ONAP Security Best Practices.

1. Introduction

The ONAP Security Best Practices is a list of Best Practices recommended by the ONAP sub-committee. These best practices have the following states:

- Draft: It is still under discussion in the ONAP security sub-committee
- Recommended: It is recommended by the security sub-committee but not yet approved by the TSC
- Approved: It is approved by the TSC.

2. CII Badging Program.

Status: Approved

Best Practice:

It is recommended that the ONAP projects are certified as part of the CII badging program. A gold badge is recommended, however the basic passing badge is the starting point.

This is currently being introduced slowly with 2 projects undergoing certification.

•Basic introduction can be found here: https://github.com/coreinfrastructure/best-practices-badge/blob/master/doc/criteria.md
•Silver/Gold criteria can be found here: https://github.com/coreinfrastructure/best-practices-badge/blob/master/doc/other.md

The CLAMP project wrote about their early experiences applying the CII badging program procedures, captured here: https://wiki.onap.org/display/DW/ONAP+security+Recomendation+Development?src=contextnavpagetreemode

Project	Progress	Project	Progress
CLAMP	blocked URL	DCAE	blocked URL
Policy	blocked URL	CCSDK	blocked URL
AAF	blocked URL	SDNC	blocked URL

3. Credential Protection and Management

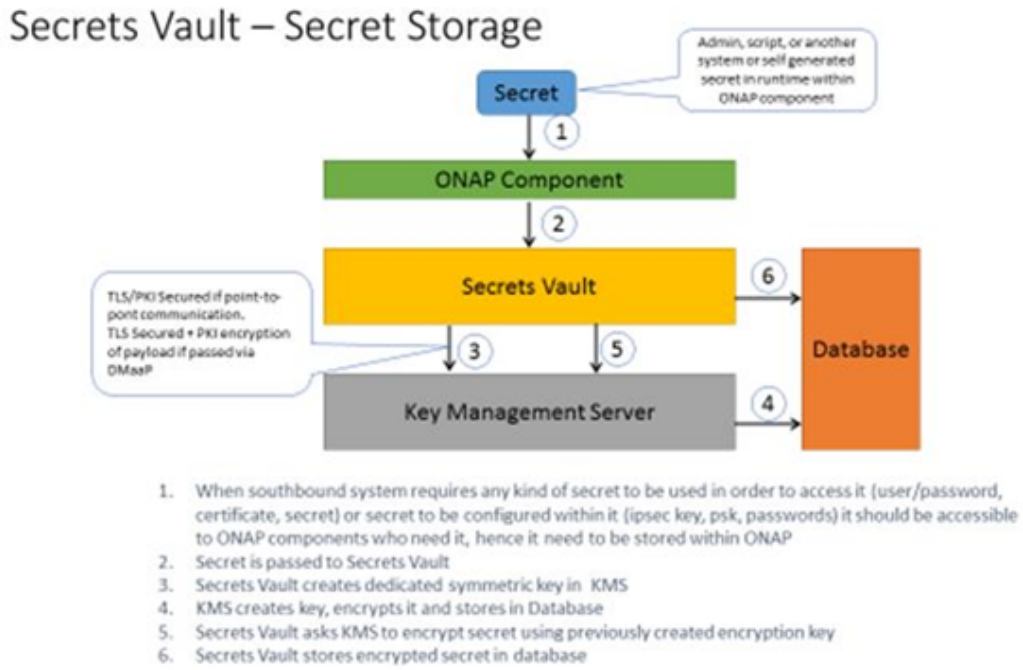
Status: Draft

Best Practice:

ONAP requires two components to improve the security of credentials used in orchestration.

1. a secrets vault to store credentials used by ONAP
2. a process to instantiate credentials

Component 1: Secrets Vault - A service that can be integrated with ONAP that provides secure storage of the credentials used by ONAP to authenticate to VNFs.



1. When southbound system requires any kind of secret to be used in order to access it (user/password, certificate, secret) or secret to be configured within it (ipsec key, psk, passwords) it should be accessible to ONAP components who need it, hence it need to be stored within ONAP
2. Secret is passed to Secrets Vault
3. Secrets Vault creates dedicated symmetric key in KMS
4. KMS creates key, encrypts it and stores in Database
5. Secrets Vault asks KMS to encrypt secret using previously created encryption key
6. Secrets Vault stores encrypted secret in database

Components

1) Vault for secret storage: storage of passwords, secrets, certificates or anything else that might be used in order to access or authenticate to some system

2) Key Management Server: used to generate/manage crypro keys that are used by other parts of the system (for example Vault) and perform crypto operations (in case we don't

want keys to leave the server)

High high level architecture approach for both components

1) External interface – for consumption by the system

2) Internal implementation interface/plugin system – to enable integration with pre-existing solutions

3) "Naïve" Native implementation – does all the stuff that required in order for system to be fully operational and secure out of box without any external systems To be used during testing/demoes or by people without hardware solutions

at place.

Use cases

Use case: provisioning SD-[W,L]AN service over already instantiated Juniper VNFs (or PNFs)

using IPsec tunnel

1) ONAP got request to service provisioning. Information includes IDs of VNFs that

service should be provisioned over

2) MSO/SA forwards request to SDN-C

3) SDN-C sends request to Secrets Vault to retrieve credentials to abovementioned VNFs by their ID

4) Secrets Vault sends back to SDN-C credentials

5) SDN-C generates pre-shared key for IPsec tunnel provisioning

6) SDN-C sends to Secrets Vault request to store pre-shared key

7) SDN-C using retrieved credentials connects via SSH to VNFs and configures

IPsec tunnel using PSK

Use case extension: at later timeframe SD-[W,L]an service is extended to new branch where VNF needs

to be instantiated

1) ONAP got request to service provisioning.

2) MSO instantiates new VNF and passes it to APP-C for initial configuration

3) APP-C generates random user name and password (aka credentials)

4) APP-C sends request to Secrets Vault to store credentials by VNF ID

5) APP-C configures new VNF to use new credentials

6) APP-C notifies MSO that it finished

7) MSO/SA forwards request to SDN-C

8) SDN-C sends request to Secrets Vault to retrieve credentials to abovementioned VNF by his ID

9) Secrets Vault sends back to SDN-C credentials

10) SDN-C sends request to Secrets Vault to retrieve pre shared IPsec key for IPsec tunnel

11) Secrets Vault sends back IPsec key

12) SDN-C using retrieved credentials connects via SSH to VNFs and configures IPsec tunnel using retrieved PSK

13) Everybody happy and everything is secured

- OpenStack's Barbican: specific to OpenStack, not a mature service
- Various commercial services such as LastPass

Recommendation: ONAP should provide a reference implementation of a secrets vault service as an ONAP project.

Next Steps:

- Find a project lead for a reference implementation.

Component 2: A process to provision ONAP instances with credentials. These credentials may be used for interprocess communication (e.g., APPC calling A&AI) or for ONAP configuring VNFs.

Automatic provisioning of certificates and credentials to ONAP components: AAF can provision certificates. ECOMP DCAE is currently using AAF to provision certificates.

Next steps:

- Work with the AAF team to include this functionality in Release 2. It is important to understand that the AAF solution depends on the CA supporting the SCEP protocol.
- Enhance AAF to provision userIDs & passwords to ONAP instances and VNFs. Most VNFs only support userID/password authentication today. ETSI NFV SEC may issue a spec in the future on a more comprehensive approach to using PKI for NFV which can be visited by ONAP SEC when released. Steve is working on this right now but doesn't know when he'll be done.

4. Static Code Scans

Status: Draft

Best Practice:

Recommendation to the TSC

- Use Coverity Scan https://scan.coverity.com/ to perform static code scans on all ONAP code.
- Automate scanning by enabling Jenkins to trigger weekly scans with Coverity Scan.
- Deliver scan reports to the PTLs for each project PTLs will be responsible for getting the vulnerabilities resolved (fixed or designated as false positive).
- All projects in a release must have the high vulnerabilities resolved by MS-3.
- All projects in a release must have the high and medium vulnerabilities resolved by MS-4.
- The Security Committee will host session to help projects walk through the scanning process and reports.

Next Steps

- Review the OPNFV scanning process at https://wiki.opnfv.org/display/security/Security+Scanning to see if it can be adopted as the ONAP static code scanning process.

Tools that have been assessed: Coverity Scan (LF using the tool in OPNFV and other projects), HP Fortify (AT&T evaluation), Checkmarx (AT&T evaluation), Bandit (AT&T evaluation)

Preliminary Decision: Coverity Scan https://scan.coverity.com/

Description: Coverity Scan is a service by which Synopsys provides the results of analysis on open source coding projects to open source code developers that have registered their products with Coverity Scan. Coverity Scan is powered by Coverity® Quality Advisor. Coverity Quality Advisor surfaces defects identified by the Coverity Static Analysis Verification Engine (Coverity SAVE®). Synopsys offers the results of the analysis completed by Coverity Quality Advisor on registered projects at no charge to registered open source developers.

Open Source use: 4000+ open source projects use Coverity Scan

Languages supported: C/C++, C#, Java, Javascript, Python, Ruby (Question: what about Groovy, Erlang?)

Current Activity: In conversations with Coverity to understand the definition of “project” – does it refer to ONAP or the projects under an ONAP release to ensure that the limitation on free scans does not lead to bottlenecks in submissions and commits. (Coverity response included below)

Coverity Scanning Process:

Coverity static analysis works by instrumenting through build capture. The components which make up the ONAP project can be managed a number of ways:

- If the ONAP project can be built from source in a single command, then Coverity can to create component maps.
- If the separate components are built individually, then each component can be submitted as a separate project.
- Coverity recommends storing the projects in a hierarchical structure in Github with the ONAP parent project referring to the project (i.e. ONAP /component_name). There are a few projects already in Scan which follow this structure. (is ONAP stored this way?) Each ONAP project has it's own hierarchy in Gerrit (its own Git tree). Can they do a Git Pull, Git Clone on an arbitrary git repository?

Restrictions on builds: (from https://scan.coverity.com/)

Maximum Lines of Code in Project	Frequency of scans
<100K lines of code	Up to 28 builds per week, with a maximum of 4 builds per day
100K to 500K lines of code	Up to 21 builds per week, with a maximum of 3 builds per day
500K to 1 million lines of code	Up to 14 builds per week, with a maximum of 2 build per day
>1 million lines of code	Up to 7 builds per week, with a maximum of 1 build per day

> Once a project reaches the maximum builds per week, additional build requests will be rejected. The submitter will be able to re-submit the build request the following week.

Scan is self-service: Coverity provides the analysis infrastructure and results, but the submitter must provide the instrumented artifacts to analysis. Scan provides integration with TravisCI/Github.

To use Scan, the submitters will have to create an account and submit their project at https://scan.coverity.com/projects

Coverity requires a code contributor to submit a project because of their responsible disclosure process for issues the tool may identify within the code.

Next Steps:

- Meet with Coverity (schedule call, include Tony Hansen , someone from Linux Foundation)
  - Will Scan integrate with Gerrit? (Coverity Scan tool indicates that it does integrate with Gerrit.)
  - Can it integrate with Jenkins (use resources from Linux Foundation to assist)?
  - How long does it take to run a scan and get results?
  - Lead time with Coverity to use Scan?
  - Mass registration of all ONAP subcomponents (approximately 30 projects, 210 subprojects)?

- Identify an open source project actively using Coverity Scan to get their feedback on the integration of Scan with their code development lifecycle
- Determine whether or not the restrictions on scan frequency will cause a problem for any of the ONAP projects
- Identify an ONAP project willing to test Scan (possibly CLAMP since they are also going through CII badging)
- Integrate Scan with ONAP code development (if Scan is determined to be a viable product)

5. Security of the xNF Package and the Artifacts

Status: Priority 1 approved by TSC

Priority 1: xNF Package Verification (Committed for Dublin as part of 5G use case)

Integrity of the xNF package needs to be verified prior to, or at the time of onboarding. The purpose is to ensure that the xNF package originates from the vendor, and that the content has not been tampered with. The verification is done against the signature provided by the vendor. Reference [ETSI NFV SOL004] contains the detailed specifications on VNF package. As of March 2019 this is being implemented for Dublin release in SDC and VNF SDK (VNF SDK includes partial implementation from Casablanca).

Priority 2: Integrity Verification at Instantiation (Release TBD)

Reference [ETSI NFV SEC021] is the main specification of this feature. As of March 2019 the status is 'final draft for approval', target date of publication is July 2019.
As of March 2019, ETSI NFV plans changes in [ETSI NFV SOL004] impacting this item: creation of signature per individual artifact in the VNF package (by the package vendor) is planned to be mandatory.

Priority 3: Service Provider Ability to Sign the Artifacts (Release TBD)

Reference [ETSI NFV SEC021] is the main specification of this feature. As of March 2019 the status is 'final draft for approval', target date of publication is July 2019.

References

[ETSI NFV SOL004]
ETSI GS NFV-SOL 004 V2.3.1 (2017-07): http://www.etsi.org/deliver/etsi_gs/NFV-SOL/001_099/004/02.03.01_60/gs_nfv-sol004v020301p.pdf

[ETSI NFV SEC021]
The latest draft can be found in: https://portal.etsi.org/webapp/WorkProgram/Report_WorkItem.asp?WKI_ID=53601