Security Levels

Level Definitions

- *Project-level requirements*
    - Level 0: None
    - Level 1: CII Passing badge
        - Including no critical and high known vulnerabilities > 60 days old
    - Level 2: CII Silver badge, plus:
        - All internal/external system communications shall be able to be encrypted.
        - All internal/external service calls shall have common role-based access control and authorization using CADI framework.
    - Level 3: CII Gold badge

ONAP Platform-level requirements per release

- Level 1: 70 % of the projects passing the level 1
    - with the non-passing projects reaching 80% passing level
    - Non-passing projects MUST pass specific cryptography criteria outlined by the Security Subcommittee*
- Level 2: 70 % of the projects passing silver
    - with non-silver projects:
        - completed passing level and 80% towards silver level
        - internal/external system communications shall be able to be encrypted
- Level 3: 70% of the projects passing gold
    - with non-gold projects achieving silver level and achieving 80% towards gold level
- Level 4: 100 % passing gold.

Minimum Levels

- Platform Level 2
- Additional recommendations:
    - All projects SHOULD migrate from the Jackson Data Processor packages to the GSON packages unless the Jackson dependency is inherited from an outside project such as ODL.
    - All projects SHOULD provide the ability to turn on and turn off Secure communication. Secure communication is on by default.

Guidance for Implementation

- Refer to the Security Subcommittee

Contacts

- Refer to the Security Subcommittee

*Specific cryptography requirements for security level 1:

- The software produced by the project MUST use, by default, only cryptographic protocols and algorithms that are publicly published and reviewed by experts (if cryptographic protocols and algorithms are used).
- If the software produced by the project is an application or library, and its primary purpose is not to implement cryptography, then it SHOULD only call on software specifically designed to implement cryptographic functions; it SHOULD NOT re-implement its own.
- The security mechanisms within the software produced by the project MUST use default key lengths that at least meet the NIST minimum requirements through the year 2030 (as stated in 2012). It MUST be possible to configure the software so that smaller key lengths are completely disabled.
- The default security mechanisms within the software produced by the project MUST NOT depend on broken cryptographic algorithms (e.g., MD4, MD5, single DES, RC4, Dual_EC_DRBG) or use cipher modes that are inappropriate to the context (e.g., ECB mode is almost never appropriate because it reveals identical blocks within the cipher text as demonstrated by the ECB penguin, and CTR mode is often inappropriate because it does not perform authentication and causes duplicates if the input state is repeated).
- The default security mechanisms within the software produced by the project SHOULD NOT depend on cryptographic algorithms or modes with known serious weaknesses (e.g., the SHA-1 cryptographic hash algorithm or the CBC mode in SSH).
- If the software produced by the project causes the storing of passwords for authentication of external users, the passwords MUST be stored as iterated hashes with a per-user salt by using a key stretching (iterated) algorithm (e.g., PBKDF2, Bcrypt or Scrypt).