Workgroup: RATS Working Group
Internet-Draft:
draft-birkholz-rats-epoch-markers-06
Published: 23 October 2023
Intended Status: Standards Track
Expires: 25 April 2024
Authors: H. Birkholz
Fraunhofer SIT
T. Fossati
Arm Limited
W. Pan
Huawei Technologies
C. Bormann
Universität Bremen TZI

Epoch Markers

Abstract

This document defines Epoch Markers as a way to establish a notion of freshness among actors in a distributed system. Epoch Markers are similar to "time ticks" and are produced and distributed by a dedicated system, the Epoch Bell. Systems that receive Epoch Markers do not have to track freshness using their own understanding of time (e.g., via a local real-time clock). Instead, the reception of a certain Epoch Marker establishes a new epoch that is shared between all recipients.

About This Document

This note is to be removed before publishing as an RFC.

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-birkholz-rats-epoch-markers/.

Discussion of this document takes place on the rats Working Group mailing list (mailto:rats@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/rats/. Subscribe at https://www.ietf.org/mailman/listinfo/rats/.

Source for this draft and an issue tracker can be found at https://github.com/ietf-rats/draft-birkholz-rats-epoch-marker.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 25 April 2024.

Copyright Notice

Table of Contents

1. Introduction

Systems that need to interact securely often require a shared understanding of the freshness of conveyed information. This is certainly the case in the domain of remote attestation procedures. In general, securely establishing a shared notion of freshness of the exchanged information among entities in a distributed system is not a simple task.

The entire Appendix A of [RFC9334] deals solely with the topic of freshness, which is in itself an indication of how relevant, and complex, it is to establish a trusted and shared understanding of freshness in a RATS system.

This document defines Epoch Markers as a way to establish a notion of freshness among actors in a distributed system. Epoch Markers are similar to "time ticks" and are produced and distributed by a dedicated system, the Epoch Bell. Systems that receive Epoch Markers do not have to track freshness using their own understanding of time (e.g., via a local real-time clock). Instead, the reception of a certain Epoch Marker establishes a new epoch that is shared between all recipients. In essence, the emissions and corresponding receptions of Epoch Markers are like the ticks of a clock where the ticks are conveyed by the Internet.

In general (barring highly symmetrical topologies), epoch ticking incurs differential latency due to the non-uniform distribution of receivers with respect to the Epoch Bell. This introduces skew that needs to be taken into consideration when Epoch Markers are used.

While all Epoch Markers share the same core property of behaving like clock ticks in a shared domain, various "epoch id" types are defined to accommodate different use cases and diverse kinds of Epoch Bells.

While Epoch Markers are encoded in CBOR [STD94], and many of the epoch id types are themselves encoded in CBOR, a prominent format in this space is the Time-Stamp Token defined by [RFC3161], a DER-encoded TSTInfo value wrapped in a CMS envelope [RFC5652]. Time-Stamp Tokens (TST) are produced by Time-Stamp Authorities (TSA) and exchanged via the Time-Stamp Protocol (TSP). At the time of writing, TSAs are the most common providers of secure time-stamping services. Therefore, reusing the core TSTInfo structure as an epoch id type for Epoch Markers is instrumental for enabling smooth migration paths and promote interoperability. There are, however, several other ways to represent a signed timestamp, and therefore other kinds of payloads that can be used to implement Epoch Markers.

To inform the design, this document discusses a number of interaction models in which Epoch Markers are expected to be exchanged. The top-level structure of Epoch Markers and an initial set of epoch id types are specified using CDDL [RFC8610]. To increase trustworthiness in the Epoch Bell, Epoch Markers also provide the option to include a "veracity proof" in the form of attestation evidence, attestation results, or SCITT receipts [I-D.ietf-scitt-architecture] associated with the trust status of the Epoch Bell.

1.1. Requirements Notation

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

In this document, CDDL [RFC8610] is used to describe the data formats. The examples in Appendix A use CBOR diagnostic notation as defined in Section 8 of [STD94] and Appendix G of [RFC8610].

2. Epoch IDs

The RATS architecture introduces the concept of Epoch IDs that mark certain events during remote attestation procedures ranging from simple handshakes to rather complex interactions including elaborate freshness proofs. The Epoch Markers defined in this document are a solution that includes the lessons learned from TSAs, the concept of Epoch IDs defined in the RATS architecture, and provides several means to identify a new freshness epoch. Some of these methods are introduced and discussed in Section 10.3 of the RATS architecture [RFC9334].

3. Interaction Models

The interaction models illustrated in this section are derived from the RATS Reference Interaction Models. In general, there are three interaction models:

* ad-hoc requests (e.g., via challenge-response requests addressed at Epoch Bells), corresponding to Section 7.1 in [I-D.ietf-rats-reference-interaction-models]

* unsolicited distribution (e.g., via uni-directional methods, such as broad- or multicasting from Epoch Bells), corresponding to Section 7.2 in [I-D.ietf-rats-reference-interaction-models]

*solicited distribution (e.g., via a subscription to Epoch Bells), corresponding to Section 7.3 in [I-D.ietf-rats-reference-interaction-models]

In all three interaction models, Epoch Markers can be used as content for the generic information element 'handle'. Handles are most useful to establish freshness in unsolicited and solicited distribution by the Epoch Bell. An Epoch Marker can be used as a nonce in challenge-response remote attestation (e.g., for limiting the number of ad-hoc requests by a Verifier). Using an Epoch Marker requires the challenger to acquire an Epoch Marker beforehand, which may introduce a sensible overhead compared to using a simple nonce.

4. Epoch Marker Structure

At the top level, an Epoch Marker is a CBOR array carrying the actual epoch id (Section 4.1) and an optional veracity proof about the Epoch Bell.

```
epoch-marker = [
  $tagged-epoch-id
  ? bell-veracity-proof
]

; veracity of the bell
bell-veracity-proof = non-empty<{
  ? remote-attestation-evidence ; could be EAT or Concise Evidence
  ? remote-attestation-result ; hopefully EAT with AR4SI Claims
  ? scitt-receipt ; SCITT receipt
}>

remote-attestation-evidence = (1: "PLEASE DEFINE")
remote-attestation-result = (2: "PLEASE DEFINE")
scitt-receipt = (3: "PLEASE DEFINE")

; epoch-id types independent of interaction model
$tagged-epoch-id /= cbor-epoch-id
$tagged-epoch-id /= #6.26980(classical-rfc3161-TST-info)
$tagged-epoch-id /= #6.26981(TST-info-based-on-CBOR-time-tag)
$tagged-epoch-id /= #6.26982(epoch-tick)
$tagged-epoch-id /= #6.26983(epoch-tick-list)
$tagged-epoch-id /= #6.26984(strictly-monotonic-counter)
```

Figure 1: Epoch Marker definition

The veracity proof can be encoded in an Evidence or Attestation Result conceptual message [RFC9334], e.g., using [I-D.ietf-rats-eat], [TCG-CoEvidence], [I-D.ietf-rats-ar4si], or SCITT receipts [I-D.ietf-scitt-architecture].

4.1. Epoch Marker Payloads

This memo comes with a set of predefined payloads.

4.1.1. CBOR Time Tags

A CBOR time representation choosing from CBOR tag 0 (tdate, RFC3339 time as a string), tag 1 (time, Posix time as int or float) or tag 1001 (extended time data item), optionally bundled with a nonce.

See Section 3 of [I-D.ietf-cbor-time-tag] for the (many) details about the CBOR extended time format (tag 1001). See [STD94] for tdate (tag 0) and time (tag 1).

```
cbor-epoch-id = [
   cbor-time
   ? nonce
]

etime = #6.1001({* (int/tstr) => any})

cbor-time = tdate / time / etime

nonce = tstr / bstr / int
```

The following describes each member of the cbor-epoch-id map.

etime: A freshly sourced timestamp represented as either time or tdate ([STD94], [RFC8610]) or etime [I-D.ietf-cbor-time-tag].

nonce: An optional random byte string used as extra data in challenge-response interaction models (see [I-D.ietf-rats-reference-interaction-models]).

4.1.1.1. Creation

To generate the cbor-time value, the emitter **MUST** follow the requirements in Section 4.2.

If a nonce is generated, the emitter **MUST** follow the requirements in Section 4.3.

4.1.2. Classical RFC 3161 TST Info

DER-encoded [X.690] TSTInfo [RFC3161]. See Appendix A.1 for the layout.

```
classical-rfc3161-TST-info = bytes
```

The following describes the classical-rfc3161-TST-info type.

classical-rfc3161-TST-info:
The DER-encoded TSTInfo generated by a [RFC3161] Time Stamping Authority.

4.1.2.1. Creation

The Epoch Bell **MUST** use the following value as MessageImprint in its request to the TSA:

```
SEQUENCE {
  SEQUENCE {
    OBJECT      2.16.840.1.101.3.4.2.1 (sha256)
    NULL
  }
  OCTET STRING BF4EE9143EF2329B1B778974AAD445064940B9CAE373C9E35A7B23361
}
```

This is the sha-256 hash of the string "EPOCH_BELL".

The TimeStampToken obtained by the TSA **MUST** be stripped of the TSA signature. Only the TSTInfo is to be kept the rest **MUST** be discarded. The Epoch Bell COSE signature will replace the TSA signature.

4.1.3. CBOR-encoded RFC3161 TST Info

Issue tracked at: https://github.com/ietf-rats/draft-birkholz-rats-epoch-marker/issues/18

The TST-info-based-on-CBOR-time-tag is semantically equivalent to classical [RFC3161] TSTInfo, rewritten using the CBOR type system.

```
TST-info-based-on-CBOR-time-tag = {
  &(version : 0) => v1
  &(policy : 1) => oid
  &(messageImprint : 2) => MessageImprint
  &(serialNumber : 3) => integer
  &(eTime : 4) => profiled-etime
  ? &(ordering : 5) => bool .default false
  ? &(nonce : 6) => integer
  ? &(tsa : 7) => GeneralName
  * $$TSTInfoExtensions
}

v1 = 1

oid = #6.111(bstr) / #6.112(bstr)

MessageImprint = [
  hashAlg : int
  hashValue : bstr
]

profiled-etime = #6.1001(timeMap)
timeMap = {
  1 => ~time
  ? -8 => profiled-duration
  * int => any
}
profiled-duration = {* int => any}

GeneralName = [ GeneralNameType : int, GeneralNameValue : any ]
; See Section 4.2.1.6 of RFC 5280 for type/value
```

The following describes each member of the TST-info-based-on-CBOR-time-tag map.

version:
The integer value 1. Cf. version, Section 2.4.2 of [RFC3161].

policy:
A [RFC9090] object identifier tag (111 or 112) representing the TSA's policy under which the tst-info was produced. Cf. policy, Section 2.4.2 of [RFC3161].

messageImprint:
A [RFC9054] COSE_Hash_Find array carrying the hash algorithm identifier and the hash value of the time-stamped datum. Cf. messageImprint, Section 2.4.2 of [RFC3161].

serialNumber:
A unique integer value assigned by the TSA to each issued tst-info. Cf. serialNumber, Section 2.4.2 of [RFC3161].

eTime:
The time at which the tst-info has been created by the TSA. Cf. genTime, Section 2.4.2 of [RFC3161]. Encoded as extended time [I-D.ietf-cbor-time-tag], indicated by CBOR tag 1001, profiled as follows:

* The "base time" is encoded using key 1, indicating Posix time as int or float.

* The stated "accuracy" is encoded using key -8, which indicates the maximum allowed deviation from the value indicated by "base time". The duration map is profiled to disallow string keys. This is an optional field.

* The map **MAY** also contain one or more integer keys, which may encode supplementary information Allowing unsigned integer (i.e., critical) keys goes counter interoperability.

ordering:
boolean indicating whether tst-info issued by the TSA can be ordered solely based on the "base time". This is an optional field, whose default value is "false". Cf. ordering, Section 2.4.2 of [RFC3161].

nonce:
int value echoing the nonce supplied by the requestor. Cf. nonce, Section 2.4.2 of [RFC3161].

tsa:
a single-entry GeneralNames array Section 11.8 of [I-D.ietf-cose-cbor-encoded-cert] providing a hint in identifying the name of the TSA. Cf. tsa, Section 2.4.2 of [RFC3161].

$$TSTInfoExtensions:
A CDDL socket (Section 3.9 of [RFC8610]) to allow extensibility of the data format. Note that any extensions appearing here **MUST** match an extension in the corresponding request. Cf. extensions, Section 2.4.2 of [RFC3161].

4.1.3.1. Creation

The Epoch Bell **MUST** use the following value as messageImprint in its request to the TSA:

```
cbor-diag [ / hashAlg / -16, / sha-256 / / hashValue /
h'BF4EE9143EF2329B1B778974AAD445064940B9CAE373C9E35A7B23361282698F'
]
```

This is the sha-256 hash of the string "EPOCH_BELL".

4.1.4. Epoch Tick

An Epoch Tick is a single opaque blob sent to multiple consumers.

```
; Epoch-Tick

epoch-tick = tstr / bstr / int
```

The following describes the epoch-tick type.

epoch-tick: Either a string, a byte string, or an integer used by RATS roles within a trust domain as extra data included in conceptual messages [RFC9334] to associate them with a certain epoch.

4.1.4.1. Creation

The emitter **MUST** follow the requirements in Section 4.3.

4.1.5. Multi-Nonce-List

A list of nonces send to multiple consumer. The consumers use each Nonce in the list of Nonces sequentially. Technically, each sequential Nonce in the distributed list is not used just once, but by every Epoch Marker consumer involved. This renders each Nonce in the list a Multi-Nonce

```
; Epoch-Tick-List

epoch-tick-list = [ + epoch-tick ]
```

The following describes the multi-nonce type.

multi-nonce-list:
A sequence of byte strings used by RATS roles in trust domain as extra data in the production of conceptual messages as specified by the RATS architecture [RFC9334] to associate them with a certain epoch. Each nonce in the list is used in a consecutive production of a conceptual messages. Asserting freshness of a conceptual message including a nonce from the multi-nonce-list requires some state on the receiver side to assess if that nonce is the appropriate next unused nonce from the multi-nonce-list.

4.1.5.1. Creation

The emitter **MUST** follow the requirements in Section 4.3.

4.1.6. Strictly Monotonically Increasing Counter

A strictly monotonically increasing counter.

The counter context is defined by the Epoch bell.

```
strictly-monotonic-counter = uint
```

The following describes the strictly-monotonic-counter type.

strictly-monotonic-counter: An unsigned integer used by RATS roles in a trust domain as extra data in the production of of conceptual messages as specified by the RATS architecture [RFC9334] to associate them with a certain epoch. Each new strictly-monotonic-counter value must be higher than the last one.

4.2. Time Requirements

Time **MUST** be sourced from a trusted clock.

4.3. Nonce Requirements

A nonce **MUST** be freshly generated. The generated value **MUST** have at least 64 bits of entropy (before encoding). The generated value **MUST** be generated via a cryptographically secure random number generator.

A maximum nonce size of 512 bits is set to limit the memory requirements. All receivers **MUST** be able to accommodate the maximum size.

5. Security Considerations

TODO

6. IANA Considerations

RFC Editor: please replace RFCthis with the RFC number of this RFC and remove this note.

6.1. New CBOR Tags

IANA is requested to allocate the following tags in the "CBOR Tags" registry [IANA.cbor-tags], preferably with the specific CBOR tag value requested:

Tag	Data Item	Semantics	Reference
26980	bytes	DER-encoded RFC3161 TSTInfo	Section 4.1.2 of RFCthis
26981	map	CBOR-encoding of RFC3161 TSTInfo semantics	Section 4.1.3 of RFCthis
26982	tstr / bstr / int	a nonce that is shared among many participants but that can only be used once by each participant	Section 4.1.4 of RFCthis
26983	array	a list of multi-nonce	Section 4.1.5 of RFCthis
26984	uint	strictly monotonically increasing counter	Section 4.1.6 of RFCthis

Table 1: New CBOR Tags

6.2. New EM CWT Claim

This specification adds the following value to the "CBOR Web Token Claims" registry [IANA.cwt].

*Claim Name: em

*Claim Description: Epoch Marker

*Claim Key: 2000

*Claim Value Type(s): CBOR array

*Change Controller: IESG

*Specification Document(s): Section 4 of RFCthis

7. References

7.1. Normative References

[I-D.ietf-cbor-time-tag]

Bormann, C., Gamari, B., and H. Birkholz, "Concise Binary Object Representation (CBOR) Tags for Time, Duration, and Period", Work in Progress, Internet-Draft, draft-ietf-cbor-time-tag-11, 22 October 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-cbor-time-tag-11>.

[I-D.ietf-cose-cbor-encoded-cert] Mattsson, J. P., Selander, G., Raza, S., Höglund, J., and M. Furuhed, "CBOR Encoded X.509 Certificates (C509 Certificates)", Work in Progress, Internet-Draft, draft-ietf-cose-cbor-encoded-cert-07, 20 October 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-cose-cbor-encoded-cert-07>.

[IANA.cbor-tags] IANA, "Concise Binary Object Representation (CBOR) Tags", <http://www.iana.org/assignments/cbor-tags>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC3161] Adams, C., Cain, P., Pinkas, D., and R. Zuccherato, "Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)", RFC 3161, DOI 10.17487/RFC3161, August 2001, <https://www.rfc-editor.org/rfc/rfc3161>.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <https://www.rfc-editor.org/rfc/rfc5652>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[RFC8610] Birkholz, H., Vigano, C., and C. Bormann, "Concise Data Definition Language (CDDL): A Notational Convention to Express Concise Binary Object Representation (CBOR) and JSON Data Structures", RFC 8610, DOI 10.17487/RFC8610, June 2019, <https://www.rfc-editor.org/rfc/rfc8610>.

[RFC9054] Schaad, J., "CBOR Object Signing and Encryption (COSE): Hash Algorithms", RFC 9054, DOI 10.17487/RFC9054, August 2022, <https://www.rfc-editor.org/rfc/rfc9054>.

[RFC9090] Bormann, C., "Concise Binary Object Representation (CBOR) Tags for Object Identifiers", RFC 9090, DOI 10.17487/RFC9090, July 2021, <https://www.rfc-editor.org/rfc/rfc9090>.

[STD94] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", STD 94, RFC 8949, DOI 10.17487/RFC8949, December 2020, <https://www.rfc-editor.org/rfc/rfc8949>.

[STD96] Schaad, J., "CBOR Object Signing and Encryption (COSE): Structures and Process", STD 96, RFC 9052, DOI 10.17487/RFC9052, August 2022, <https://www.rfc-editor.org/rfc/rfc9052>.

[X.690] International Telecommunications Union, "Information technology — ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", ITU-T Recommendation X.690, August 2015, <https://www.itu.int/rec/T-REC-X.690>.

7.2. Informative References

[I-D.ietf-rats-ar4si] Voit, E., Birkholz, H., Hardjono, T., Fossati, T., and V. Scarlata, "Attestation Results for Secure Interactions", Work in Progress, Internet-Draft, draft-ietf-rats-ar4si-05, 30 August 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-rats-ar4si-05>.

[I-D.ietf-rats-eat] Lundblade, L., Mandyam, G., O'Donoghue, J., and C. Wallace, "The Entity Attestation Token (EAT)", Work in Progress, Internet-Draft, draft-ietf-rats-eat-22, 14 October 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-rats-eat-22>.

[I-D.ietf-rats-reference-interaction-models] Birkholz, H., Eckel, M., Pan, W., and E. Voit, "Reference Interaction Models for Remote Attestation Procedures", Work in Progress, Internet-Draft, draft-ietf-rats-reference-interaction-models-08, 10 September 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-rats-reference-interaction-models-08>.

[I-D.ietf-scitt-architecture] Birkholz, H., Delignat-Lavaud, A., Fournet, C., Deshpande, Y., and S. Lasker, "An Architecture for Trustworthy and Transparent Digital Supply Chains", Work in Progress, Internet-Draft, draft-ietf-scitt-architecture-03, 16 October 2023, <https://

datatracker.ietf.org/doc/html/draft-ietf-scitt-architecture-03>.

[IANA.cwt] IANA, "CBOR Web Token (CWT) Claims", <http://www.iana.org/assignments/cwt>.

[RFC9334] Birkholz, H., Thaler, D., Richardson, M., Smith, N., and W. Pan, "Remote ATtestation procedureS (RATS) Architecture", RFC 9334, DOI 10.17487/RFC9334, January 2023, <https://www.rfc-editor.org/rfc/rfc9334>.

[TCG-CoEvidence] Trusted Computing Group, "TCG DICE Concise Evidence Binding for SPDM", June 2023, <https://trustedcomputinggroup.org/wp-content/uploads/TCG-DICE-Concise-Evidence-Binding-for-SPDM-Version-1.0-Revision-53_1August2023.pdf>.

Appendix A. Examples

The example in Figure 2 shows an epoch marker with a cbor-epoch-id and no bell veracity proof.

```
[
  / cbor-epoch-id / [
    / 1996-12-19T16:39:57-08:00[America//Los_Angeles][u-ca=hebrew] /
    / etime / 1001({
        1: 851042397,
      -10: "America/Los_Angeles",
      -11: { "u-ca": "hebrew" }
    })
  ]
  / no bell veracity proof /
]
```

Figure 2: CBOR epoch id without bell veracity proof

A.1. RFC 3161 TSTInfo

As a reference for the definition of TST-info-based-on-CBOR-time-tag the code block below depects the original layout of the TSTInfo structure from [RFC3161].

```
TSTInfo ::= SEQUENCE  {
   version                      INTEGER  { v1(1) },
   policy                       TSAPolicyId,
   messageImprint               MessageImprint,
     -- MUST have the same value as the similar field in
     -- TimeStampReq
   serialNumber                 INTEGER,
    -- Time-Stamping users MUST be ready to accommodate integers
    -- up to 160 bits.
   genTime                      GeneralizedTime,
   accuracy                     Accuracy                 OPTIONAL,
   ordering                     BOOLEAN             DEFAULT FALSE,
   nonce                        INTEGER                  OPTIONAL,
     -- MUST be present if the similar field was present
     -- in TimeStampReq.  In that case it MUST have the same value.
   tsa                          [0] GeneralName          OPTIONAL,
   extensions                   [1] IMPLICIT Extensions   OPTIONAL  }
```

Acknowledgements

TBD

Authors' Addresses

Henk Birkholz
Fraunhofer SIT
Rheinstrasse 75
64295 Darmstadt
Germany

Email: henk.birkholz@sit.fraunhofer.de

Thomas Fossati
Arm Limited
United Kingdom

Email: Thomas.Fossati@arm.com

Wei Pan
Huawei Technologies

Email: william.panwei@huawei.com

Carsten Bormann
Universität Bremen TZI
Bibliothekstr. 1
D-28359 Bremen
Germany

Phone: +49-421-218-63921

Email: cabo@tzi.org