Lecture Outline

- Protocols for detecting manipulation
- How to think about “architecture”
  - In general systems terms
  - For security implications
- “Tussles” in architectures that affect multiple stakeholders
- Ethane: the good and the could-have-been-better

Detecting manipulation, con’t:

… by destroying information an attacker needs
… by creating information an attacker can’t destroy

Alice

Bob

Alice wants to pair with Bob via D-H exchange.

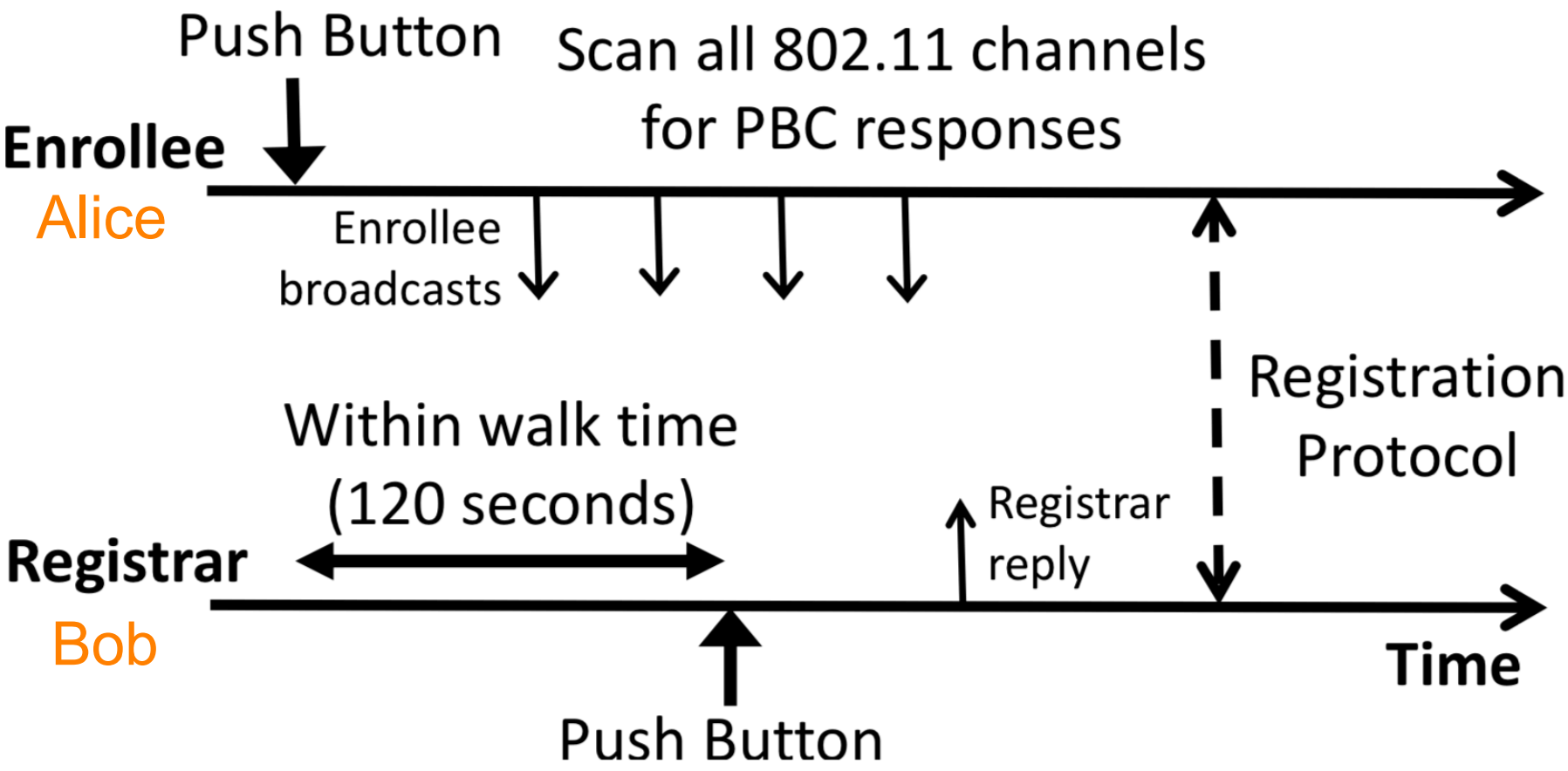
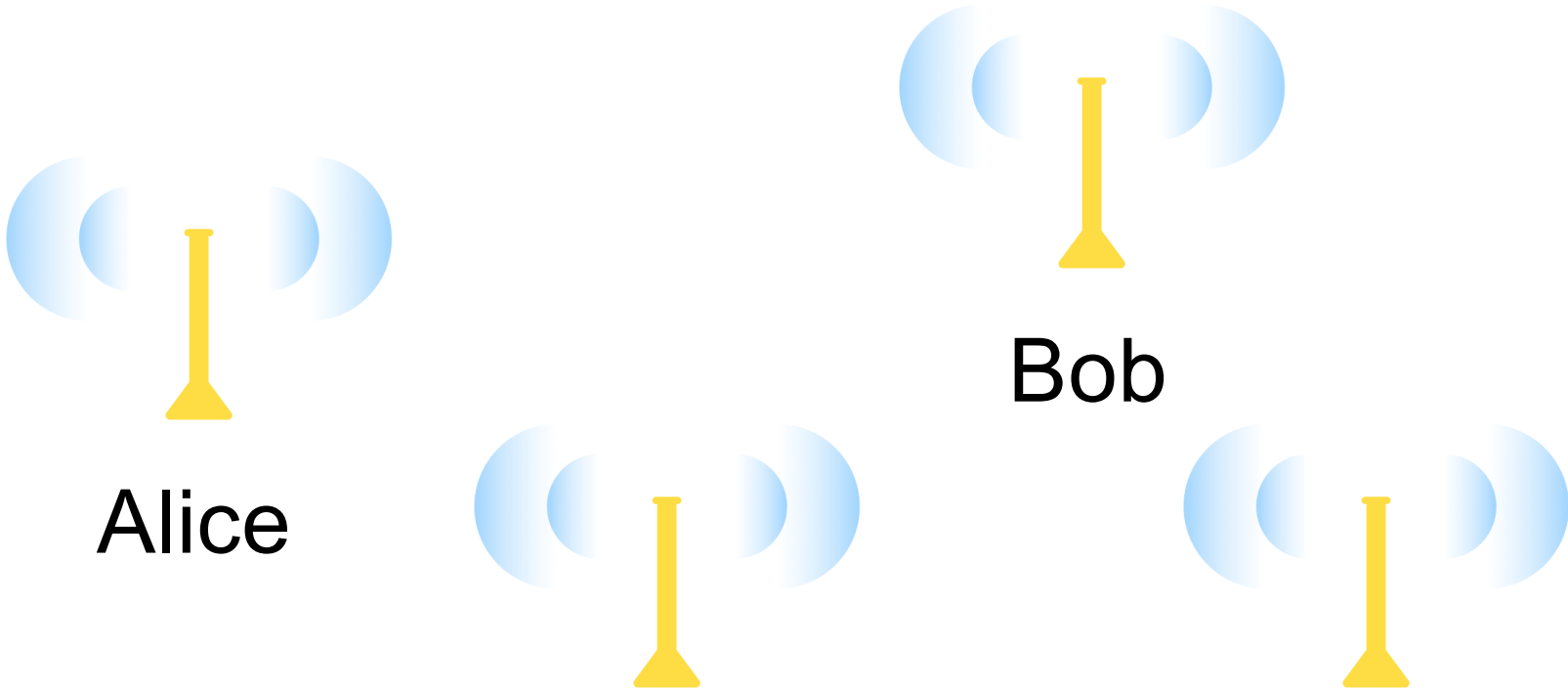


Figure 2: A timeline depicting the operation of Push Button Configuration (PBC) between an enrollee and a registrar.



Charlie is a benign third party.
Everyone can hear everyone else.
Including Mallory. Mallory can cheat.
Mallory wants Alice to mistakenly pair w/ Mallory.

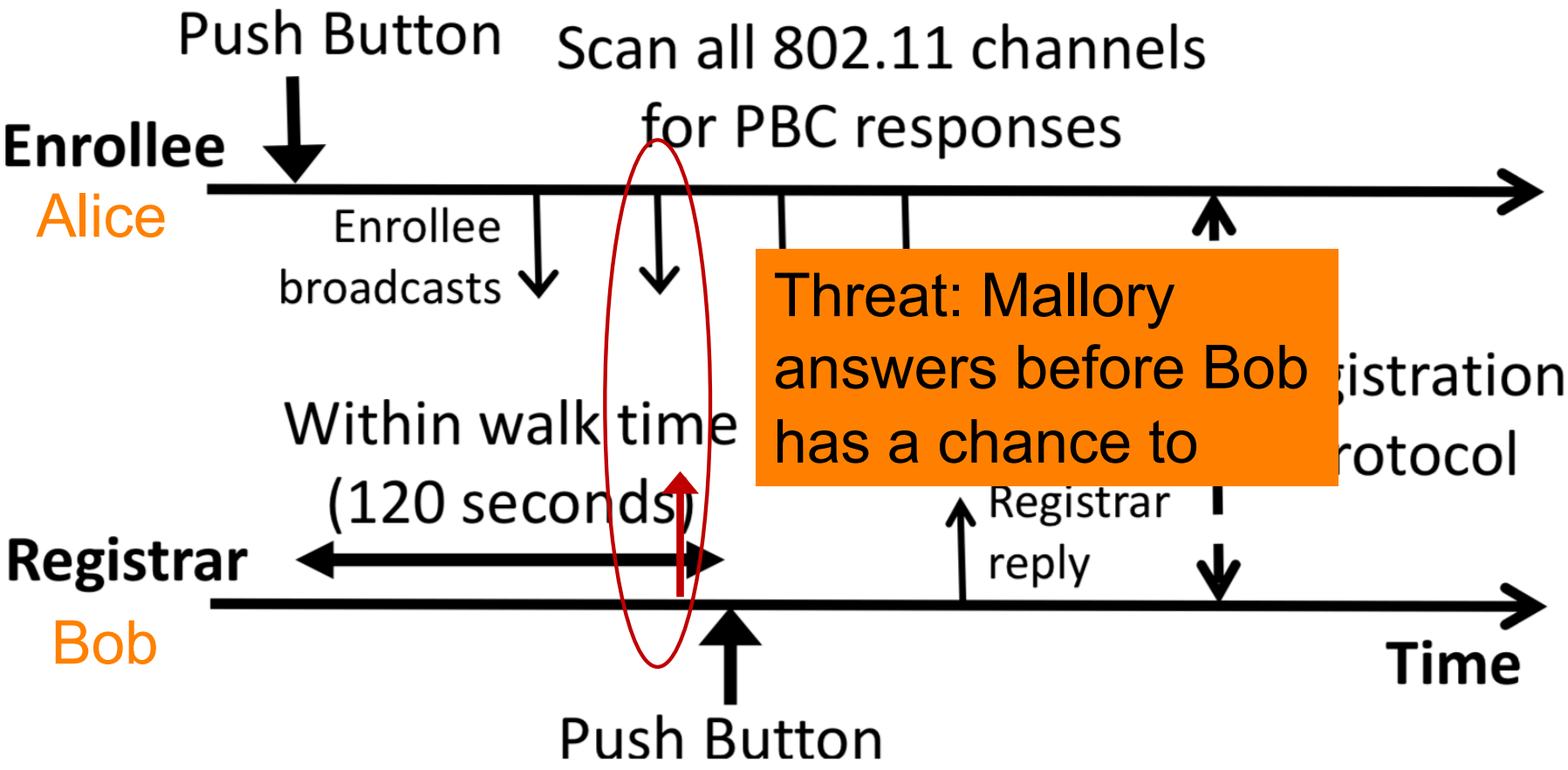
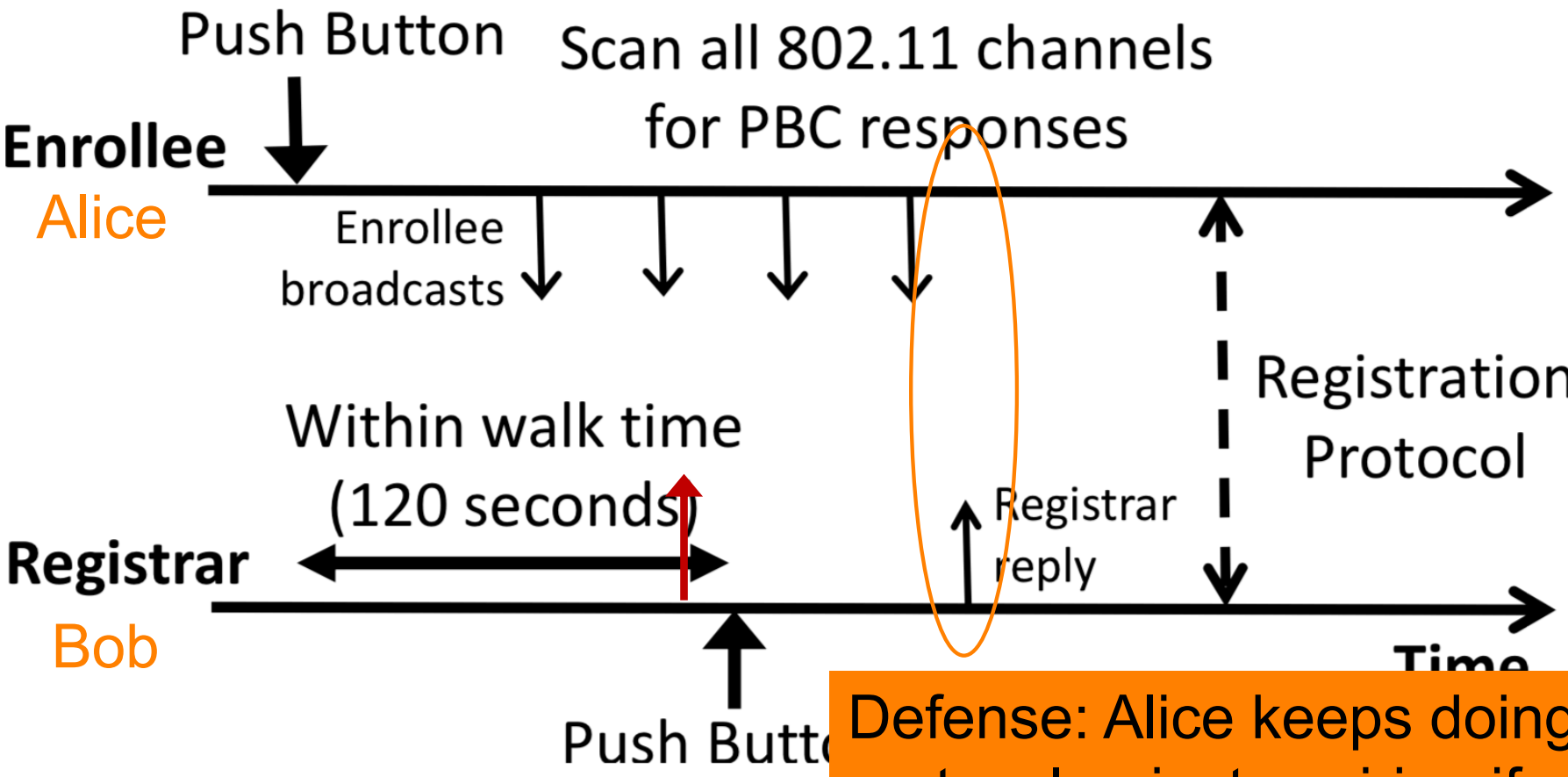


Figure 2: A timeline depicting the operation of Push Button Configuration (PBC) between an enrollee and a registrar.



Defense: Alice keeps doing protocol, rejects pairing if hears more than one

Figure 2: A timeline depicting th n-figuration (PBC) between an enrollee and a registrar.

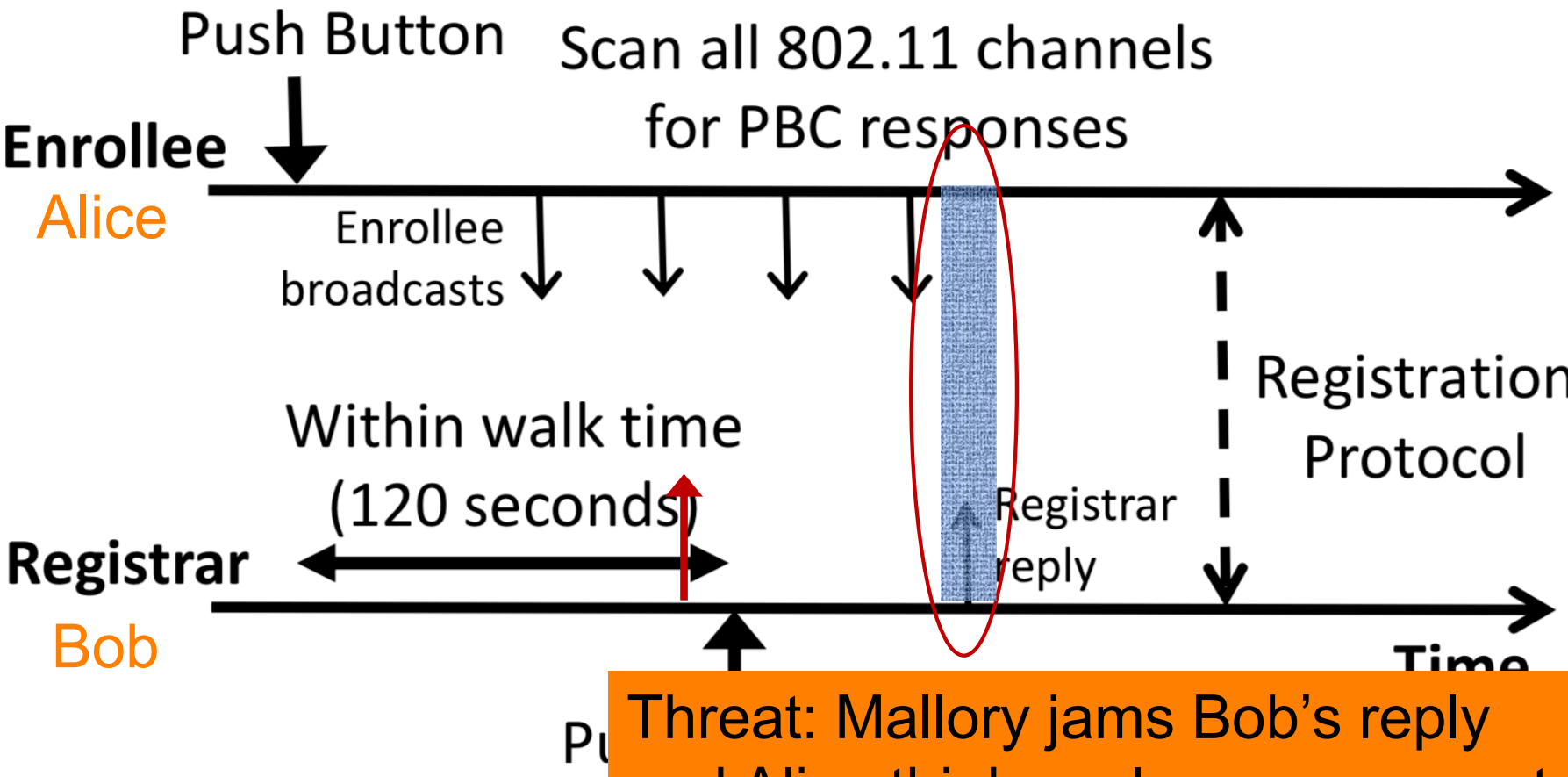
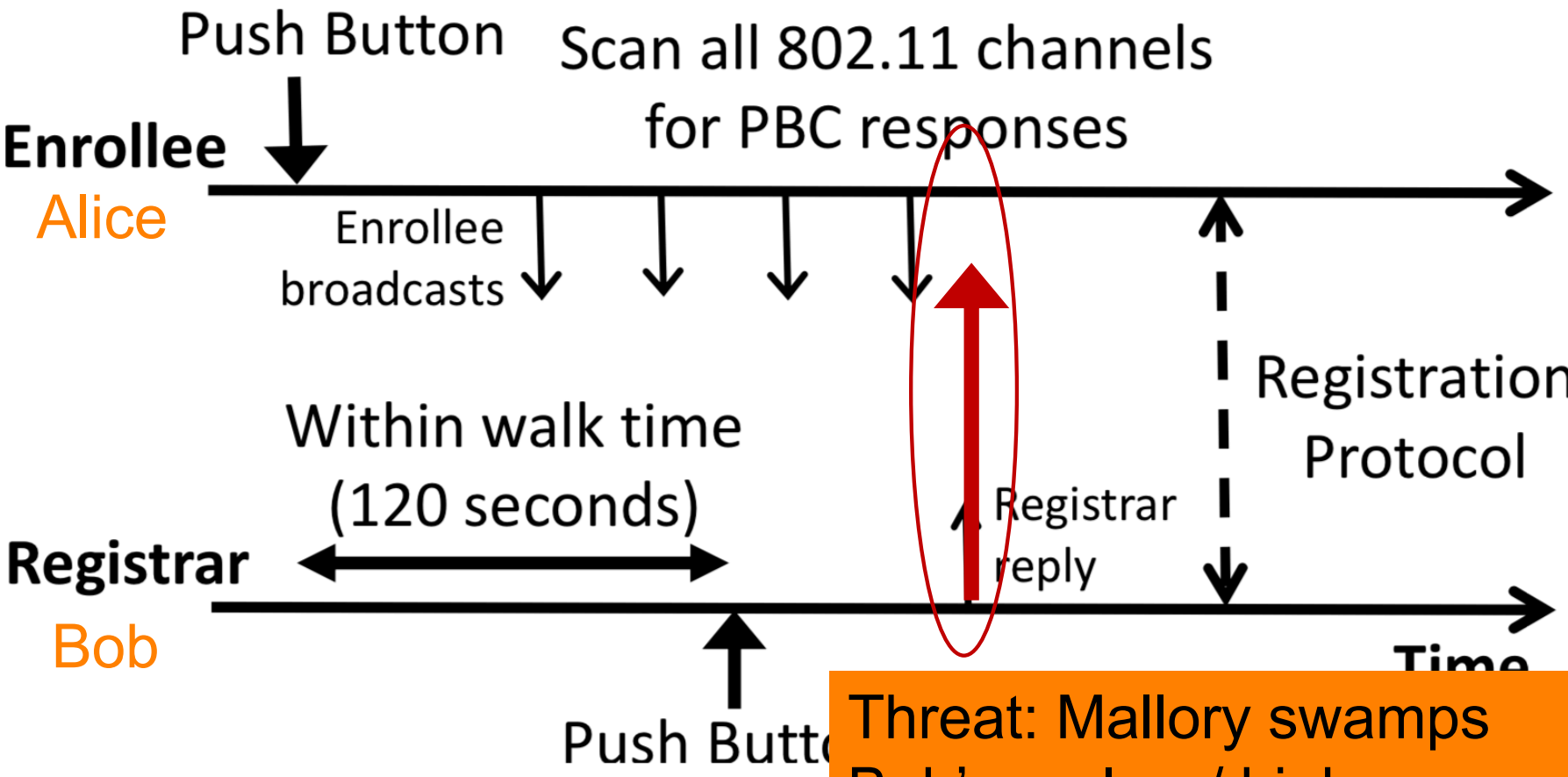


Figure 2: A timeline dep … figuration (PBC) between …

Threat: Mallory jams Bob's reply and Alice thinks only one was sent. This can happen for benign reasons (Charlie), so Alice doesn't know it's manipulation.

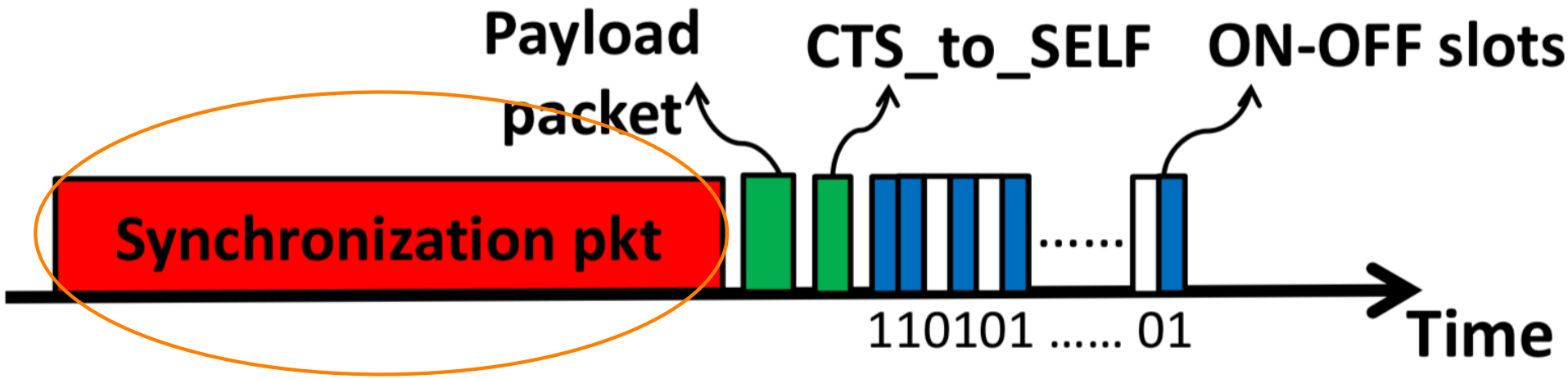


Threat: Mallory swamps Bob's reply w/ higher-powered responses

Figure 2: A timeline depicting th … n-figuration (PBC) between an enrollee and a registrar.

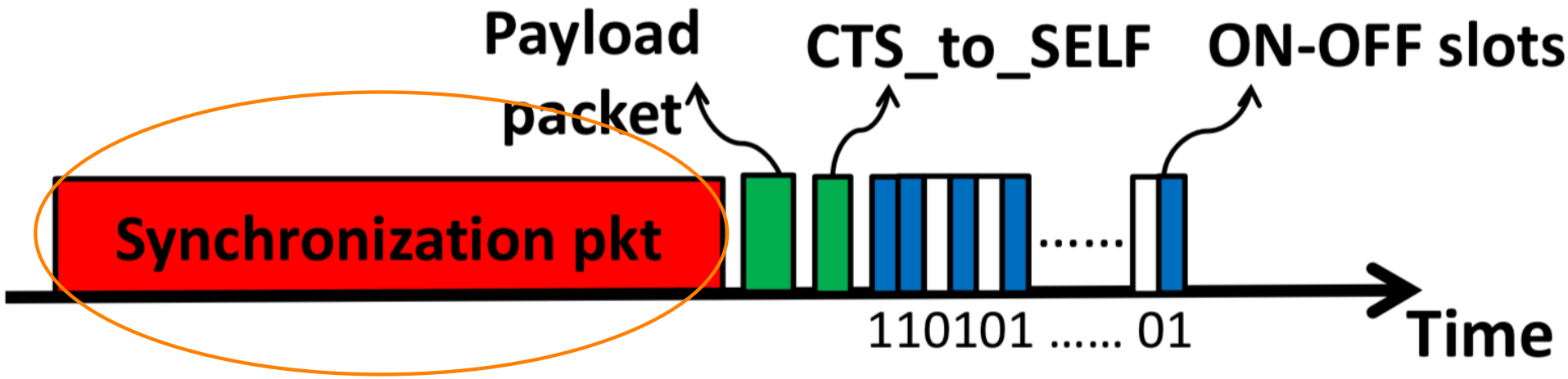
Goal: *tamper-evident* pairing
Alice can tell someone is messing with her attempt, and will try again later, until no evident tampering

Idea: extend protocol with extra packet slots, some of which **must** be *empty* (silent)



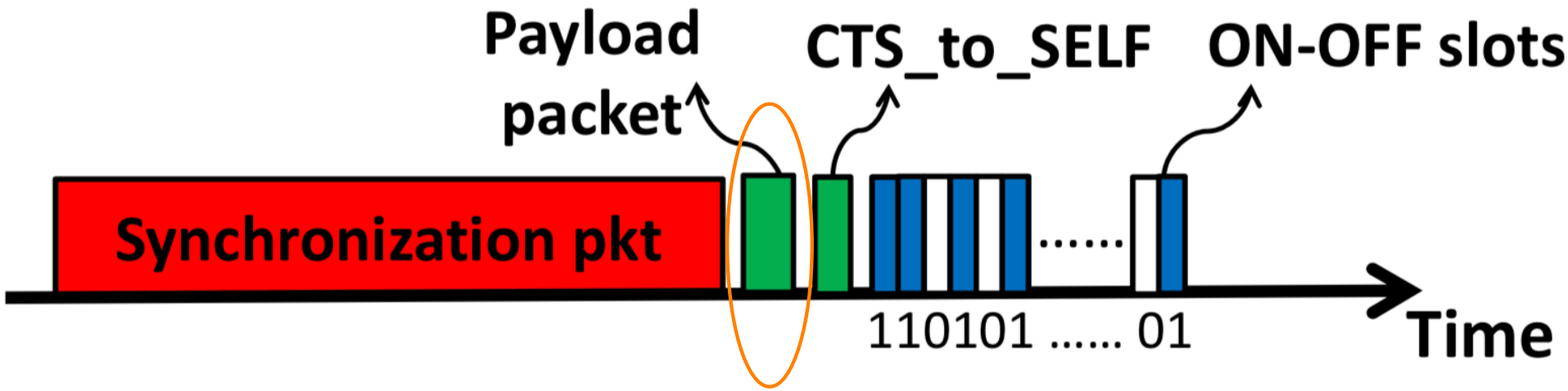
Long burst that ensures all legit sources will be quiet in the next slot

Idea: extend protocol with extra packet slots, some of which **must** be *empty* (silent)



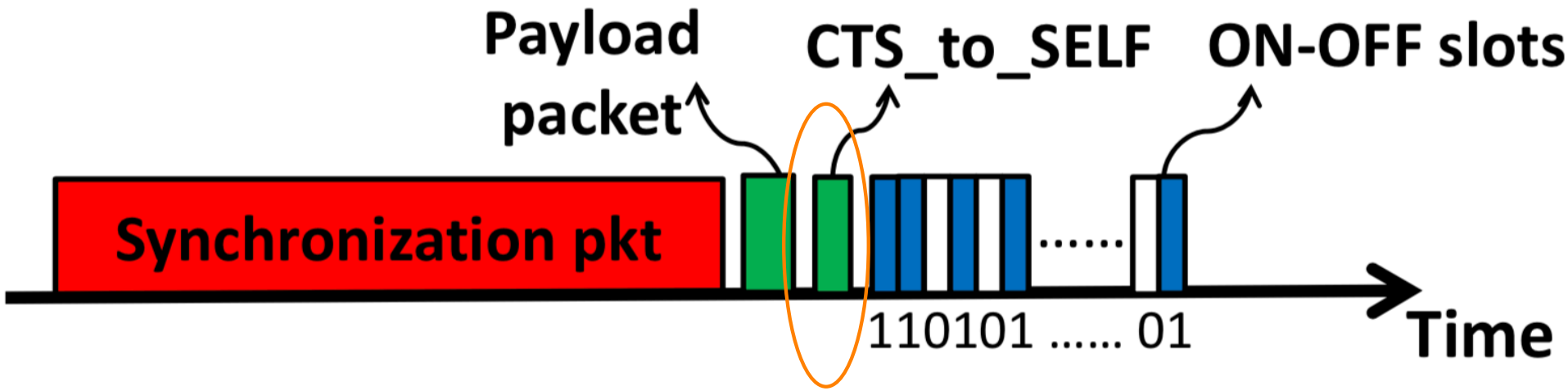
Upon seeing this, Bob knows the protocol's in effect

Idea: extend protocol with extra packet slots, some of which **must** be *empty* (silent)



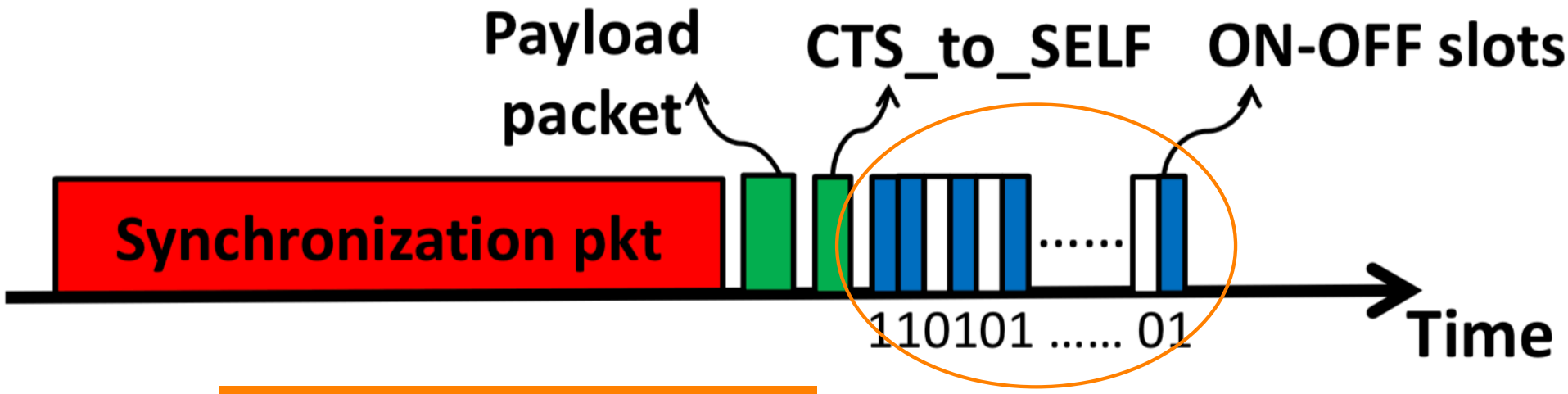
Alice's D-H data

Idea: extend protocol with extra packet slots, some of which **must** be *empty* (silent)



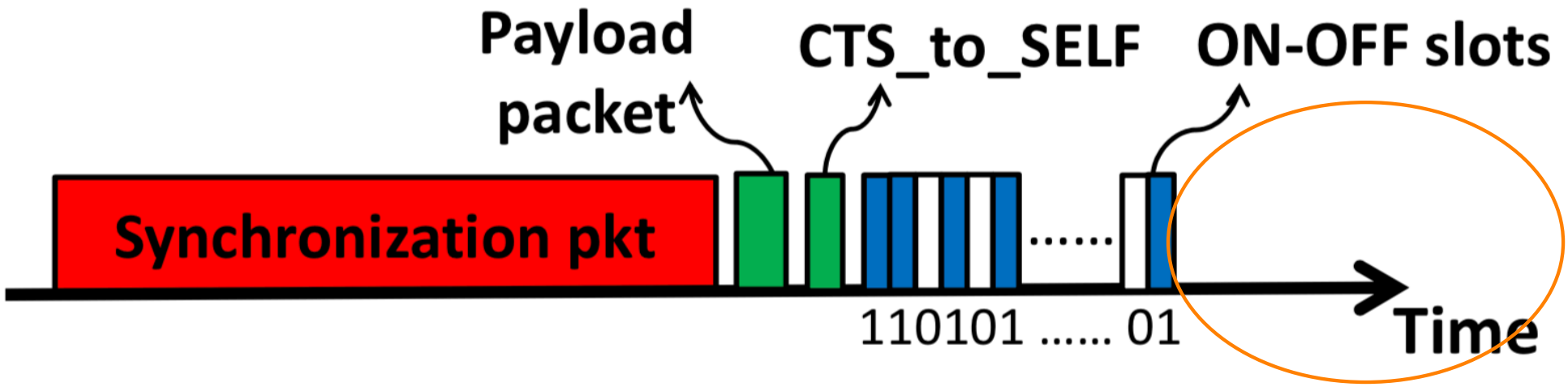
Alice *reserves* a bunch of packet slots

Idea: extend protocol with extra packet slots, some of which **must** be *empty* (silent)



Alice sends a *hash* of D-H data …
… *encoded* as
packet-sent = 1 bit,
no-packet-sent = 0 bit

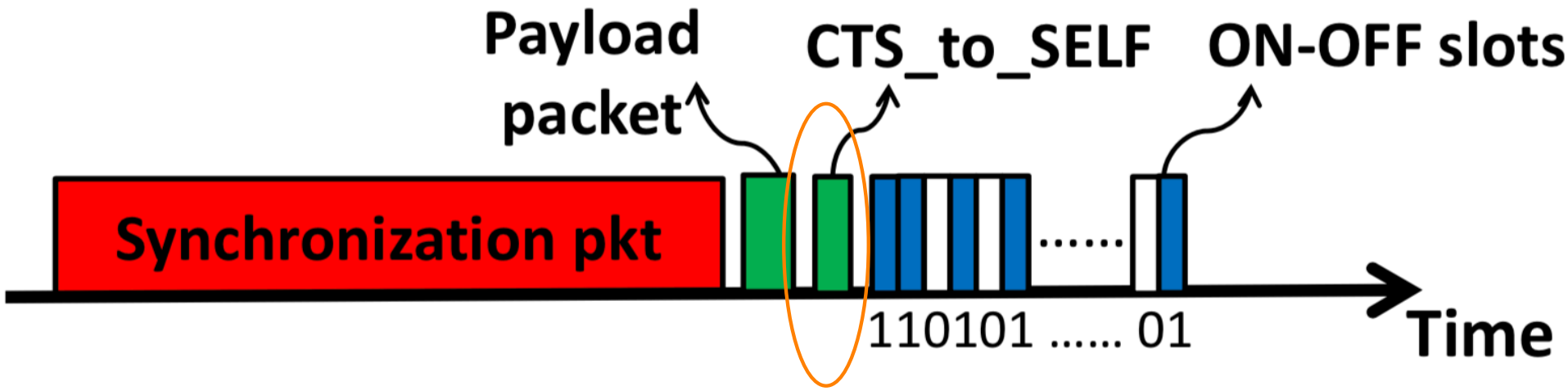
Idea: extend protocol with extra packet slots, some of which **must** be *empty* (silent)



Bob does the same for Bob's D-H data

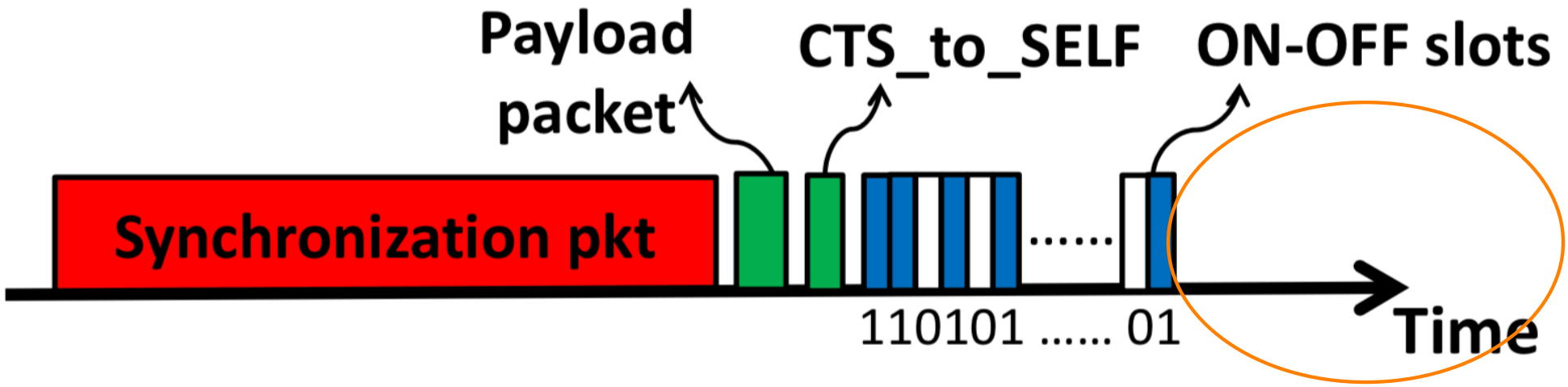
Threat: Mallory sends *early* and now jams Bob's reply so Alice thinks earlier one was the only one sent.
Can happen for benign reasons (Charlie), so Alice doesn't know it's manipulation.

?



Alice knows collision is violation of her slot reservation: **tampering**

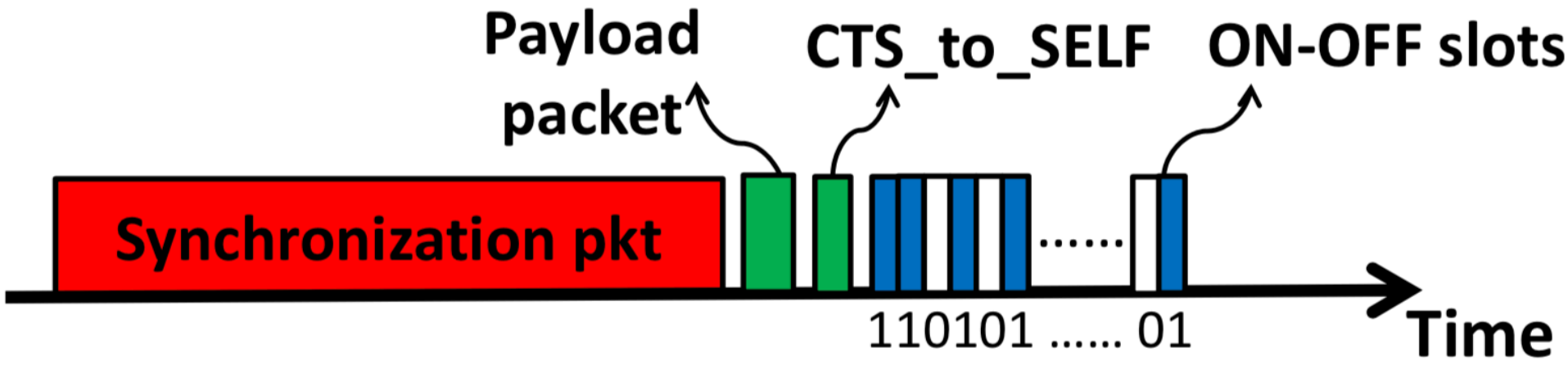
Threat: Mallory swamps Bob's reply w/ higher-powered responses ?



Bob will "step on" some of Mallory's 0-bit hash slots due to Bob's own hash having 1-bits in those slots …
Alice will see that hash doesn't match: **tampering**

New threat: Mallory precomputes D-H data w/ a *hash of nearly all 1-bits*

?



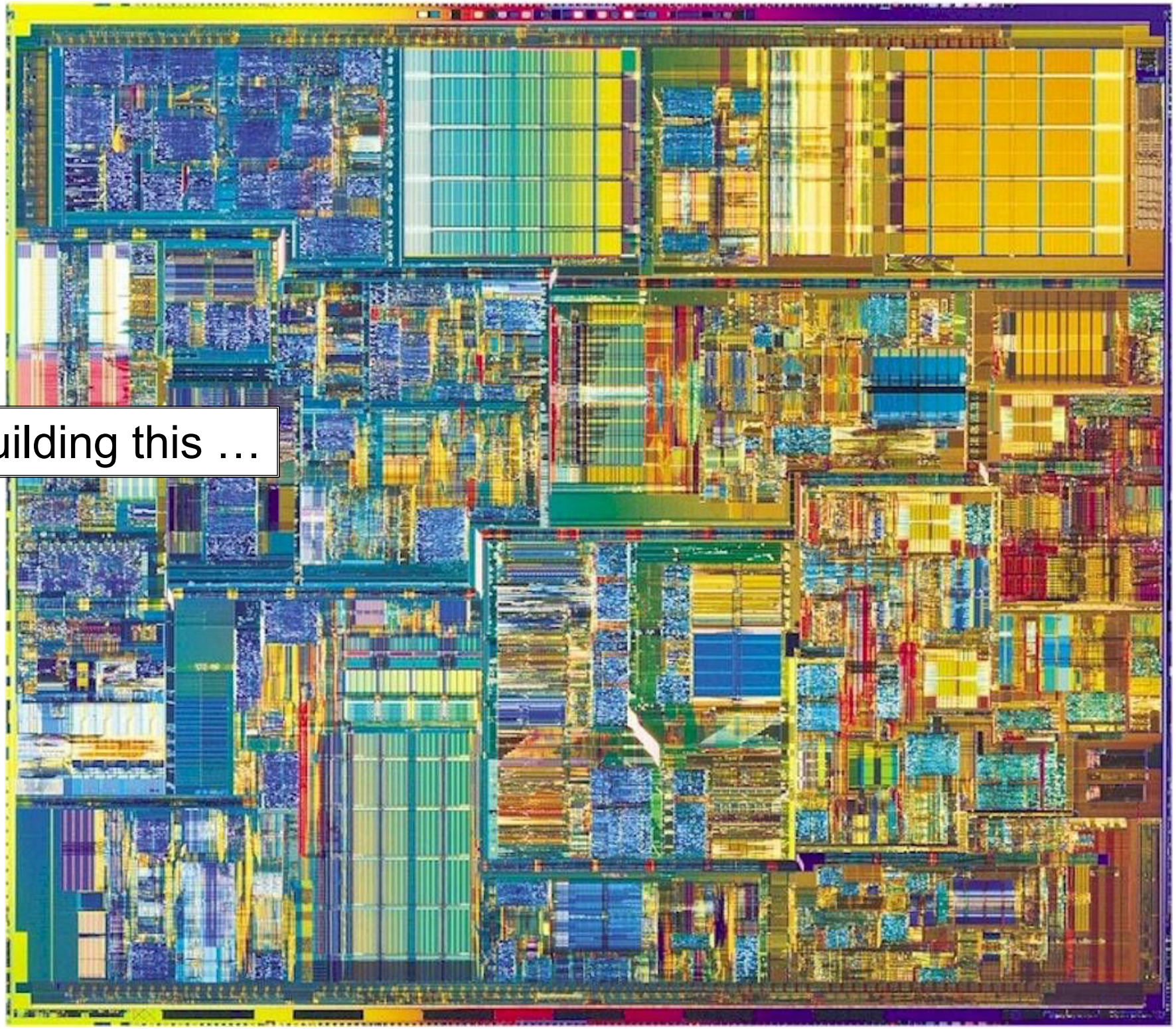
Solution: don't directly encode 0/1 bits

Instead: (something like) *Manchester encoding*:

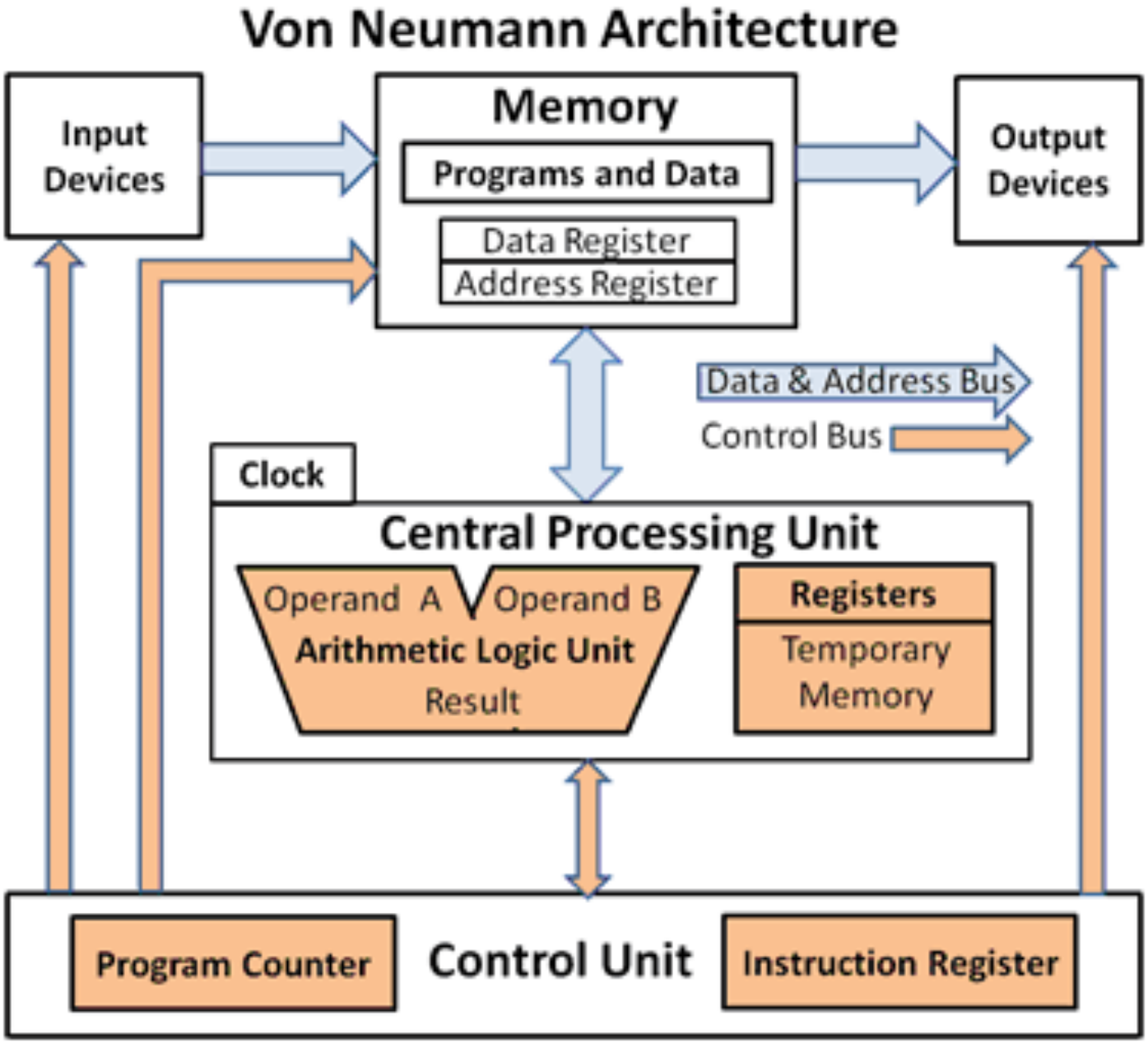
Encode **0** bit as **0** || **1**

Encode **1** bit as **1** || **0**

Q’s before moving on to Architecture?


Building this …

… takes high-level thinking like this:



Architecture

- Engineering = “obtaining predictable & desirable behavior”
- To engineer complex systems requires designing *overarching structure*
  - Abstractions
  - Placement of functionality
  - State management
  - Naming
- Good architecture aligns mechanism with functionality/enforcement

Architecture, con't

- High-level/abstract nature can make it hard to "get"
- Has a flavor of "think outside the box": in fact, "design the box"
- In security, we're used to intensely *scrutinizing the box*
  - Rather than stepping back to consider its design properties / how it could have been different

⇒ Ask questions!

Abstractions?

Policy-neutral, strongly-typed asynchronous *events*

Employ *filtering* and *reduction* to balance processing load

Connection-oriented (e.g. TCP bytestreams emphasized over packets)

Self-describing log files *linked together* by opaque identifiers

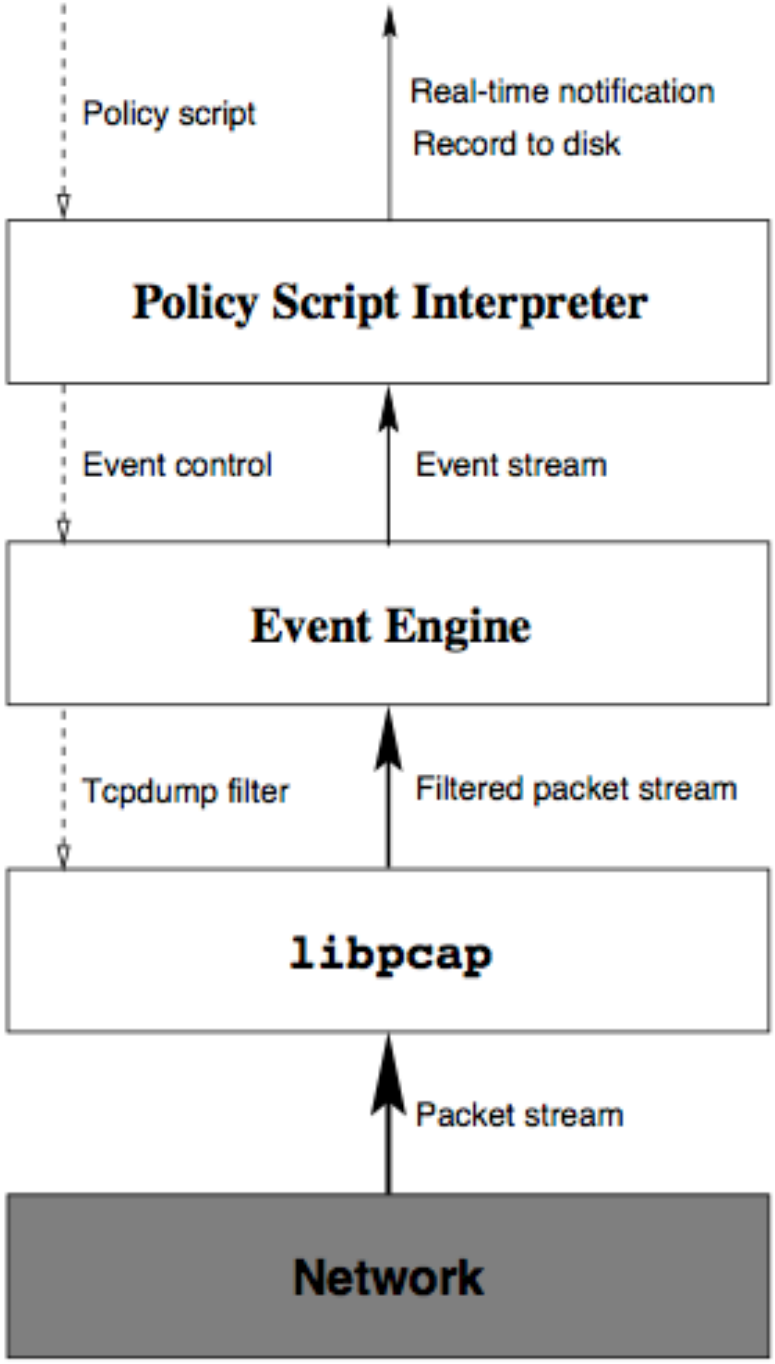


Figure 1: Structure of the Bro system

```
#separator \x09
#set_separator  ,
#empty_field    (empty)
#unset_field    -
#path   conn
#open   2016-07-13-16-16-57
#fields ts      uid     id.orig_h       id.orig_p       id.resp_h
        id.resp_p       proto   service duration        orig_bytes      resp_bytes
        conn_state      local_orig      local_resp      missed_bytes
        history orig_pkts       orig_ip_bytes   resp_pkts
        resp_ip_bytes   tunnel_parents
#types  time    string  addr    port    addr    port    enum    string
        interval        count   count   string  bool    bool    count   string
        count   count   count   count   set[string]
1324071333.493287       CHhAvVGS1DHFjwGM9       192.168.1.79    51880
        131.159.21.1    22      tcp     ssh     6.159326        2669    2501    SF
        -       -       0       ShAdDaFf        25      3981    20      3549    -
1409516196.337184       ClEkJM2Vm5giqnMf4h      10.0.0.18       40184
        128.2.6.88      41644   tcp     ssh     2.079071        3813    3633    SF
        -       -       0       ShADadFf        22      4965    26      5017    -
1419870189.485611       C4J4Th3PJpwUYZZ6gc      192.168.2.1     57189
        192.168.2.158   22      tcp     ssh     6.641754        5253    3489    SF
        -       -       0       ShADadFf        38      7241    29      5005    -
1419870206.101883       CtPZjS20MLrsMUOJi2      192.168.2.1     57191
        192.168.2.158   22      tcp     ssh     3.862198        576     813     SF
        -       -       0       ShAdDaFf        23      1784    16      1653    -
```

```
#separator \x09
#set_separator  ,
#empty_field    (empty)
#unset_field    -
#path   ssh
#open   2018-10-23-15-34-42
#fields ts      uid     id.orig_h       id.orig_p       id.resp_h
        id.resp_p       version auth_success    auth_attempts   direction
        client  server  cipher_alg      mac_alg compression_alg kex_alg
        host_key_alg    host_key
#types  time    string  addr    port    addr    port    count   bool
        count   enum    string  string  string  string  string  string
        string  string
1324071333.792887       CHhAvVGS1DHFjwGM9       192.168.1.79    51880
        131.159.21.1    22      2       -       0       -       SSH-2.0-
OpenSSH_5.9     SSH-2.0-OpenSSH_5.8     aes128-ctr      hmac-md5
        zlib@openssh.com ecdh-sha2-nistp256     ecdsa-sha2-nistp256
        a7:26:62:3f:75:1f:33:8a:f3:32:90:8b:73:fd:2c:83
1409516196.413240       ClEkJM2Vm5giqnMf4h      10.0.0.18       40184
        128.2.6.88      41644   2       T       1       -       SSH-2.0-
OpenSSH_6.6     SSH-2.0-OpenSSH_5.9p1 Debian-5ubuntu1.1  aes128-ctr
        hmac-md5 none   ecdh-sha2-nistp256      ssh-rsa
        8a:8d:55:28:1e:71:04:99:94:43:22:89:e5:ff:e9:03
1419870189.489202       C4J4Th3PJpwUYZZ6gc      192.168.2.1     57189
        192.168.2.158   22      2       T       3       -       SSH-2.0-
OpenSSH_6.2     SSH-1.99-OpenSSH_6.6.1p1 Ubuntu-2ubuntu2 aes128-ctr
        hmac-md5-etm@openssh.com none   diffie-hellman-group-exchange-sha256
        ssh-rsa 28:78:65:c1:c3:26:f7:1b:65:6a:44:14:d0:04:8f:b3
```

Opaque identifier for linking to other logs associated w/ same connection

```
#separator \x09
#set_separator   ,
#empty_field     (empty)
#unset_field     -
#path    conn
#open    2016-07-13-16-16-57
#fields  ts      uid     id.orig_h       id.orig_p       id.resp_h
         id.resp_p       proto   service duration        orig_bytes      resp_bytes
         conn_state      local_orig      local_resp      missed_bytes
         history orig_pkts       orig_ip_bytes   resp_pkts
         resp_ip_bytes   tunnel_parents
#types   time    string  addr    port    addr    port    enum    string
         interval        count   count   string  bool    bool    count   string
         count   count   count   count   set[string]
1324071333.493287        CHhAvVGS1DHFjwGM9       192.168.1.79    51880
         131.159.21.1    22      tcp     ssh     6.159326        2669    2501    SF
         -       -       0       ShAdDaFf        25      3981    20      3549    -
1409516196.337184        ClEkJM2Vm5giqnMf4h      10.0.0.18       40184
         128.2.6.88      41644   tcp     ssh     2.079071        3813    3633    SF
         -       -       0       ShADadFf        22      4965    26      5017    -
1419870189.485611        C4J4Th3PJpwUYZZ6gc      192.168.2.1     57189
         192.168.2.158   22      tcp     ssh     6.641754        5253    3489    SF
         -       -       0       ShADadFf        38      7241    29      5005    -
1419870206.101883        CtPZjS20MLrsMUOJi2      192.168.2.1     57191
         192.168.2.158   22      tcp     ssh     3.862198        576     813     SF
         -       -       0       ShAdDaFf        23      1784    16      1653    -
```

```
#separator \x09
#set_separator   ,
#empty_field     (empty)
#unset_field     -
#path    ssh
#open    2018-10-23-15-34-42
#fields  ts      uid     id.orig_h       id.orig_p       id.resp_h
         id.resp_p       version auth_success    auth_attempts   direction
         client  server  cipher_alg      mac_alg compression_alg kex_alg
         host_key_alg    host_key
#types   time    string  addr    port    addr    port    count   bool
         count   enum    string  string  string  string  string  string
         string  string
1324071333.792887        CHhAvVGS1DHFjwGM9       192.168.1.79    51880
         131.159.21.1    22      2       -       0       -       SSH-2.0-
OpenSSH_5.9     SSH-2.0-OpenSSH_5.8     aes128-ctr      hmac-md5
         zlib@openssh.com ecdh-sha2-nistp256      ecdsa-sha2-nistp256
         a7:26:62:3f:75:1f:33:8a:f3:32:90:8b:73:fd:2c:83
1409516196.413240        ClEkJM2Vm5giqnMf4h      10.0.0.18       40184
         128.2.6.88      41644   2       T       1       -       SSH-2.0-
OpenSSH_6.6     SSH-2.0-OpenSSH_5.9p1 Debian-5ubuntu1.1  aes128-ctr
         hmac-md5 none   ecdh-sha2-nistp256      ssh-rsa
         8a:8d:55:28:1e:71:04:99:94:43:22:89:e5:ff:e9:03
1419870189.489202        C4J4Th3PJpwUYZZ6gc      192.168.2.1     57189
         192.168.2.158   22      2       T       3       -       SSH-2.0-
OpenSSH_6.2     SSH-1.99-OpenSSH_6.6.1p1 Ubuntu-2ubuntu2 aes128-ctr
         hmac-md5-etm@openssh.com none   diffie-hellman-group-exchange-sha256
         ssh-rsa 28:78:65:c1:c3:26:f7:1b:65:6a:44:14:d0:04:8f:b3
```

Same identifier

Placement of functionality?

Layered design with instructions/control passed “down” and data stream flowing “up”

Security analysis only occurs at script layer

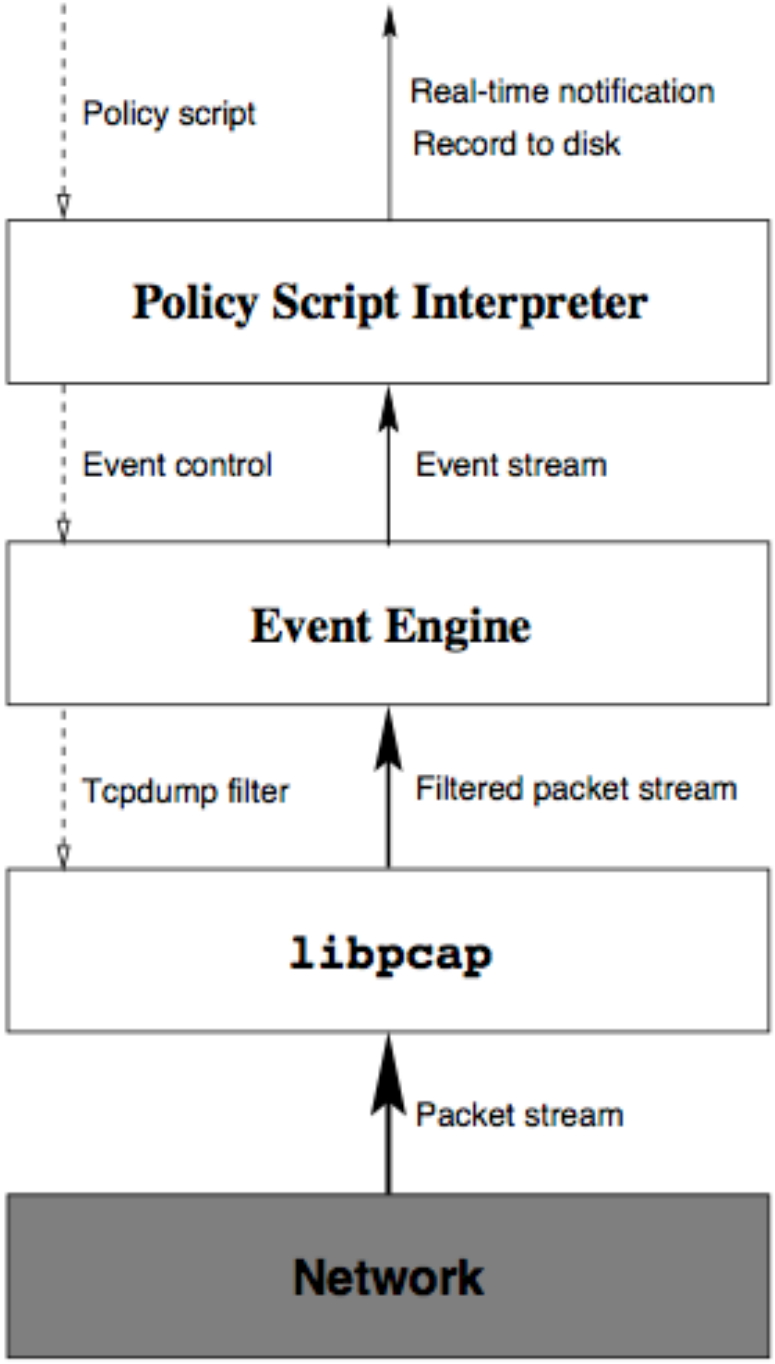


Figure 1: Structure of the Bro system

State management?

Stateless, other than BPF filter.

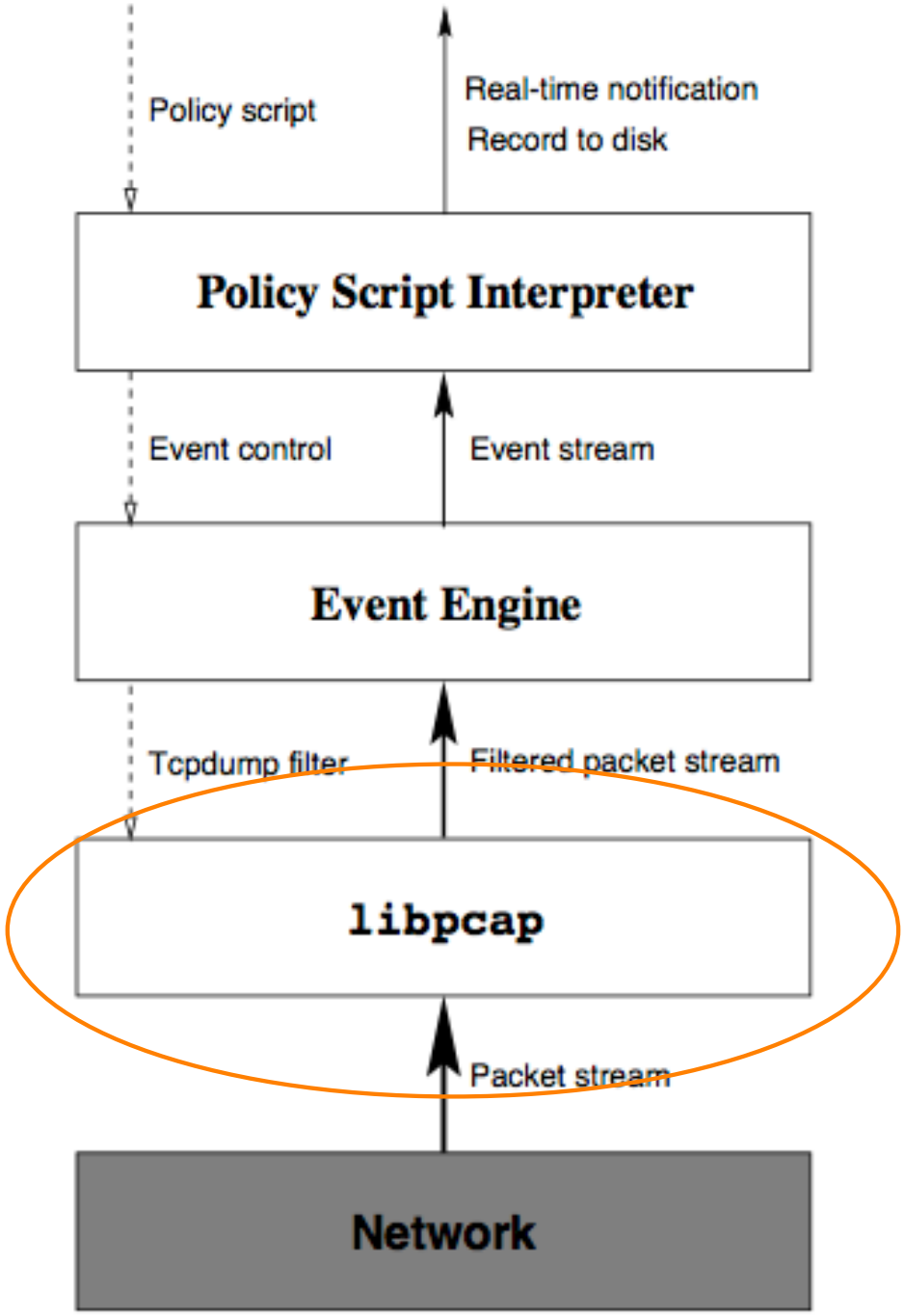


Figure 1: Structure of the Bro system

State management?

Per-flow protocol state. Managed using reference-counting.

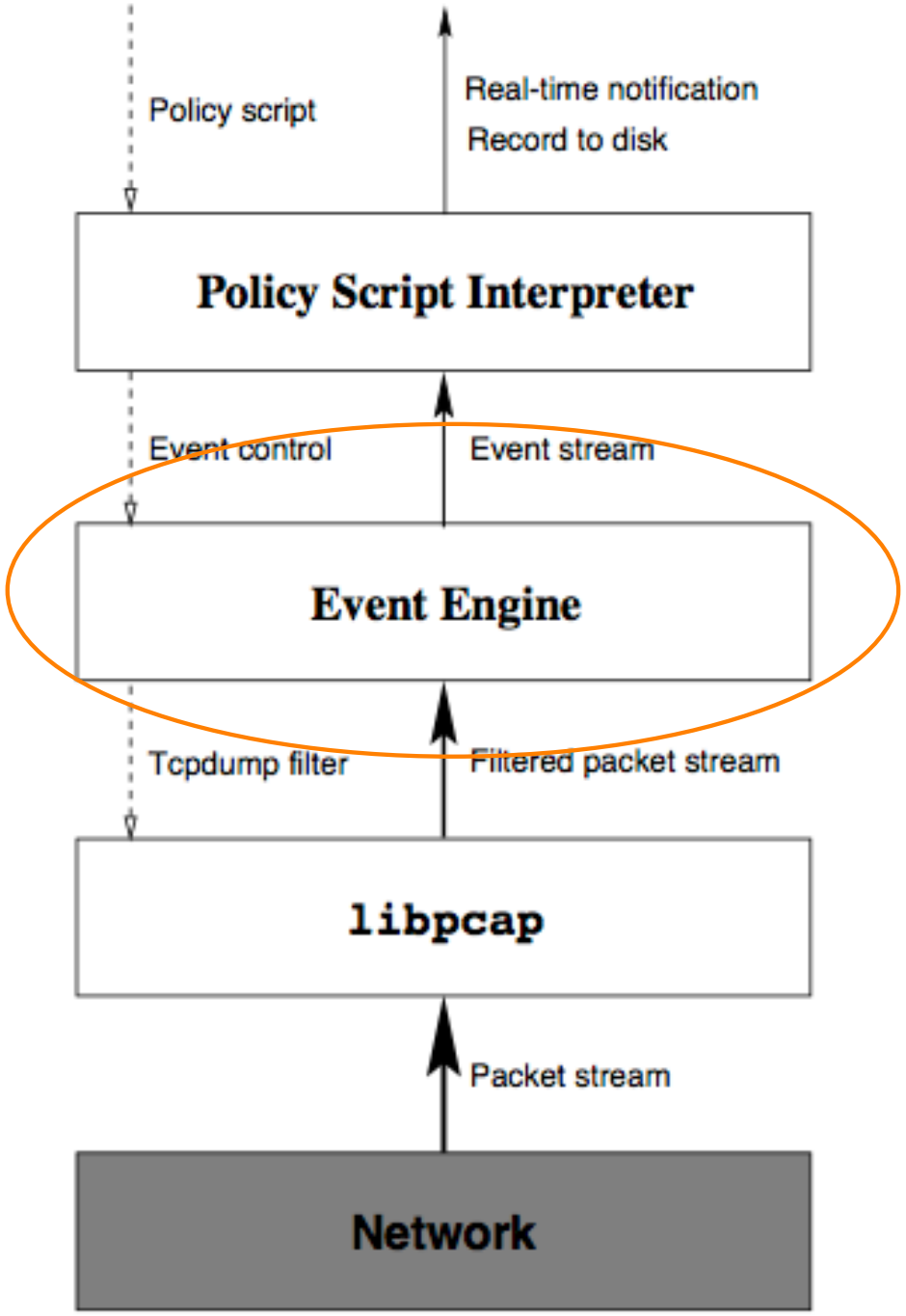


Figure 1: Structure of the Bro system

State management?

Extensive long-lived state kept in script variables.
Expiration either via explicit “delete” or timer-driven (delta T after creation/read/write).

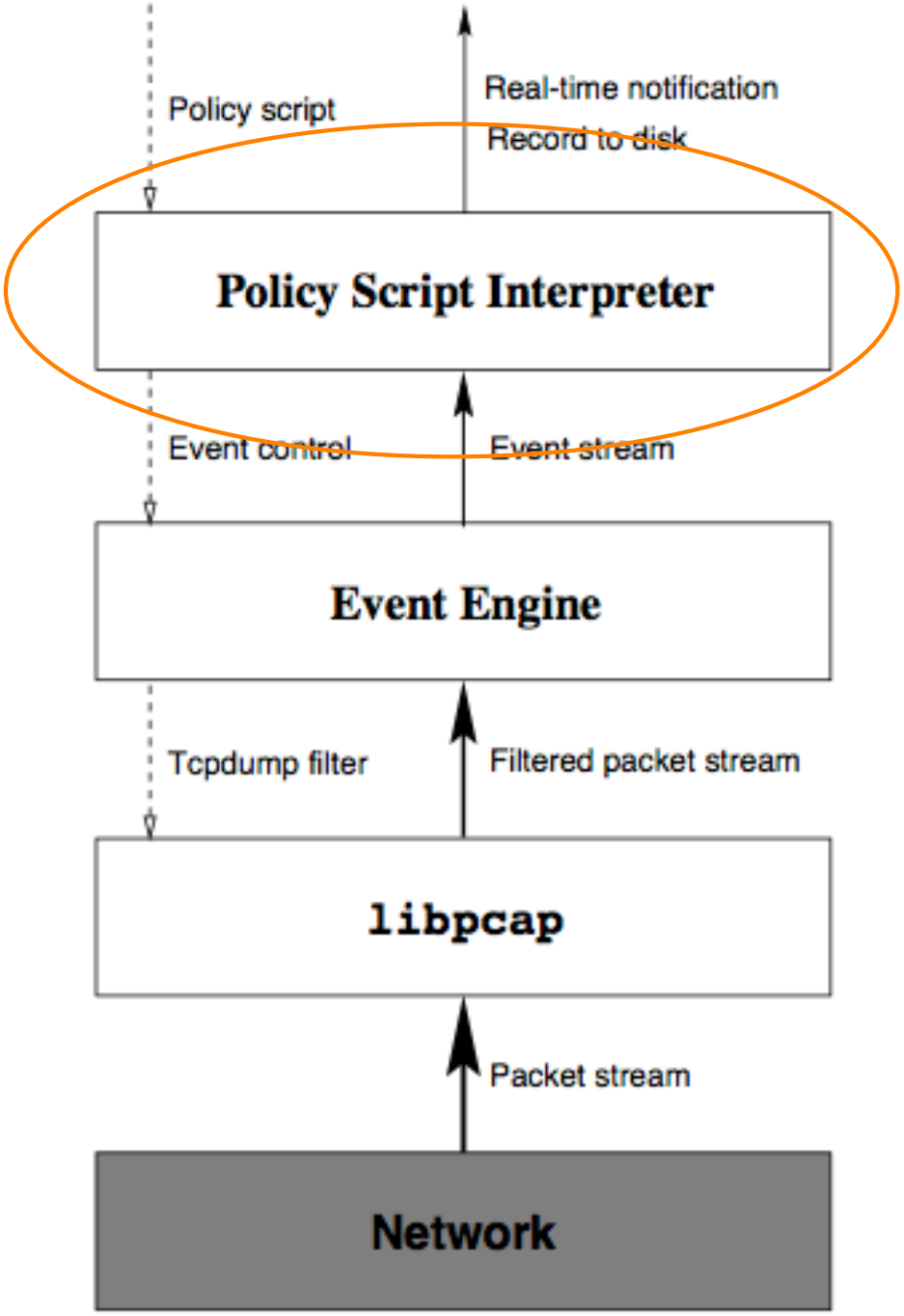


Figure 1: Structure of the Bro system

State management?

Even longer-lived state resides on disk …
... Or, today, in a “data lake” such as Splunk/Elastic.

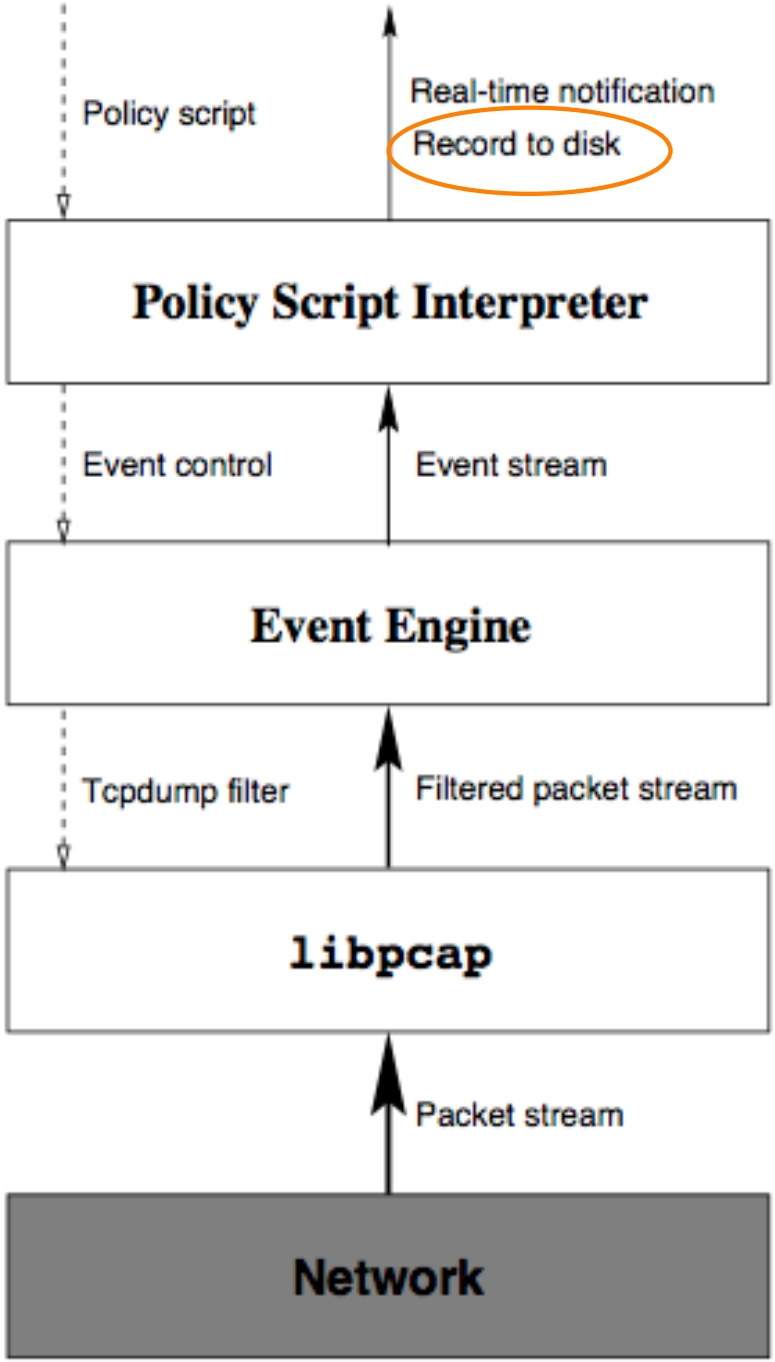


Figure 1: Structure of the Bro system

Architecture, con't

- Can both provide security properties …
  - We'll see an example next week
- … and pose security issues
- E.g.: which components are trusted to behave in what ways?

- Exploring an architecture's emergent security properties: IPv4 addressing

IPv4 Addressing Architecture

- High-level architecture of IPv4 addresses?
- Abstraction: addresses are both *locators* and *identifiers*
  - *Locators*: bits are topologically relevant
    - Includes: multicast, broadcast, private networks
  - *Identifiers*: addresses used to identify connection endpoints
    - Have global meaning
- Naming: addresses are associated with NICs rather than end systems or people

IPv4 Addressing: Mechanisms

- Addresses are represented with 32 bits
  - Limited room available for topological structure
  - Possible (today) to *fully enumerate*
  - Limited supply ⇒ *architectural stress* (NATs)
- Bit patterns have topological significance
  - Original design: class A/B/C networks
  - Current design: CIDR
- Packets carry source addresses
  - Which are set by sending system