Heartbeat Mechanism
draft-ietf-dots-signal-channel

IETF#105 Montreal, July 2019
M. Boucadair, T. Reddy, J. Shallow
Why Heartbeats are Needed?

• Assess if the remote peer is defunct or alive
• Maintain any state in on-path NATs or firewalls

To that aim, *bidirectional HBs* are exchanged between DOTS peers
  – DOTS agents regularly send heartbeats to each other
On The Importance of Client-Initiated Heartbeats

Server-initiated HBs (Only)

(1) A binding entry is instantiated

DOTS Client Domain

C

NAT and/or FW

Network

sHB

DOTS Server Domain

S

No binding entry

Client-initiated HBs to help with NAT/FW traversal

DOTS Client Domain

C

NAT and/or FW

Network

sHB

(3) A binding entry is found

DOTS Server Domain

S

(2) A binding entry is instantiated

Then, why do we need server-initiated HBs in the presence of client-initiated HBs?
On The Importance of Server-Initiated Heartbeats for DOTS

(1) The DOTS server can identify:
- The DOTS client domain is under attack
- The inbound link to the DOTS client domain is saturated.

(2) If an attack mitigation is in progress, it implies the applied DDoS mitigation actions are not yet effective to handle the DDoS attack volume.
On The Importance of Server-Initiated Heartbeats for DOTS

If no traffic is received from the client and missing-hb-allowed is reached, the DOTS server triggers **automatic pre-configured mitigation requests** for this DOTS client (if any).

DOTS Signal Channel

Network

Attack Traffic
DOTS Session Configuration

```
+--:(signal-config)
  |  +--rw sid         uint32
  |  +--rw mitigating-config
  |         +--rw heartbeat-interval
  |         +--rw missing hb-allowed
  |         +--rw max retransmit
  |         +--rw ack timeout
  |         +--rw ack random-factor
  |  +--rw idle-config
```

A DOTS server might want to **reduce** heartbeat frequency or **cease** heartbeat exchanges when an active DOTS client has not requested mitigation (RFC8612)

DOTS agents **automatically switch** to the other configuration upon a change in the mitigation activity
DOTS Session Configuration

Unreliable Transport

+-:(signal-config)
  |   +-rw sid uint32
  |   +-rw mitigating-config
  |     |   +-rw heartbeat-interval
  |     |   +-rw missing-hb-allowed
  |     |   +-rw max-retransmit
  |     |     +-rw heartbeat-interval
  |     |     +-rw missing-hb-allowed
  |     |     +-rw max-retransmit
  |     |     +-rw ack-timeout
  |     |     +-rw ack-random-factor
  |     +-rw idle-config
  |     |   +-rw heartbeat-interval
  |     |   +-rw missing-hb-allowed
  |     |   +-rw max-retransmit
  |     |   +-rw ack-timeout
  |     |   +-rw ack-random-factor

A maximum number of missing heartbeats is allowed.

HBs can be disabled.
DOTS Session Configuration

Flexible Retry Configuration for Unreliable Transports (1)

```
+--:(signal-config)
  |    +--rw sid uint32
  |    +--rw mitigating-config
  |    |    +--rw heartbeat-interval
  |    |    +--rw missing-hb-allowed
  |    |    +--rw max-retransmit
  |    |    +--rw ack-timeout
  |    |    +--rw ack-random-factor
  |    +--rw idle-config
  |        +--rw heartbeat-interval
  |        +--rw missing-hb-allowed
  |        +--rw max-retransmit
  |        +--rw ack-timeout
  |        +--rw ack-random-factor
```

![Diagram showing heartbeat intervals](image)

- **HB**: Heartbeat
- **T0**: Initial Time
- **T0 + interval**: First heartbeat
- **T0 + 2*interval**: Second heartbeat
- **T0 + 3*interval**: Third heartbeat

**No Loss**

- e.g., Set to 5
- e.g., Set to 3
DOTS Session Configuration

Flexible Retry Configuration for Unreliable Transports (2)

```
+--:(signal-config)
    |  +--rw sid         uint32
    |  +--rw mitigating-config
    |      |  +--rw heartbeat-interval
    |      |  +--rw missing-hb-allowed
    |      |  +--rw max-retransmit
    |      |  +--rw ack-timeout
    |      |  +--rw ack-random-factor
    |  +--rw idle-config
    |      |  +--rw heartbeat-interval
    |      |  +--rw missing-hb-allowed
    |      |  +--rw max-retransmit
    |      |  +--rw ack-timeout
    |      |  +--rw ack-random-factor
```

- **heartbeat-interval**: Set to 45 seconds.
- **missing-hb-allowed**: Set to 5.
- **max-retransmit**: Set to 3.

**MUST NOT transmit a HB while waiting for the previous HB.**

![Diagram showing heartbeat intervals and retransmissions](image-url)
DOTS Session Configuration

Flexible Retry Configuration for Unreliable Transports (3)

```plaintext
+- -:(signal-config)
  |  +--rw sid uint32
  |  +--rw mitigating-config
  |  |  +--rw heartbeat-interval
  |  |  +--rw missing-hb-allowed
  |  |  +--rw max-retransmit
  |  |  +--rw ack-timeout
  |  |  +--rw ack-random-factor
  |  +--rw idle-config
  |  |  +--rw heartbeat-interval
  |  |  +--rw missing-hb-allowed
  |  |  +--rw max-retransmit
  |  |  +--rw ack-timeout
  |  |  +--rw ack-random-factor
```

Loss

1st HB | 2nd HB | 3rd HB | 4th HB
---|---|---|---
TO | retrans. | retrans. | retrans. | retrans.

- Loss: Set to 1
- mitigating-config: Set to 15
DOTS Session Configuration

Flexible Retry Configuration for Unreliable Transports (4)

```plaintext
+-+(signal-config)
 |   +-rw sid           uint32
 |     +-rw mitigating-config
 |       |   +-rw heartbeat-interval
 |       |   +-rw missing-hb-allowed
 |       |   +-rw max-retransmit
 |       |   +-rw ack-timeout
 |       |   +-rw ack-random-factor
 |     +-rw idle-config
 |       |   +-rw heartbeat-interval
 |       |   +-rw missing-hb-allowed
 |       |   +-rw max-retransmit
 |       |   +-rw ack-timeout
 |       |   +-rw ack-random-factor
```

- **Loss**: Example values for different configurations.
  - 1st HB: Set to 7
  - 2nd HB: Set to 1
  - 3rd HB: Set to 10
  - 4th HB: Set to 1.5

- **Diagram**:
  - 1st HB: 45s
  - 2nd HB: 30s
  - 3rd HB: 30s
  - 4th HB: 30s
  - T0: 15s retrans.
DOTS Session Configuration

Flexible Retry Configuration for Unreliable Transports

NEW text (-36):

The specification allows for a flexible retry configuration when an unreliable transport is in use. For example, a server may be tweaked to return a lower 'missing-hb-allowed' (e.g., 5) value but delegate the retransmission to the underlying CoAP library by setting 'max-retransmit' to a high value (e.g., 3). The server may also be configured to return a 'max-retransmit' set to '1' and higher 'missing-hb-allowed' value (e.g., 15).
DOTS Session Configuration
Reliable Transport

```
+---:(signal-config)
    |   +---rw sid                        uint32
    |   +---rw mitigating-config
    |     |   +---rw heartbeat-interval
    |     |     +---rw missing-hb-allowed
    |     |     +---rw max-retransmit
    |     |     +---rw ack-timeout
    |     |     +---rw ack-random-factor
    |   +---rw idle-config
    |     +---rw heartbeat-interval
    |     +---rw missing-hb-allowed
    |     +---rw max-retransmit
    |     +---rw ack-timeout
    |     +---rw ack-random-factor
```

Since the underlying TCP connection provides retransmissions, HBs can be disabled.
When the DOTS signal channel is established over a reliable transport (e.g., TCP), there is no need for the reliability mechanisms provided by CoAP over UDP since the underlying TCP connection provides retransmissions and deduplication [RFC8323]. As a reminder, CoAP over reliable transports does not support Confirmable or Non-confirmable message types. As such, the transmission-related parameters (missing hb allowed and acceptable signal loss ratio) are negotiated only for DOTS over unreliable transports.
DOTS Session Configuration
Reliable Transport

Echoing this text from RFC8323:

“CoAP over reliable transports does not support
Confirmable or Non-confirmable message types”

NEW text (-36):

When the DOTS signal channel is established over a reliable transport (e.g., TCP), there is no need for the reliability mechanisms provided by CoAP over UDP since the underlying TCP connection provides retransmissions and deduplication [RFC8323]. As a reminder, CoAP over reliable transports does not support Confirmable or Non-confirmable message types. As such, the transmission-related parameters (missing-hb-allowed and acceptable signal loss ratio) are negotiated only for DOTS over unreliable transports.
When the DOTS signal channel is established over a reliable transport (e.g., TCP), there is no need for the reliability mechanisms provided by CoAP over UDP since the underlying TCP connection provides retransmissions and deduplication [RFC8323]. As a reminder, CoAP over reliable transports does not support Confirmable or Non-confirmable message types. As such, the transmission-related parameters (missing-hb-allowed and acceptable signal loss ratio) are negotiated only for DOTS over unreliable transports.
Which Heartbeat for DOTS?

- DOTS over reliable transports
  - Connection Health based on Ping/Pong messages defined in RFC8323

- DOTS over unreliable transports
  - Relies upon CoAP Ping: Empty Confirmable message and the peer DOTS agent will respond by sending a Reset message
When to Declare Failure During an Attack? Reliable Transport

• The DOTS application *has control over the Pong timeout*; hence when to declare failure based on heartbeat-interval

Quoting RFC8323:

“the present specification does not define any specific maximum time that the sender of a Ping message has to allow when waiting for a Pong reply. Any limitations on patience for this reply are a matter of the application making use of these messages, as is any approach to recover from a failure to respond in time.”
When to Declare Failure During an Attack? Unreliable Transport

• The DOTS application *owns the decision when to declare failure* based on various parameters, e.g., missing-hb-allowed, attack mitigation, etc.

![Diagram showing DOTS client keeping on using the current DOTS signal channel session even after a maximum 'missing-hb-allowed' threshold and trying to resume the (D)TLS session. The network link is saturated due to attack traffic, leading to the DOTS server concluding the session is disconnected.]

The DOTS client **keeps on** using the current DOTS signal channel session even after a maximum 'missing-hb-allowed' threshold and try to resume the (D)TLS session.

If it does not receive any traffic from the DOTS client and maximum 'missing-hb-allowed' threshold is reached, the DOTS server concludes the session is disconnected.
An Alternative Approach? (1)

• A Proposal from Mirja Kuehlewind
  – “I believe there are flaws in the design. First it’s a layer violation, but if more an idealistic concern but usually designing in layers is a good approach. But more importantly, you end up with unfrequent messages which may still terminate the connection at some point, while what you want is to simply send messages frequently in an unreliable fashion but a low rate until the attack is over”

That is, use non-confirmable messages
An Alternative Approach? (2)

• Requires the DOTS server to send **non-confirmable messages**, but
  – Given that DOTS client is a CoAP Client and DOTS server is a CoAP Server
  – And Section 1.2 of RFC7252 indicates:
    • `Client`: The originating endpoint of a request; the destination endpoint of a response.
    • `Server`: The destination endpoint of a request; the originating endpoint of a response.
    • `Empty Message`: A message with a Code of 0.00; *neither a request nor a response*.
  – The server can **only send Empty requests**

But, is it possible to send non-confirmable empty requests?
An Alternative Approach? (3)

• Section 4.3 in RFC7252:
  – A Non-confirmable message always carries either a request or response and MUST NOT be Empty
Summary

• The intended heartbeat functionality is naturally provided by existing CoAP messages
  – Informed WG decision (next slide, for example)
  – Implemented
  – Tested with interoperable implementations
  – The DOTS application has the full control on the intended functionality

• The proposed alternative approach violates RFC7252

• Any objection with the assessment?

• What’s Next for handling Mirja’s pending DISCUSS point?
  – Report to Mirja the decision of the WG
  – Ben/Chairs?
From an Email sent by Med to the List (10/2017)

• [https://mailarchive.ietf.org/arch/msg/dots/3mL8TjLlipWU8YOd6FRwWqd9vj8](https://mailarchive.ietf.org/arch/msg/dots/3mL8TjLlipWU8YOd6FRwWqd9vj8)
  – “Should we rely solely on the missing-hb-allowed to detect a session problem?
  – Should we get rid of missing-hb-allowed, but rely on the retransmission to declare failure or not?
  – What is the advantage of cumulating both missing-hb-allowed and the retransmission procedure to declare a channel out?”