IEC/IEEE 60802 Security Slice

Contributors
Fischer, Kai <kai.fischer@siemens.com>
Furch, Andreas <andreas.furch@siemens.com>
Pfaff, Oliver <oliver.pfaff@siemens.com>
Pössler, Thomas <thomas.poessler@siemens.com>
Steindl, Günter <guenter.steindl@siemens.com>

Abstract
The purpose of this text is to establish a common understanding for TSN-IA security. An incremental procedure is applied in bottom-up style:

i. First increment (V0.1, this version): message exchange protection for network configuration with NETCONF-over-TLS

ii. Second increment (V0.2, later): resource access authorization for network configuration with NETCONF-over-TLS

iii. Further increments: to-be-defined

Elaborations of this text provide a skeleton for the security profile text in D1.3 of TSN Profile for Industrial Automation. It also provides a background for describing the security use cases.

Log

v0.1	2021-05-21	Initial draft

Contents

References

1. IETF RFC 4949: Internet Security Glossary, Version 2, 2007
2. IETF RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2, 2008
3. IETF RFC 5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, 2008
4. IETF RFC 6125: Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS), 2011
5. IETF RFC 8572: Using the NETCONF Protocol over Transport Layer Security (TLS) with Mutual X.509 Authentication, 2015
6. IEEE 802.1AR-2018: IEEE Standard for Local and Metropolitan Area Networks–Secure Device Identity, 2018
7. IETF RFC 8572: Secure Zero Touch Provisioning (SZTP), 2019

Abbreviations

ASCII	American Standard Code for Information Interchange
CA	Certification Authority
CN	(X.500) Common Name
DN	(X.500) Distinguished Name
DNS	Domain Name Service
EE	End Entity
FQDN	Fully Qualifier Domain Name
HW	HardWare
IA	Industrial Automation
IDevID	Initial Device IDentifier
LDevID	Locally significant Device IDentifier
NETCONF	NETwork CONFiguration
SZTP	Secure Zero Touch Provisioning
TLS	Transport Layer Security
URL	Uniform Resource Locator
YANG	Yet Another Next Generation

Preconditions

Following preconditions are assumed:

- IA systems are equipped with system components from multiple manufacturers.
- Each individual system component has a housing that carries an end station or bridge component.
- By the time a system component is shipped by its manufacturer, it is assumed to comprise:
    - **Secure element** component: generic or dedicated HW (the exact form factor is out-of-scope for IEC/IEEE 60802) providing:
        - Persistent storage for keys and credentials esp. IDevID/LDevID credentials and corresponding trust anchors (see below)
        - Execution environment for these keys and credentials
    - **IDevID credential** object: defined by **IEEE 802.1AR**, to be further profiled by IEC/IEEE 60802. This object encompasses:
        - Private key
        - End entity (EE) certificate (plus intermediate CA certificates) containing **product master data** identifying the physical instance of this component according to manufacturer knowledge e.g., product serial number in an eternal manner.

      Note: IDevID EE certificates cannot contain deployment master data e.g., application name(s) or IP address(es)

      Hint: *IDevID EE certificates can be thought of as "birth certificates" - they contain data that is known by the time-of-birth.*
    - Corresponding **trust anchor**: defined by IEEE 802.1AR. This object represents the manufacturer certification authority (CA) in the form of a self-signed CA certificate. It is used to initialize the validation of certification paths of peers, see **IETF RFC 5280**.

Goal

A system component (that fulfills the prerequisites above) shall participate in protected network configuration.

- Assumptions:
  - This uses NETCONF as application protocol and YANG as data model
  - Message exchange protection uses TLS according **IETF RFC 7589**
  - The system component acts in (NETCONF and TLS) server role - push supply
- Plain vanilla tasks: using NETCONF-over-TLS is straightforward provided:
  - The NETCONF-over-TLS server (i.e., the to-be-managed system component) possesses a credential (private key, EE certificate [plus intermediate CA certificates]) that matches the requirements in RFC 7589 as well as trust anchor(s) that allows to validate the EE certificates (plus intermediate CA certificates) of its clients.
  - Vice versa for NETCONF-over-TLS clients that (want to) manage the network configuration of the considered system component.
- Provisioning challenge: supply the **LDevID-NETCONF** credential and corresponding **trust anchor** in a secure manner to the system component that shall be managed
  - The shorthand term LDevID-NETCONF is used for an LDevID credential according to IEEE 802.1AR which also matches the requirements that are set forth in section 6 of RFC 7589: the component's FQDN shall be part of the subjectAltName extension in the EE certificate
  - In general, LDevID credentials encompass:
    - Private key
    - EE certificate containing **deployment master data** identifying the component according to deployment knowledge e.g., application name(s) or IP address(es) and in a time-limited manner.

      Hint: *LDevID EE certificates can be thought of as "driving licenses" - they contain info that is unknown when "birth certificates" are issued e.g., driving license classes*

Solving this Provisioning Challenge

Suggested approach for solving this provisioning challenge[1]:

- Use NETCONF-over-TLS for supplying the LDevID-NETCONF credential and corresponding trust anchor as NETCONF payload.
- Use a YANG-based info model to store/address the LDevID-NETCONF credential and corresponding trust anchor.
- Bootstrapping challenge: the LDevID-NETCONF credential and corresponding trust anchor supply happens in NETCONF payload. When this provisioning is happening, the to-be-provisioned objects cannot be simultaneously used in the TLS layer.

[1] NETCONF SZTP in **IETF RFC 8572** is no (full) solution for this provisioning challenge: it does not cover the credential portion. The trust anchor portion is covered but SZTP uses pull or physical push (*Removeable Storage*)

Solving this Bootstrapping Challenge

Suggested approach: use the IDevID credential and corresponding trust anchor (see prerequisites) on TLS protocol level when performing the NETCONF-over-TLS exchanges to provision the LDevID-NETCONF credential and corresponding trust anchor.

Resulting challenges:

- Server identity checking challenge: the matching rule in RFC 7589 is geared towards the "*all is setup*" scenario (post provisioning). Adaptations of the matching rule need to be considered for exchanges that do this provisioning. TODO: follow-up (later)
- Client identity verification challenge: clients that call the component for doing the provisioning must be assumed to be equipped with credentials from another authority, not yet known by the to-be-provisioned component. The imprinting of common trust anchors and/or provisional acceptance of clients for which the server has not yet a matching trust anchor needs to be considered. TODO: follow-up (later)
- Client authorization challenge: TODO: follow-up (part of V0.2)

Annex A IEEE 802.1AR 'Secure Device Identity'

A.1 IDevID Objects

- Abbreviation for: **I**nitial **Dev**ice **ID**entifier
- Definition (somewhat rephrased for simplicity): a manufacturer-generated and installed object that is cryptographically bound to the component, and that comprises (see **IEEE 802.1AR** for all applicable details):
    - An asymmetric **private key**
    - An **EE certificate** which binds the corresponding public key to information about the component and that is stated by its manufacturer. This certificate is assumed to be:
        - Valid eternally (notAfter=99991231235959Z)
        - Have an X.500 subject field (DN) carrying a unique product serial number
        - Not self-signed
    - A **certificate chain** i.e., a list of intermediate CA certificates that links the EE certificate to the trust anchor (self-signed root CA certificate) of the manufacturer
- Quantity: IEEE 802.1AR-2018 allows one component to possess one or more IDevIDs (IEEE 802.1AR-2009 did limit this to one IDevID).
- Important:
    - IDevID issuance and supply is meant to happen once in the lifetime of the component (during its manufacturing and before its shipment). Typically, the IDevID object is never updated or erased.
    - Since IDevID objects are created at component manufacturing time they can only contain information known at manufacturing time (these items are called 'product master data' herein).
    - System integrators and owner/operators do not have to worry about IDevID object production - they consume IDevIDs only.
    - Invalidation of an IDevID credential does not (have to) prevent the usage of the component:
        - This only prevents the use of this IDevID object. This affects usages of this IDevID after the invalidation event, not (or not necessarily) earlier usages of this IDevID before its invalidation event.
        - This does not affect the usage of other IDevID credentials - if there are multiple IDevID credential objects for a specific component.

A.2 LDevID Objects

- Abbreviation for: **L**ocally significant **Dev**ice **ID**entifier
- Definition (somewhat rephrased for simplicity): a system integrator or owner/operator-generated and installed object that is cryptographically bound to the component, and that comprises (see **IEEE 802.1AR** for all applicable details):
    - An asymmetric **private key**

  - An **EE certificate** which binds the corresponding public key to information about the component and that is stated by its system integrator or owner/operator. This certificate is assumed to be:
    - Not eternal (no [notBefore, notAfter] interval length is suggested)
    - Not self-signed
  - A **certificate chain** i.e., a list of intermediate CA certificates that links the EE certificate to the trust anchor (self-signed root CA certificate) of the system integrator or owner/operator.
- Quantity: IEEE 802.1AR-2009 and 2018 allow one component to possess one or more LDevIDs
- Important:
  - LDevID issuance and supply is meant to happen one or more times during the lifetime of the component (during bootstrapping or even operation phases). The LDevID objects can be updated or erased. A security model is needed to prevent attackers from supplying or managing LDevID objects.
  - The LDevID objects are created at bootstrapping or even operation time of the component. Hence, they can and shall contain information known when this component is bootstrapped or operated but which is not known when the component is manufactured (this is also called ‘deployment master data’ herein).
  - Manufacturers do not have to worry about LDevID supply. With respect to LDevIDs their “only” concern is supplying (protected and initially empty) storage and means to support system integrators and owners/operators e.g., building blocks for cryptographic operations such as random number generation, key pair generation, object signing and validating.
  - Invalidation of an LDevID credential does not (have to) prevent the usage of the component:
    - This only prevents the use of this LDevID credential. This affects usages of this LDevID credential after the invalidation event, not (or not necessarily) earlier usages of this IDevID before its invalidation event.
    - This does not affect the usage of other LDevID credentials - if there are multiple LDevID credential objects for a specific component.
    - Although this reads equivalent to the corresponding section for IDevIDs, the consequences of a LDevID invalidation are more severe than IDevID invalidation. This is due to following:
      - LDevIDs should be assumed to be used often (hint: “daily use”)
      - IDevIDs can be assumed to be used occasionally (hint: “annual use”)

Annex B IETF RFC 6125

IETF RFC 6125 is mandated for checking the identity of a NETCONF-over-TLS server by RFC 7589 '*Using the NETCONF Protocol over Transport Layer Security (TLS) with Mutual X.509 Authentication*'.

RFC 6125 requires the name of an application service to be (or to be based on) a DNS domain name in one of the following forms:

- **Traditional domain name**: a FQDN with labels constrained to ASCII letter, digits and hyphen (further small-print applies)
- **Internationalized domain name**: a FQDN with at least one Unicode label (further small-print applies)

Following 'actual vs. expected'-matching rules apply for checking the identity of a NETCONF-over-TLS server based on their application names:

- Actual (FQDN in subjectAltName extension of the EE certificate) is a traditional domain name: case-insensitive ASCII comparison against expected (from address info e.g., request URL)
- Actual (FQDN in subjectAltName extension of the EE certificate) is an internationalized domain name: case-insensitive ASCII comparison against expected (from address info e.g., request URL) after performing any U-label to an A-label (see RFC 5890 and 5891 for details)
- Actual (FQDN in subjectAltName extension of the EE certificate) contains a wildcard in its leftmost label:
    - "*" always matches e.g., foo.example.com matches *.example.com (does not match foo.example.net or foo.superexample.com)
    - "<abc>*<xyz>" matches when it matches e.g., foobar.example.com matches foo*.example.com (small-print applies, see RFC 6125)
- Actual (CN in subject field [this is an X.500 DN] of the EE certificate) is a traditional domain name: case-insensitive ASCII comparison against expected (from address info e.g., request URL)

As a *last resort check* (if no FQDN can be found in the subjectAltName extension of the EE certificate) these matching rules can be applied to the CN portion of the subject DN value (small-print applies, see RFC 6125).