

---

# Exploring Symmetric Cryptography for Secure Network Reprogramming



---

Donnie H. Kim, Rajeev Gandhi, Priya Narasimhan  
Carnegie Mellon University

---

## *Problem*

- **Designing a secure code-update protocol for sensor network**

## *Problem*

- Designing a secure **code-update** protocol for sensor network

# Problem

- Designing a secure **code-update** protocol for sensor network
- Every system needs software updates



CartoonChurch.com

## *Problem*

- **Designing a secure code-update protocol for sensor network**

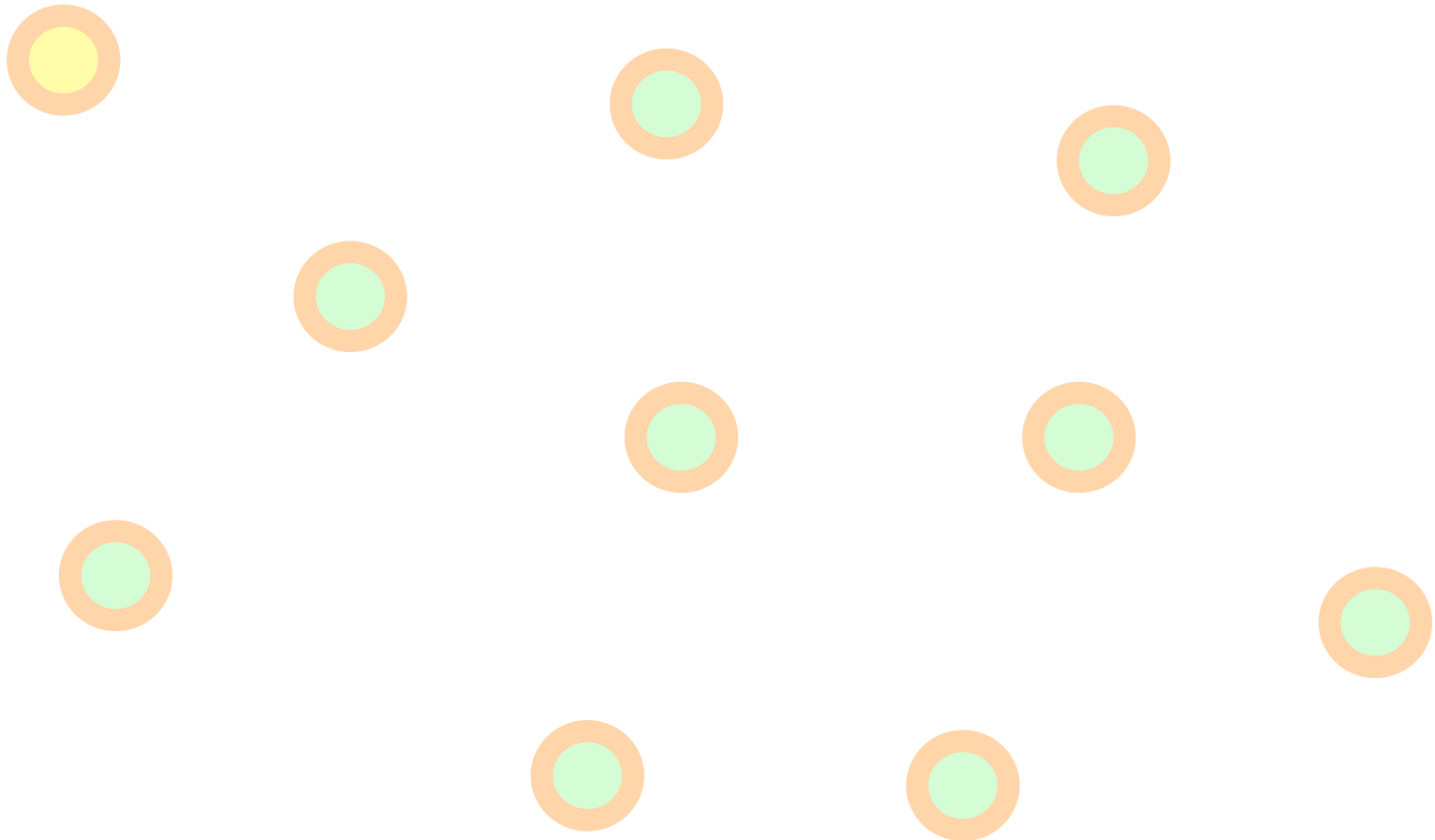
## *Problem*

- Designing a **secure** code-update protocol for sensor network

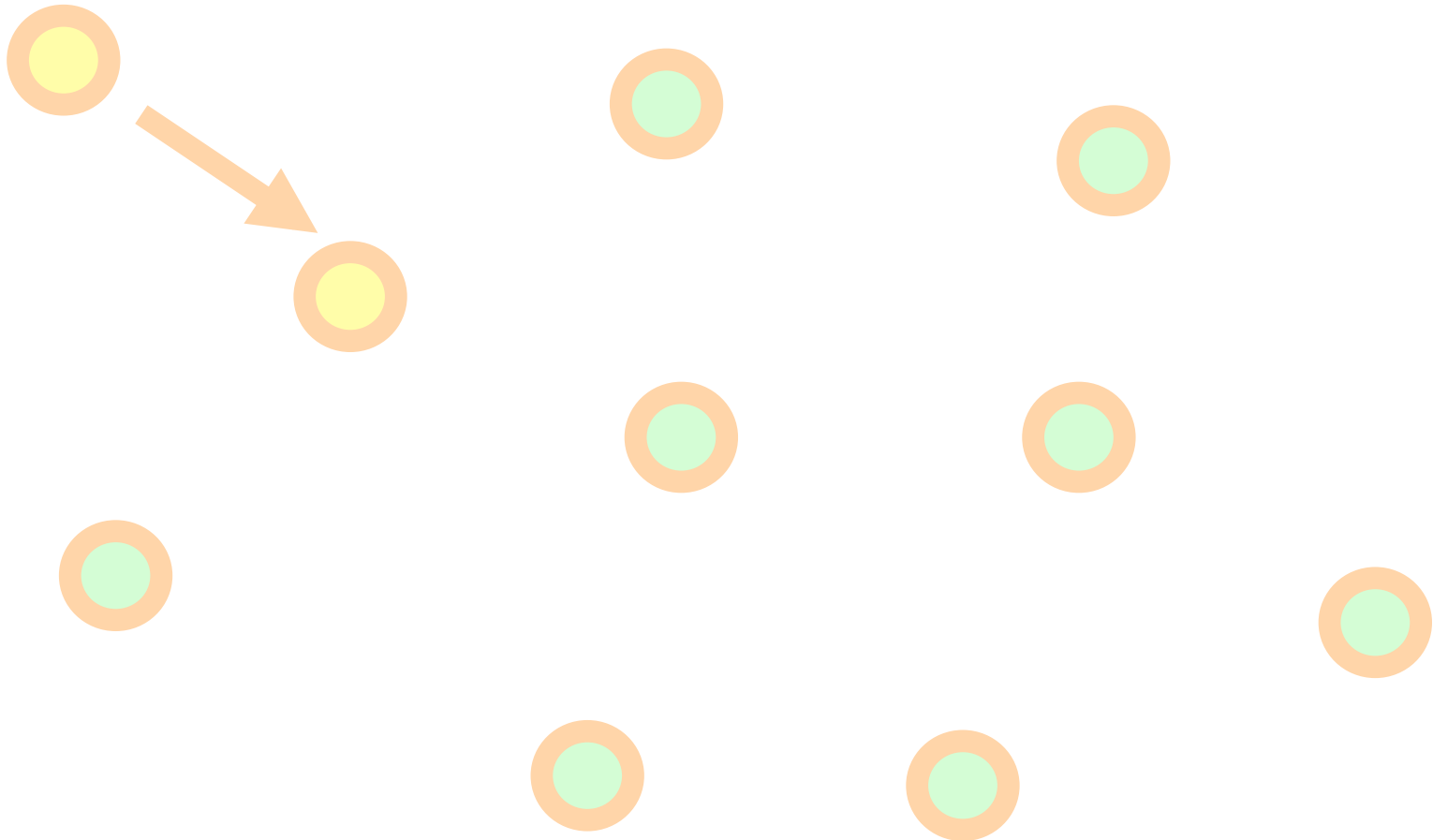
## *Problem*

- Designing a **secure** code-update protocol for sensor network
  - **Authentication**, but not necessarily secrecy
  - Network reprogramming protocols are epidemic in nature, so could the infection

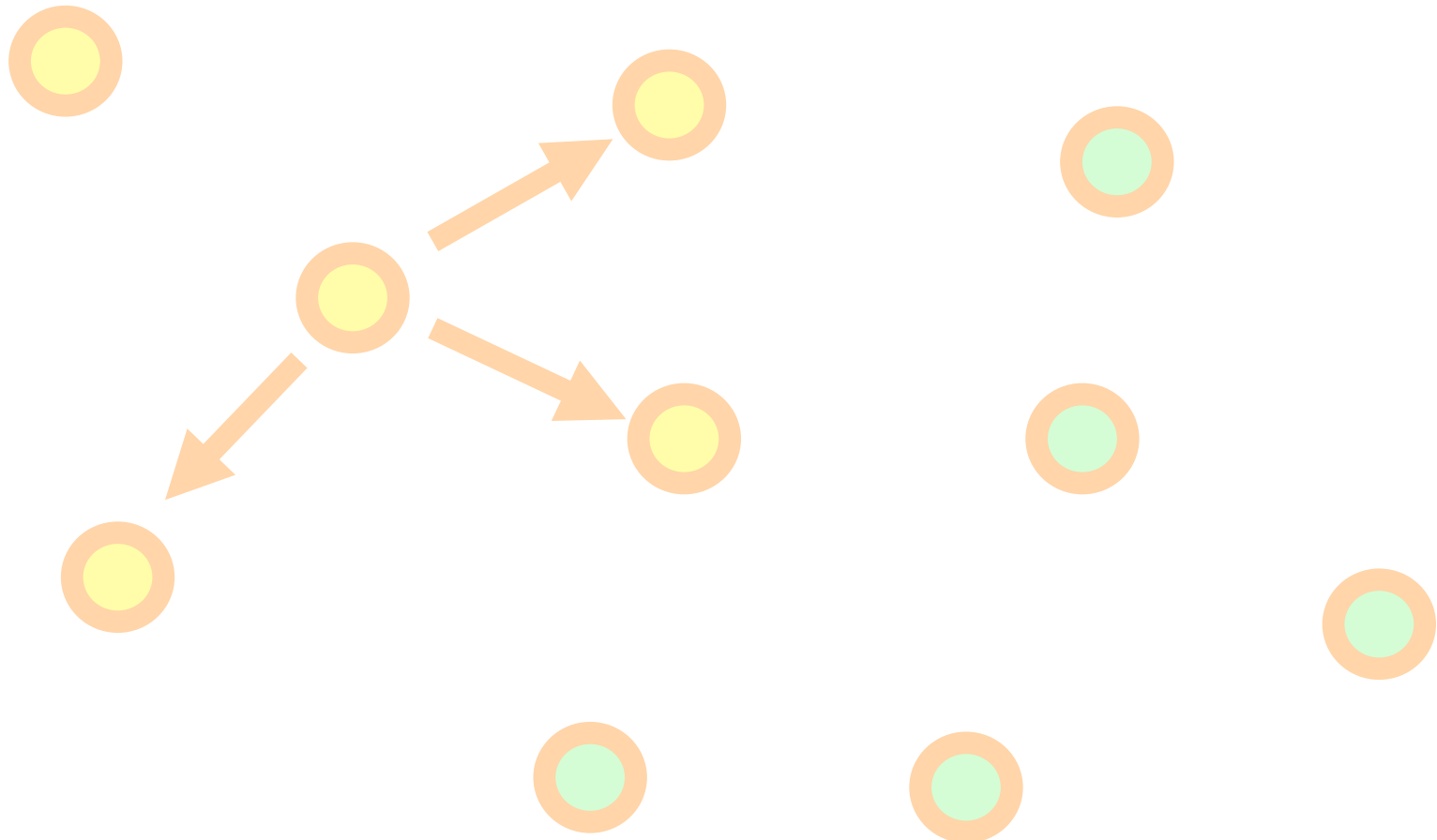
# *Network Reprogramming*



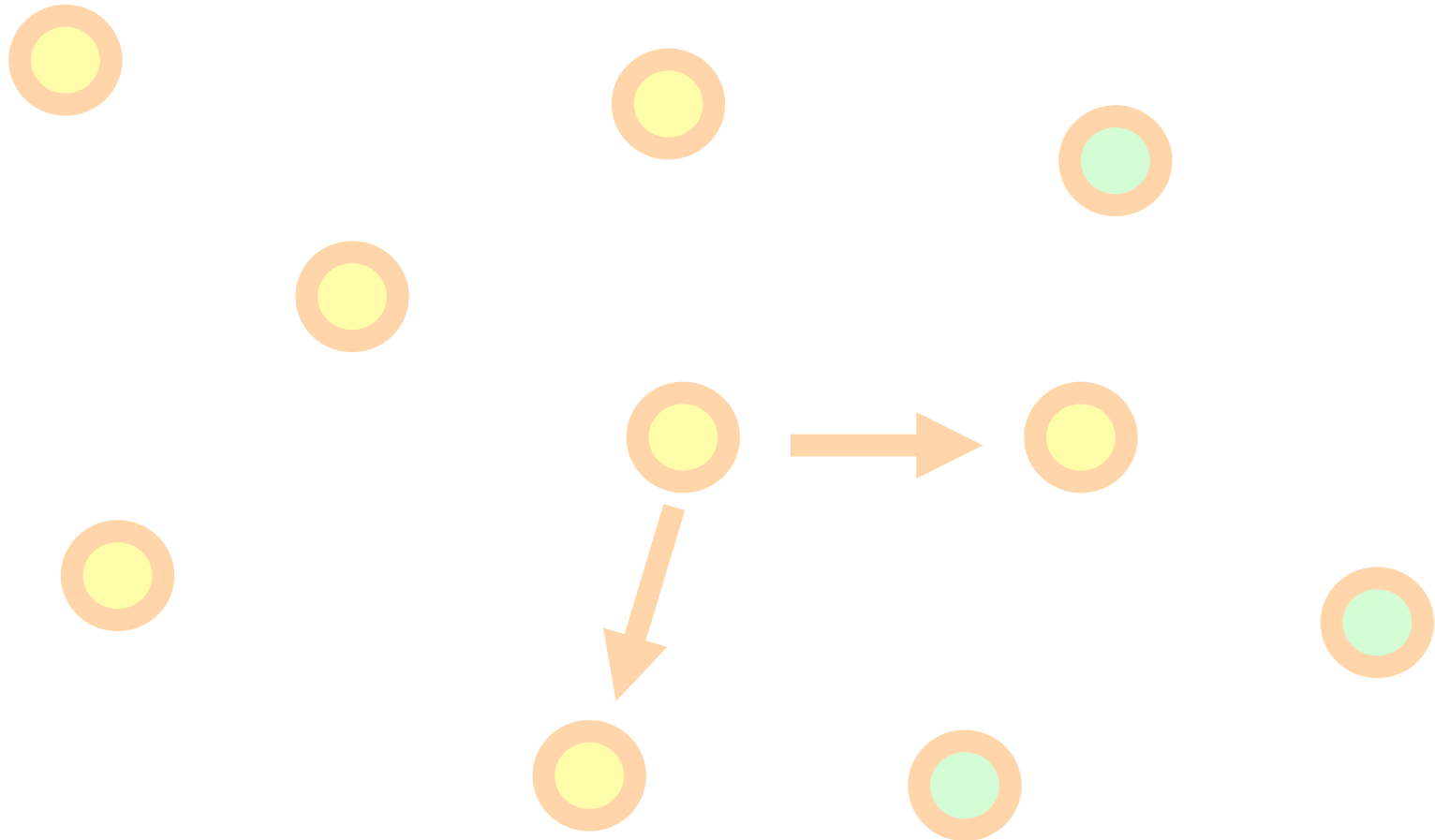
# *Network Reprogramming*



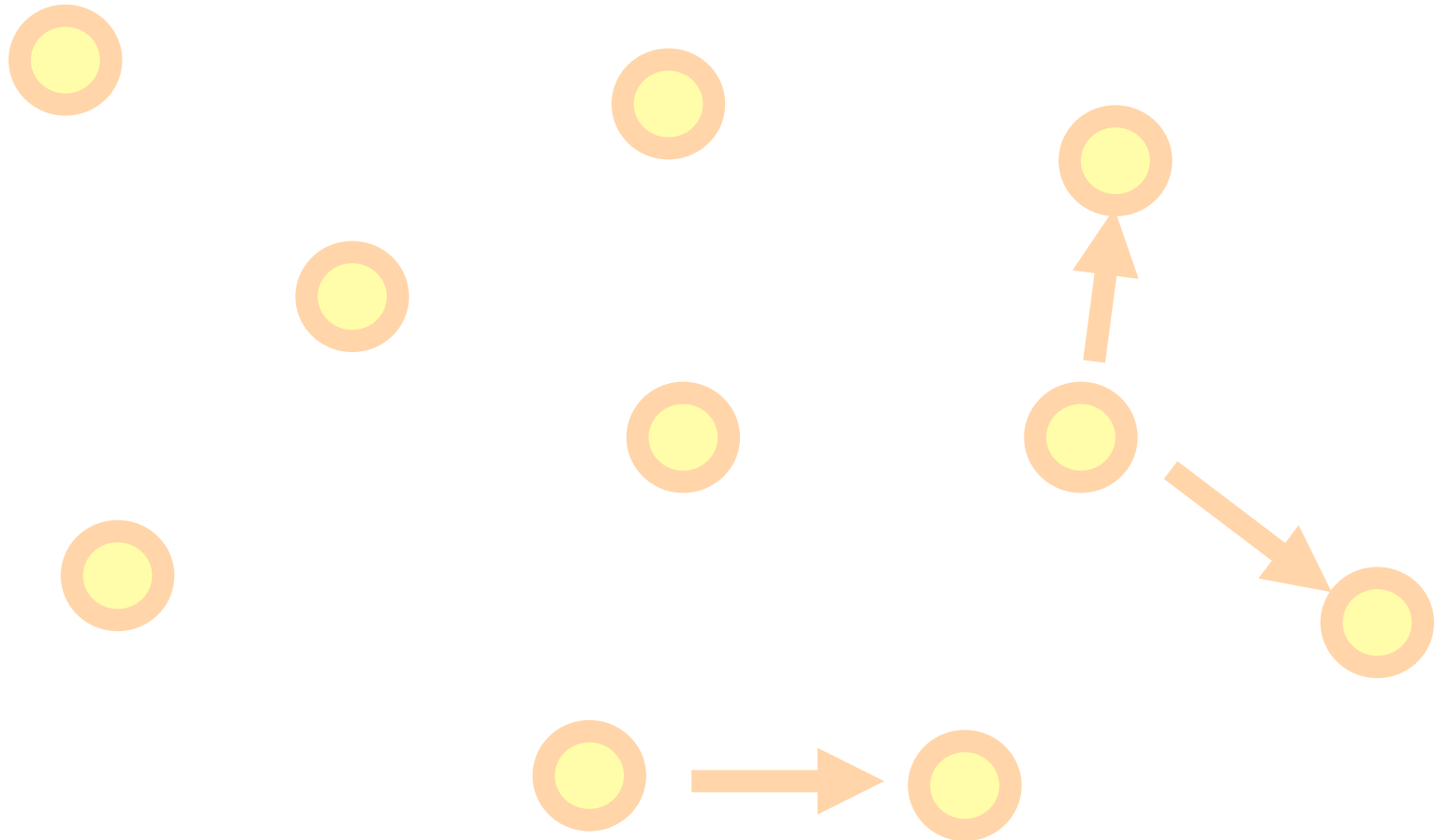
# Network Reprogramming



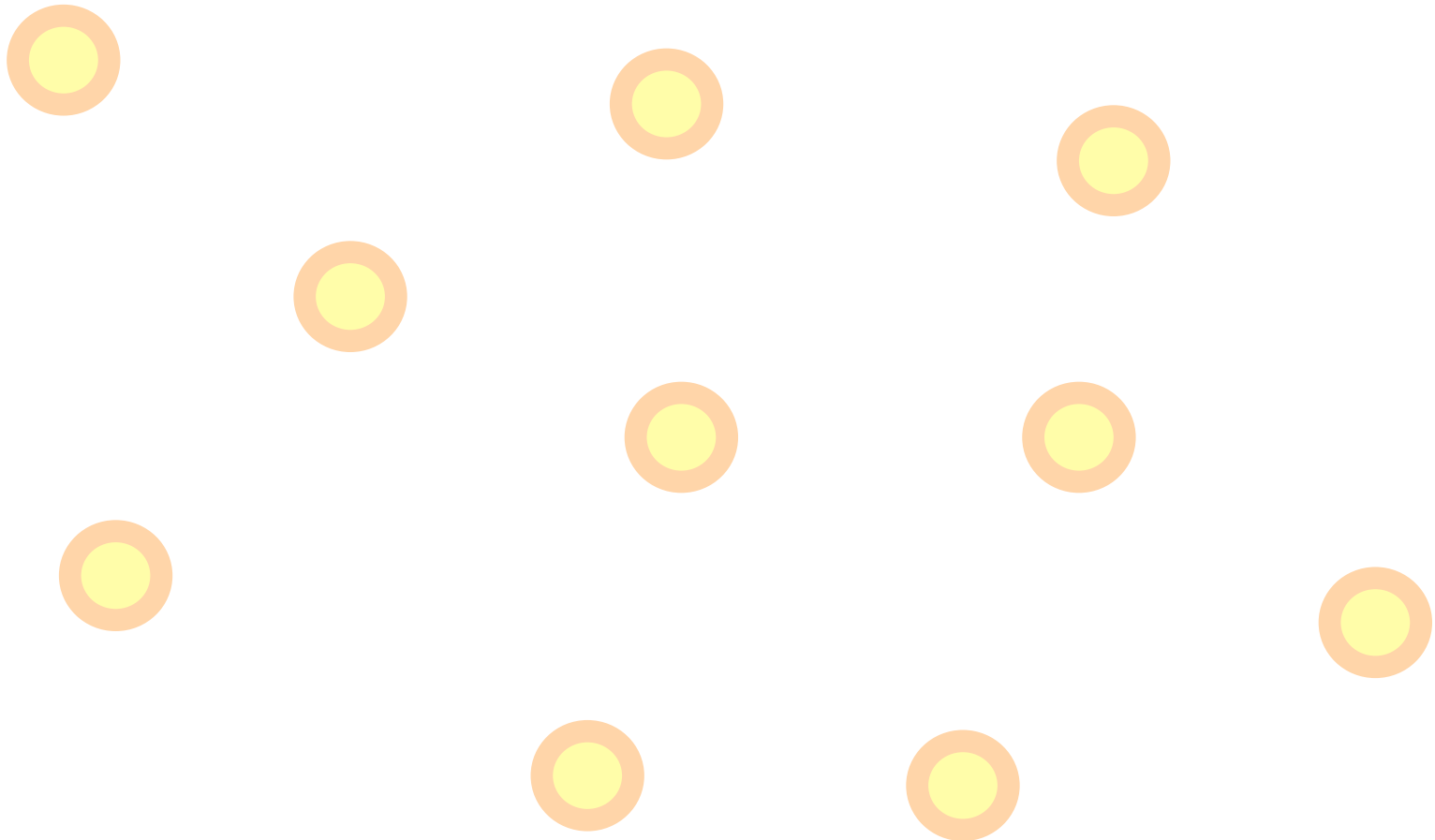
# Network Reprogramming



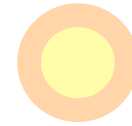
# Network Reprogramming



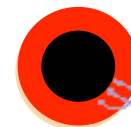
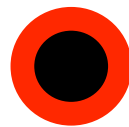
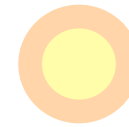
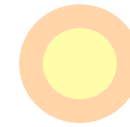
# *Network Reprogramming*



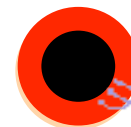
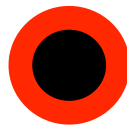
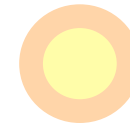
*But!*



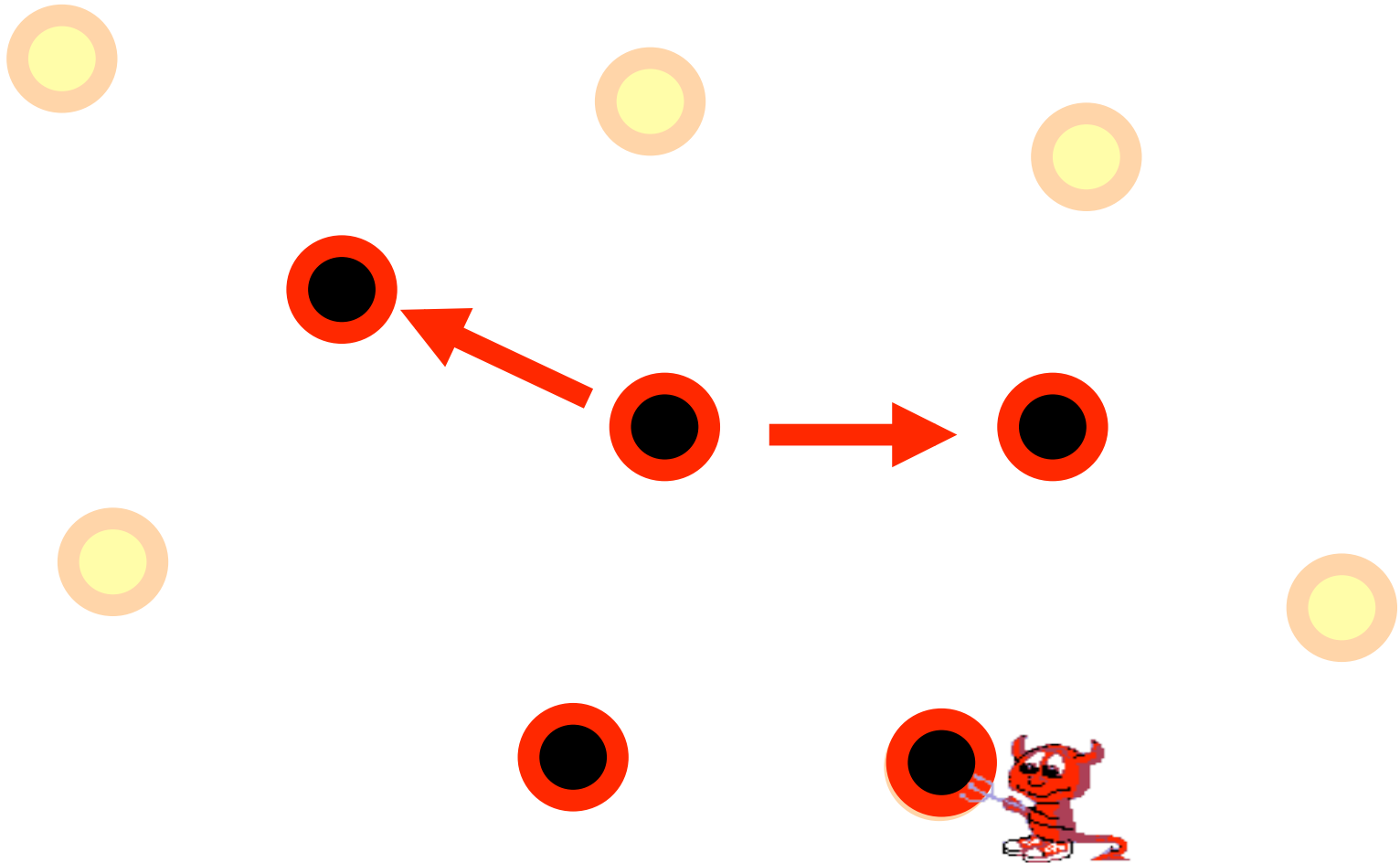
*Attack!*



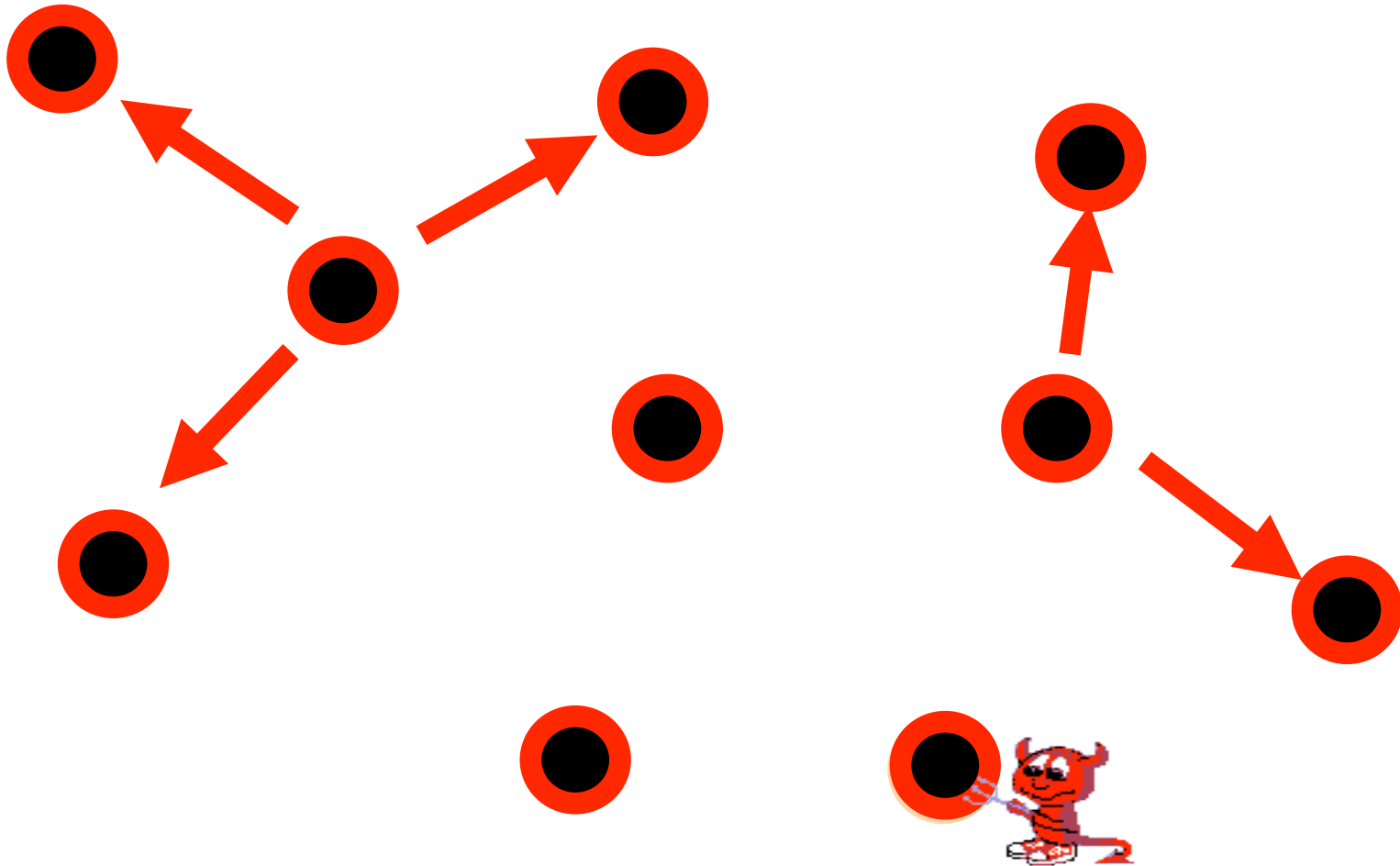
# Attack!



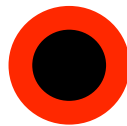
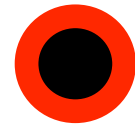
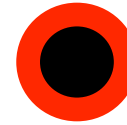
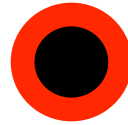
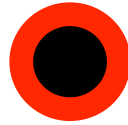
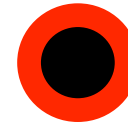
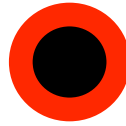
# Attack!



*Attack!*



*Attack!*



# Attack!



## *Problem*

- **Designing a secure code-update protocol for sensor network**

## *Problem*

- **Designing a secure code-update protocol for sensor network**

## *Problem*

- **Designing a secure code-update protocol for sensor network**
  - **Limited Resources (ex: Sky Tmote)**
    - **CPU:** ~ 8 mhz
    - **Radio Chip:** ~ 250 kbps
    - **Memory:** ~ 10 kb (RAM), ~ 48 kb (program EEPROM)
    - **Battery:** two AA batteries

# Comparison

- Previous approaches
  - Judicious use of **asymmetric cryptosystems**
    - Requiring only **a single signature** over each new code-images

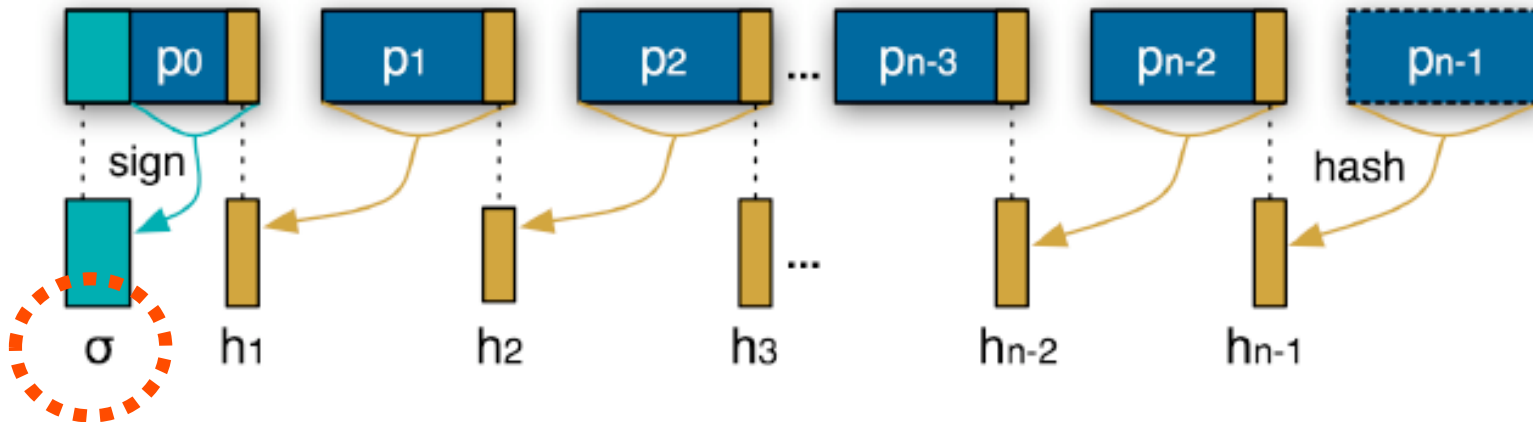
	Sluice	Secure Deluge	Deng, et al.
Authenticity Commitment	Digital Signature	Digital Signature	Digital Signature
Hash Structure	Page-level chain	Packet-level chain	Packet-level tree

# Comparison

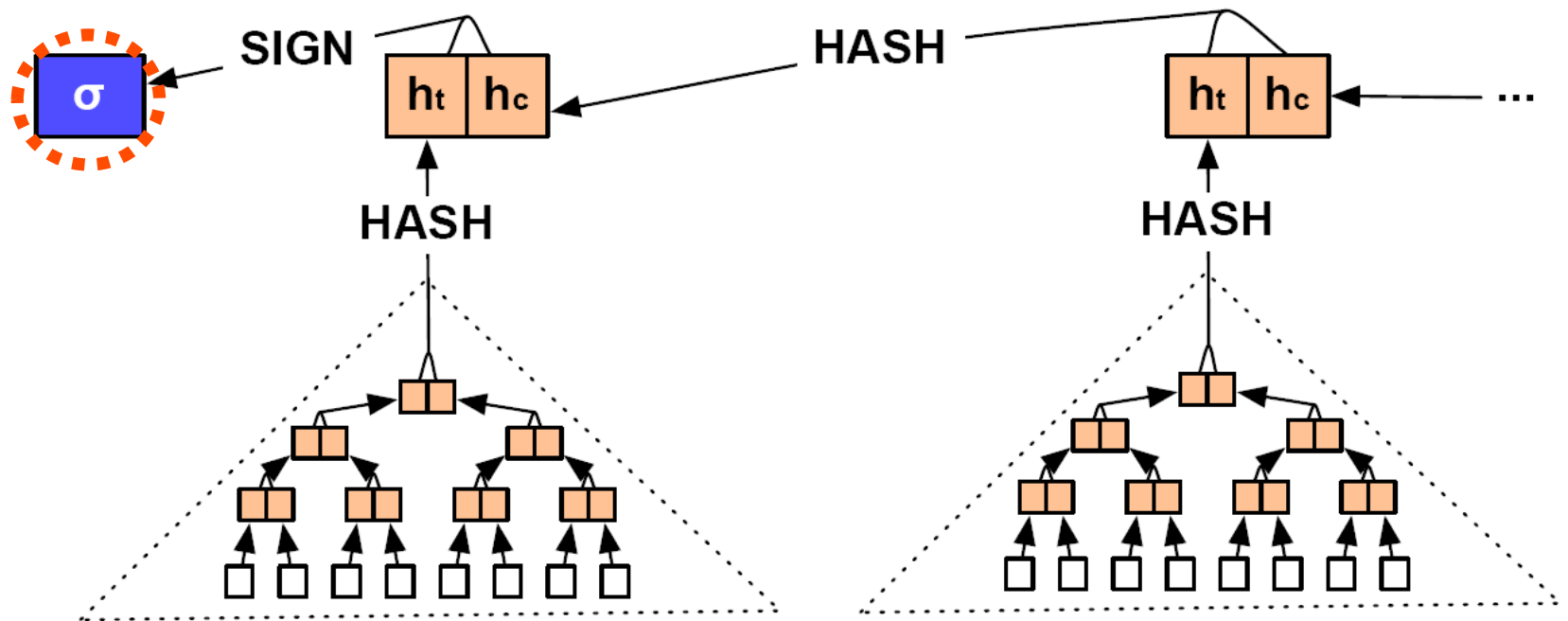
- Previous approaches
  - Judicious use of **asymmetric cryptosystems**
    - Requiring only **a single signature** over each new code- images

	Sluice	Secure Deluge	Deng, et al.
Authenticity Commitment	Digital Signature	Digital Signature	Digital Signature
Hash Structure	Page-level chain	Packet-level chain	Packet-level tree

# Sluice – Hash chain over pages



# Deng, et al. – Hash tree over packets



## Asymmetric cryptography vs. Symmetric cryptography

	Digital Signature (ECC)	MAC (CBC-MAC)
Verification	≈ 5 seconds	≈ 5 milliseconds
Signature/MAC size	44 bytes	4 ~ 8 bytes
Key size	≥ 16 bytes	≥ 8 bytes

MAC : Message Authentication Code

## *More specific problem*

- Can we replace digital signatures?

	Sluice	Secure Deluge	Deng, et al.
Authenticity Commitment	Digital Signature	Digital Signature	Digital Signature
Hash Structure	Page-level chain	Packet-level chain	Packet-level tree

## *More specific problem*

- Can we replace digital signatures?

	Sluice	Secure Deluge	Deng, et al.
Authenticity Commitment	MAC	Digital Signature	Digital Signature
Hash Structure	Page-level chain	Packet-level chain	Packet-level tree

## *More specific problem*

- Can we replace digital signatures?

	Sluice	Secure Deluge	Deng, et al.
Authenticity Commitment	MAC	MAC	Digital Signature
Hash Structure	Page-level chain	Packet-level chain	Packet-level tree

## *More specific problem*

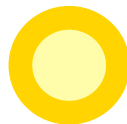
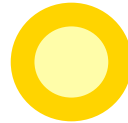
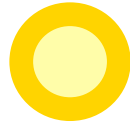
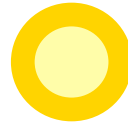
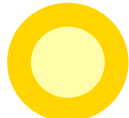
- Can we replace digital signatures?

	Sluice	Secure Deluge	Deng, et al.
Authenticity Commitment	MAC	MAC	MAC
Hash Structure	Page-level chain	Packet-level chain	Packet-level tree

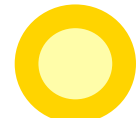
*And now, we present Castor*



# Castor



Now! That's very good sweetly!



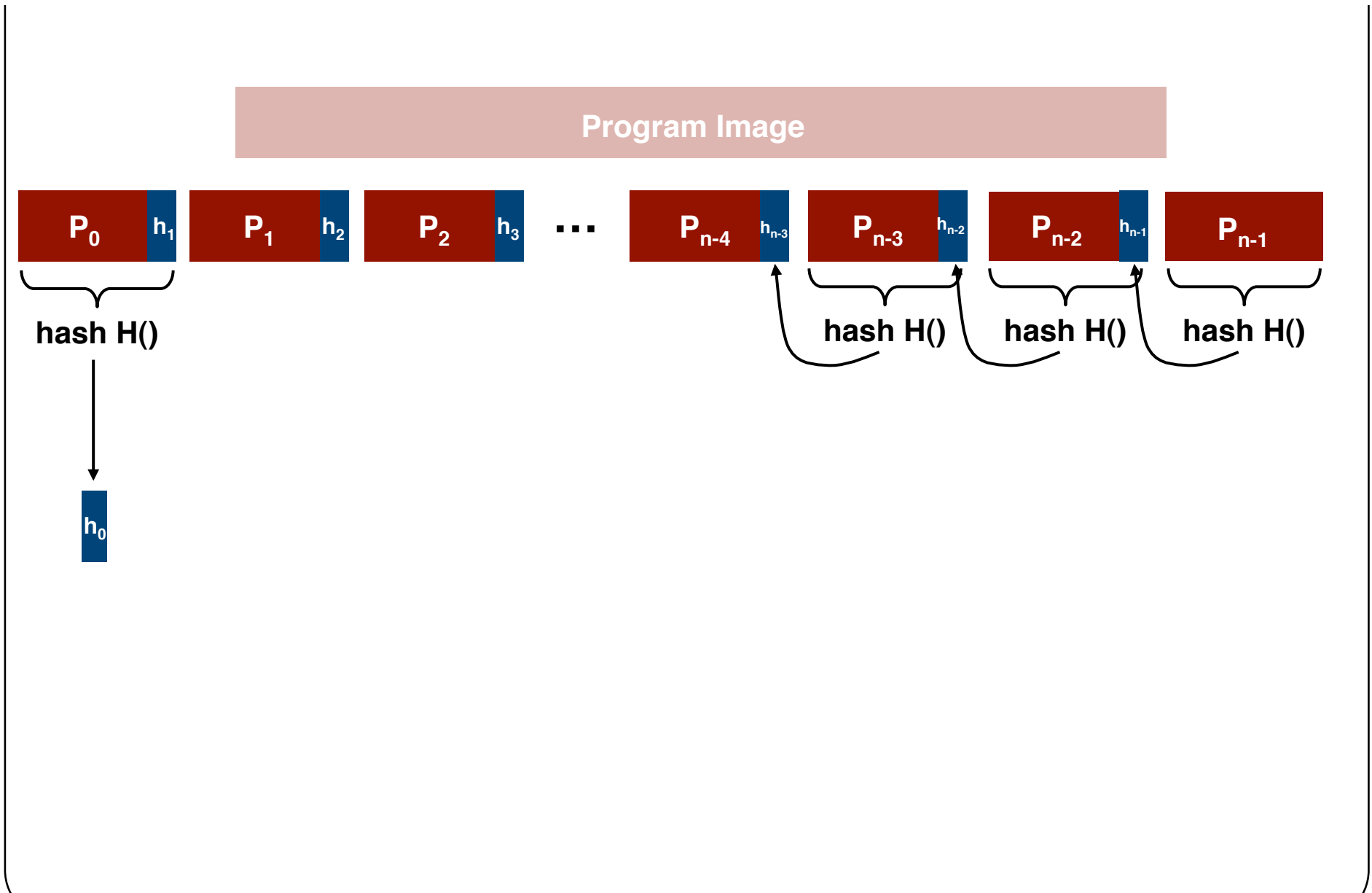
## *Assumptions and attacker model*

- **Assumption**
  - **Single trusted sender** (base station) updates the network
  - **Base station and every sensor nodes are bootstrapped with:**
    - **T (the key disclosure interval)**,  **$k_0$  (the head of the key chain)**, one-way function  $K()$  and one-way hash function  $H()$
  - **Loose time synchronization**
- **Attacker model – Attackers can:**
  - inject an arbitrary number of packets
  - withhold/delay/modify an arbitrary number of packets
  - tamper any sensor nodes and reveal crypto-materials (keys)

**Program Image**



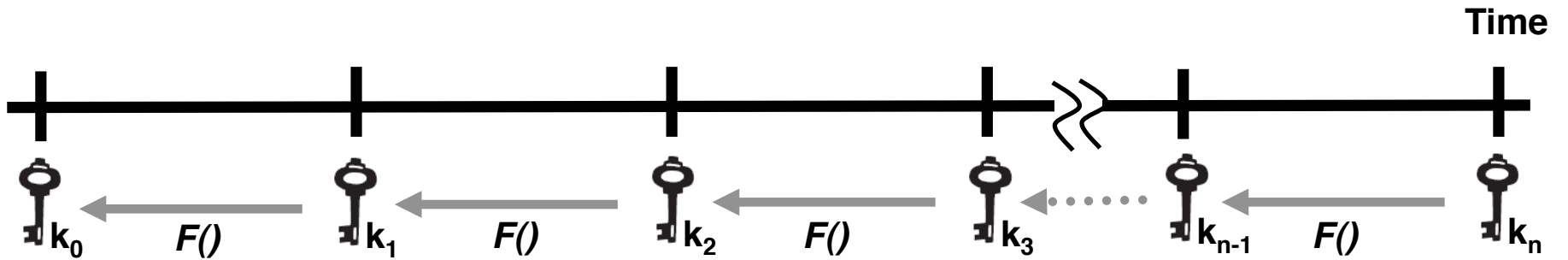
Program Image

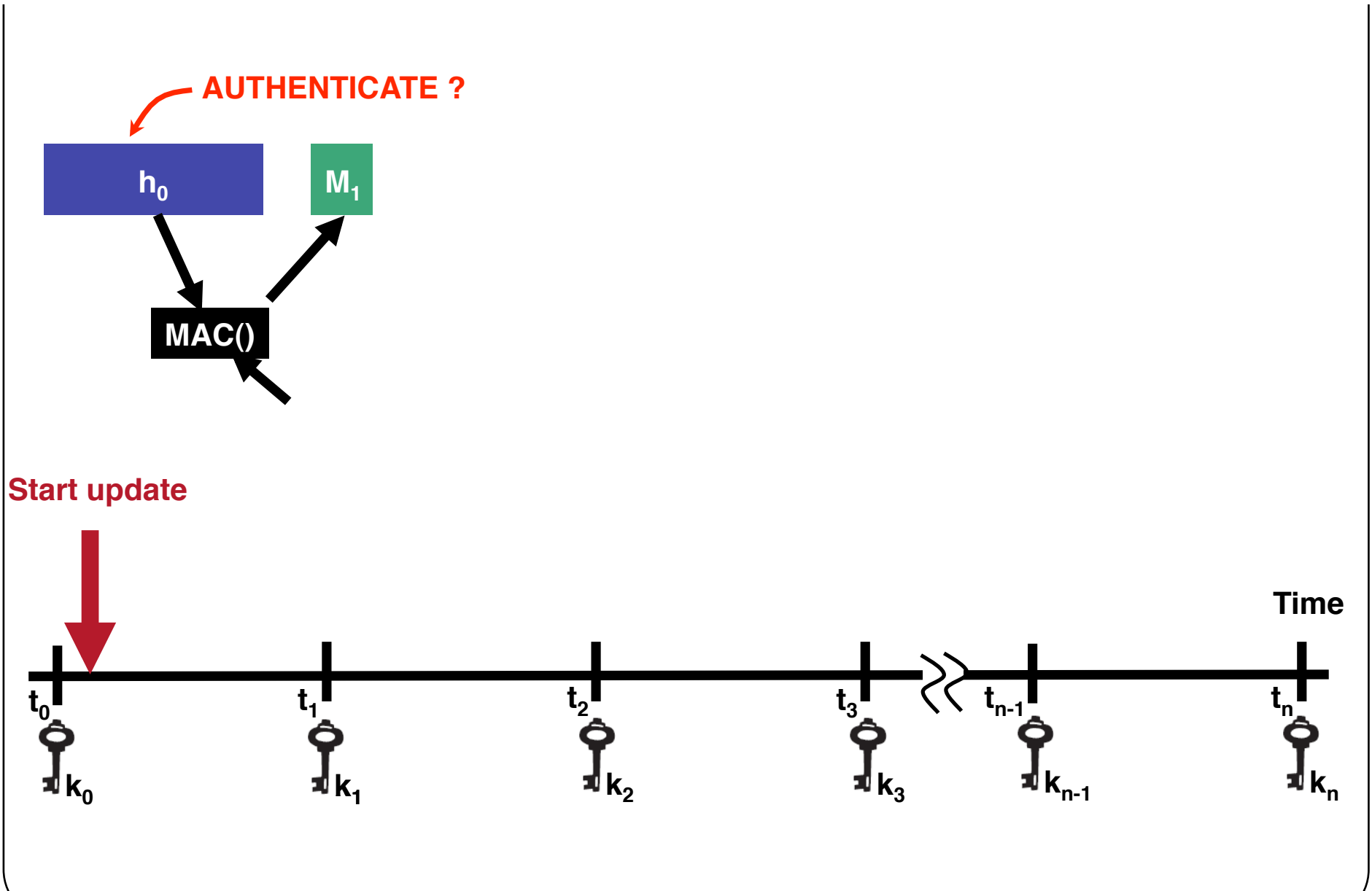


**AUTHENTICATE ?**

$h_0$

$h_0$



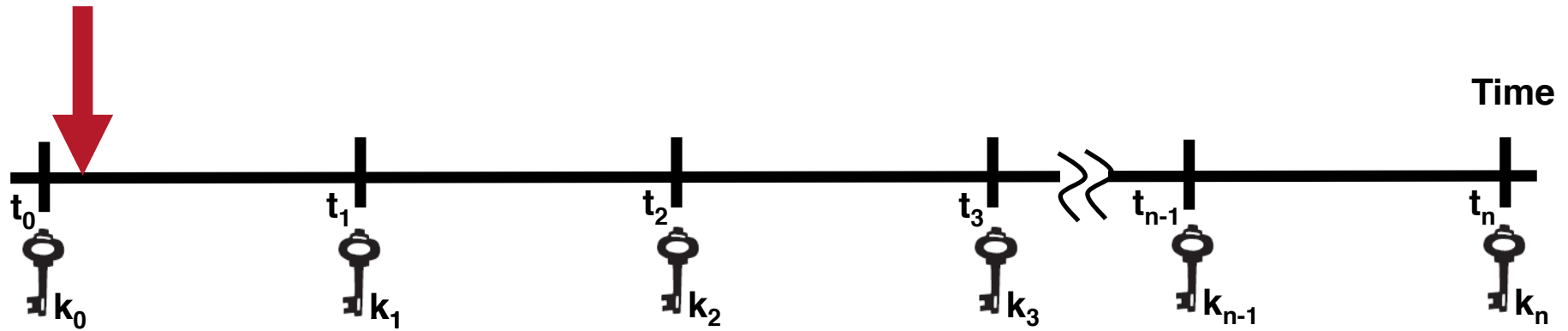


**AUTHENTICATE ?**

$h_0$

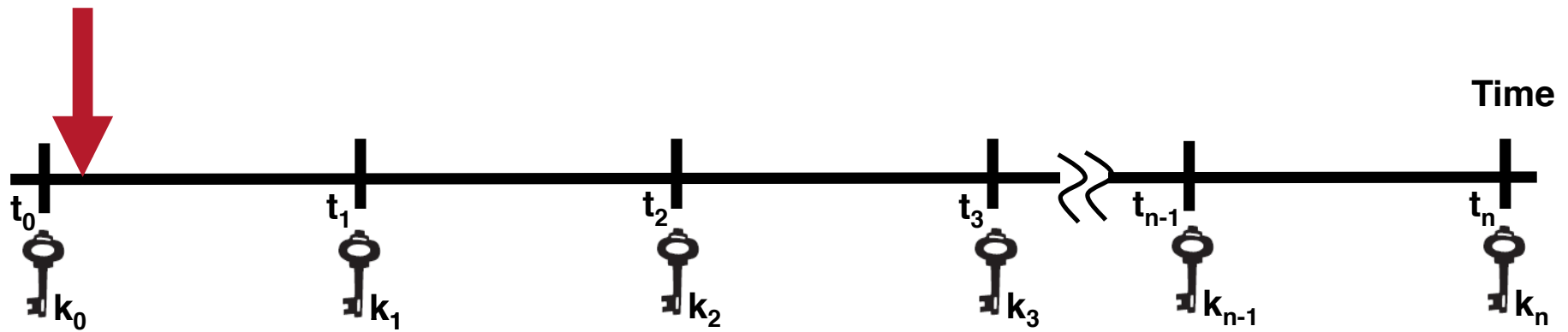
$M_1$

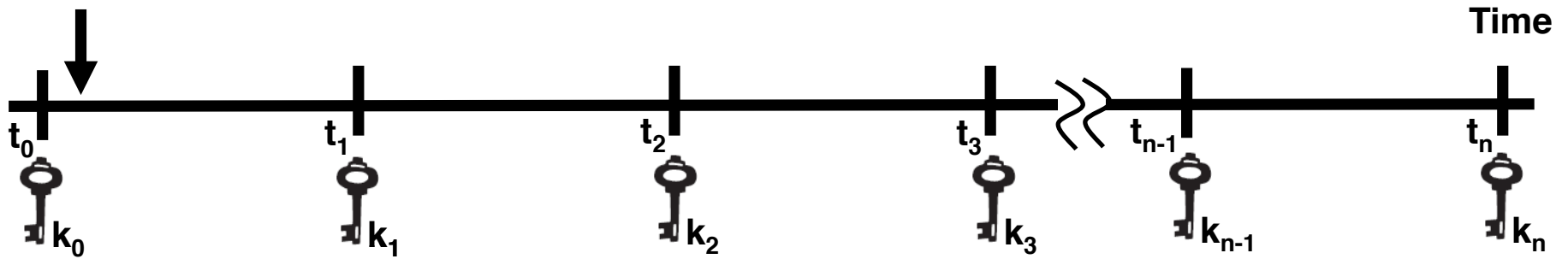
**Start update**

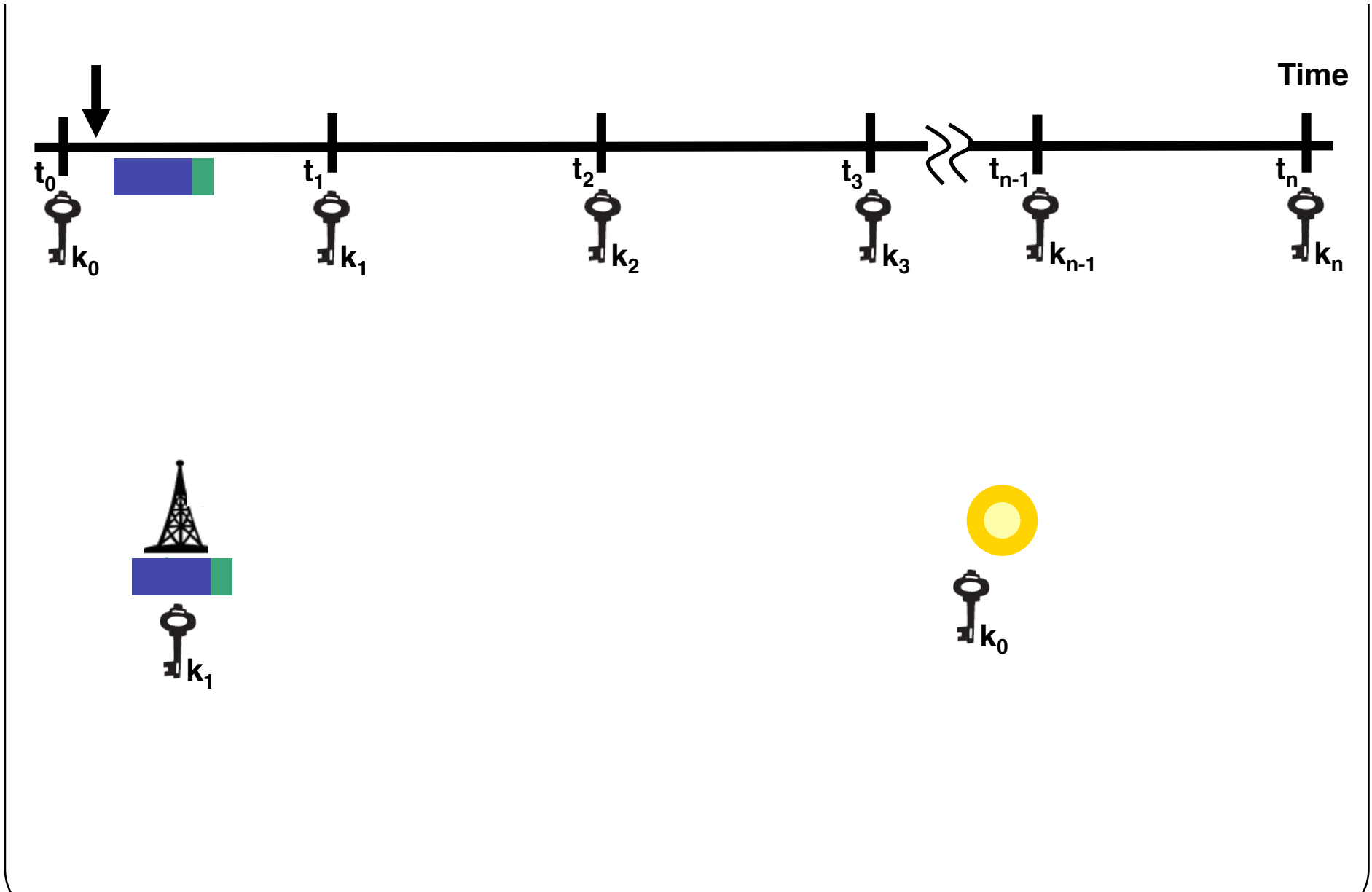


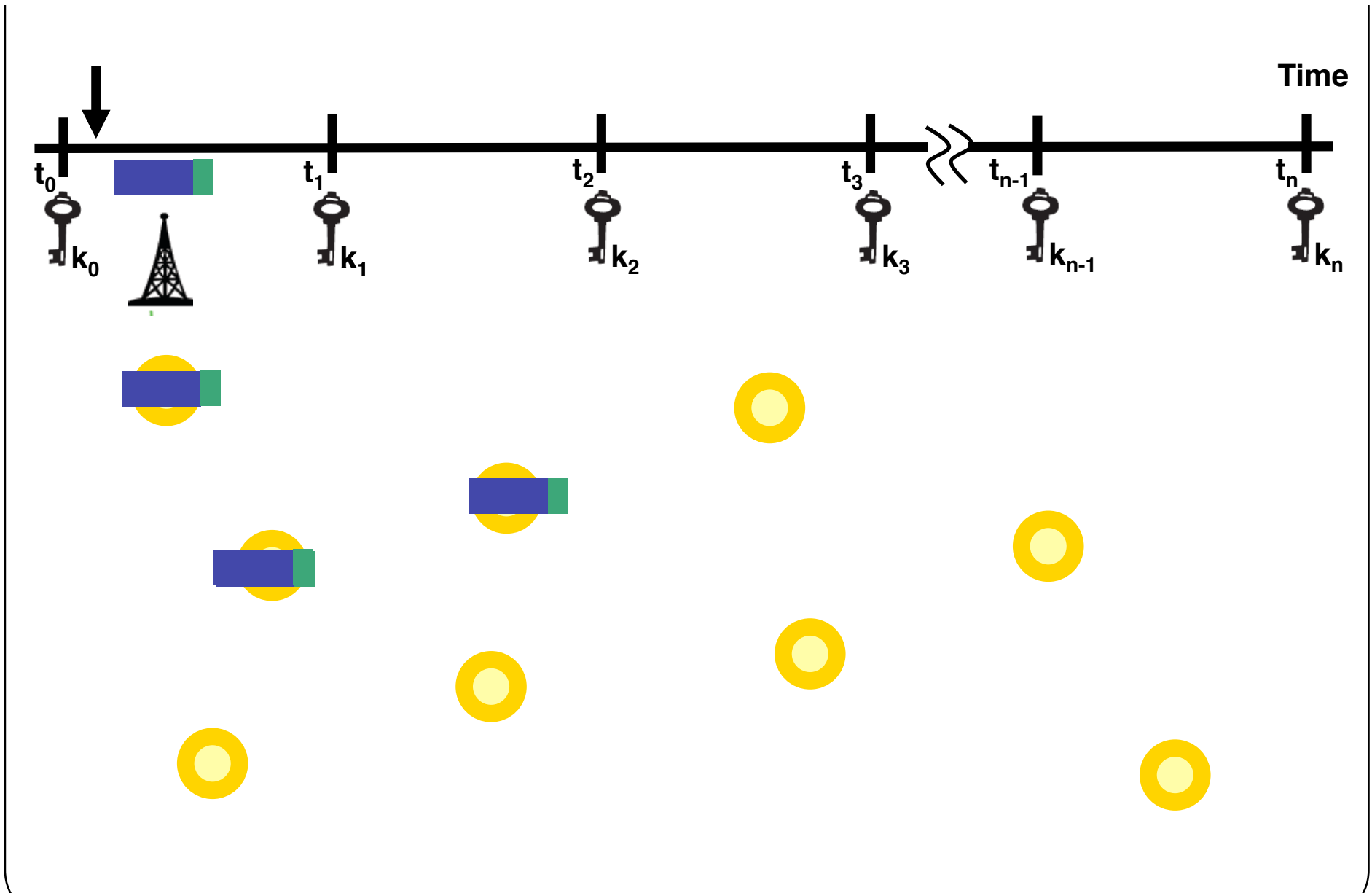


Start update





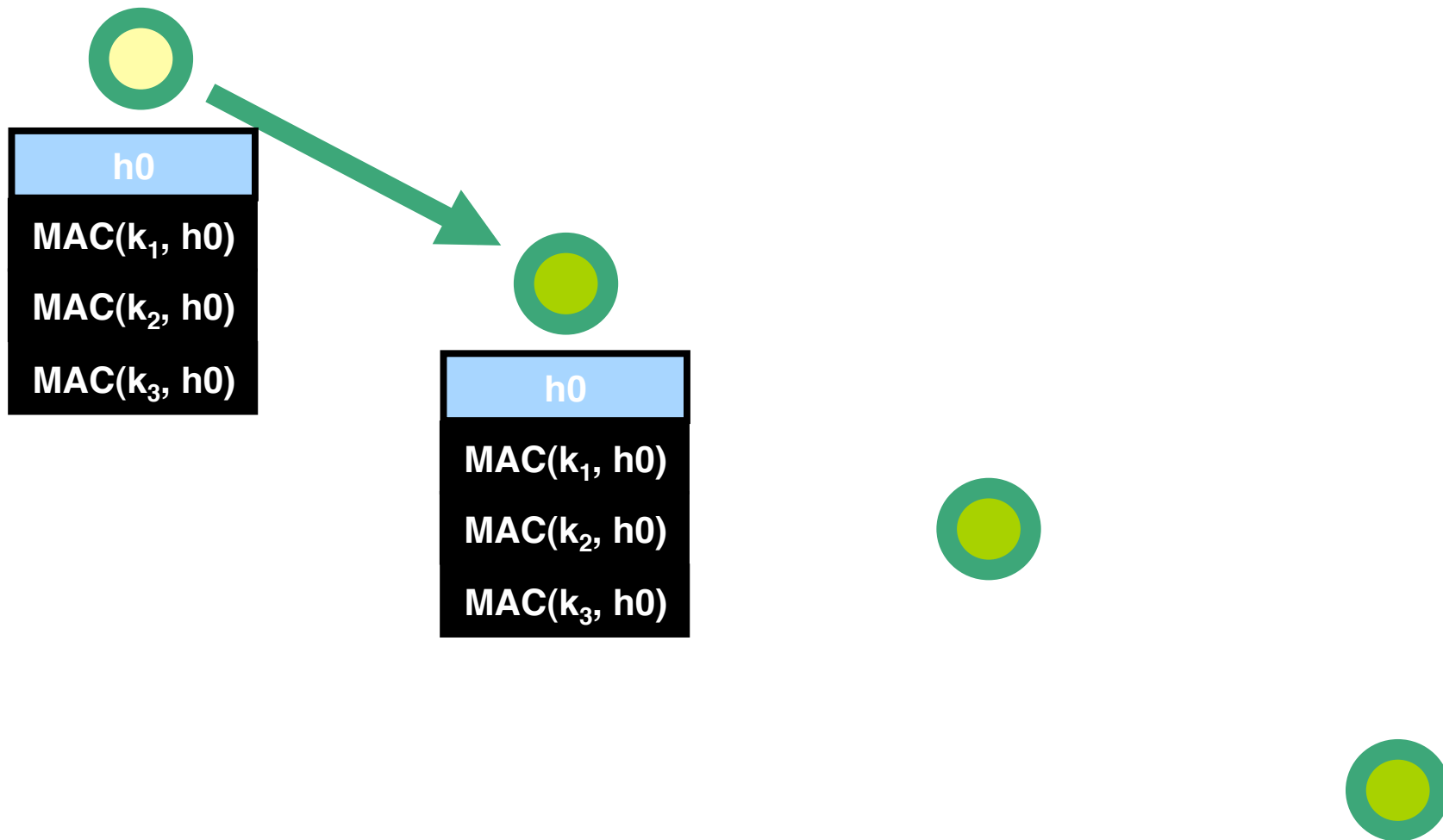


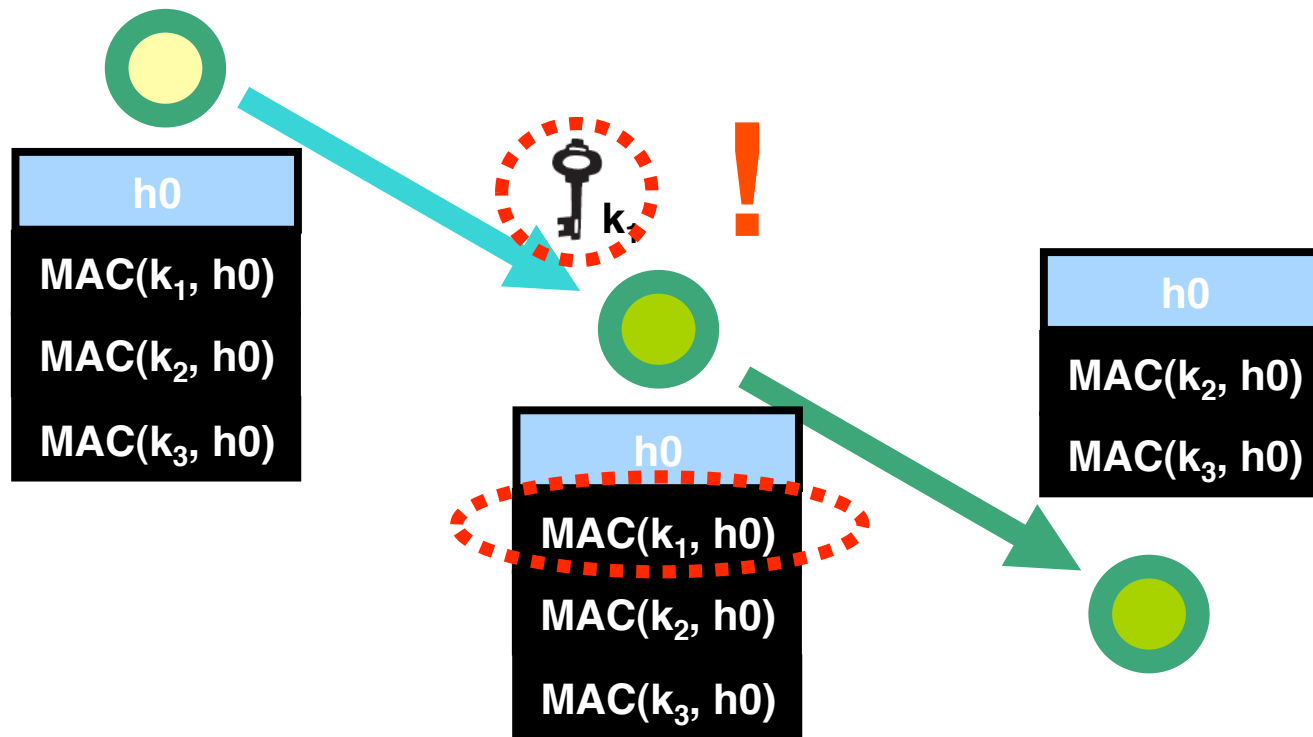


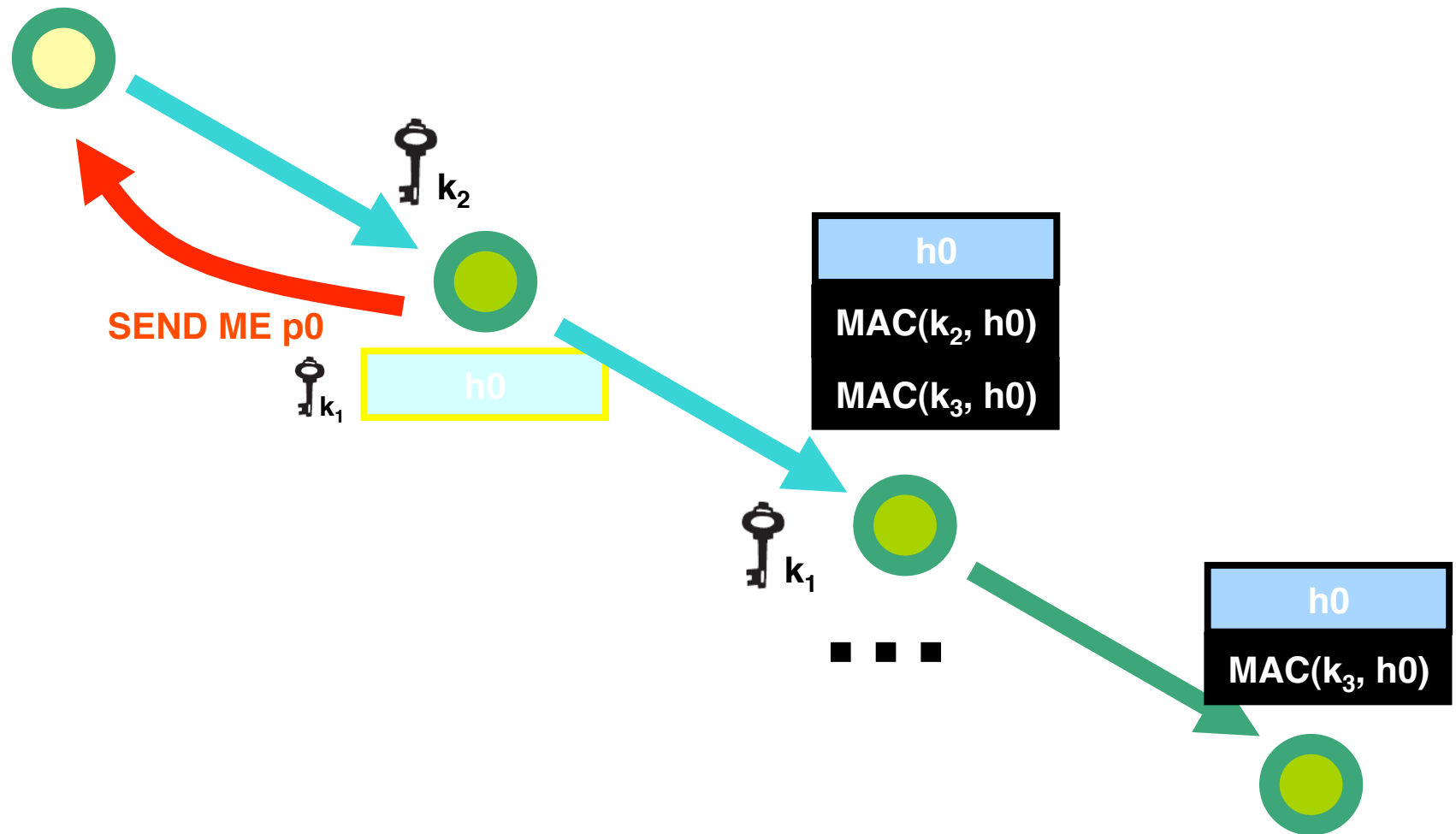


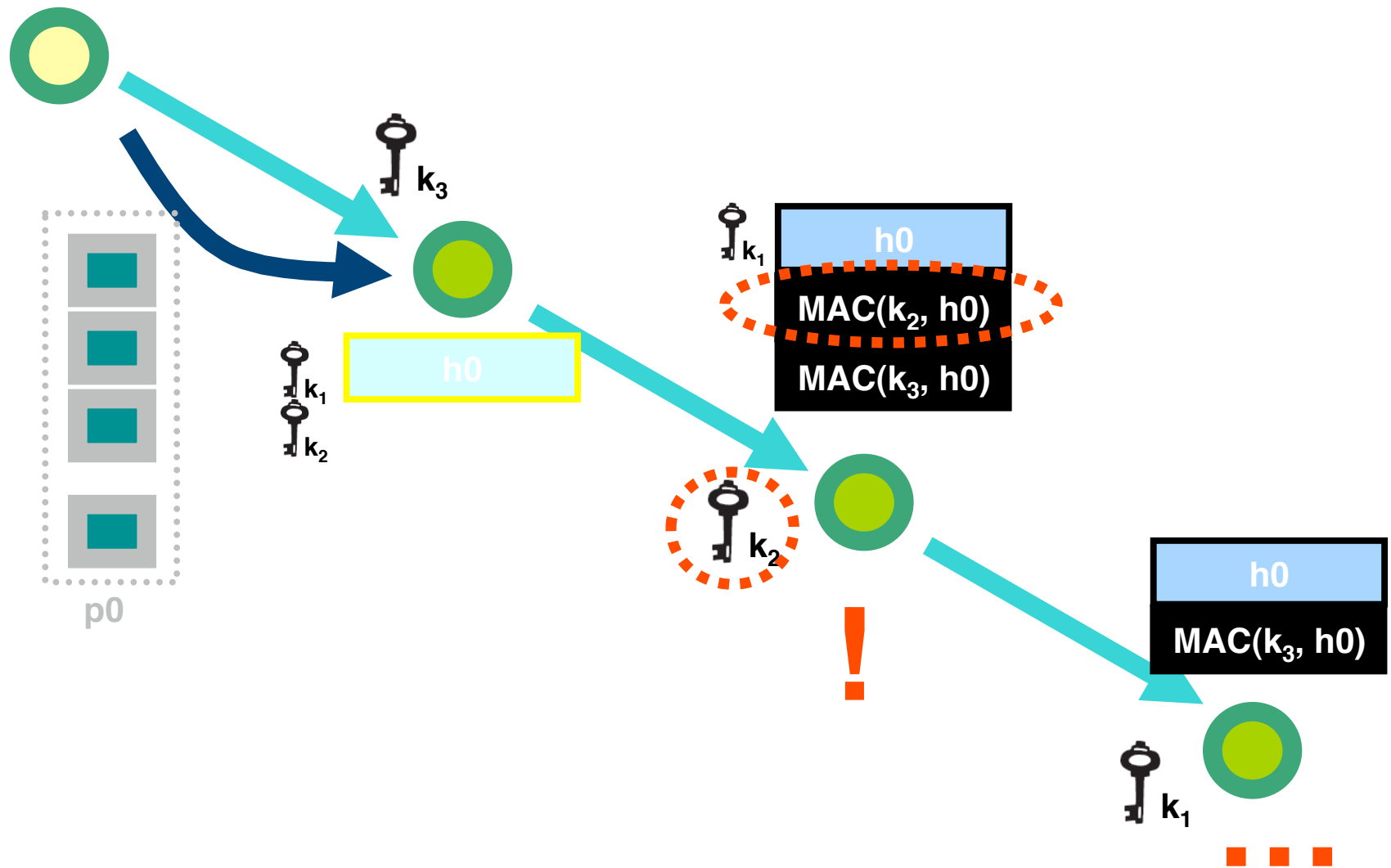
$h_0$
$\text{MAC}(k_1, h_0)$
$\text{MAC}(k_2, h_0)$
$\text{MAC}(k_3, h_0)$

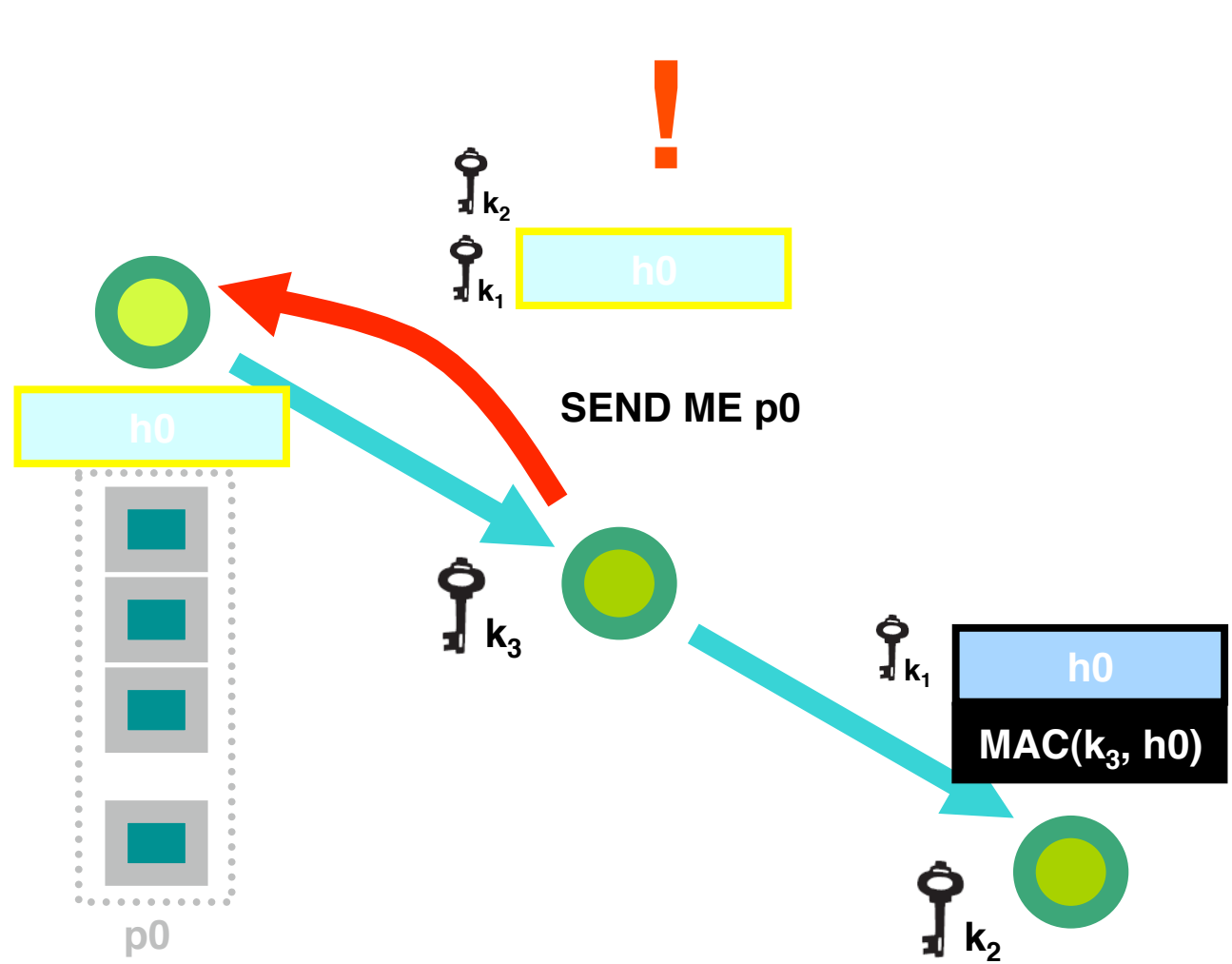


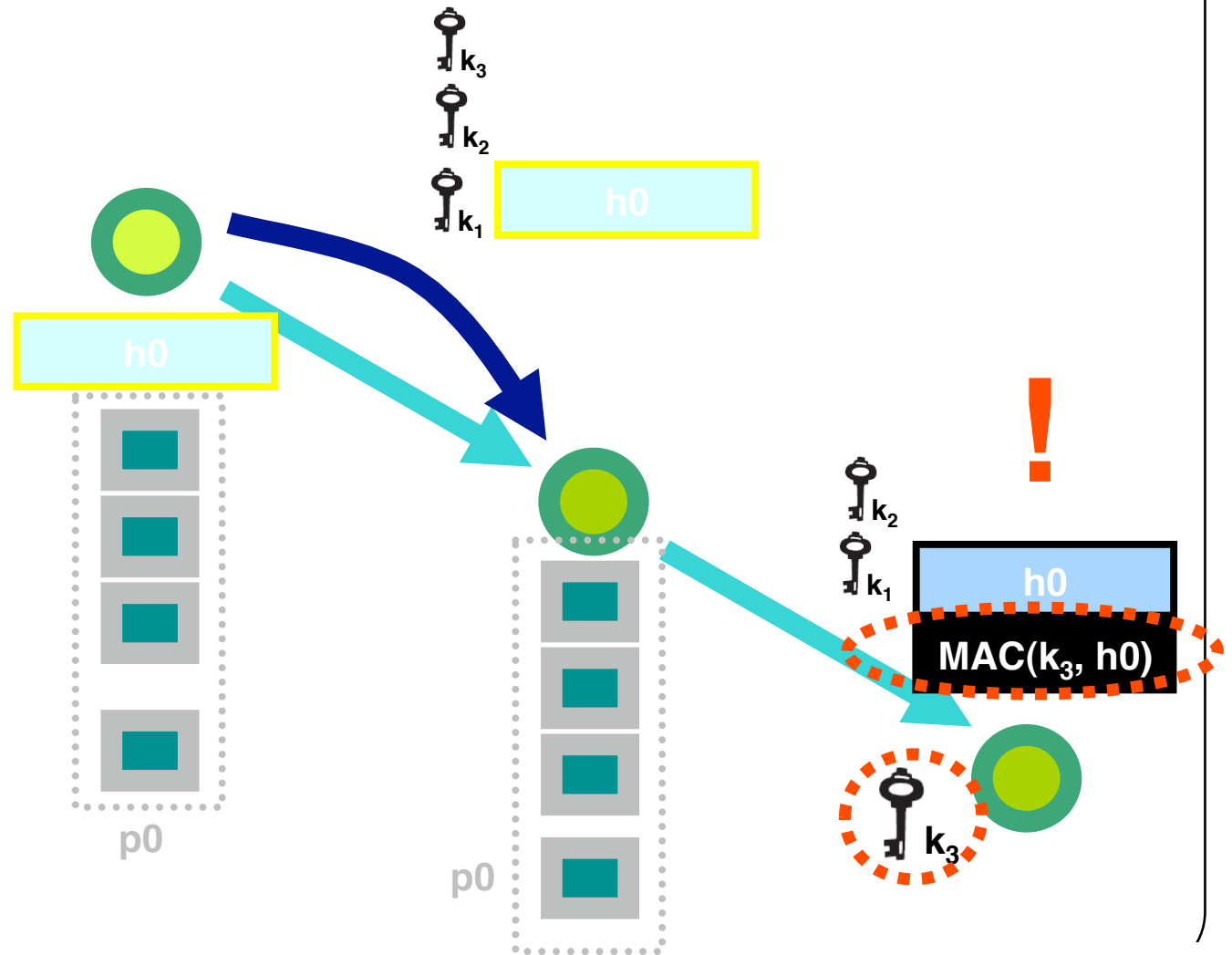


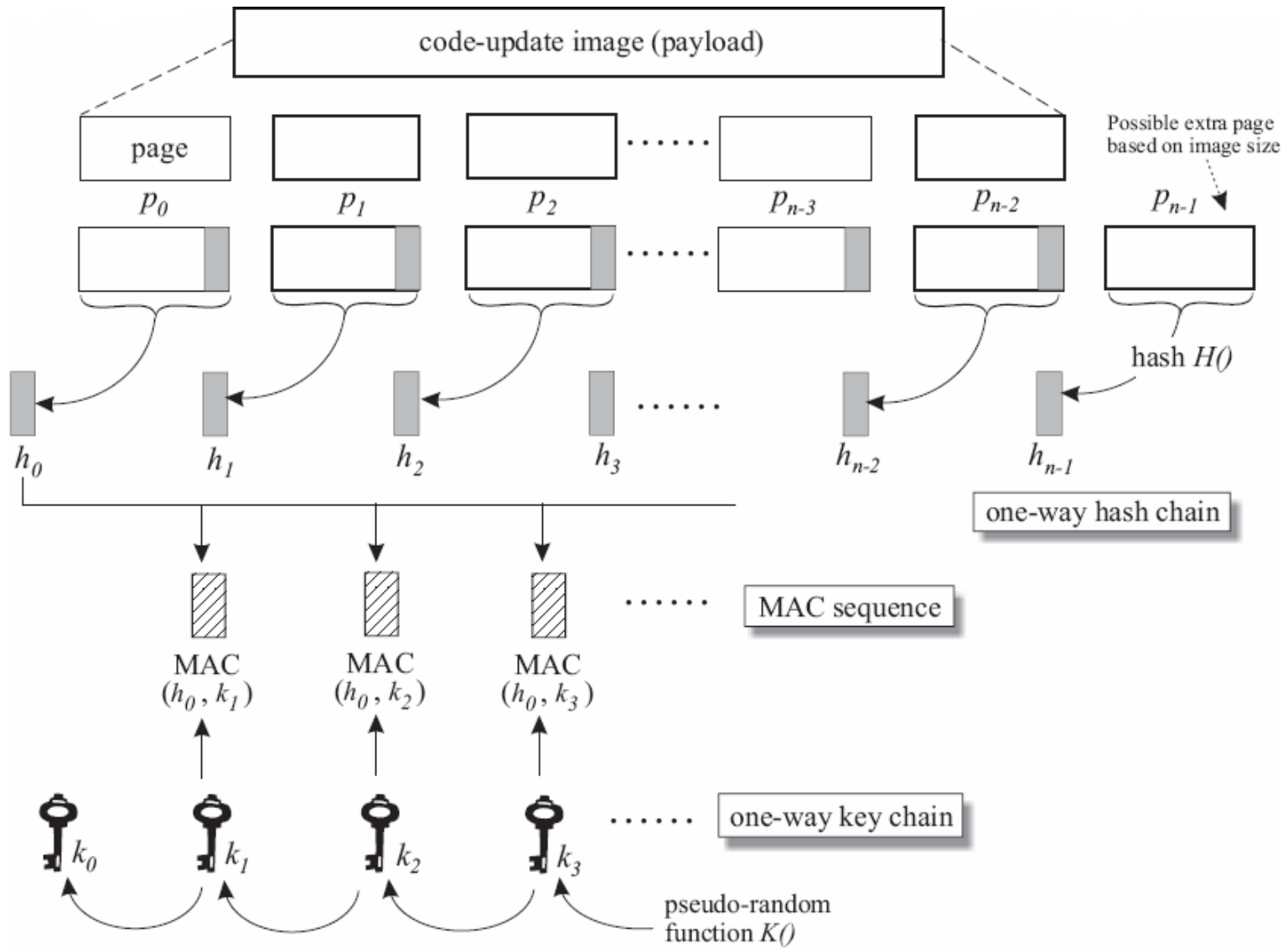












## *Castor: Summary*

- Authenticates code-updates with MAC
  - Exploits delayed key disclosure scheme to protect from compromised sensor nodes
- Uses a sequence of MAC
  - Ensures that all of the nodes are able to verify the MAC without adversely impacting the end-to-end update latency

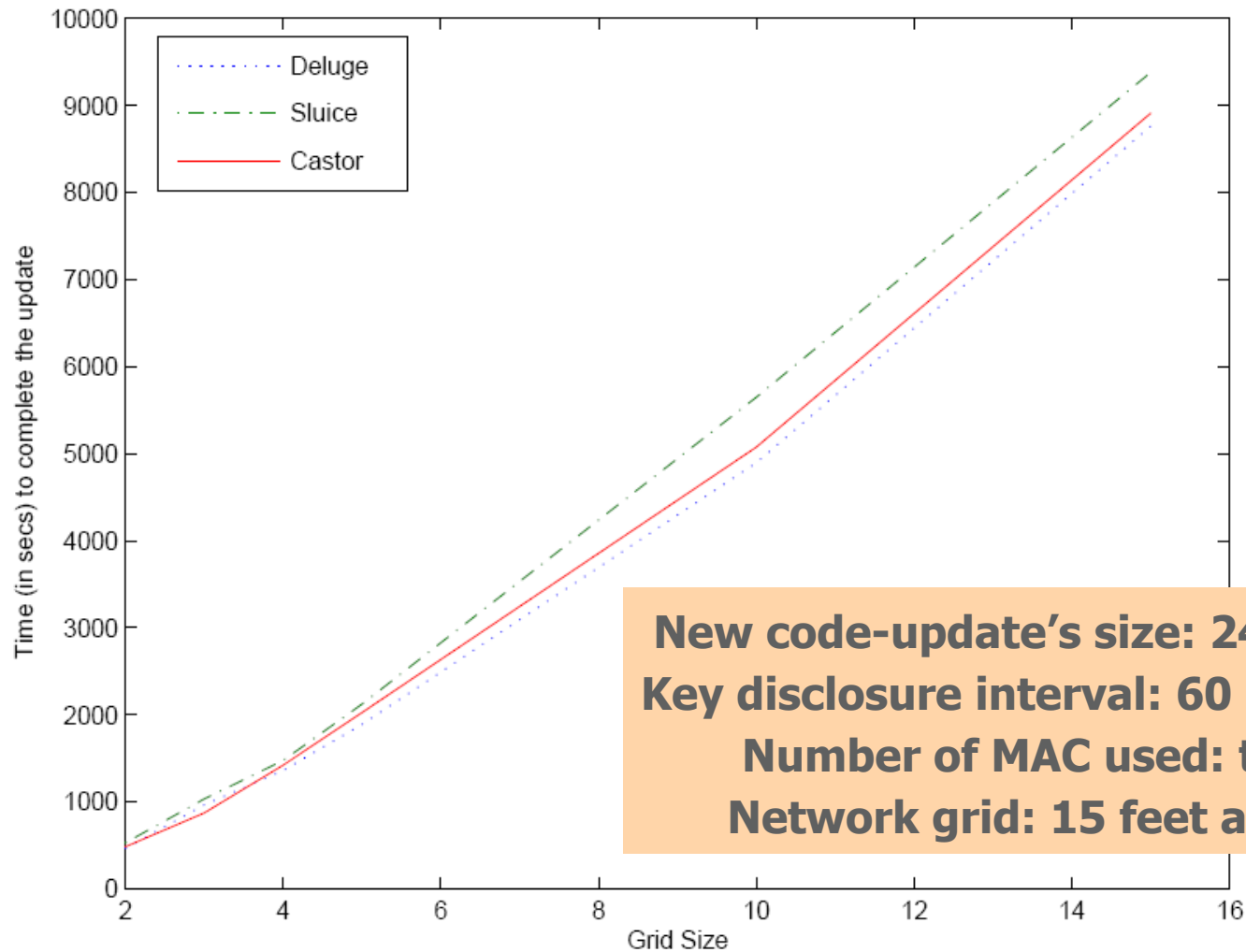
## *Implementation*

- Wrote as an extension of Deluge
- Hash: 160-bit SHA1 digests
- Block cipher: RC5
- MAC: 4-byte CBC-MAC and 8-byte key
- Generated separate packets to disseminate crypto-materials

## *Evaluation*

- Used TOSSIM to assess Castor's overhead
- Network topology
  - N by N grid topology (N varying from 2 to 15, 15ft)
  - Generated by `LossyBuilder` (based on empirical measurements)

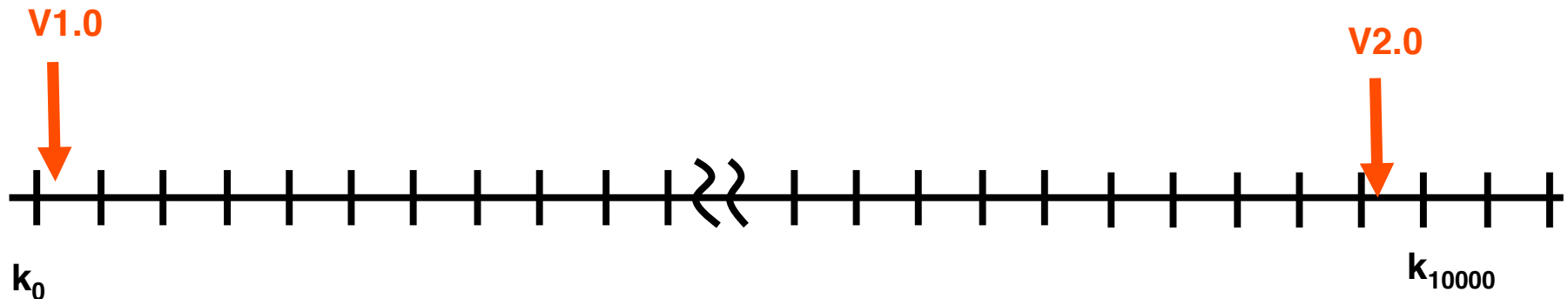
## Result – end-to-end update latency



**New code-update's size: 24 pages**  
**Key disclosure interval: 60 seconds**  
**Number of MAC used: two**  
**Network grid: 15 feet apart**

## Looking ahead

- Castor's scheme might be computationally inefficient across multiple distinct updates



How many pseudo-random function operations are required to verify  $k_{10000}$ ?

**Additional one-way key-chain**

## *Conclusion*

- Existing secure network reprogramming protocols make use of asymmetric cryptography
- Castor
  - Exploits symmetric cryptography, instead
  - Low end-to-end latency
  - Low energy consumption
  - Guarantees only legitimate senders to update the network

*Thank You*

**Questions?**

[dhjkim@cs.cmu.edu](mailto:dhjkim@cs.cmu.edu)

**[www.cs.cmu.edu/~dhjkim](http://www.cs.cmu.edu/~dhjkim)**