Requirements we need to change

Wednesday, April 04, 2018

10:11 AM

# Bulleted Requirements

* R-69565 The xNF Package MUST include documentation describing xNF Management APIs. The document must include information and tools for:
  + ONAP to deploy and configure (initially and ongoing) the xNF application(s) (e.g., NETCONF APIs). Includes description of configurable parameters for the xNF and whether the parameters can be configured after xNF instantiation.
  + ONAP to monitor the health of the xNF (conditions that require healing and/or scaling responses). Includes a description of:
    - Parameters that can be monitored for the xNF and event records (status, fault, flow, session, call, control plane, etc.) generated by the xNF after instantiation.
    - Runtime lifecycle events and related actions (e.g., control responses, tests) which can be performed for the xNF.
* R-89571 The xNF **MUST** support and provide artifacts for configuration management using at least one of the following technologies:
  + Netconf/YANG
  + Chef
  + Ansible

Note: The requirements for Netconf/YANG, Chef, and Ansible protocols are provided separately and must be supported only if the corresponding protocol option is provided by the xNF providor.

* R-01556 The xNF Package **MUST** include documentation describing the fault, performance, capacity events/alarms and other event records that are made available by the xNF. The document must include:
  + A unique identification string for the specific xNF, a description of the problem that caused the error, and steps or procedures to perform Root Cause Analysis and resolve the issue.
  + All events, severity level (e.g., informational, warning, error) and descriptions including causes/fixes if applicable for the event.
  + All events (fault, measurement for xNF Scaling, Syslogs, State Change and Mobile Flow), that need to be collected at each VM, VNFC (defined in [VNF Guidelines](http://onap.readthedocs.io/en/latest/submodules/vnfrqts/guidelines.git/docs/vnf_guidelines/vnf_guidelines.html#a-glossary) ) and for the overall xNF.
* R-01478 The xNF Package **MUST** include documentation describing all parameters that are available to monitor the xNF after instantiation (includes all counters, OIDs, PM data, KPIs, etc.) that must be collected for reporting purposes. The documentation must include a list of:
  + Monitoring parameters/counters exposed for virtual resource management and xNF application management.
  + KPIs and metrics that need to be collected at each VM for capacity planning and performance management purposes.
  + The monitoring parameters must include latencies, success rates, retry rates, load and quality (e.g., DPM) for the key transactions/functions supported by the xNF and those that must be exercised by the xNF in order to perform its function.
  + For each KPI, provide lower and upper limits.
  + When relevant, provide a threshold crossing alert point for each KPI and describe the significance of the threshold crossing.
  + For each KPI, identify the suggested actions that need to be performed when a threshold crossing alert event is recorded.
  + Describe any requirements for the monitoring component of tools for Network Cloud automation and management to provide these records to components of the xNF.
  + When applicable, provide calculators needed to convert raw data into appropriate reporting artifacts.
* R-97102 The VNF Package **MUST** include VM requirements via a Heat template that provides the necessary data for:
  + VM specifications for all VNF components - for hypervisor, CPU, memory, storage.
  + Network connections, interface connections, internal and external to VNF.
  + High availability redundancy model.
  + Scaling/growth VM specifications.

Note: Must comply with the *Heat requirements in 5.b*.

* R-26508 The xNF **MUST** support NETCONF server that can be mounted on OpenDaylight (client) and perform the following operations:
  + Modify, update, change, rollback configurations using each configuration data element.
  + Query each state (non-configuration) data element.
  + Execute each YANG RPC.
  + Receive data through each notification statement.
* R-67114 The xNF **MUST** be installed with:
  + Chef-Client >= 12.0
  + Chef push jobs client >= 2.0
* R-19624 The xNF **MUST** encode and serialize content delivered to ONAP using JSON (RFC 7159) plain text format. High-volume data is to be encoded and serialized using [Avro](http://avro.apache.org/), where the Avro [[5]](http://onap.readthedocs.io/en/latest/submodules/vnfrqts/requirements.git/docs/Chapter7.html#id11) data format are described using JSON.
  + JSON plain text format is preferred for moderate volume data sets (option 1), as JSON has the advantage of having well-understood simple processing and being human-readable without additional decoding. Examples of moderate volume data sets include the fault alarms and performance alerts, heartbeat messages, measurements used for xNF scaling and syslogs.
  + Binary format using Avro is preferred for high volume data sets (option 2) such as mobility flow measurements and other high-volume streaming events (such as mobility signaling events or SIP signaling) or bulk data, as this will significantly reduce the volume of data to be transmitted. As of the date of this document, all events are reported using plain text JSON and REST.
  + Avro content is self-documented, using a JSON schema. The JSON schema is delivered along with the data content (<http://avro.apache.org/docs/current/> ). This means the presence and position of data fields can be recognized automatically, as well as the data format, definition and other attributes. Avro content can be serialized as JSON tagged text or as binary. In binary format, the JSON schema is included as a separate data block, so the content is not tagged, further compressing the volume. For streaming data, Avro will read the schema when the stream is established and apply the schema to the received content.
* R-98191 The xNF **MUST** vary the frequency that asynchronous data is delivered based on the content and how data may be aggregated or grouped together. For example, alarms and alerts are expected to be delivered as soon as they appear. In contrast, other content, such as performance measurements, KPIs or reported network signaling may have various ways of packaging and delivering content. Some content should be streamed immediately; or content may be monitored over a time interval, then packaged as collection of records and delivered as block; or data may be collected until a package of a certain size has been collected; or content may be summarized statistically over a time interval, or computed as a KPI, with the summary or KPI being delivered.
  + We expect the reporting frequency to be configurable depending on the virtual network function’s needs for management. For example, Service Provider may choose to vary the frequency of collection between normal and trouble-shooting scenarios.
  + Decisions about the frequency of data reporting will affect the size of delivered data sets, recommended delivery method, and how the data will be interpreted by ONAP. These considerations should not affect deserialization and decoding of the data, which will be guided by the accompanying JSON schema or GPB definition files.
* R-08312 The xNF **MAY** use other options which are expected to include
  + REST delivery of binary encoded data sets.
  + TCP for high volume streaming asynchronous data sets and for other high volume data sets. TCP delivery can be used for either JSON or binary encoded data sets.
  + SFTP for asynchronous bulk files, such as bulk files that contain large volumes of data collected over a long time interval or data collected across many xNFs. This is not preferred. Preferred is to reorganize the data into more frequent or more focused data sets, and deliver these by REST or TCP as appropriate.
  + REST for synchronous data, using RESTCONF (e.g., for xNF state polling).

# Noted Requirements

* R-85633 The VNF MUST implement Data Storage Encryption (database/disk encryption) for Sensitive Personal Information (SPI) and other subscriber identifiable data.
  + Note: subscriber’s SPI/data must be encrypted at rest, and other subscriber identifiable data should be encrypted at rest. Other data protection requirements exist and should be well understood by the developer.
* R-14853 The VNF MUST respond to a “move traffic” [2] command against a specific VNFC, moving all existing session elsewhere with minimal disruption if a VNF provides a load balancing function across multiple instances of its VNFCs.
  + Note: Individual VNF performance aspects (e.g., move duration or disruption scope) may require further constraints.
* R-47849 The xNF provider MUST support the metadata about licenses (and their applicable entitlements) as defined in this document for xNF software, and any license keys required to authorize use of the xNF software. This metadata will be used to facilitate onboarding the xNF into the ONAP environment and automating processes for putting the licenses into use and managing the full lifecycle of the licenses. The details of this license model are described in Appendix C.
  + Note: License metadata support in ONAP is not currently available and planned for 1Q 2018.
* R-82018 The VNF SHOULD load the SSH key onto VNF VM(s) as part of instantiation. This will allow the Ansible Server to authenticate to perform post-instantiation configuration without manual intervention and without requiring specific VNF login IDs and passwords.
  + CAUTION: For VNFs configured using Ansible, to eliminate the need for manual steps, post-instantiation and pre-configuration, to upload of SSH keys, SSH keys loaded during (heat) instantiation shall be preserved and not removed by (heat) embedded scripts.
* R-51442 The xNF SHOULD use playbooks that are designed to automatically ‘rollback’ to the original state in case of any errors for actions that change state of the xNF (e.g., configure).
  + NOTE: In case rollback at the playbook level is not supported or possible, the xNF provider shall provide alternative locking mechanism (e.g., for a small xNF the rollback mechanism may rely on workflow to terminate and re-instantiate VNF VMs and then re-run playbook(s)). Backing up updated files also recommended to support rollback when soft rollback is feasible.
* R-58301 The VNF **SHOULD NOT** use playbooks that make requests to Cloud resources e.g. Openstack (nova, neutron, glance, heat, etc.); therefore, there is no use for Cloud specific variables like Openstack UUIDs in Ansible Playbooks.
  + Rationale: Flows that require interactions with Cloud services e.g. Openstack shall rely on workflows run by an Orchestrator or other capability (such as a control loop or Operations GUI) outside Ansible Server which can be executed by a Controller such as APPC. There are policies, as part of Control Loop models, that send remediation action requests to APPC; these are triggered as a response to an event or correlated events published to Event Bus.

# Paragraph Requirements

* R-88199 The VNF MUST utilize a persistent datastore service that can meet the data performance/latency requirements. (For example: Datastore service could be a VNFC in VNF or a DBaaS in the Cloud execution environment)
* R-84473 The VNF MUST enable DPDK in the guest OS for VNF’s requiring high packets/sec performance. High packet throughput is defined as greater than 500K packets/sec.
* R-41159 The VNF MUST deliver any and all functionality from any VNFC in the pool (where pooling is the most suitable solution). The VNFC pool member should be transparent to the client. Upstream and downstream clients should only recognize the function being performed, not the member performing it.
* R-52870 The VNF MUST provide a method of metrics gathering and analysis to evaluate the resiliency of the software from both a granular as well as a holistic standpoint. This includes, but is not limited to thread utilization, errors, timeouts, and retries.
* R-23740 The VNF MUST accommodate the security principle of “least privilege” during development, implementation and operation. The importance of “least privilege” cannot be overstated and must be observed in all aspects of VNF development and not limited to security. This is applicable to all sections of this document.
* R-23882 The VNF SHOULD be scanned using both network scanning and application scanning security tools on all code, including underlying OS and related configuration. Scan reports shall be provided. Remediation roadmaps shall be made available for any findings
* R-55830 The VNF MUST distribute all production code from NCSP internal sources only. No production code, libraries, OS images, etc. shall be distributed from publically accessible depots.
* R-99771 The VNF MUST provide all code/configuration files in a “Locked down” or hardened state or with documented recommendations for such hardening. All unnecessary services will be disabled. VNF provider default credentials, community strings and other such artifacts will be removed or disclosed so that they can be modified or removed during provisioning.
* R-49956 The VNF MUST pass all access to applications (Bearer, signaling and OA&M) through various security tools and platforms from ACLs, stateful firewalls and application layer gateways depending on manner of deployment. The application is expected to function (and in some cases, interwork) with these security tools.
* R-69649 The VNF MUST have all vulnerabilities patched as soon as possible. Patching shall be controlled via change control process with vulnerabilities disclosed along with mitigation recommendations.
* R-68589 The VNF MUST, if not using the NCSP’s IDAM API, support User-IDs and passwords to uniquely identify the user/application. VNF needs to have appropriate connectors to the Identity, Authentication and Authorization systems that enables access at OS, Database and Application levels as appropriate.
* R-79107 The VNF MUST, if not using the NCSP’s IDAM API, enforce a configurable maximum number of Login attempts policy for the users. VNF provider must comply with “terminate idle sessions” policy. Interactive sessions must be terminated, or a secure, locking screensaver must be activated requiring authentication, after a configurable period of inactivity. The system-based inactivity timeout for the enterprise identity and access management system must also be configurable.
* R-46908 The VNF MUST, if not using the NCSP’s IDAM API, comply with “password complexity” policy. When passwords are used, they shall be complex and shall at least meet the following password construction requirements: (1) be a minimum configurable number of characters in length, (2) include 3 of the 4 following types of characters: upper-case alphabetic, lower-case alphabetic, numeric, and special, (3) not be the same as the UserID with which they are associated or other common strings as specified by the environment, (4) not contain repeating or sequential characters or numbers, (5) not to use special characters that may have command functions, and (6) new passwords must not contain sequences of three or more characters from the previous password.
* R-39342 The VNF MUST, if not using the NCSP’s IDAM API, comply with “password changes (includes default passwords)” policy. Products will support password aging, syntax and other credential management practices on a configurable basis.
* R-41994 The VNF MUST, if not using the NCSP’s IDAM API, comply with “No Self-Signed Certificates” policy. Self-signed certificates must be used for encryption only, using specified and approved encryption protocols such as TLS 1.2 or higher or equivalent security protocols such as IPSec, AES.
* R-73541 The VNF MUST use access controls for VNFs and their supporting computing systems at all times to restrict access to authorized personnel only, e.g., least privilege. These controls could include the use of system configuration or access control software.
* R-15671 The VNF MUST NOT provide public or unrestricted access to any data without the permission of the data owner. All data classification and access controls must be followed.
* R-21652 The VNF MUST implement the following input validation control: Check the size (length) of all input. Do not permit an amount of input so great that it would cause the VNF to fail. Where the input may be a file, the VNF API must enforce a size limit.
* R-54930 The VNF MUST implement the following input validation control: Do not permit input that contains content or characters inappropriate to the input expected by the design. Inappropriate input, such as SQL insertions, may cause the system to execute undesirable and unauthorized transactions against the database or allow other inappropriate access to the internal network.
* R-84160 The VNF MUST have security logging for VNFs and their OSs be active from initialization. Audit logging includes automatic routines to maintain activity records and cleanup programs to ensure the integrity of the audit/logging systems.
* R-13151 The VNF SHOULD disable the paging of the data requiring encryption, if possible, where the encryption of non-transient data is required on a device for which the operating system performs paging to virtual memory. If not possible to disable the paging of the data requiring encryption, the virtual memory should be encrypted.
* R-02170 The VNF MUST use, whenever possible, standard implementations of security applications, protocols, and format, e.g., S/MIME, TLS, SSH, IPSec, X.509 digital certificates for cryptographic implementations. These implementations must be purchased from reputable vendors and must not be developed in-house.
* R-86758 The VNF SHOULD provide an automated test suite to validate every new version of the software on the target environment(s). The tests should be of sufficient granularity to independently test various representative VNF use cases throughout its lifecycle. Operations might choose to invoke these tests either on a scheduled basis or on demand to support various operations functions including test, turn-up and troubleshooting.
* R-06327 The VNF MUST respond to a “drain VNFC” [2] command against a specific VNFC, preventing new session from reaching the targeted VNFC, with no disruption to active sessions on the impacted VNFC, if a VNF provides a load balancing function across multiple instances of its VNFCs. This is used to support scenarios such as proactive maintenance with no user impact.
* R-66070 The xNF Package MUST include xNF Identification Data to uniquely identify the resource for a given xNF provider. The identification data must include: an identifier for the xNF, the name of the xNF as was given by the xNF provider, xNF description, xNF provider, and version.
* R-84366 The xNF Package MUST include documentation describing xNF Functional APIs that are utilized to build network and application services. This document describes the externally exposed functional inputs and outputs for the xNF, including interface format and protocols supported.
* R-30278 The xNF provider MUST provide a Resource/Device YANG model as a foundation for creating the YANG model for configuration. This will include xNF attributes/parameters and valid values/attributes configurable by policy.
* R-18525 The xNF provider MUST provide a JSON file for each supported action for the xNF. The JSON file must contain key value pairs with all relevant values populated with sample data that illustrates its usage. The fields and their description are defined in Appendix A.
* R-16777 The xNF provider MUST provide a JSON file for each supported action for the xNF. The JSON file must contain key value pairs with all relevant values populated with sample data that illustrates its usage. The fields and their description are defined in Appendix B.
* R-22888 The xNF provider MUST provide documentation for the xNF Policy Description to manage the xNF runtime lifecycle. The document must include a description of how the policies (conditions and actions) are implemented in the xNF.
* R-74763 The xNF provider MUST provide an artifact per xNF that contains all of the xNF Event Records supported. The artifact should include reference to the specific release of the xNF Event Stream Common Event Data Model document it is based on. (e.g., VES Event Listener)
* R-58775 The xNF provider MUST provide software components that can be packaged with/near the xNF, if needed, to simulate any functions or systems that connect to the xNF system under test. This component is necessary only if the existing testing environment does not have the necessary simulators.
* R-44125 The xNF provider MUST agree to the process that can be met by Service Provider reporting infrastructure. The Contract shall define the reporting process and the available reporting tools.
* R-85991 The xNF provider MUST provide a universal license key per xNF to be used as needed by services (i.e., not tied to a VM instance) as the recommended solution. The xNF provider may provide pools of Unique xNF License Keys, where there is a unique key for each xNF instance as an alternate solution. Licensing issues should be resolved without interrupting in-service xNFs.
* R-60106 The xNF MUST implement the protocol operation: get(filter) - Retrieve (a filtered subset of) the running configuration and device state information. This should include the list of xNF supported schemas.
* R-62468 The xNF MUST allow all configuration data to be edited through a NETCONF <edit-config> operation. Proprietary NETCONF RPCs that make configuration changes are not sufficient.
* R-28756 The xNF MUST support :partial-lock and :partial-unlock capabilities, defined in RFC 5717. This allows multiple independent clients to each write to a different part of the <running> configuration at the same time.
* R-83873 The xNF MUST support :rollback-on-error value for the <error-option> parameter to the <edit-config> operation. If any error occurs during the requested edit operation, then the target database (usually the running configuration) will be left unaffected. This provides an ‘all-or-nothing’ edit mode for a single <edit-config> request
* R-68990 The xNF MUST support the :startup capability. It will allow the running configuration to be copied to this special database. It can also be locked and unlocked.
* R-68200 The xNF MUST support the :url value to specify protocol operation source and target parameters. The capability URI for this feature will indicate which schemes (e.g., file, https, sftp) that the server supports within a particular URL value. The ‘file’ scheme allows for editable local configuration databases. The other schemes allow for remote storage of configuration databases.
* R-20353 The xNF MUST implement at least one of the capabilities :candidate or :writable-running. If both :candidate and :writable-running are provided then two locks should be supported.
* R-11499 The xNF MUST fully support the XPath 1.0 specification for filtered retrieval of configuration and other database contents. The ‘type’ attribute within the <filter> parameter for <get> and <get-config> operations may be set to ‘xpath’. The ‘select’ attribute (which contains the XPath expression) will also be supported by the server. A server may support partial XPath retrieval filtering, but it cannot advertise the :xpath capability unless the entire XPath 1.0 specification is supported.
* R-26115 The xNF MUST follow the data model upgrade rules defined in [RFC6020] section 10. All deviations from section 10 rules shall be handled by a built-in automatic upgrade mechanism.
* R-31809 The xNF MUST support the HealthCheck RPC. The HealthCheck RPC executes a xNF Provider-defined xNF Healthcheck over the scope of the entire xNF (e.g., if there are multiple VNFCs, then run a health check, as appropriate, for all VNFCs). It returns a 200 OK if the test completes. A JSON object is returned indicating state (healthy, unhealthy), scope identifier, time-stamp and one or more blocks containing info and fault information. If the xNF is unable to run the HealthCheck, return a standard http error code and message.
* R-72184 The xNF MUST have routable FQDNs for all the endpoints (VMs) of a xNF that contain chef-clients which are used to register with the Chef Server. As part of invoking xNF actions, ONAP will trigger push jobs against FQDNs of endpoints for a xNF, if required.
* R-15885 The xNF MUST Upon completion of the chef-client run, POST back on the callback URL, a JSON object as described in Table A2 if the chef-client run list includes a cookbook/recipe that is callback capable. Failure to POST on the Callback Url should not be considered a critical error. That is, if the chef-client successfully completes the xNF action, it should reflect this status on the Chef Server regardless of whether the Callback succeeded or not.
* R-32217 The xNF MUST have routable FQDNs that are reachable via the Ansible Server for the endpoints (VMs) of a xNF on which playbooks will be executed. ONAP will initiate requests to the Ansible Server for invocation of playbooks against these end points [3].
* R-92866 The VNF MUST include as part of post-instantiation configuration done by Ansible Playbooks the removal/update of SSH keys loaded through instantiation to support Ansible. This may include download and install of new SSH keys.
* R-49396 The xNF MUST support each xNF action by invocation of one playbook [4]. The playbook will be responsible for executing all necessary tasks (as well as calling other playbooks) to complete the request.
* R-48698 The xNF MUST utilize information from key value pairs that will be provided by the Ansible Server as extra-vars during invocation to execute the desired xNF action. If the playbook requires files, they must also be supplied using the methodology detailed in the Ansible Server API.
* R-50252 The xNF MUST write to a specific set of text files that will be retrieved and made available by the Ansible Server if, as part of a xNF action (e.g., audit), a playbook is required to return any xNF information. The text files must be written in the same directory as the one from which the playbook is being executed. A text file must be created for each host the playbook run targets/affects, with the name ‘<hostname>\_results.txt’ into which any desired output from each respective VM/xNF must be written.
* R-43353 The VNF MUST return control from Ansible Playbooks only after tasks are fully complete, signaling that the playbook completed all tasks. When starting services, return control only after all services are up. This is critical for workflows where the next steps are dependent on prior tasks being fully completed.
* R-84879 The xNF MUST have the capability of maintaining a primary and backup DNS name (URL) for connecting to ONAP collectors, with the ability to switch between addresses based on conditions defined by policy such as time-outs, and buffering to store messages until they can be delivered. At its discretion, the service provider may choose to populate only one collector address for a xNF. In this case, the network will promptly resolve connectivity problems caused by a collector or network failure transparently to the xNF.
* R-81777 The VNF MUST be configured with initial address(es) to use at deployment time. Subsequently, address(es) may be changed through ONAP-defined policies delivered from ONAP to the VNF using PUTs to a RESTful API, in the same manner that other controls over data reporting will be controlled by policy.
* R-03070 The xNF MUST, by ONAP Policy, provide the ONAP addresses as data destinations for each xNF, and may be changed by Policy while the xNF is in operation. We expect the xNF to be capable of redirecting traffic to changed destinations with no loss of data, for example from one REST URL to another, or from one TCP host and port to another.
* R-11240 The xNF MUST respond with content encoded in JSON, as described in the RESTCONF specification. This way the encoding of a synchronous communication will be consistent with Avro.
* R-70266 The xNF MUST respond to an ONAP request to deliver the current data for any of the record types defined in Event Records - Data Structure Description by returning the requested record, populated with the current field values. (Currently the defined record types include fault fields, mobile flow fields, measurements for xNF scaling fields, and syslog fields. Other record types will be added in the future as they become standardized and are made available.)
* R-43327 The xNF SHOULD use Modeling JSON text with YANG, If YANG models need to be translated to and from JSON{RFC7951]. YANG configuration and content can be represented via JSON, consistent with Avro, as described in “Encoding and Serialization” section.