefore can exist within a single service provider or between service providers. Functional blocks within an administrative domain are considered to have a stable trust relationship, while a transient, specific trust relationship should be established among functional blocks across different administrative domains. Administrative domains can be collapsed, resulting in different deployment options. Figure 1 depicts the two main administrative domains associated with the scenarios considered in ETSI GS NFV 001 [1]: the "Infrastructure Domain" and the "Tenant Domain". These administrative domains are described in the following clauses.

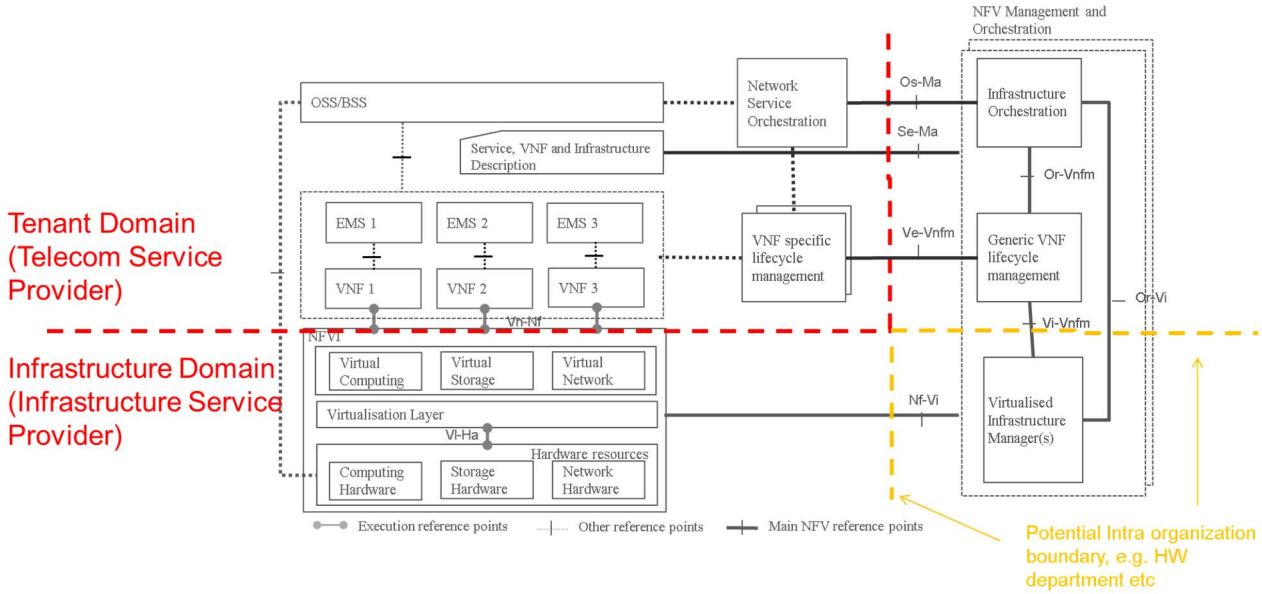

Figure 3: Administrative Domains

4.4.4.3 Infrastructure Domain

The Infrastructure Domain provides virtual (infrastructure) resources such as computer, networking and storage or a composition of those to a Tenant Domain. The Infrastructure Domain may be split into sub domains for e.g. physical networking, compute and storage such as in traditional datacentre environments. Two well-defined reference points exist between the Infrastructure Domain and the Tenant Domain.

4.4.4.4 Tenant Domain

The tenant combines and configures VNFs to form a network service. Therefore the Tenant Domain consumes resources from one or more Infrastructure Domains using the Infrastructure Domain orchestration functionality to orchestrate and operate virtual infrastructure resources required by VNFs and associated network services. Typical tasks within the Tenant Domain are VNF functional configuration, software upgrades and FCAPS. Typical tasks requiring the involvement of both the Tenant Domain and the Infrastructure Domain are on-boarding of VNFs, instantiation of VNFs and scaling of VNFs.

4.4.4.5 Implications

To be able to provide simple and well-defined reference points between the Tenant and Infrastructure Domain a separation of the Orchestration functionality and the VNF lifecycle management functionality into a tenant and an infrastructure part is proposed. Therefore four functional blocks are introduced as a result of the separation:

- Network Service Orchestration.
- Infrastructure Orchestration.
- VNF-specific lifecycle management.
- Generic VNF lifecycle management.

The Ve-Vnfm reference point and the Os-Ma reference point are reference points between functional blocks, which are in scope of NFV and therefore have well defined endpoints. The Reference points are depicted in figure 1. The scope of both reference points is targeted to virtual resources or compositions of virtual resources. As the endpoints and the scope of both reference points are well defined, clear requirements on the secure interaction between the Tenant Domain and the Infrastructure Domain can be defined.

4.4.4.6 Inter-Domain functional blocks and reference points

4.4.4.6.1 Network Service Orchestration

The Network Service Orchestration uses one or more Infrastructure Domains to create the required resources for a network service. Additionally the Network Service Orchestration facilitates the configuration of VNFs to form a network service. The Network Service Orchestration functionality can be merged with OSS/BSS functionality and can therefore be seen as an evolution of those systems.

4.4.4.6.2 Infrastructure Orchestration

The Infrastructure Orchestration provides functionality to orchestrate and manage virtual resources or compositions of virtual resources. The Infrastructure Orchestration is not aware of the actual network service the virtual resources are providing.

4.4.4.6.3 VNF-Specific Lifecycle Management

The VNF-Specific Lifecycle Management provides functionality to configure and operate VNFs. It hides the VNF internal configuration towards the Infrastructure Domain. The VNF-Specific Lifecycle Management can manage virtual infrastructure resources after being authorized by the Infrastructure Domain. The VNF-Specific Lifecycle Management can be merged with EM functionality or a VNF and therefore can be seen as an evolution of those functional blocks. If a VNF does not implement VNF-Specific Lifecycle Management it can use VNF lifecycle management functionalities provided by the Generic VNF Lifecycle Management.

4.4.4.6.4 Generic VNF Lifecycle Management

The Generic VNF Lifecycle Management provides functionality to perform VNF lifecycle management. The Generic VNF Lifecycle Management can provide generic capabilities to configure VNFs, yet being unaware of the actual semantics of the configuration data. In case a VNF-Specific Lifecycle Management is present the Generic VNF Lifecycle Management facilitates requests from one or more Tenant Domains.

4.4.4.6.5 Inter-Orchestration (Os-Ma)

To enable a multi-tenant environment the Os-Ma reference point needs to provide authorization of the Tenant Domain towards the Infrastructure domain. The Os-Ma reference point needs to support access from multiple tenants and provide sufficient isolation of management operations on resources from different tenants.

4.4.4.6.6 Inter-VNFM (Ve-Vnfm)

To enable a multi-tenant environment the Ve-Vnfm reference point needs to provide authorization of the Tenant Domain towards the Infrastructure Domain and vice versa. A token-based authentication mechanism such as Kerberos may be used between the Tenant Domain and Infrastructure Domain. The token may be derived from the initial interactions on the Os-Ma reference point and "pushed down" to be used by the entities attached at the Ve-Vnfm reference point, or may be requested for each operation, or be subject to time-based constraints.

4.4.4.7 VNF Package and Image Management

When a new VNF is to be onboarded, NFV management and orchestration entities may wish to make a number of security-related checks before deciding whether to start the process of instantiation. These can be broken down into several types:

1) *Integrity checks* - does the VNF Package include the various components expected, and are they free of tampering.

2) *Trust checks* - does the VNF package consist of components from trusted vendors/suppliers.

In both of these cases, the use of cryptographic signing and certificates can provide assurances.

A VNF Package is expected to comprise the following pieces, as files (see the document Software Architecture; Virtual Network Functions Architecture):

- VNFC images
- VNFC configuration data
- VM configuration data

In addition, a VNF Descriptor (VNFD) provides references to all of these and details of how the VNF should be instantiated from the available pieces, including internal network graph(s).

NOTE: This description assumes that a VNF Package is supplied by a single entity. For more complex relationships, further complexity in the signing and provenance process will be required.

4.4.4.7.1 Integrity checks

In order to ensure the integrity of the first three types of component, a cryptographic hash should be created of each unique component, and the hashes listed in the VNFD entry associated with each component. A cryptographic hash of the VNFD should also be created, and the hash published by the VNF supplier. It is strongly recommended that the VNFD is versioned, and that the version number is reflected internally with text changes, and that each version of the VNFD (which reflects a different VNF version) is hashed and the hash published.

Although it would be possible to create a single hash across the entire VNF, this type of hierarchical hashing allows the easy updating of separate components - or configurations - of the VNF without the need to create a single hash each time. It also allows for simpler downloading and checking of newer versions of specific components.

4.4.4.7.2 Trust checks

In order to establish trust, the standard mechanism is the use of cryptographic certificates. Using the hash calculated for each of the entities in the VNF Package (including the VNFD), a digital signature can be created for each, and the associated public key certificate(s) made available. The relevant management and orchestration entities wishing to instantiate a new VNF should validate the relevant public key certificates against their existing trust framework:

- Include virtual appliances and security virtual appliances

4.4.4.8 VNFC Security Overview

Clauses of the Security Problem Statement expected to be of particular relevance to VNFCs and their lifecycle are:

- Trustworthy Boot and Instantiation.
- Secured Crash and Recovery.
- Private Key Management within Cloned Images.
- Private Keys Management within Migrated Images.

NOTE: Performance isolation of the Security Problem Statement, though important to the operation of VNFCs and their placement on hosts, is not considered in the present document.

4.4.4.8.1 VNFC security scope

This clause addresses trust state changes related to lifecycle events for a VNF that contains multiple VNFCs. Various components within the lifecycle process should have - or delegate - specified levels of trust to other components, and these relationships are explored. It also discusses mechanisms for cryptographic key management that are relevant for intra-VNF communication (between VNFCs), since these are relevant to establishing and maintaining trust. The mechanisms and protocols for intra-VNF communication are not addressed; though it is assumed that they will require cryptographic keys. No specific recommendations are made around protocols or cryptographic suites: standard industry techniques should be employed, and relevant recommendations are out of scope of the present document.

There are many other security issues related to VNFCs: these are not discussed in the present document, which concentrates on the problems presented by trust within an NFV deployment and supporting technologies and techniques that need to be developed or employed.

The main use case is one of infrastructure-supported virtualisation containers consisting of virtual machines running on a hypervisor platform. That said, a great many of the mechanisms and procedures detailed or outlined in the present document are applicable to other container approaches, such as Linux® Containers (LXCs), Java Virtual Machines (JVMs), virtual appliances (VAs) and Security Virtual Appliances (SVAs).

The following Operations in the VNFC lifecycle are covered in this clause:

- VNFC Creation.
- VNFC Deletion.
- VNFC Configuration Management.
- VNFC Migration.
- VNFC Operational State Changes (e.g. hibernation, sleep, resumption, abort, restore, power-on, and power-off).
- VNFC Topology Changes.
- VNFC Scale-Up and Scale-Down.
- VNFC Scale-In and Scale-Out.

This clause will address the above requirements, from a security perspective, in the use cases below. Equivalent operations, at least in terms of security concerns, will also be identified. Work needs to be undertaken to explore and identify these equivalences and document gaps.

Scope:

- SLAs and business contracts/agreements will allow the description of responsibilities and auditable points of trust. The details of these contracts are out of scope, but the expression and instantiation of their descriptions will generally be in scope.

Lifecycle use cases which may or may not be in scope:

- Auto-healing, where the system will need to recover from a large number of faults, due to the destruction of the inherent infrastructure of the system.
- VNF maintenance functions:
  - Back-ups and archiving.
  - Snapshots and mirroring.
  - Updates to and upgrades of VNF functions.
- Data store maintenance functions - changing a LUN, may be owned by a different function.

4.4.4.9 VNFC Lifecycle Security - Statement of the problem

When a VNFC is created, it is part of a VNF. It may be:

- The initial VNFC for that VNF.
- The only VNFC for that VNF.
- Joining a set of other VNFCs for the VNF.
- Replacing one or more VNFCs for that VNF.

In all of these cases, it does not exist as a single entity: it needs to be able to communicate with entities external to the VNF - including other VNFs - and it may need to be able to communicate with other VNFCs within the VNF. In order to be able to do that in a way that allows the security of the system to be known and maintained requires trust state to be established and maintained.

This trust needs to be maintained - at both the VNF and VNFC level - over various lifecycle events during VNF's lifetime, including (but not necessarily limited to) the following:

- Creation of VNFCs.
- Deletion of VNFCs.
- Migration of VNFCs.
- Suspension and resumption of VNFCs.

A key point to consider is that it not just the booting up of a VNFC within a virtualisation container that needs to be managed in order to ensure a known level of trust, but the entire environment within which is operates. The following clauses of the present document provide guidance as to how to manage this process.

There are a set of challenges raised by these lifecycles, and clause 4.4.4.8 addresses these. Without solutions to these challenges, trust cannot be fully established and/or maintained. They are:

1) establishment the initial identity of the VNFC (see clause 4.4.3.2.2);
2) establishment an initial root of trust with an external entity (see clause 5.4.3.1);
3) assurance of the security of the boot process (see clause 4.4.5.1);
4) assessment of the impact that the crash of one or more VNFCs has on the state of trust (see clause 4.4.6.4.1);
5) assessment of the impact of cloning on the security of private keys and associated trust and identity issues (see clause 4.4.3.3.1);
6) assessment of the impact of migration on the security of private keys and associated trust and identity issues (see clause 4.4.3.3.2);
7) choice of appropriate models for control over lifecycle events and associated authorization (see clause 4.4.6.2).

4.4.4.10 Security Approach

The principles used in defining content for the present document include:

- we wish to state deployment options and assumptions where possible, not define a shared framework;
- it is up to the implementer to make implementation-specific risk decisions: we can only point out general classes and instances of risk and make general recommendations;
- informative, not formative (descriptive, not prescriptive), built on a catalogue of recommendation options;
- always be careful to describe elements and components of a solution and audit points (for instance), rather than specifying a particular technology or vendor.

The present document approaches the question of the management of trust states for VNFCs within a single VNF from the viewpoint of the lifecycle of a VNF and its constituent VNFCs. See clause 5.2 - for a list of the components considered and their responsibilities.

Note that the present document does not cover the full lifecycle of a VNF from the point of view of the wider NFV: this process is more fully described in Network Functions Virtualisation (NFV); Virtual Network Functions Architecture (ETSI GS NFV-SWA 001 [i.3]) Rather, we examine the core steps in the process which are of relevance to the security lifecycle of the VNF. Nor does it cover more general security requirements of VNFs, except where they touch on areas and actions that are in scope.

4.4.5 VNF Instantiation

This clause describes the security processes associated with starting a VNF.

4.4.5.1 Trustworthy Boot

Trustworthy Boot encompasses the technologies and methods for validation and assurance of boot integrity. Boot integrity validation can invoke numerous assurance factors, including local integrity enforcement, remote attestation, attribution, authenticity, configuration management, certificates, cryptographic keys, digital signatures and hardware-specific features such as HSM, TPM and Dynamic Root of Trust for Measurement (DRTM). The Trustworthy Boot process is inclusive of hardware, firmware, hypervisor and OS image validation and assurance factors. The desired level of assurance is achieved through the selection of appropriate factors.

Trustworthy Boot, as defined by NFV, is contrasted with the industry definition of "Secure Boot" and "Measured Boot". Secure Boot is a security concept for validating that a platform boots using only firmware and software that is trusted by the platform manufacturer and/or the platform owner. The integrity checks are based on digital signatures of the firmware and software, leveraging available hardware-based Roots of Trust (RoT) and upon an integrity validation failure, the boot process will be halted. Trust in the boot process is derived from its completion status. A successfully completed boot process is inherently trusted.

Measured Boot, as defined by the TCG, differs from Secure Boot in that it aims to only record integrity information on all code (firmware and software) executed during the boot process. No integrity validation is performed during the boot process. Instead, the recorded integrity data is meant to be retrieved by a remote verifier sometime after the boot process has been completed. The remote verifier is responsible for validating the integrity data and assigning an appropriate trust level for the boot process.

An implementation of Secure Boot and/or Measured Boot, based on DRTM (Dynamic Root of Trust for Management) technology, is also possible. An example of that is TBOOT, which leverages DRTM technology to perform a measured and/or verified launch of an OS kernel or hypervisor.

Note that the NFV Trustworthy Boot process is appropriately inclusive of Secure Boot, Measured Boot and TBOOT, while not exclusive and prescriptive for these technologies and methods.

It is sometimes assumed that Secure Boot is fully defined by the homonymous mechanism in the UEFI ("Unified Extensible Firmware Interface") standard [i.4]. However, the subject is much broader and more nuanced than UEFI, which is a specific solution to the integrity validation problem of only a part of the boot process. Thus, NFV has adopted the term "Trustworthy Boot" to reflect all relevant methods for implementing a VNFCs and NFVI components (i.e. hypervisor) boot process, from which a trust level can be derived.

Trustworthy Boot embodies the requirement that the controlling entity for a VNFC instance - typically the VNF Manager (VNFM) - can have a certain level of assurance that the boot process for the VNFC has completed in a trustworthy manner, be that through Measured Boot, Secure Boot or both combined. Whereas, in a standalone NF, this requirement extends no further than the booting of the hardware and OS of the NF, in a VNF, this assurance needs to extend down through the INF layer to the hypervisor layer.

Steps for trustworthy boot in the virtualised case:

1) Hardware
2) Hypervisor & platform
3) VNFC virtualisation container (by hypervisor)
4) VNFC OS
5) VNFC application
6) Deployment state application to VNFC

Note that just because the integrity validation during a Secure Boot type of boot process does not yield the expected results - and that, therefore, the boot cannot be considered entirely "secure" (trustworthy) - this does not mean that operation of an entity should necessarily be stopped. There are occasions where less secure situations may be acceptable, such as when maintenance is underway or is known to be required. This is especially true, if a Measured Boot process is employed along the Secure Boot, thus enabling the trust level to be determined not only based on the completion status of the (Secure) boot process. Policies for dealing with a "failed" integrity validation may include (for example):

- Do not boot.
- Allow to boot, but with reduced privileges.
- Allow to boot, but restrict access to other entities, network, etc.
- Allow to boot, but flag for investigation.

4.4.5.2 VTPM (Virtual Trusted Platform Module)

The secured storage of measurements, credentials and secrets in a virtualised trusted platform module is highly desired when the use of a physical TPM is limited or undesirable. Standards for virtual TPMs are developing and the use of standards-based vTPM will be referenced in the present document when available.

The use of vTPM is optional and is an evolving area that will be described with further NFV-specific guidance as commercial implementations are available.

4.4.5.3 Attestation

- Measurements.
- Local attestation of VNFCI.
- Remote attestation of VNFCI.

4.4.5.4 Attribution

- Attribution to source - company, project, open-source.

4.4.5.5 Authenticity

- Is package, module, update, patch genuine?

4.4.5.6 Authentication

4.4.5.6.1 User/Tenant Authentication, Authorization and Accounting

The introduction of NFV brings new security issues when it comes to AAA, as it implies using the current identity and accounting facilities at least at two layers: the network infrastructure (identifying the tenant) and the network function (identifying the actual user). What is more: the E2E Use Cases document describes architectural patterns (NFVIaaS, VNFaaS, VNPaaS, VNF Forwarding Graphs, etc.) that suggest the stacking of identities can occur at more layers if all patterns become commonplace, as shown in figure 4.

While the figure does only show what we could call an infrastructure (vertical) integration, it is worth noting that collaborative (horizontal) is also possible. A couple of examples could be the federation of different NFVI providers, or the traversing of several NSPs to provide VNPaaS. Issues related with identity will need to be solved potentially at any kind of horizontal and vertical integration patterns.

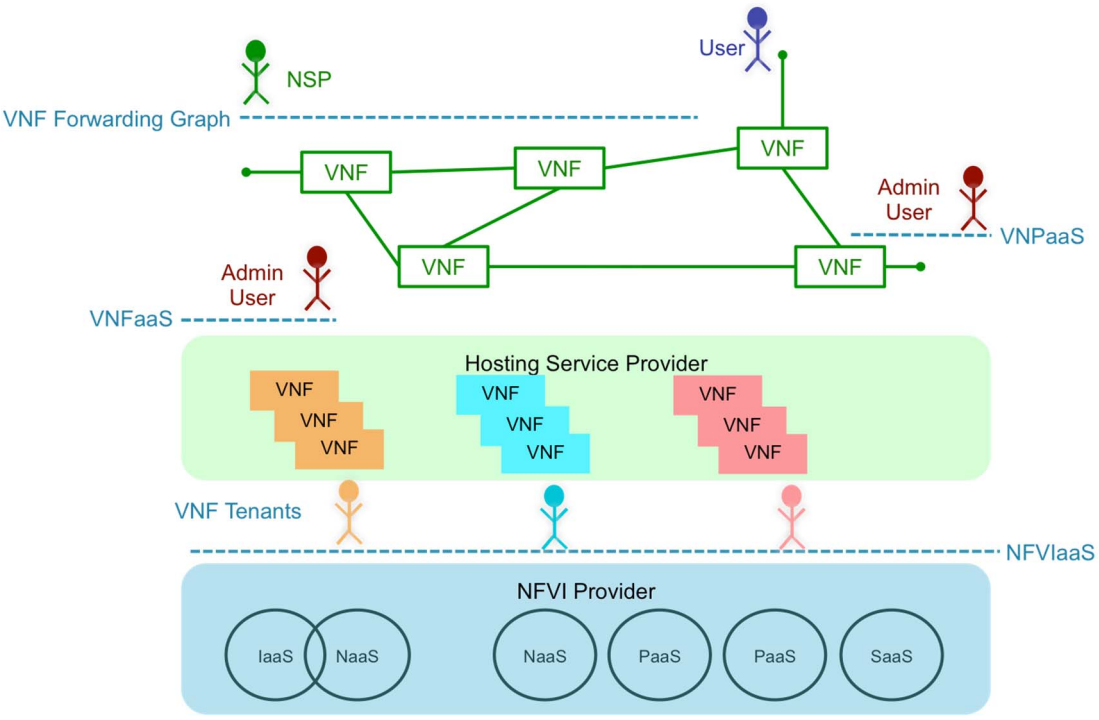

Figure 4: Infrastructure Integration

A generalized AAA schema for identifying users utilizing a particular tenant infrastructure and/or a tenant acting on behalf a user is required to support these patterns and the new operational an business models they will bring. Current AAA mechanisms assume there will be a single identity, single policy decision and enforcement points, a single level of policy, and a single accounting infrastructure. Even if a strict separation by tunnelling (as current practice seems to suggest) would be feasible without breaking some of the promised NFV enhancements in what relates to scalability, agility and resiliency, there exist risks related to each one the three components in AAA:

1) Authentication procedures can imply privacy breaches associated to the disclosure of user information at layers that are not intended to consume certain identity attributes.

2) Authorization risks are mostly related to privilege escalation produced by wrapping unrelated identities not verifiable at a given layer. A well-known example in telecommunications infrastructures is the multi-frequency phone hacking of not that many years ago.

3) Accounting should be performed at all the underlying infrastructure layer(s), and they will not only require accounting at the granularity of virtualised applications that use the infrastructures, but also at the coarser granularity of the tenants running virtualised network functions. In summary, the capability of billing the billers.

To address privacy issues in authentication, mechanisms for the generation and validation of identity tokens, as well as for the exchange of identity, attributes need to be validated in a multi-layered multi-tenant environment. Directly connected to this attribute exchange, appropriate authorization processes will require the availability of methods for policy routing (or migration) and composition, applicable not only to users and tenants, but to VNF deployment units as well. These requirements claims for a consistent composition of current cloud AuthN/AuthZ techniques (like Keystone) and network practices (mostly based on RADIUS and DIAMETER), probably through proposals intended to go beyond the state of the art in identity federation.

For security accounting, apart from the existing cloud mechanisms on resource usage (processing, storage, memory, local network capacity, licences, etc.) an infrastructural network function will be needed that consists of the following functional blocks (FBs):

- Traffic packet acquisition at full-line rate.
- Traffic classification and accounting per service provider, per user and per application, verified against the aggregate to assure integrity.

- Policy decision function, intended to raise an alarm whenever a specific threshold has been achieved according to the detected traffic and the policies.
- Policy enforcement for integrity, intended to enforce the rule in the data plane.

While traffic acquisition FBs should be distributed due to their nature, traffic classification/interpretation FBs need to be consolidated from different nodes into a central location, to achieve gains in scale and operational simplification.

All this requires the support of a trust framework that the experience has shown cannot be derived hierarchically from a single global root, while it does not seem realistic to expect that pure peer-to-peer agreements will be able to support the highly dynamic scenarios NFV will bring. It is crucial to find an appropriate combination of flexibility, manageability and security properties for the trust framework supporting the AAA infrastructural services.

- Authentication of VNF images.
- Credentials in suspended/offline images.
- Between different functional blocks. When intra-domain in may be less relevant (it is up to the operator, they may not require it), but when it is inter-domain it is mandatory.
- Authentication of the user/customer that makes the request to the NFV-management and orchestration functional blocks.

4.4.5.7 Authorization

- Updates to authorized users in suspended/offline images.
- Updates to authorized managers in suspended/offline images.
- On all operations on interfaces/APIs between different functional blocks. Simply because operators may want to restrict access in some cases, or enforce a specific flow when multiple options are possible in the architecture.

4.4.5.8 Interface Instantiation

Virtual MAC addresses, VLAN assignment and update from image configuration.

4.4.5.9 Levels of assurance

It is important to note that security of a system always involves trade-offs, typically between usability (or ease of management), cost, and assurance of security. It is acknowledged that adoption of the measures proposed in the present document may not be appropriate or relevant to all deployments, and that any organization planning to integrate VNFs or VNFCs from one or more vendors within a broader NFV infrastructure may not be in a position to implement all of the possible options. The present document recommends broader and deeper understanding of the security issues, to facilitate better informed deployments of real-life components. The present document therefore attempts to delve deeply into the relevant security topics, providing options that range from a baseline level of security to high assurance.

As a matter of principle, the present document attempts to describe high assurance security requirements for deployment in hostile environments that are constantly under advanced attack, and highlights the associated pathological requirements or risks. This does not imply that all the measures described will be relevant to all - or even most - deployments.

4.4.5.10 Logging, Reporting, Analytics and Metrics

- Support for real-time monitoring, SLAs, reliability, performance.

4.4.6 VNF Operation

4.4.6.1 Planned operational lifecycle events

- VNF/VNFC healing.
- VNFC scaling.
- VNFC - if it has new resources added to itself, the following operations may have security implication:
  - Safe memory allocation, de-allocation, etc.
  - Adding LUN volumes.
  - Infrastructure manipulation of a VNF: any changes to the underlying infrastructure.

4.4.6.2 VNFC Lifecycle control and authorization

At one level, MANO, in conjunction with the hypervisor, has ultimate control over VNFCs that are instantiated within VMs, and therefore over the location of the constituent VNFCs in a specific VNF. However, many lifecycle events will need a control plane to manage the workings of the VNF's constituent VNFCs themselves, to tell them that they are about to be moved, destroyed, have peers added, a master upgraded, etc.

A key question that arises, and which will likely be addressed differently by different vendors for different VNF types, is "who is authorized to instigate particular lifecycle events?" Several approaches exist, each of which bring different failure modes:

- All control should be via a master VNFC, which talks to a VNF Manager (VNFM).
- Control is directly into a set of peer VNFCs via a VNFM.
- Control is directly from a VNFM.
- Control is by other entities.

The different approaches are an area for further discussion and study, but any model selected for a particular deployment will be subject to the same issues, which are discussed here.

In all these approaches, RBAC (Role-Based Access Control) or ABAC (Attribute-Based Access Control) will be appropriate for most deployments. For both RBAC and ABAC the accessing entity should to be authorized (or not) based on authentication based on a cryptographic certificate. The actual authorization of operations may either be controlled by an entity in the chain of trust for the VNFC (such as a AAA ("Authentication, Authorization, Access") server) or the VNFC should be provided with enough information to make decisions about authorization itself. This information should, likewise, come from an entity.

Note that this entity (whether a delegating or provisioning entity) is not yet believed to have been identified in the NFV architecture and this is a therefore noted as a GAP.

Two key, and inter-related, issues need to be addressed, for whichever set of models that are chosen:

- Where the database of record resides?
- What auditable points exist?

In general, the use of a single, long-lived entity for control is preferred for the following reasons:

- Single database of record.
- Single point of control.
- The ability to consolidate auditable events.
- Simplicity of authentication, revocation, etc.

Some VNFCs will be short-lived, and availability of audit logging will be reduced.

4.4.6.3 Dynamic State Management

Online and offline security operations, suspending an image, updating Access Control Lists (ACLs), etc. in suspended images, migrations.

4.4.6.3.1 Provision by trusted party - network

Once the VNFC has been provisioned with an initial root of trust (see clause 5.4.3.1), then it is possible for it to contact this initial root of trust and accept keys from it. This might not seem plausible because the entity acting as the root of trust might be considered unable to trust the VNFC which is contacting it, but this may not necessarily be the case. Since MANO, in conjunction with the hypervisor, is able:

- to identify the VNFC uniquely;
- to know, if Trustworthy Boot has been completed successfully (see clause 4.4.5.1 Trustworthy Boot that the VNFC's state is pristine;
- to control all communications between the initial root of trust and the VNFC;
- to have a defined trust relationship with the certificate and key provider chain then it can be assured of the identity and trustworthiness of the VNFC, and can act as a trust broker, assuring the initial root of trust the identity of the VNFC and ongoing trust management throughout the lifecycle.

4.4.6.3.2 Provision by trusted party - storage

As management and orchestration, in conjunction with the hypervisor, controls the storage to which a VNFC has access, it would be possible for an entity which also has write access to the same storage to provision it with a key pair. This, however, requires that access to protected credential storage be kept confidential, and that access to it - both read and write - is strictly controlled.

4.4.6.4 Dynamic Integrity Management

4.4.6.4.1 Secured crash and recovery

See Security Problem Statement and clause 4.4.7.2.

It is expected that erroneous software, hardware, configuration, administration and power events will have an adverse effect on the VNF and cause a crash. A crash event results in the VNF and associated components being in an unknown and unexpected state, which can cause security concerns. In the event of a crash, several decisions need to be orchestrated:

- Is the primary goal to recover from the crash and restore availability as soon as possible?
- Is the primary goal to gather information (crash dumps) to analyse the crash event?
- How is the security of private information such as keys and passwords maintained during the crash and subsequent reporting and analysis?
- Does the crash involve external influence that should be mitigated to restore service, as in a DoS (Denial of Service) attack?
- Does information involve confidential hardware, hypervisor, VNF, OS, application and user state?

The posture on crash management and recovery should be part of the overall VNF configuration.

4.4.6.5 Application Programming Interfaces (APIs)

Define the guidelines for securing a rich set of standard and extensible APIs for NFV operations and management.

4.4.7 VNF Retirement

This clause provides guidance on all aspects of the decommissioning of a VNF and associated VNFCs.

4.4.7.1 License retirement

- Process to unregister from operational and suspended VNFs.
- Include DRM.

4.4.7.2 Secured wipe

There are three types of memory or storage that are most likely to retain data after the retirement of a VNFCI or set of VNFCIs. These are:

1) RAM on the hypervisor platform. When a VM is removed (either destroyed or migrated) from a hypervisor, it is the hypervisor's responsibility to ensure that any RAM which is freed up is wiped before reuse by another VM. This is standard functionality for the leading hypervisor platforms.

2) Storage local to the hypervisor platform ("local storage"). This may be of two types:

   a) Swap space: generally, VNFCIs should avoid use of swap space due to the performance issues associated with its use. However, if it used, it is the responsibility of the hypervisor platform to ensure that it is wiped before reuse by another VM.

   b) Local storage space may be preferred over remote storage for data which is short-lived and/or requires short access times. In this case, it is the responsibility of the hypervisor platform to ensure that it is wiped before reuse by another VM.

3) Storage remote from the hypervisor platform ("remote storage"). It will not always be possible or beneficial for a hypervisor to wipe remote storage as it may be used for a migrated VM (a single hypervisor only has knowledge of its local context), or used for long-term data storage between VNF instantiations. In this case, it considered best practice to have the VNFM coordinate with the VIM and any other Management and Orchestration components to manage the wiping of remote storage.

In all these cases, the assumption is made that the hypervisor has direct access to the storage being employed. This may not be the case where "pass-through" techniques are used to allow the VMs (and VNFCIs) being hosted to access the hardware or interfaces directly. In this case, the technique employed for the wiping of remote storage is preferred.

Note that there are always possible fault scenarios when a malfunction - for instance a crash - will leave unwiped data on storage. In these cases, care should be taken to ensure that processes are in place - which may need to be manually performed - to wipe allocated storage which may be subject to such issues. RAM, being ephemeral, is usually immune to such issues, though in exceptional cases, certain attacks may be performed which provide access to this data. These attacks are best mitigated by restricting physical access to the NFV Infrastructure.

A further case where possible attacks on unwiped data are possible are when a hypervisor is compromised while it still has data from previously retired VNFCI which it has not yet wiped. This is not considered to present any higher threat, however, to the compromise of a hypervisor with currently executing VNFCIs.

4.5 NVF Security Technologies

This clause is intended to represent a *technology* view of NFV security and supply associated technology guidance.

The security stack for Network Function Virtualisation is summarized as follows:

Layer	Key Security Measures
Management	Secured API's
Data	Encryption, Metadata security
VNFC	
OS	Trustworthy boot, hardening, patching
Hypervisor	Trustworthy boot, hardening, patching
Platform	Hardware-assisted virtualisation, UEFI, TPM, HSM

Figure 5: NFV Security Stack

4.5.1 Technologies and Processes

The following are descriptions of security technologies and processes with aspects unique to NFV. Descriptions include threats mitigated:

- Trustworthy Boot
- Hardening
- Patching
- Identity and Access Management
- RBAC specific to NFV roles
- Cryptographic Management
- Encryption
- Digital Rights Management (DRM)
- Analytics
- Privacy Protections, abstraction, tokenization and masking
- Threat Modelling
- APIs and Programmatic Interfaces
- Data protection and Metadata security
- Security automation

5 Trusted Network Function Virtualisation

5.1 NFV High-Level Trust Goals

Measured and Validated Trust Enabling Reliance

Trust is defined as confidence in the integrity of an entity for reliance on that entity to fulfil specific responsibilities.

Trust is highly dynamic and described in assurance levels based on specific measures that identify when and how a relationship or transaction can be relied upon. Trust measures can combine a variety of assurance elements that include identity, attribution, attestation and non-repudiation.

Additional high-level trust goals for NFV include:

- Establish guidance for NFV trust in platform, software, policies, processes, practices and interoperability.
- Define areas of consideration where technologies, practices and processes have different requirements than non-NFV systems and operations.
- Supply guidance for the operational environment that supports and interfaces with NFV systems and operations, but avoid redefining any security considerations that are not specific to NFV.
- The ability to specify and enforce detailed trust relationships for and between virtualisation resources for End-to-End Trust Lifecycle Management.

Trust is a complex issue, but in many cases, the decisions that are required within a particular NFV deployment will be simple. The present document does not attempt to enforce complex frameworks for NFV deployments, nor to insist that sophisticated runtime decisions need to be made at every stage in every NFV component's lifecycle. It does, however, attempt to explain and discuss the complexities that may arise within NFV deployments and to give guidance as to what issues should be considered, and to provide starting points for future work on trust within NFV.

Some myths or commonly ignored features about trust:

- Having a secured communications channel with another entity is never sufficient reason to trust that entity, even if you trust the underlying security primitives on which that communications channel is based.
- Trust is not a binary operation. There may be various levels of trust that an entity has for another.
- Trust may be relative, not absolute. Entity A may trust Entity C more than Entity B, without trusting either absolutely.
- Trust is rarely symmetric. Entity A may trust Entity B completely, whereas the amount of trust that B has for A may be very low. This does not always matter: a schoolchild may trust a schoolteacher, for instance, without any requirement for that trust to be reciprocated.
- One of the axes for trust is almost always time, and the trust relationship between two entities may be highly dynamic. Just because a certain level of trust was established at point T, it does not mean that that level will continue. Trust sometimes wanes - I may trust you less to repair my car if you do not maintain levels of training - and sometimes waxes - I trust you more to look after my money because you have never abused that trust in the past.

As noted above, trust is defined as confidence in the integrity of an entity for reliance on that entity to fulfil specific responsibilities. An entity A has no need to have a direct trust relationship with another entity B if B's operation has no direct impact on A. It may be that entity C is affected by entity B's operations, and that entity A relies on entity C, but this does not affect entity A directly, and therefore the trust relationship can be considered separate.
See clause 5.1.6 for further discussion of this topic.

There are other occasions on which entity A may choose to trust entity B to some extent, based on the trust relationship which entity C has with entity B and the trust relationship which entity A has with entity C. This is a subtly different case, and is defined as transitive trust. See clause 5.1.6.3, under clause 5.1.6 for further discussion of this topic.

5.1.1 Assigning trust

5.1.1.1 Why assign trust?

Whether explicit or not, there are two trust relationships which are available whenever an entity communicates with another:

- Trust the other entity completely.
- Distrust the other entity completely.

Of these two options, the first is usually the most useful. It is also often the most appropriate. There is, in the vast majority of cases, no need to perform complex calculations - often costly both in terms of computational and human resources - in order to explore appropriate levels of trust in a trust relationship.

Trust should be associated with threats and risk, and the need for complexity should be balanced against the real likelihood of business impact - as is the case for all security. Simply, if a trust metric does not defend against a threat or mitigate against a risk, it is not worthwhile considering it.

5.1.1.2 How to assign trust

The assignment of trust is the decision that an entity A should trust entity B in one or more particular contexts. Assigning trust is a matter of deciding what types of context are appropriate for a particular trust relationship between two entities at a particular time (time being one of the possible contexts for trust). Key criteria for assigning trust are:

- The identity of the entity to be trusted

- The contexts within which the trust should be constrained

Beyond these two criteria, other parameters will depend on the entities and their respective contexts.

The choice of contexts will impact on how trust is validated (see clause 5.1.2), and it is important to understand what data points and levels of granularity will be available to the entity making the validating choices (which may not be the trusting entity: see clause 5.1.6). For example, if one context of trust is a shared time with another entity, the accuracy of the time source available to each entity should be assessed, as well as possible network latency between the entities.

Different criteria may hold different weight relative to each other, and it may be possible to substitute one trust context for another if the requisite information is available to make a decision. It will not always be necessary to have complex logic built into the trusting entity in order to make such decisions: it may be that simple actuarial-style tables can be made available for certain decisions at run-time, simplifying the logic and improving reusability.

Another key question that should be addressed is what measures should be taken if trust breaks down (and cannot be re-established), or if it cannot be established in the first place. Options include:

- Attempt to contact another trusted entity (preferably up the chain of trust) to communicate the failure.
- Wait and attempt to re-establish trust after a defined amount of time.
- Increase logging levels.
- Continue to work as normal.
- Reduce operational parameters.
- Attempt to contact an alternative entity (e.g. in the case of a load-balancer).
- Cease operation.
- Destroy self.

The selection of all of these criteria should be performed before deployment: at design time (as the decisions will depend on the type of entity being designed), and in preparation for deployment (as the decisions will also depend on the environment into which the entity will be deployed).

5.1.2 Evaluating and validating trust

Validating trust is the exercise of going through the various measures of trust applicable for a particular trust relationship, evaluating the levels of trust assurance and, if they meet the criteria set, validating that trust relationship. This exercise may be carried out by the trusting entity itself, or with the aid of other entities: see clause 5.1.6 for more details of the latter. Even if trust is not delegated, it may be that the entity relies on results or information from other entities in order to make its evaluation: if this is the case, then it needs to trust these other entities in a relevant context.

There may be a cost associated with evaluating trust, and latency associated with it. It may be that checking trust requires a suspension of normal operations, or that evaluation requires communications with various other entities, and there may be associated delays. It is recommended that entities operating under such constraints monitor the time context of trust and attempt to re-validate trust relationships before they expire, parallelising queries with third party entities where possible.

5.1.2.1 Parameters for trust evaluation

Decisions on trust are rarely made on a single parameter, and trust is always contextual. The most commonly ignored parameter for machine-mediated trust is that of time: in human trust contexts, our trust for particular individuals tends to decay if we do not see them for a while, or if we are unsure if they have kept a skill current, for instance. The same should generally be true of trust relationships in the computational realm: as time passes, the chances that a trusted party has suffered a failure or been compromised increases. There are multiple other parameters, however, that are relevant within NFV, including:

- Geographical location
- Jurisdiction/regulatory location (public/private)

- Logical (network) location
- Hardware capabilities
- Hardware provenance and history
- Software capabilities
- Software provenance
- Execution instance history
- Chain of trust
- Time elapsed since last trust audit/check
- Date
- Time of day
- Ownership of hardware
- Ownership of software
- Security of network
- Appropriate use of encryption techniques
- Extent to which the software was initially hardened
- Measures in place to maintain the integrity of the software
- Physical security of the various locations over which the NFV components are deployed

For any particular NFV deployment, many of these axes will be examined before runtime, and trust relationships defined through contracts, policies and guidelines. Others will require revisiting by human players at various times, whereas others again will be subject to dynamic decision-making. The present document concentrates focuses on dynamic trust decisions within software at run-time, though some references are made to the other scenarios mentioned above.

5.1.2.2 Methods for trust evaluation

There are many methods that can be used for trust evaluation, and the choice will depend on available resources and the threats and risks relevant to the entity and the specific NFV deployment.

Techniques include:

- Reputational approaches: evaluating across a set of different elements, leading to a calculation of "reputation" (though the term and its usage is often loosely defined).
- Game theoretical approaches.
- Probabilistic approaches.
- Look-up tables.

5.1.3 Re-evaluating trust

Many of the issues that need to be addressed revolve around establishment, re-establishment or revocation of trust. The requirement to re-evaluate trust may be prompted by a variety of different events, including time-based contexts such as a time-out or set frequency. The list below addresses events that may be associated with life cycle events, and acts to allow a categorization and simplification to a smaller set of well-defined trust use cases:

- Disappearance of an entity

- Appearance of an entity
- Movement of an entity - e.g. migration
- Duplication of an entity
- Re-configuration of an entity
- Changes to the description of trust measures
- Changes to the repudiation of roots of trust

In most cases, some or all of the following parameters may apply:

1) Expected/Unexpected.
2) Availability/unavailability of communications with CA and/or Management and Orchestration entities (see clause 5.1.5).

One example of an expected movement of an entity would be the migration of the VM containing a VNFCI. In this case, the migration is planned, and another entity in the chain of trust - a VNFM or Orchestrator, for example, may inform the trusting entity that the other entity is about to move and inform it of the new location. Alternatively, if it is aware of a pending move, the trusted entity may inform the trusting entity itself. Once the trusted entity has moved, the trusting entity may re-evaluate the trust relationship, bearing in mind information such as the jurisdiction, geographical, and location of the entity, the level it has in the provenance of that information, etc.

Disappearances of the trusted entity are a special case. For most trust relationships, there is no need to maintain a connection between the two entities at all times, or even to institute a regular heartbeat. Where this is not instituted, there is knowledge of possible network topology failures (see Problem Statement: Network topology validation are relevant, and it may be appropriate to inform trusting parties - or their delegation partners (see clause 5.1.6) of failures which may have impacted connectivity, but may not otherwise have been detected. This means that it is not always possible for one of the two members of a trust relationship to ascertain whether the other has been restarted: the key reason to re-establish trust. There are a number of cases when a restart of an entity might occur:

- Infrastructure failure
- Software failure
- Planned maintenance:
  - Typically for software upgrade or update
- Emergency maintenance

In some of these cases (e.g. planned maintenance), the trusting entity may be informed of the expected restart.

The re-evaluation of trust implies that the trust relationship has not completely broken down: it is a change in an existing trust relationship. Where the relationship has completely broken down, it will need to be re-established: see clause 5.1.5.

5.1.4 Invalidating trust

There are some cases where trust relationships are invalidated on purpose:

- Notification from the trusted entity that it should no longer be trusted - this is most likely due to an expected destruction, decommissioning or retirement, but could be if the entity believes that it has been compromised.
- Notification from another entity up the chain of trust that a trust relationship should be invalidated.

In these cases, the trusting entity should generally not attempt to re-establish the trust relationship.

Other cases may be due to a failure to re-validate a trust relationship after a re-evaluation or due to some external factors. All of the reasons listed in clause 5.1.3 may lead to this taking place, and where this has happened, the trusting entity may make the decision to invalidate the trust relationship.

In many cases, the (previously) trusting entity will not be the sole entity with an interest in this trust relationship. Many trust relationships have an element of delegated trust (see clause 5.1.6, and it is important to ensure that entities from whom trust was delegated, or to whom trust was delegated, are aware of the change in trust relationship in order to allow them to re-evaluate or invalidate any relevant trust relationships themselves.

5.1.5 Re-establishing trust

Re-establishing trust is defined here as the process of recreating a trust relationship between entity A and entity B which has previously existed, but which has, for whatever reason, failed.

Other scenarios which may require the trusting entity to re-establish trust with another are:

- Loss of network connection between the entities, which could lead to spoofing or man-in-the-middle attacks.
- Compromise of the trusted entity.

Where such events occur, they may or may not be detected by the trusting entity. There exist, therefore two sets of options: delegation up the chain of trust and peer-mediated distrust.

5.1.5.1 Delegation up the chain of trust

Delegation up the chain of trust is when the trusting entity delegates to an entity further up the chain of trust the decision of whether it should trust an entity between them in the chain. For instance, if a VNFCI trusts a VNFM, but has a Certificate Authority as its initial root of trust, delegation would involve the VNFCI reaching the decision, based on the CA's decision, that it should no longer trust the VNFM (or, in the contrary case, that it should trust the VNFM after all). There are three ways in which this decision could be taken:

1) Proactively by the trusting entity, by contacting the initial root of trust or its agents (such as a Certificate Revocation service).
2) Reactively by the trusting entity, when the initial root of trust contacts it.
3) Passively by the trusting entity, based on other information, such as the expiry date on a previously issued certificate.

Rather than communicating with the initial link in the chain of trust - which may not be possible due to networking policies, the trusting entity may communicate with a Trust Manager - see clauses 5.1.6 and 5.1.7.1.

5.1.5.2 Peer-mediated distrust

In certain cases, an entity (A) may suspect that another entity (B) that it trusts should no longer be trusted (due, for instance, to B having been compromised). It may have the option to delegate a trust decision up the chain of trust (see clause 5.1.5.1), but another alternative is to consult peer entities (for instance, a VNFCI consulting other VNFCIs in the same VNF instance). Where a number of peers are trying to make a decision as to whether to trust one another, or another mutually trusted entity, this is called a Byzantine Generals problem. In cases where delegation up the chain of trust is not appropriate of available, implementing a Byzantine fault tolerant system between various peer entities may provide opportunities for trust decisions to be made (see http://en.wikipedia.org/wiki/Byzantine_generals).

5.1.6 Delegating trust

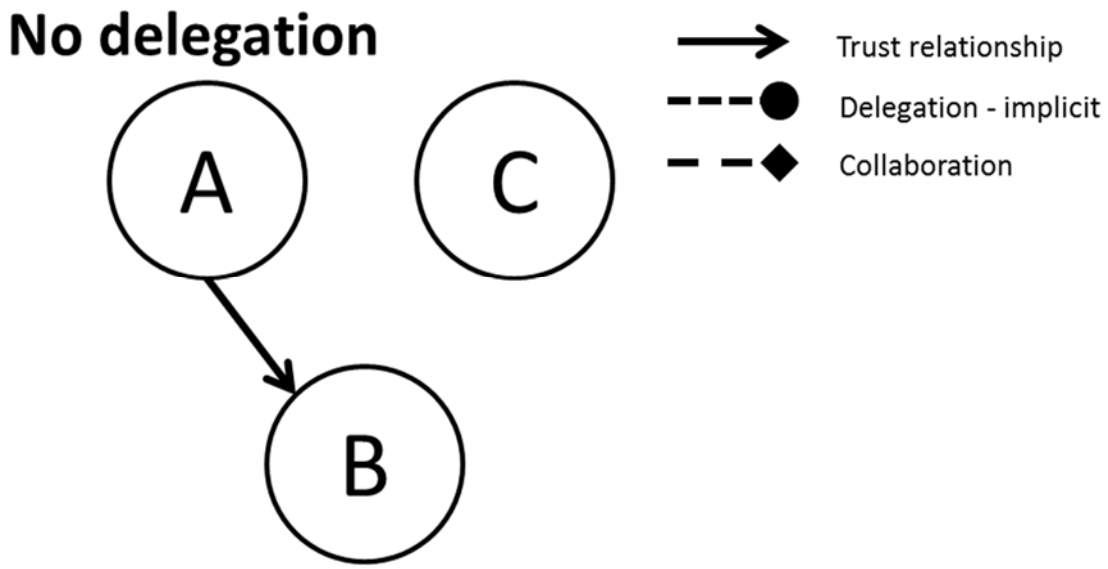

Figure 6: Trust Delegation - No Delegation

Sometimes an entity A needs to establish a trust relationship with an entity B, but lacks some or all of the necessary capabilities to evaluate the appropriate level of trust. This lack could be due to a variety of issues, for example:

- Lack of access to historical information about entity B's behaviour.
- Lack of framework to evaluate B's trustworthiness.
- Lack of direct network access to a Revocation Authority to check whether a certificate that B has presented is current.

Quite often, it is inappropriate to include sophisticated trust logic within a component such as a VNFCI which has very specific duties, and where duplication of such logic across multiple components would be computationally wasteful or architecturally messy.

5.1.6.1 Directly delegated trust

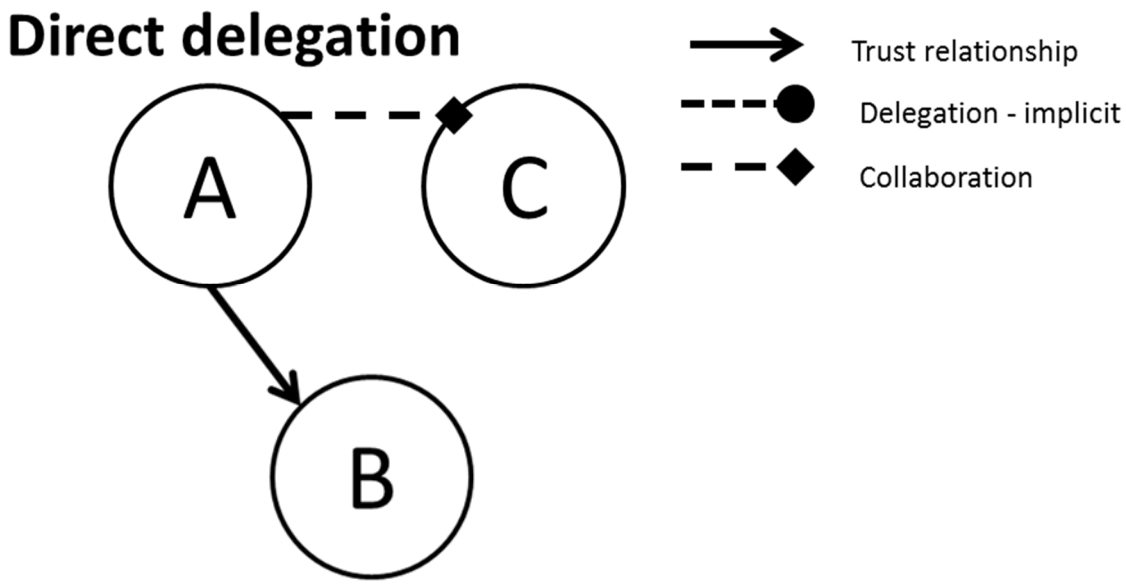

Figure 7: Trust Delegation - Direct

Where an entity A is unable to evaluate the appropriate level of trust for a relationship with another entity B, A may choose to delegate the decision to another entity C, which is in a better position to make such a decision. In this case, there should be an explicit element to the trust relationship from entity A to entity C that it is happy for entity C to make such decisions for entity B, or components of entity B's type: this is a particular context within the trust relationship from entity A to entity C.

The delegation of this trust may not be an explicit one, but may be implicit in the design and/or deployment options of entity A. For example, it may be that a VNFM makes decisions about trust relationships between VNFC Instances as they are being instantiated, and the VNFCIs themselves have no need to have any involvement in the decision at all. The delegation, then, is in the deployment and configuration options of the VNF Package.

5.1.6.2 Collaborative trust

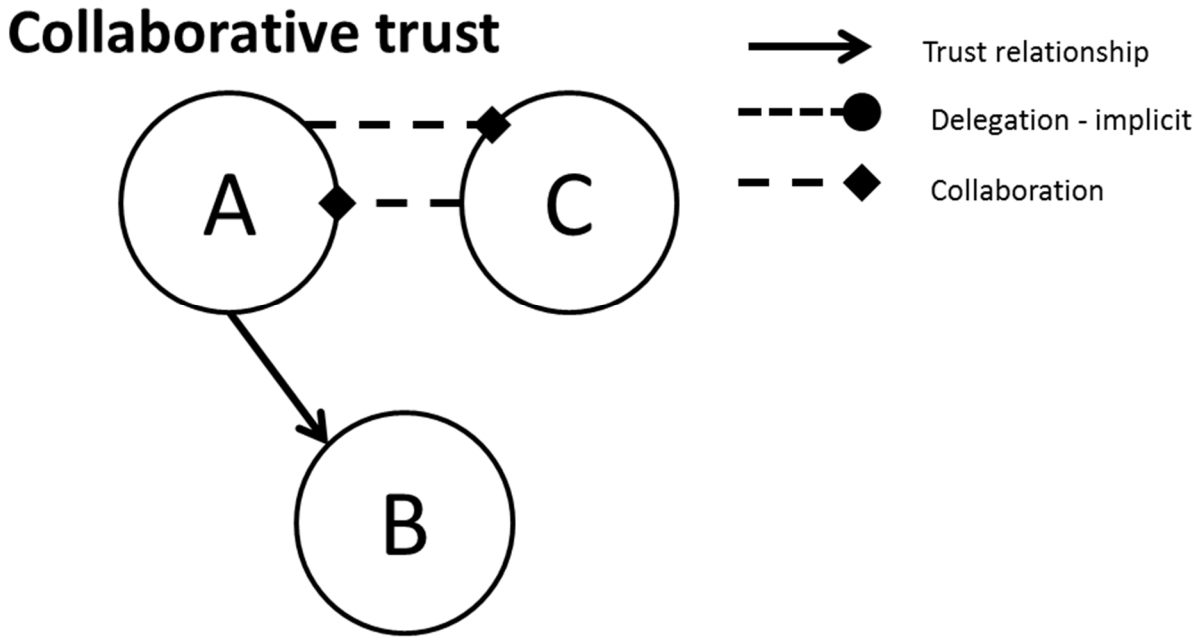

Figure 8: Trust Delegation - Collaborative Trust

Collaborative trust involves two more entities (entities A and C) working together to decide whether to trust another (entity B) - the final goal may be for both entity A and entity C to have a trust relationship with entity B, or just one of them. The expectation is that entities A and C may have different information available to them which will help them to make a more informed decision about the trust relationship with B. This may be because they have more information either:

- across the same parameters because entities A and C are peers (they might both interact with entity B in the same way, but over different networks, for instance);
- across different parameters because their interactions with entity C are different (entity A might provide data to entity B for consumption, whereas entity C might be a consumer of data from entity B). In this case, there is not even any requirement that entities A and C are in the same NFV architectural domain or share the same abstracted view of entity C.

The expectation with collaborative trust is that contexts of trust will be shared, but parameters may be different. There should also be opportunities for entities A and C to communicate if trust levels - or the parameters on which they are based - change, so that re-evaluation can be performed by all relevant parties.

A good example of collaborative trust might be that several VNFCIs on a single hypervisor host might collaborate to decide that they will trust another VNFCI which resides on the same hypervisor host more than they might if it were on a separate host because:

- they have a trust relationship with the hypervisor already;
- they share the same geographical location;
- communication channels with this VNFCI may be local and are therefore, compared to off-host communications:
  - more secure;
  - more reliable;
  - of lower latency.

5.1.6.3 Transitive trust

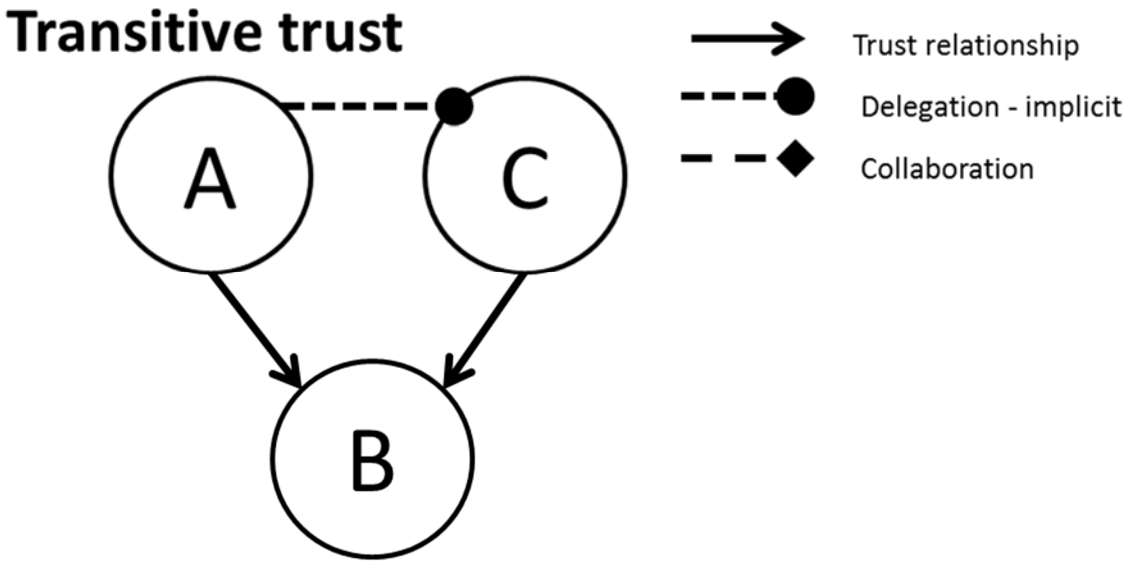

Figure 9: Trust Delegation - Transitive Trust

Transitive trust is the decision by an entity A to trust entity B because entity C trusts it. This is not the same as a pure delegation of trust, as entity C may be unaware of entity A's reliance on it: in other words, entity C is no way brokering the trust relationship from entity A to entity B. Unlike in "pure" delegated trust (see clause 5.1.6.1, there is no explicit element to the trust relationship to entity A to entity C (that the latter is aware of) that entity B is trusted due to A->C relationship.

The key danger of transitive trust is that because there is no explicit element of this type, entity A cannot be certain that the contexts for trust - and associated parameters - from entity C to entity B are entirely aligned with entity A's contexts. Nor can it be sure that the relationship from entity C to entity B is still current unless it has methods by which is can examine any re-evaluations, re-validations and invalidations of the C-> trust relationship.

The ssh model for transitive trust is notoriously prone to the dangers noted above.

5.1.6.4 Reputational trust

Reputational trust is a specific instance of transitive trust, where entity A takes a view on the trustworthiness of C based on a rating of B's trust relationship with C. Usually, there will be many other entities trust C (say D, E, F, G, etc.), and some algorithm will be applied to the various ratings published by these entities in order to allow A to make a decision about trusting B. This algorithm may be applied by A (in which case A needs access the ratings of the various parties C, D, E, F, G, etc.) or by a third party - see, for example, clause 5.1.7.1 in the present document). A distinguishing point about this type of transitive trust is that it is almost always explicit: the entities C, D, E, F, G, etc. are likely to be aware that they are participating in a reputational trust scheme.

5.1.7 Scope of trust

Trust is, like many other aspects of an architecture, layered. Direct trust relationships should generally not extend beyond the following bounds:

- Trust within an architectural layer.
- Trust up one architectural layer.
- Trust down one architectural layer.

This allows architectural abstractions to be maintained. The key techniques to allow broader trust to be built up are chains of trust and the delegation of trust between multiple entities (see clause 5.1.6). It is also more likely that relevant communications will be available between the various entities involved in forming trust relationships.

5.1.7.1 Trust manager

There are many occasions when placing significant trust determination logic in entities - which are generally of very specific function, and may be designed to be as lightweight as possible - is not appropriate. Indeed, it may be that even with sophisticated trust delegation models (see clause 5.1.6) within the scope of trust suggested in clause 5.1.7, "low level" entities will not have appropriate information to make informed decisions about their trust relationships. In this case, the availability of an deployment-wide Trust Manager entity is suggested. This should sit within the Management and Orchestration administrative domain, and should be a long-lived entity (see clause 5.3.2.

The availability of this entity allows for a great simplification in the trust relationships for a single NFV deployment, and is very attractive for exactly this reason. It may be appropriate, particularly in simple deployments, for the Trust Manager to contain all of the explicit trust logic for the entire deployment, but a Trust Manager may have an important role to play even in complex deployments.

Benefits of a Trust Manager:

- Less logic required by other entities within the deployment.
- Can act as a *deus ex machina*, providing information across different architectural layers.
- Act as an interface between different administrative domains and operators.
- Act as repository for trust around VNF Packages and vendors.
- Provide historical data about entities that are more long-lived than the trusting entity.

Drawbacks of a Trust Manager:

- Single point of failure.
- Single point of attack.
- May require communications channels across architectural boundaries which are not easily maintained.
- Encourages "crunching" of trust contexts in a smaller set of implicit contexts.
- Encourages assumptions that all entities share the same trust contexts.

5.2 NFV Trust Use Case Summaries

NOTE: Throughout this clause, it is important to be aware that a VNF (a virtual network function) is an abstraction of a collection of VNFCIs (virtual network function component instances). The VNCIs make up an executing resource which provides the instantiation of the service - the software contract - offered by the VNF. We can therefore talk about trust between VNFs, for instance, but the actual entities which are establishing the trust relationships are VNFCIs. This is brought out explicitly in some of the examples below - on other occasions, the abstraction is used as convenient shorthand for discussion and description.

5.2.1 Intra-VNF Trust: Trust within Virtual Network Functions

Within the VNF, trust establishment, validation and management are required for VNF operations, for contained VNFC operations, and for secured interface with external assets and services. See clause 5.4 for more details.

5.2.2 Inter-VNF Trust: Trust between Virtual Network Functions

There are at least three types of trust that may be required between different network functions:

1) VNF A trusts the syntactic correctness of the output from VNF B according to the Software Contract offered by VNF B.

2) VNF A trusts VNF B to perform the correct operations on VNF A's output.

3) VNF A trusts VNF B to perform operations which have indirect impact on data or state that VNF A holds.

These types of trust may be combined. Type 3 requires care, as standard Object-Oriented design principles would suggest that an entity performing operations which have side effects on another should generally be considered a single object or component, and should generally therefore be deployed as a single unit - as a single VNF in NFV nomenclature.

In all these examples, however, the preferred mechanism is for the VNFM for each VNF to make the trust decision (and review it as required), rather than the VNFCIs which comprise the VNFs. This assumes timely communication between the VNF and its VNFM. In this case, there is a strong transitive trust relationship between the VNFCIs and their VNFM. It makes little sense to work around the VNFM, which is supposed to act as the Management and Operations "brain" of a VNF, and as the VNFM will, by necessity, be involved in the initial set-up and configuration of a VNF, it should also make the initial trust assessment and pass the relevant VNFCIs the required information (certificates and capability approvals) for making relevant communication connections between VNFs.

Trust maintenance through VNF lifecycles

As Virtual Network Functions are managed, orchestrated and retired, they need to ensure that the other entities with which they interact can be trusted to perform their expected functions with the appropriate assurance through the lifecycles not only of the VNFs themselves but also of the different components. The types of metrics on which trust relationships will be based will intersect with those required at VNF instantiation, but will be different and may be more complex as different entities interact with the VNF.

Without appropriate trust relationships, entities cannot be certain that the correct operation of their own components or those with which they interact can be assured, and service providers cannot be sure that the services which they are providing will act correctly. The maintenance of trust relationships - some short-lived, some long-lived, some direct, some delegated - between the various NFV entities includes components such as management and orchestration software, VNF abstractions, VNFCs, virtual machines, hypervisors, network elements and NFV infrastructure hardware.

The security and trust document will describe the building blocks required to maintain trust relationships within an NFV environment, and techniques and processes that can be employed to maintain appropriate levels of assurance for service providers.

5.2.2.1 Managing trust between a VNF instance and its VNFM

The functionality of a VNFM - an entity that controls and manages the various components of a VNF instance and communicates with the various parts of the Management and Orchestration domain which are stakeholders in the VNF instance's operation - can either be included within the VNF or, more typically, exist as a separate entity, within the Management and Orchestration domain.

The trust relationship between a VNF instance and its managing VNFM (where this exists as a separate entity) is one of the key relationships that the VNF has, as the VNFM is generally expected to provide transitive trust services for trust between the VNF instance and other VNF instances (see clause 5.2.2.2). If trust in this relationship fails, then, the assurance that the VNF instance can hold in other trust relationships cannot be easily renewed, and should be expected to decay, as it there is no easy way to maintain it.

It is important, therefore, to establish and maintain the trust relationship between a VNF instance and its VNFM. This should be a bi-directional trust relationship. This trust relationship should be checked as required, as trust should generally be expected to decay over time unless further checks are made. In particular, there are various lifecycle events which should occasion a re-evaluation of the trust relationship:

- On the migration of VNFC instances within the VNF instance.
- On the suspend/resume of VNFC instances within the VNF instance.
- On the creation of VNFC instances within the VNF instance.
- On the deletion of VNFC instances within the VNF instance.
- On the failure of VNFC instances within the VNF instance.

Further work and specification is expected on this topic.

5.2.2.1.1 VNF instance's trusting of the VNFM

Each VNF instance requires either a set of VNFC instances or a Master VNFC instance: we can refer to this outward-facing VNFC instance as the VNF instance for simplicity. In both of these cases, if the VNFM is not the VNF instance's initial root of trust (see clause 5.4.3.1), then it should be one of the first relationships that is established. The key assurance that the VNF instance needs to establish is that of the identity of the VNFM: a VNFM should be provisioned with a list (which may have only one entry) of acceptable VNFM identities. These are unlikely to be unique: rather, the VNFM will have a certificate signed by a party which should be trusted by the VNF instance.

5.2.2.1.2 VNFM's trusting of the VNF instance

For a VNFM to trust a VNF instance, the core criteria are:

- Trust in the provenance of the VNF Package (see clause 5.2.3.1).
- Trust in the Trustworthy Boot process (see clause 4.4.5.1).

5.2.2.2 Managing trust between VNF instances

For the purposes of this discussion, it is assumed that each VNF has a VNF Manager (VNFM). This may not always be the case, as several deployment scenarios are actually possible:

1) A VNF instance has its own VNFM: there is a 1:1 mapping between VNF instances and VNFMs.

2) Several VNF instances share the same VNFM.

3) A VNF instance has no external VNFM entity, but keeps this capability within itself. Despite this, the VNF instance will still require the ability to communicate with the Management and Orchestration domain, and therefore the VNFM functionality still exists - the Management and Orchestration domain functionality just extends into the particular VNF instance. This case is therefore functionally equivalent to the 1:1 mapping case.

The key parameters to be considered when establishing the correct level of trust for a VNF instance to have in another are:

- Assurance of identity
- Historical behaviour of the VNF instance
- Historical behaviour of other instantiations of the VNF
- The infrastructure (NFVI) hosting the VNF instance
- The security of the network between the two VNF instances
- The legislative/jurisdictional location of the other VNF instance

The target VNF instance may be able to make representations about some of these (for instance, its identity, the infrastructure on which it is hosted, and its legislative/jurisdictional location), in order to test these, both VNFs need to share a trust infrastructure. This is a function which should fit in the Management and Orchestration domain, and to which the VNFs may not have a direct connection. For these reasons, as because the target VNF instance cannot make reliable or informed representations about the other parameters, decisions about whether VNF instance can have - and maintain - trust in another should be made by the VNFM, not by the VNF instance itself.

This transitive trust relationship is entirely appropriate, as the VNFM is entirely in keeping with the different functions of the VNF instance and the VNFM. It is the latter which provides the interface with the Management and Operations domain, and should have access to the appropriate trust infrastructure. It is also the VNFM which will have interfaces and communications with the Orchestration layer, which maintains information about the service graph which makes up the connections between the VNFs. In the other direction, the VNFM will have interfaces and communications with the VIM layer, which will be able to provide information about the NFVI.

Where several VNF instances share a single VNFM instance, there are some decisions that it will be able to make without the need to consult with other entities. As long as it has a high enough level of trust in the various VNF instances that it controls (see clause 5.2.2.1), it should be equipped with enough information to manage the continued trust relationships between them. It should, however, still consult with the relevant entities within the Management and Orchestration domain at instantiation time to ensure that it has up-to-date information on parameters such as historical behaviour of different VNF instances of the VNFs that it is about to deploy.

There currently exists no trust infrastructure, nor function for maintaining historical data about the behaviour of VNF instances. This is identified as a GAP.

5.2.3 Extra-VNF Trust: Trust external to Virtual Network Functions

Trust establishment at VNF instantiation

As Virtual Network Functions are created, they need to ensure that the other entities with which they interact can be trusted to perform their expected functions. Without appropriate trust relationships, entities cannot be certain that the correct operation of their own components or those with which they interact can be assured, and service providers cannot be sure that the services which they are providing will act correctly. The establishment of trust relationships - some short-lived, some long-lived, some direct, some delegated - between the various NFV entities includes components such as management and orchestration software, VNF abstractions, VNFCs, virtual machines, hypervisors, network elements and NFV infrastructure hardware.

The security and trust document will describe the building blocks required to create trust relationships within an NFV environment, and techniques and processes that can be employed to create appropriate levels of assurance for service providers.

5.2.3.1 Establishing trust in a VNF Package for deployment

As noted in clause 4.4.4.7, there is a trust component to the process of managing VNFs and their components. The clause 4.4.4.7.2 briefly describes the technical process for checking the integrity of a VNF package, but does not explore the processes required.

A key point to understanding the trust relationships required is understanding the actors. The following actors can be identified:

- NFVI domain.
- Management and Operations domain.
- NFV provider - the vendor who provides the VNF Package.

This analysis assumes that the operator is operating the Management and Operations domain: if a third party is employed in this role, appropriate changes should be considered.

5.2.3.1.1 NFVI domain

In a standard, "pure" virtualisation deployment, the infrastructure (in this case, the NFVI actor) generally has little need to have a great deal of assurance that the VMs which it is hosting (the components of the VNF Package which will be instantiated as VNFC Instances) are well-behaved (and suitable for deployment), as the encapsulation provided by virtualisation gives the hypervisor high levels of control over the use of resources by the VMs. A variety of the use cases and deployment scenarios under consideration for NFV, however, require that the VNFCIs have direct access to certain NFVI infrastructure resources, rather than their being mediated through the hypervisor. This means that there are increased opportunities for malfeasance by the VNFCIs, and that the NFVI will require appropriate levels of assurance before agreeing to execute the components of a VNF Package. Since it is the Management and Operations domain which will have control over the service catalogue holding the VNF Package, and it most likely to be in a position to make assurance checks on it, this implies a trust relationship from the NFVI to the Management and Operations domain in this context: this is an example of a transitive trust relationship.

It should be noted that there is the possibility of a channel back from the NFVI to the Management and Operations domain to inform the latter if there is malfeasance (or attempted malfeasance) on the part of an instantiated and deployed VNF. This is most likely to be performed procedurally, however, rather than automatically in real-time, and would be expected to affect the trust relationship between the Management and Operations domain and the VNF provider. An example of the type of misbehaviour that might be reported is a "noisy neighbour" VNFC Instance within the VNF Instance. This is where a particular VNFC Instance uses - or attempts to use - more network or storage bandwidth than expected, with the result that other VNFC Instances share that the same I/O channel may be starved of bandwidth if resource constraints are not - or cannot - imposed by the NFVI. Beyond the impact that this will have on the other VNFC Instances sharing the channel - if not controlled - this is important information for the Management and Operations domain to have, since it may both have immediate operational impact on existing services and inform decisions about the trust that should be placed in the VNF Package from which the particular VNFC Instance came, and, in consequence, the NFV provider of that VNF Package.

5.2.3.1.2 Management and Operations domain

The key actor in making a decision on whether to instantiate a VNF from the VNF Package is the Management and Operations domain. The decision may need to be made from the point of view of two separate actors: the Management and Operations domain itself, and the NFVI domain. This latter may require assurances from the Management and Operations domain that the VNF package is suitable for deployment (see above).

There are two decisions which need to be made, and they are subtly different:

- around the *trustworthiness* of the VNF Package to run on the NFVI: whether the components of the VNF Package will "behave" appropriately, and not misuse - or attempt to misuse - the resources provided by the NFVI;
- around the *suitability* to perform the service required by the Management and Orchestration domain: whether the components of the VNF Package will meet the Software Contract advertised by the VNF package's VNFD, as implemented by the NFV Package vendor.

The tests for these will be different, but both will depend crucially on the trust relationship with the VNF provider. Importantly, changes to the trust levels for either of these are likely to affect the trust relationship between the Management and Orchestration domain and the VNF provider, thereby affecting the levels of trust associated with the other. For instance, if a VNF Package, once instantiated, violates the RAM constraints advertised by the VNF Descriptor, this may cast doubt on whether the stated measures of performance are correct.

Trustworthiness

The first test for trustworthiness is to check the provenance of the VNF Package. Use of certificate schemes and PKI, along with the techniques described in clause 4.4.4.7, allow the Management and Orchestration domain to check that the VNF Package is from the expected vendor. Beyond that, however, there are broader measures of trustworthiness, which are likely to be built on a set of agreed components and acceptable behaviours. For example:

- Use of a particular Operating System version.
- RAM and CPU constraints.
- Use of restricted set of hardware drivers.
- Use of set of hardened libraries or a previously hardened OS image.
- Requesting of only a particular set of hypervisor capabilities.
- Restrictions on the resources to be requested at instantiation of VNFCIs and over their lifecycles.

An operator is likely to have a published set of standards for these parameters, and to insist that a VNF provider guarantees that their VNF Package meets them. A certain amount of introspection into the VNF Package - particularly at the VNF Descriptor level - is possible, but the Church-Turing thesis tells us that it will not be possible to be certain of the trustworthiness of a VNF Package without executing it, or even of future behaviour when it is executing.

There are a variety of checks that can be performed on the VNF Package or its components in an attempt to increase assurance that it is trustworthy. These may include checks before the instantiation of the VNF Package and checking after instantiation. The former should generally be performed as part of the due diligence before accepting the VNF Package into the service catalogue.

Checks that may be performed before instantiation:

- Examination of the VNF Descriptor to ascertain expected behaviours.
- Examination of the image files for VNFCs, looking for expected binaries, libraries.
- Checking logs of historical behaviour by other VNF instances from the same VNF Package.

Checks that may be performed after instantiation:

- Introspection of running VNFC Instances by the hypervisor platform.
- Trustworthy Boot and attestation from within the VNFC Instances, if available.

- External monitoring of service quality and behaviour.
- Monitoring of VNFC Instances by the hypervisor.
- Reports from the VNFC Instances via the VNFM.

Suitability

Suitability revolves around whether the VNF Package, once instantiated, will provide the required functionality to perform the service required. The main measures for this are likely to include:

- Incoming and outgoing flows (syntactic correctness).
- Management and control APIs.
- Reporting APIs: monitoring, audit, etc.
- Measures of reliability.
- Measures of performance.

Although these are basic measures of fitness for purpose, unless the Management and Operations domain can be assured that a VNF Package is suitable for deployment, it should not instantiate it. This is a about meeting the Software Contract which the VNF provider has associated with the VNF package.

5.2.3.1.3 VNF provider

The identification of a single VNF provider does not imply that all the components of the VNF package come from a single vendor. It may be that there are components from multiple vendors, but the vendor of the VNF Package is a single entity, and it is with this entity (actor) that contractual agreements will be made around fitness for purpose, support, maintenance, etc. In the case where multiple vendors provide different components of a single VNF package, it will be the responsibility of the VNF provider to manage any other contractual arrangements with these parties. There is an element of transitive trust through the VNF provider through to these entities, but this is managed through standard business processes, and is not expected to be dynamic or tested at runtime.

Maintaining the trustworthiness and suitability for deployment is a key reputational requirement for VNF providers. Since virtualisation means that the NFVI provides the execution environment for Network Functions, rather than a vendor-controlled hardware environment, exhaustive testing in-house, support for staging deployments and real-time support agreements will need to be carefully considered and may make up part of the contractual arrangements between the VNF providers and the operator.

5.3 Trust between Management and Orchestration entities

The key entities currently defined in the Management and Orchestration domain are:

- NFV Orchestrator (NFVO)
  - The various catalogues which the NFVO controls are considered subsidiary to the NFVO.
- VNF Manager (VNFM)
- Virtual Infrastructure Manager (VIM)

To this list may be added two further entities which are not currently defined, but whose presence has been noted as a gap:

- Certificate Authority - note that this does not necessarily need to be a full Certificate Authority instance run by the NFV deployment operator. What is required is that Certificate Authority capabilities are available to the NFV deployment, and that they are administered from within the Management and Operations domain.
- Trust manager for delegating trust decisions (see clause 5.1.7.1).

All of these elements need to establish trust relationships with each other.

There are two key points about entities within the Management and Orchestration domain which are important when considering their trust relationships:

1) They will not be hosted on the NFVI infrastructure.

2) They will generally be long-lived entities.

5.3.1 Management and Orchestration infrastructure

The statement that Management and Orchestration entities will not be hosted on the NFVI infrastructure is sometimes misunderstood, requires careful clarification. There is no assertion being made that these entities will not, cannot, or should not be virtualised. There are many deployments where the decision to virtualise some or all of the components making up the various Management and Orchestration domain entities will be a sensible one, and should be subject to standard operational decision-making processes around the virtualisation and deployment of key infrastructure.

However, the platforms on which these entities will be deployed should not be part of the NFVI. This is for several reasons:

- The specifications for the NFVI are carefully tuned to the requirements of VNFs, and are different to the requirements of Management and Orchestration entities.
- Administration of the NFVI is under the control of the VIM, and virtualising the VIM could therefore lead to circular dependencies.
- The operational management of network services may be under the control of a different group to that for Management and Operational entities, with correspondingly different service characteristics (see, for example, the point above around the long-lived nature of the latter).
- Management and Operational entities may be sited in a separate geographical location to the services that they are controlling.

The administration of the infrastructure for Management and Orchestration entities is considered, therefore, to be separate to the administration of the NFVI. The security of the infrastructure used to host these entities - whether virtualised or not - should be subject to the same best practices employed for existing systems. Where these entities are hosted by an external provider - such as in a third party Data Centre - care should be taken to ensure that relevant contractual and audit processes are followed to ensure appropriate levels of trust and security.

5.3.2 Implications of long-lived entities

It is important that the lifetime of Management and Orchestration entities should be long, relative to the lifetime of entities which they control, such as VNFs, and VNFCIs. In order to allow for roots of trust to remain as such, they will need to be long-lived so that complex and computationally expensive processes do not need to be applied, should a trusting entity need to re-establish trust with a trusted entity which has restarted (see clause 5.1.5 for more details).

Trust, left unattended, should decay, as one of the parameters of trust is time. The rate of decay need not be fast if initial parameters of trust are appropriate, and if the initial level of trust is high, those entities trusting Management and Orchestration entities will not necessarily need to re-evaluate trust frequently. Re-establishment of trust (see clause 5.1.5) will need to occur if the relationship has failed completely, but in most cases, performing a number of checks to re-evaluate the trust relationship may be enough to reset it to an acceptable level.

Within the Management and Orchestration domain, the key entities for trust establishment are the Certificate Authority and the NFV Orchestration (NFVO). As they are the roots of trust for the entire NFV deployment, processes involving human actors should be instituted. Best practice processes for Certificate Authorities are standardised within the industry, and should be followed where applicable. Similar processes should be established for NFVOs. In both cases, failover procedures - manual or automatic - will usually also be required. It is important to stress that the final trust of the entire NFV deployment lies not in computational or virtual components, but in humans and real organizations: in this way, it is exactly the same as non-virtualised operator deployments. Such trust relationships are beyond the scope of the present document.

Where re-evaluation of trust is required, the key test is likely to be that the identity of the trusted entity has remained the same. This does not necessarily mean that the trusting entity is communicating with exactly the same entity (the identity does not map directly to a uniquely component or UUID, for instance), as fail-over may have occurred, but in this case either the identity of the fail-over paired entities will need to be the same, or there techniques such as cross-signing of certificates should allow the trusting entity to be assured that the new entity is an acceptable alternative. In some cases, however, such techniques may not be available, or may not be considered robust enough, in which case, the re-establishment of trust will be required (see clause 5.1.5).

A final implication of the fact that an entity is long-lived - concomitant with its being a key link in the chain of trust, in the case of the Certification Authority and NFVO - is that great measures should be taken to protect it. Physical, procedural, legal and organizational protections should all be put in place to provide the highest levels of protection for all entities of this type, and not just the Certificate Authority.

5.4 NFV Trusted Lifecycle Management

The Trust Lifecycle represents the trust processes spanning from NVF platform guidance, through VNF instantiation, operation and retirement.

This clause is intended to represent a *process* view of NFV trust and supply associated process guidance.

5.4.1 Objectives and Policy

Includes the definition of trust relationships, transactions, actors, identity, reputation, mutual and transitive trust. Based on NFV use cases.

Also, organizational aspects of Trust, Multiparty and Multitenant trust, Subscriber trust (primarily concerned with logging and auditing identification and privacy issues), Trust across domains, transparency, privacy, assurance, governance.

Throughout the trust lifecycle, trust is *described*. Describing trust allows an understanding of what attributes and elements were used to build and verify a chain of trust and the conditions under which they are valid. Attributes are a quality, property, or characteristic of somebody or something. Elements are a separate identifiable part of something, or a distinct group within a larger group.

Attributes of trust include:

- State of trusted computing base.
- Trustworthy Boot status.
- Attribution of package source.
- Attestation of package/module/patch validity.

Elements of trust include:

- Non-repudiation of a transaction.

Trust measures can combine a variety of assurance elements that include identity, attribution, attestation and non-repudiation. Assurance levels can be described that include the concept of baseline, high-assurance, immutability and non-repudiation.

Attributes and elements each have an assurance measure associated with them that incorporate reputation, authenticity, risk posture, SLAs and repudiation, Assurance is one of the most basic levels of trust, with multiple assurance factors often combined to build an appropriate representation of trust.

Measuring and verifying trust are processes that are auditable, built upon actuarial principles, empirical, incorporates legal definitions, includes dependencies and describes relationships to the overall chain of trust.

The CSA CloudTrust project [i.2] should be reviewed for alignment of requirements.

5.4.2 Defining a Chain of Trust

The chain of trust consists of all the attributes and elements used to build and verify trust, as well as a description of the relationships between the links in the chain. The links can be considered as individual domains of trust, with the elements and attributes that constitute a link existing for a specific purpose that can stand on its own with a specific definition of an individual trust relationship.

A chain of trust begins with a root of trust. A Root of Trust can be based in hardware, software, policy, cryptographic secrets and signatures or through attestation.

Represents the end-to-end nature of trust and describes dependencies.

- Multi-party, dynamic.

5.4.3 Establishing Roots of Trust for VNFs

Defines the basis of trust.

5.4.3.1 Initial VNFC root of trust establishment

In order to be able to be able to form trust relationships with other entities, a VNFC requires an initial root of trust. Without this, it cannot be sure whether it can trust any other entity: this is sometimes known as the "turtle's problem" (see note) within the IT security community. The standard entity to perform this function is a Certificate Authority ("CA"). The process overhead of running a CA is high, however, and for a system with a great number of VNFCs it may not be considered a requirement that each of them use a CA as their individual root of trust. Specifically, for VNFCs which do not need to communicate with external entities, but only with other VNFCs within their own VNF, or only with their own VNF Manager, the use of an external CA may not require, as the trust relationships that they are required to set up and participate in are clearly constrained. It is important, however, to ensure that if any such entities *do* need to communicate with other entities, that circular trust relationships are not established through accident or ignorance.

NOTE: http://en.wikipedia.org/wiki/Turtles_all_the_way_down.

The key information that is required to set up an initial root of trust is a secured method to contact the trusted party. Additionally, if there is no trusted communications channel with that party, a cryptographic certificate is required as well. As noted in clause 4.4.6.3.1, as management and orchestration can trust the identity of the VNFC and can therefore act as a trust broker to the initial root of trust, there is no need for authentication between the two parties. However, without initial certificates, establishment of an encrypted communications channel between the two parties is best left to the infrastructure: once this has been established, a cryptographic certificate identifying the initial root of trust can be provided to the VNFC.

A variety of mechanisms could be deployed to allow a VNFC to be provisioned with a way to contact the trusted party (typically a MAC or IP address, and possibly a port). These may also include the ability to provide a certificate identifying the trusted party (though other techniques may be applicable - see above):

- Multicast.
- Hypervisor injection.
- Initial image.

5.4.3.1.1 Multicast

Multicast allows for an entity to send out a request for information across a particular network subnet, and then await a reply. The most obvious implementation of this is DHCP, but the security of DHCP as a method for providing trusted information is low in the standard implementation, as it is very easy to spoof. Although it might be possible to implement a secured DHCP server, this would rely on the pre-existence of a certificate within the VNFC to allow it to check the DHCP server's identity, and is therefore not appropriate for this case (it is a "turtle's problem"). In order to allow multicast to work, it should be possible for MANO to ensure that there is only one responding server which could possibly respond to the VNFC's request: the trusted one. This could be attempted operationally - by attempting to ensure that rogue responding servers cannot be created - or by controlling the breadth of the multicast by manipulating the network(s) to which the VNFC initially has access. Wireless discovery protocols may likewise create secured services for multicast trust instantiation.

5.4.3.1.2 Injection by hypervisor

All the techniques identified in this clause are also applicable to this case.

5.4.3.1.3 Initial image

Although in the case of identity provision, the information in each VNFC should be unique, so placing keys in an image is inappropriate, in the case of an initial root of trust, it may be that the same information - e.g. network location and a cryptographic certificate - may be shared across a complete NFV deployment, or at least all the VNFCs of a particular type for that deployment. In this case, placing the information in the VNFC image may be entirely appropriate.

Note, however, that if this choice is taken, the assurance of the integrity of the image may be reduced, as the image is likely to require changing from the stock image provided by a VNFC vendor and changes to be made by the operator. Although not necessarily a problem, this introduces another link in the chain of trust for secure boot.

5.4.3.1.4 Hypervisor

The Trustworthy Boot of the various parts of the hypervisor is the key part of the Trustworthy Boot process, and is made up of two parts: the hardware and the hypervisor and associated platform. There are a number of techniques that can be leveraged, the most applicable being measured boot with remote attestation.

Secured configuration choices should be made to all underlying hardware, firmware and configurations that comprise the hypervisor platform. The selection of tamper-resistant hardware components may also require visibility by the hypervisor or OS for alerting and reporting. Ownership of a physical TPM can only be by one entity and this choice needs to be made before the hypervisor is configured.

The hypervisor should be installed, patched and hardened with the recommended security configuration.

5.4.3.1.5 VNFC OS and application

In the non-virtualised case, the most troublesome parts of a Trustworthy Boot would typically be the configuration of applications and the OS, but in the case of a VNF, this secured configuration is simpler. Since the creation of a VNFC instance results from booting up an image (as specified in the VNFD), there will be no dynamic state added to a VNFC image above and beyond the image state information referenced in the VNFD. As long as the image is trusted, therefore, the booting up of the VNFC itself can be relegated to the hypervisor, assuming a sufficient hypervisor trust model. Whether or not a VNFC boots up via UEFI or a similar process is therefore irrelevant in this case.

This pushes the problem to another entity or set of entities by requiring a store of images whose integrity is secured (or can be attested to (e.g. with one-way hashing functions)). Integrity management has the advantage of centralizing a function and allowing for hierarchical schemes for access and changes to the set of available images. A key decision is what level of assurance is required before an image is trusted and accepted into this image store. Levels of assurance might include:

- Accepted if from an approved vendor.
- Accepted if digitally signed by an approved vendor.
- Accepted if digitally signed by a certificating body.

- Accepted if digitally signed by a certificating body and based on a hardened OS template.

There are two measures introduced in the list above which do not currently exist: a certificating body and the availability of one or more hardened OS templates. Currently no bodies exist to service either of these needs. This leads to their being identified as GAPS.

This method of ensuring the known level of assurance for VNFC OS and application boot is only applicable when a clean VNFC image is used on each occasion that a new VNFC instance is created. One of the capabilities offered by hypervisors is the ability to take "snapshots" of the state of an existing VM: clearly, when a VNFC instance is created from a snapshot, rather from a certified image, the level of assurance that the VNFM can have is constrained by the knowledge it (the VNFM) has - if any - about the provenance and history of the snapshot. It will generally be difficult for a VNFM to be able to gain full knowledge of such a snapshot and it is therefore recommended that snapshots are not used for VNFs where high levels of assurance are required. Any snapshots will also include private keys which need to be protected. Cloning of images creates similar issues, with the added complication of possible duplication of private keys - see clause 4.4.3.3.1.

The situation with regards to other hypervisor-managed VM (and therefore VNFC) lifecycle events is similar but subtly different. Snapshots act as points to which VMs can be taken back, or to act as templates for new VMs. VMs which have been suspended and resumed, however, are the *same* instance of an existing VM, and therefore VNFC, and should maintain and existing UUID. As long as the VNFM lifecycle spans the suspend and resumption of a VNFC-bearing VM, then, it should be possible to trust it.

There is a further complication, however, as it is possible that a VM may be migrated to a different hypervisor host which is less trusted, and then resumed. The same, in fact, goes for VMs which are migrated whilst alive. The key point here, then, is the extent to which the VNFC trusts the live management and orchestration infrastructure entities with which it is in contact. There is a danger that this trust relationship, if not continuously maintained, could become the weak link in the chain of trust. It might not seem important for the VNFM to maintain a high level of trust in its coordinating MANO entities except for specific points of operation, but this danger - of a management and orchestration entity migrating a VNFC out from under the VNFM's nose to a hypervisor host with a different trust assurance level - shows that it is a key trust relationship.

5.4.3.1.6 Deployment state

In order for a VNFC to be able to function, it will require state in order to perform its function. This state will usually be required from an external source, or derived from its environment. As such, it is not strictly part of the Trustworthy Boot process.

State management encompasses the lifecycle for controlling system-state information that is specific to instantiation, running processes, suspended processes and for secured retirement.

Annex A (informative): Authors & contributors

Document Change Request Log

V0.0.x - Initial development of NFV Security and Trust Guidance document

V0.1.0 - Initial Stable Draft, incorporating editorial updates and review by SWA and MANO groups

V1.1.2 – Updates to clause 4.4.5.1, renamed "Trustworthy Boot"

Recognition of Contributors

The following individuals and organizations are recognized for their contributions to NFV Security and Trust Guidance:

- Mike Bursell, Intel [Rapporteur]
- Ashutosh Dutta, AT&T
- Diego Lopez, Telefonica
- Hui-Lan (Huilan) Lu, Alcatel-Lucent
- Marie-Paule Odini, HP
- Kurt Roemer, Citrix [Rapporteur]
- Kapil Sood, Intel
- Marcus Wong, Huawei
- Peter Wörndle, Ericsson
- Mihai Serb, Huawei

Annex B (informative): Bibliography

- NIST SP800-125 (Jan 2010): "Guide to Security for Full Virtualization Technologies".
- ETSI GS NFV-SEC 001: "Network Functions Virtualisation (NFV); NFV Security; Problem Statement".
- Popek and Goldberg 1974 paper: "Formal Requirements for Virtualizable Third Generation Architectures".

History

Document history		
V1.1.1	December 2014	Publication as ETSI GS NFV-SEC 003
V1.2.1	August 2016	Publication