Polka attack simulations

Henrique Coutinho Layber[1]

June 27 2024

[1]henrique.layber@edu.ufes.br

Baseline measurements

Before we start attacking the network and comparing digests, we need to establish a baseline for correct operation.

Baseline measurements

Before we start attacking the network and comparing digests, we need to establish a baseline for correct operation.

For that, we will ping host `h10` from host `h1` and measure the packet through all controlled interfaces. Always using the hardcoded timestamp `0x61E8D6E7` unless said otherwise.

In a linear topology passing though 10 core switches, this generates 88 packets:

In a linear topology passing though 10 core switches, this generates 88 packets:

- one digest for entrance edge + 10 core switches (11)

In a linear topology passing though 10 core switches, this generates 88 packets:

- one digest for entrance edge + 10 core switches (11)
- two captures per switch (one for each interface) (22)

In a linear topology passing though 10 core switches, this generates 88 packets:

- one digest for entrance edge + 10 core switches (11)
- two captures per switch (one for each interface) (22)
- two time that for the reply (44)

In a linear topology passing though 10 core switches, this generates 88 packets:

- one digest for entrance edge + 10 core switches (11)
- two captures per switch (one for each interface) (22)
- two time that for the reply (44)
- two times that for the ping and reply the other way (88)

So in a trace, we expect the packets to appear in the following order:

A =

```
0x61E8D6E7 e1
0xAE91434C s1
0x08C97F5F s2
0xEFF1AAD2 s3
0x08040C89 s4
0xAA99AE2E s5
0x7669685E s6
0x03E1E388 s7
0x2138FFD3 s8
0x1EF2CBBE s9
0x99C5FE05 s10
```

B =

```
0x61E8D6E7 e10
0xCFFABC9F s10
0x69409E70 s9
0xF3E992E0 s8
0x8DDE192B s7
0x92B098FA s6
0x1115A62C s5
0x41E1B5E0 s4
0x227F0B72 s3
0x82FC6346 s2
0xD01E3E0F s1
```

$$\text{Order} = \text{dup}(A), \text{dup}(B), \text{dup}(B), \text{dup}(A)$$

$$\text{dup}([a, b, c]) = [a, a, b, b, c, c]$$

So in a trace, we expect the packets to appear in the following order:

A =

```
0x61E8D6E7 e1
0xAE91434C s1
0x08C97F5F s2
0xEFF1AAD2 s3
0x08040C89 s4
0xAA99AE2E s5
0x7669685E s6
0x03E1E388 s7
0x2138FFD3 s8
0x1EF2CBBE s9
0x99C5FE05 s10
```

B =

```
0x61E8D6E7 e10
0xCFFABC9F s10
0x69409E70 s9
0xF3E992E0 s8
0x8DDE192B s7
0x92B098FA s6
0x1115A62C s5
0x41E1B5E0 s4
0x227F0B72 s3
0x82FC6346 s2
0xD01E3E0F s1
```

$$\text{Order} = \text{dup}(A), \text{dup}(B), \text{dup}(B), \text{dup}(A)$$

$$\text{dup}([a, b, c]) = [a, a, b, b, c, c]$$

For brevity, dup will be ommited since it's information is irrelevant, but it will be checked regardless

Ping `h1` → `h10`

name	e1
digest	0x61E8D6E7
expected	0x61E8D6E7

name	s1
digest	0xAE91434C
expected	0xAE91434C

name	s2
digest	0x08C97F5F
expected	0x08C97F5F

name	s3
digest	0xEFF1AAD2
expected	0xEFF1AAD2

name	s4
digest	0x08040C89
expected	0x08040C89

name	s5
digest	0xAA99AE2E
expected	0xAA99AE2E

name	s6
digest	0x7669685E
expected	0x7669685E

name	s7
digest	0x03E1E388
expected	0x03E1E388

name	s8
digest	0x2138FFD3
expected	0x2138FFD3

name	s9
digest	0x1EF2CBBE
expected	0x1EF2CBBE

name	s10
digest	0x99C5FE05
expected	0x99C5FE05

Ping `h1` → `h10` Reply

name	e10
digest	0x61E8D6E7
expected	0x61E8D6E7

name	s10
digest	0xCFFABC9F
expected	0xCFFABC9F

name	s9
digest	0x69409E70
expected	0x69409E70

name	s8
digest	0xF3E992E0
expected	0xF3E992E0

name	s7
digest	0x8DDE192B
expected	0x8DDE192B

name	s6
digest	0x92B098FA
expected	0x92B098FA

name	s5
digest	0x1115A62C
expected	0x1115A62C

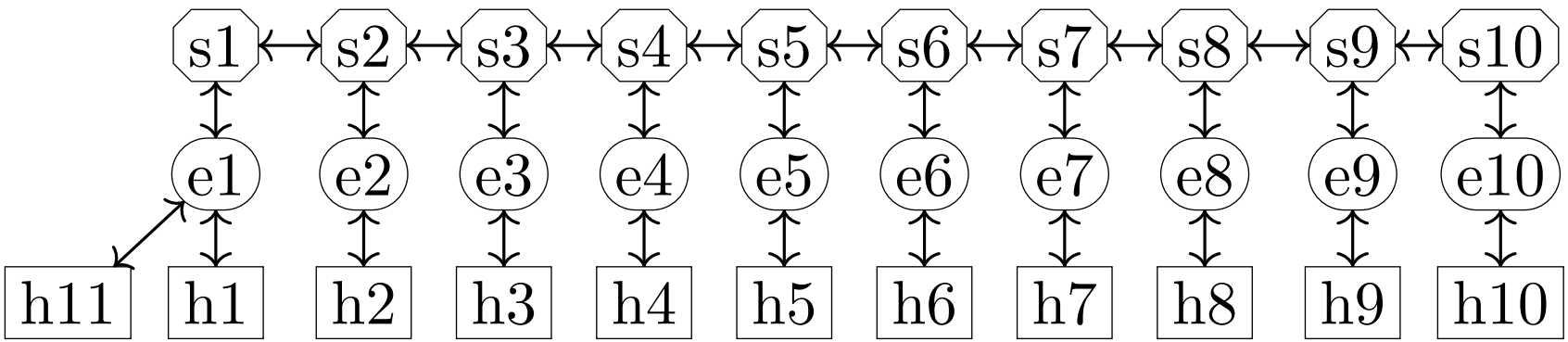
name	s4
digest	0x41E1B5E0
expected	0x41E1B5E0

name	s3
digest	0x227F0B72
expected	0x227F0B72

name	s2
digest	0x82FC6346
expected	0x82FC6346

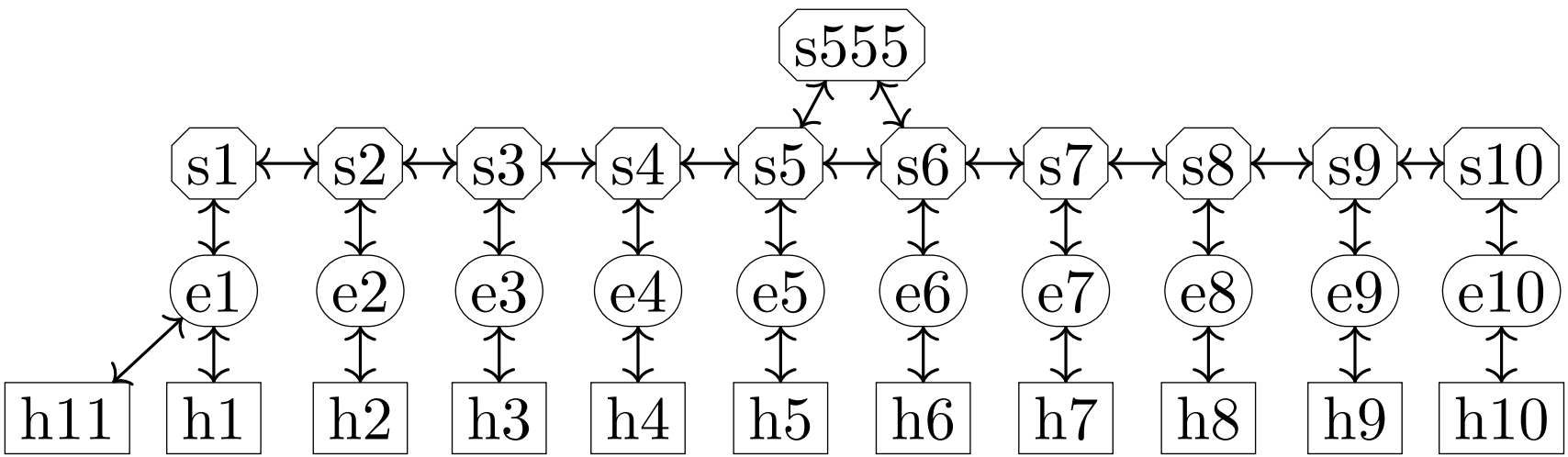
name	s1
digest	0xD01E3E0F
expected	0xD01E3E0F

Base topology



$s\{n\} \leftrightarrow s\{n+1\}$ connects through port 2:1

Attacked topology



s5 $\leftrightarrow$ s555 connects through port 2:0

s555 $\leftrightarrow$ s6 connects through port 1:2

s5 $\leftrightarrow$ s6 connects through port 3:3

Ping `h1` → `h10`

name	e1
digest	0x61E8D6E7
expected	0x61E8D6E7

name	s1
digest	0xAE91434C
expected	0xAE91434C

name	s2
digest	0x08C97F5F
expected	0x08C97F5F

name	s3
digest	0xEFF1AAD2
expected	0xEFF1AAD2

name	s4
digest	0x08040C89
expected	0x08040C89

name	s5
digest	0xAA99AE2E
expected	0xAA99AE2E

name	attacer
digest	0xC450DD37
expected	0x7669685E

name	s6
digest	0x5397C754
expected	0x7669685E

name	s7
digest	0xE21DAB66
expected	0x03E1E388

name	s8
digest	0x8C375948
expected	0x2138FFD3

name	s9
digest	0x352F1CF8
expected	0x1EF2CBBE

name	s10
digest	0x3a61c724
expected	0x99C5FE05

Ping `h1` → `h10` Reply

name	e10
digest	0x61E8D6E7
expected	0x61E8D6E7

name	s10
digest	0xCFFABC9F
expected	0xCFFABC9F

name	s9
digest	0x69409E70
expected	0x69409E70

name	s8
digest	0xF3E992E0
expected	0xF3E992E0

name	s7
digest	0x8DDE192B
expected	0x8DDE192B

name	s6
digest	0x92B098FA
expected	0x92B098FA

name	s5
digest	0x1115A62C
expected	0x1115A62C

name	s4
digest	0x41E1B5E0
expected	0x41E1B5E0

name	s3
digest	0x227F0B72
expected	0x227F0B72

name	s2
digest	0x82FC6346
expected	0x82FC6346

name	s1
digest	0xD01E3E0F
expected	0xD01E3E0F

Ping `h10` → `h1`

name	e10
digest	0x61E8D6E7
expected	0x61E8D6E7

name	s10
digest	0xCFFABC9F
expected	0xCFFABC9F

name	s9
digest	0x69409E70
expected	0x69409E70

name	s8
digest	0xF3E992E0
expected	0xF3E992E0

name	s7
digest	0x8DDE192B
expected	0x8DDE192B

name	s6
digest	0x92B098FA
expected	0x92B098FA

name	s5
digest	0x1115A62C
expected	0x1115A62C

name	s4
digest	0x41E1B5E0
expected	0x41E1B5E0

name	s3
digest	0x227F0B72
expected	0x227F0B72

name	s2
digest	0x82FC6346
expected	0x82FC6346

name	s1
digest	0xD01E3E0F
expected	0xD01E3E0F

Ping `h10` → `h1` Reply

name	e10
digest	0x61E8D6E7
expected	0x61E8D6E7

name	s10
digest	0xAE91434C
expected	0xAE91434C

name	s9
digest	0x08C97F5F
expected	0x08C97F5F

name	s8
digest	0xEFF1AAD2
expected	0xEFF1AAD2

name	s7
digest	0x08040C89
expected	0x08040C89

name	s6
digest	0xAA99AE2E
expected	0xAA99AE2E

name	attacker
digest	0xC450DD37
expected	0x7669685E

name	s5
digest	0x5397C754
expected	0x7669685E

name	s4
digest	0xE21DAB66
expected	0x03E1E388

name	s3
digest	0x8C375948
expected	0x2138FFD3

name	s2
digest	0x352F1CF8
expected	0x1EF2CBBE

name	s1
digest	0x3A61C724
expected	0x99C5FE05