SUIT Multiple Trust Domains

draft-ietf-suit-trust-domains-08

Brendan Moran and Ken Takayama

Status of draft-ietf-suit-trust-domains

- We are addressing review comments.

- Thank you for your comments!
  - Thomas Fossati
  - Marek Serafin
  - Dave Thaler
  - Russ Housley
  - Deb Cooley

Comment from Thomas Fossati #1

I started checking the examples against the extended CDDL and validation failed.

We've found that the failure was introduced by this update

suit-install = 17 → suit-install = 20

in draft-ietf-suit-manifest and fixed this.
And additionally,

Address Comment from Thomas Fossati #1

I started checking the examples against the extended CDDL and validation failed.

```
suit-manifest-multiple.cddl:
	cat draft-ietf-suit-manifest.cddl rfc-9052.cddl ../draft-ietf-suit-trust-domains.cddl > suit-manifest-multiple.cddl

.PHONY: validate
validate: suit-manifest-multiple.cddl
	RUBYOPT="-W0" cddl suit-manifest-multiple.cddl validate ../examples/example1_process.suit
	RUBYOPT="-W0" cddl suit-manifest-multiple.cddl validate ../examples/example2_integrated.suit
```

CI check is added to GitHub repo
to always validate examples agains CDDL

https://github.com/suit-wg/suit-multiple-trust-domains/pull/23

Comment from Thomas Fossati #2

Examples and CDDL need some attention - mostly on syntax and alignment with SUIT grammar.

Earlier this month, Ken and I have been exchanging messages on the topic.
IIRC, he has fixed the identified issues in his local copy.

Solved by #1

Example 0 references a dependent manifest, but this reader couldn’t find it.
For completeness, I suggest adding it.

Needed to be solved.

Address Comment from Thomas Fossati #2

Example 0 references a dependent manifest, but this reader couldn't find it.
For completeness, I suggest adding it.

Now Added

```
   The dependent Manifest (fetched from "https://example.com/
   dependent.suit"):

/ SUIT_Envelope_Tagged / 107({
  / authentication-wrapper / 2: << [
    << [
      / digest-algorithm-id: / -16 / SHA256 /,
      / digest-bytes: / h'0F02CAF6D3E61920D36BF3CEA7F862A13BB8FB1F09C3F4
    ] >>,
    << / COSE_Sign1_Tagged / 18([
      / protected: / << {
        / algorithm-id / 1: -7 / ES256 /
      } >>,
      / unprotected: / {},
      / payload: / null,
      / signature: / h'D0703EA193E12381A66FFADEF2F0949711CFE05ED2322818D
    ]) >>
  ] >>,
  / manifest / 3: << {
    / manifest-version / 1: 1,
    / manifest-sequence-number / 2: 0,
    / common / 3: << {
      / components / 2: [
        ['00']
      ]
    } >>,
    / manifest-component-id / 5: [
      'dependent.suit'
    ],
    / invoke / 9: << [
      / directive-override-parameters / 20, {
        / parameter-invoke-args / 23: 'cat 00'
      },
      / directive-invoke / 23, 15
    ] >>,
    / install / 20: << [
      / directive-override-parameters / 20, {
        / parameter-content / 18: 'hello world'
      },
      / directive-write / 18, 15
    ] >>
  } >>
})
```

Comment from Marek Serafin on 5.6.4.

If I understand correctly ①I verify the signature of the dependency manifest hash using dependency authentication block. Then ②I compare the dependent's image digest param to the value from dependency suit digest ? Then ③I calculate checksum of dependency manifest myself (as in dependent) and I compare it to the dependency suit digest from dependency manifest?

5.6.4. suit-condition-dependency-integrity

Verify the integrity of a Dependency Manifest. When a Manifest Processor executes suit-condition-dependency-integrity, it performs the following operations:

① • Verify the signature of the Manifest hash

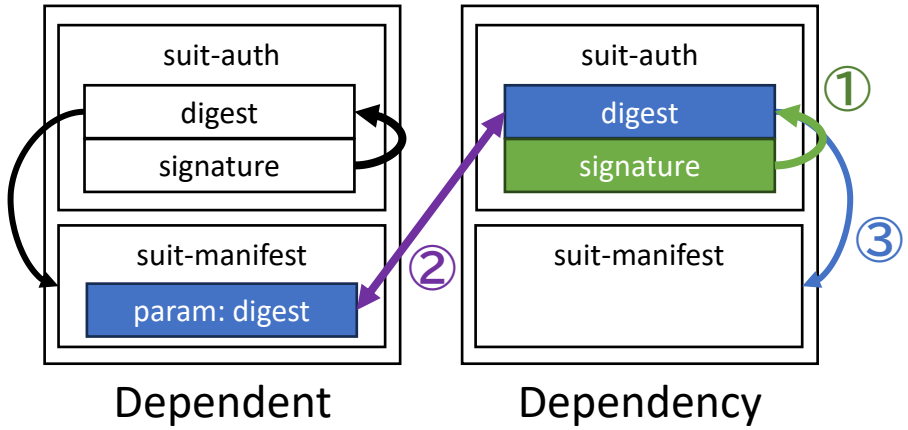
② • Compare the Manifest hash to the provided hash

③ • Verify the Manifest against the Manifest hash

Previous draft was ambiguous

Address Comment from Marek Serafin

We clarified each digest location



5.6.4. suit-condition-dependency-integrity

Verify the integrity of a Dependency Manifest. When a Manifest Processor executes suit-condition-dependency-integrity, it performs the following operations:

① • Verify the signature of the Dependency's suit-authentication-wrapper.

② • Compare the Dependency's suit-authentication-wrapper digest to the dependent's suit-parameter-image-digest

③ • Verify the Dependency Manifest against the Depedency's suit-authentication-wrapper digest

Clarified

Comment from Dave Thaler #1

The abstract seems to contain references ([I-D.ietf-suit-manifest]), which it shouldn't.
Please replace those with straight textual mentions of the documents in question.

Workgroup: SUIT
Internet-Draft:
draft-ietf-suit-trust-domains-07
Published: 6 July 2024
Intended Status: Standards Track
Expires: 7 January 2025

B. Moran
Arm Limited
K. Takayama
SECOM CO., LTD.

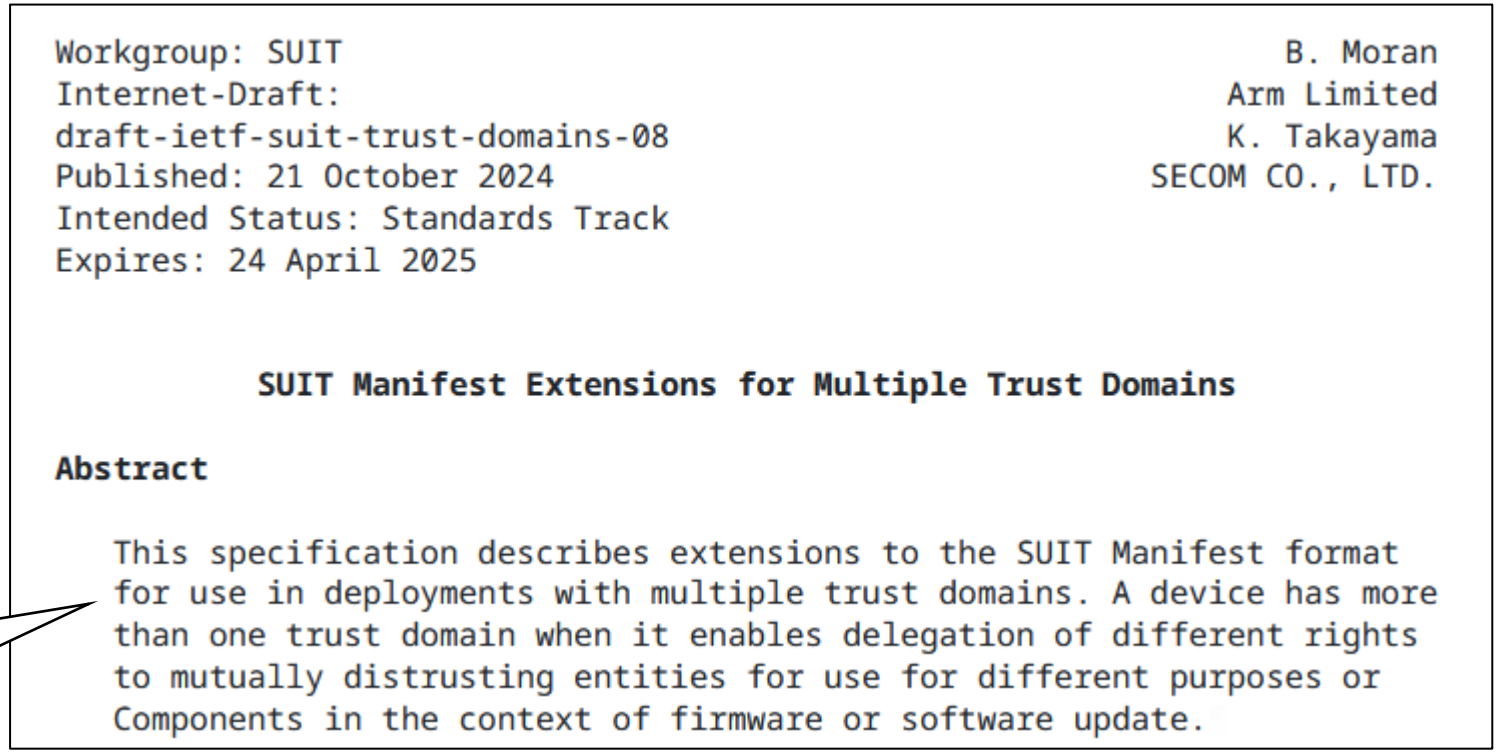
SUIT Manifest Extensions for Multiple Trust Domains

Abstract

This specification describes extensions to the SUIT Manifest format (as defined in [I-D.ietf-suit-manifest]) for use in deployments with multiple trust domains. A device has more than one trust domain when it enables delegation of different rights to mutually distrusting entities for use for different purposes or Components in the context of firmware or software update.

Address Comment from Dave Thaler #1

The abstract seems to contain references ([I-D.ietf-suit-manifest]), which it shouldn't.
Please replace those with straight textual mentions of the documents in question.



Workgroup: SUIT
Internet-Draft:
draft-ietf-suit-trust-domains-08
Published: 21 October 2024
Intended Status: Standards Track
Expires: 24 April 2025

B. Moran
Arm Limited
K. Takayama
SECOM CO., LTD.

SUIT Manifest Extensions for Multiple Trust Domains

Abstract

This specification describes extensions to the SUIT Manifest format for use in deployments with multiple trust domains. A device has more than one trust domain when it enables delegation of different rights to mutually distrusting entities for use for different purposes or Components in the context of firmware or software update.

Now Removed

https://github.com/suit-wg/suit-multiple-trust-domains/pull/22

Comment from Dave Thaler #2

I found the following so far:

1. RFC 8392 is in the normative refs section but as far as I can find, it is never cited in the document.
2. Same issue for RFC 8747.
3. Same issue for RFC 3986.
4. ietf-suit-firmware-encryption looks normatively cited in the document but is in the informative refs section.

Address Comment from Dave Thaler #2

I found the following so far:

Now Removed

1. RFC 8392 is in the normative refs section but as far as I can find, it is never cited in the document.
2. Same issue for RFC 8747.
3. Same issue for RFC 3986.
4. ietf-suit-firmware-encryption looks normatively cited in the document but is in the informative refs section.

To be Moved in the next version

https://github.com/suit-wg/suit-multiple-trust-domains/pull/25 (Open)

Comment from Dave Thaler and Russ Housley #3

In https://mailarchive.ietf.org/arch/msg/suit/aO64Ga0UysSThUqPf8mhWGYCJ_g/

Russ stated:

- There are downrefs to four Informational RFCs: RFC 6024, RFC 7228, RFC 9019, and RFC 9124.
- Looking at each one, I think these can be informational references.

I agree with that, and that feedback has also not been responded to or acted on yet.

Address Comment from Dave and Russ #3

In https://mailarchive.ietf.org/arch/msg/suit/aO64Ga0UysSThUqPf8mhWGYCJ_g/

Russ stated:

- There are downrefs to four Informational RFCs: RFC 6024, RFC 7228, RFC 9019, and RFC 9124.
- Looking at each one, I think these can be informational references.

To be Moved in the next version

https://github.com/suit-wg/suit-multiple-trust-domains/pull/26 (Open)

Comment from Dave Thaler and Deb Cooley

Deb Cooley asked in an AD review:
> Section 10:
> 1. Why bother referencing RFC9019 when it merely points to RFC9124
> (and really only to the draft of RFC9124 that existed at the time of publication).
> 2. While RFC9124 discussed security considerations in terms of threats and consequences
> in great detail, it does not address the threats/consequences of having multiple
> trust domains.
> What vulnerabilities are introduced by allowing dependencies?
> What does the implementer have to consider when adding this complexity?
>
> There are hints buried in the draft (must abort type language), but it is my opinion
> that those hints could be expanded and clarified in the security consideration section.

Personally I think referencing 9019 in section 10 is still useful but don't feel strongly either way.
But the authors should respond to that feedback.

Address Comment from Dave Thaler and Deb Cooley

Deb Cooley asked in an AD review:

> Section 10
> 1. Why
> (and rea
> 2. Whil
consequ
> in great
> trust do
> What vu
> What do
>
> There ar
> that those hints coul

n).

ple

pinion

and clarified in the security consideration section.

RFC9019 covers the architecture that references multiple trust domains and requirements that bring about multiple trust domains.
RFC9124 does cover the threats to multiple trust domains.

Personally I think referencing 9019 in section 10 is still useful but don't feel strongly either way.
But the authors should respond to that feedback.

RFC9019 coverage of multiple trust domains

- Section 3:
  A dependency resolution phase is needed when more than one component can be updated or when a differential update is used. The necessary dependencies must be available prior to installation.
- Section 5.2:
  There will generally be two images: one for secure mode and one for normal mode. In this configuration, firmware upgrades will generally be done by the CPU in secure mode, which is able to write to both areas of the flash device. In addition, there are requirements to be able to update either image independently as well as to update them together atomically, as specified in the associated manifests.

- These sections inform the security considerations in the form of requirements and architecture.

RFC9124 coverage of multiple trust domains

- 4.2.9: THREAT.IMG.NON_AUTH: Unauthenticated Images
  If an attacker can install their firmware on a device -- for example, by manipulating either payload or metadata -- then they have complete control of the device.
- 4.2.13: THREAT.MFST.OVERRIDE: Overriding Critical Manifest Elements
  An authorized actor, but not the author, uses an override mechanism (USER_STORY.OVERRIDE (Section 4.4.3)) to change an information element in a manifest signed by the author.

- More detail is presented in the security requirements and manifest elements, however it appears that some additional treatment is needed to address these threats in more detail within draft-ietf-suit-trust-domains.