

Castor: Secure Code Updates using Symmetric Cryptosystems

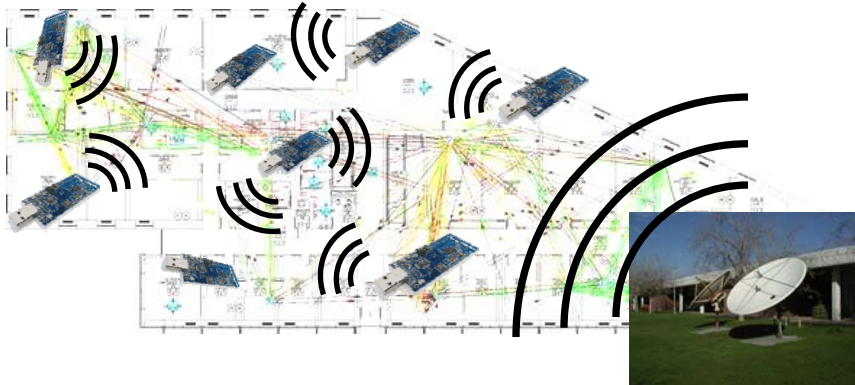
Donnie H. Kim
Rajeev Gandhi
Priya Narasimhan
Carnegie Mellon



Carnegie Mellon

Network Reprogramming

- Network (re)programming or code update protocols
 - ▼ Sensor networks are long-lived
 - ▼ Program code might need updates
 - ▼ (Re)programming the code on a number of sensors *en masse* via radio channel



2

Castor: Symmetric-Crypto Code Update Protocol

Motivation

- Several code-update protocols have emerged
 - ▼ Deluge (bundled with TinyOS), Infuse, MNP, ...
- Epidemic dissemination of the update
 - ▼ Update propagates from one sensor node to the next until all nodes have received the new version of the code
- Protocols have focused (so far) on efficient and reliable update propagation using a variety of mechanisms
 - ▼ Message suppression, sender selection, propagation of *fragments* (instead of entire images) and pipelining

3

Castor: Symmetric-Crypto Code Update Protocol

The Problem

- Epidemic protocols give rise to security concerns
 - ▼ A single malicious node can hijack the entire protocol and compromise all of the nodes in the network
- Research questions
 - ▼ Can we ensure that correct nodes in the system will never install a compromised/corrupt update?
 - ▼ Can we do this using symmetric cryptosystems?
 - ▼ Symmetric cryptosystems are lightweight as compared to asymmetric cryptosystems (i.e., digital signatures)
- And Castor was born

4

Castor: Symmetric-Crypto Code Update Protocol

Goals of Castor

- **Authenticity**
 - ▼ Verify images as originating from a trusted source
 - ▼ Stop unauthorized code from propagating through the network
- **Efficiency**
 - ▼ Allow code-update protocols to exploit existing mechanisms such as pipelining for faster propagation
- **Progressive verification**
 - ▼ Enable piece-wise verification of fragments as they are propagated
 - ▼ Do not wait for entire program image to arrive in order to detect a malicious update
- **Resource sensitivity**
 - ▼ Amortize security overheads over an entire program image
- **Ideally, no node left behind**
 - ▼ Every node receives the cryptographic material in time for verification

7

Castor: Symmetric-Crypto Code Update Protocol

System Model

- **Threat model**
 - ▼ Individual sensor nodes are untrusted, and might exhibit arbitrary behavior
 - ▼ Update originates from a trusted base station
 - ▼ An adversary can compromise an arbitrary number of sensor nodes in the system
- **NO assumptions about**
 - ▼ Number of malicious nodes, their locations, the degree of connectivity or collusion between them
- **Confidentiality of an update is outside Castor's current scope**
 - ▼ Current focus is instead on authenticating the trusted source of the update and establishing the integrity of the update image

8

Castor: Symmetric-Crypto Code Update Protocol

Twigs (Building Blocks)

■ One-way chains

- ▼ Involves a one-way function $G(\)$ that is easy to compute but hard to invert
- ▼ A one-way chain of length $m+1$ is generated by selecting an a_m randomly and then applying $G(\)$ repeatedly i.e.,

$$a_j = G(a_{j+1}) = G^{m-j}(a_m) \quad 0 \leq j \leq m-1$$

- ▼ Chain is generated in the order $a_m, a_{m-1}, \dots, a_1, a_0$ but revealed in the reverse order
- ▼ If a node confirms a_0 to be the authentic head of the chain, it can verify (but not generate) other elements of the chain by ensuring $a_0 = G^x(a_x)$

■ Twig #1 – one-way hash chain

- ▼ Elements a_i of the one-way chain are hashes



■ Twig #2 – one-way key chain

- ▼ Elements a_i of the one-way chain are keys

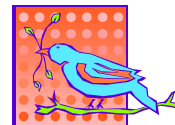
9

Castor: Symmetric-Crypto Code Update Protocol

Twigs (Building Blocks)

■ Twig #3 – symmetric-crypto authenticated broadcast

- ▼ Authenticated broadcast protocols like TESLA use symmetric keys to authenticate the sender of a message
- ▼ Source uses elements of a one-way key chain to authenticate data
- ▼ Source computes message authentication code (MAC) of the data using an unused key in the key-chain without disclosing the key
 - ▼ Allow data and MAC to propagate to all the nodes of the network
- ▼ Source discloses the keys at regular pre-defined interval
- ▼ If a node receives the data and the MAC before the source has revealed the key, it can verify the authenticity of the data by verifying that the disclosed key is an element of the one-way key chain



10

Castor: Symmetric-Crypto Code Update Protocol

Loose Time Synchronization

- Requires loose time synchronization between all the nodes of the network
 - ▼ Enables each node to verify whether it received the data before key was disclosed
- Since Castor uses authenticated broadcast, we assume that all the nodes are loosely and securely time synchronized
- If the underlying application (e.g. localization) already uses time-synchronization, Castor can simply exploit this facility

11

Castor: Symmetric-Crypto Code Update Protocol

Castor in a Nutshell

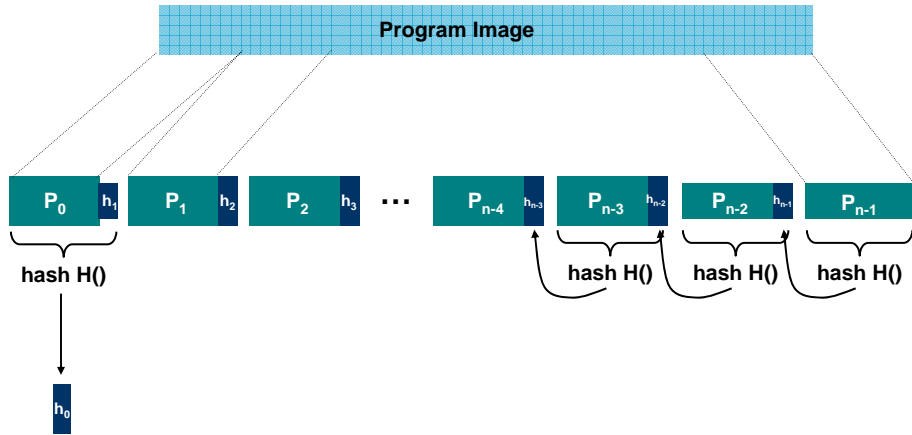
- Divide the update image into fragments (called pages)
- Uses a hash-chain \mathbf{h} across the pages of the image, with h_0 (the head element) being the hash of the first page of the update
- The rest of Castor's mechanisms ensure that h_0 is a trusted value that can indeed serve as a legitimate commitment to all of \mathbf{h}
- Rest of Castor's mechanisms
 - ▼ m -element one-way verification key-chain, \mathbf{k}
 - ▼ s -element sequence, \mathbf{M} , of MACs of h_0 where each MAC is computed over h_0 using a distinct element of \mathbf{k}
 - ▼ One-way backbone key-chain, \mathbf{K} , to mitigate the computational costs associated with authenticating successive update-images,
- Delayed key-disclosure intervals
 - ▼ T seconds for \mathbf{k}
 - ▼ T' seconds for \mathbf{K}



12

Castor: Symmetric-Crypto Code Update Protocol

Overview of Approach

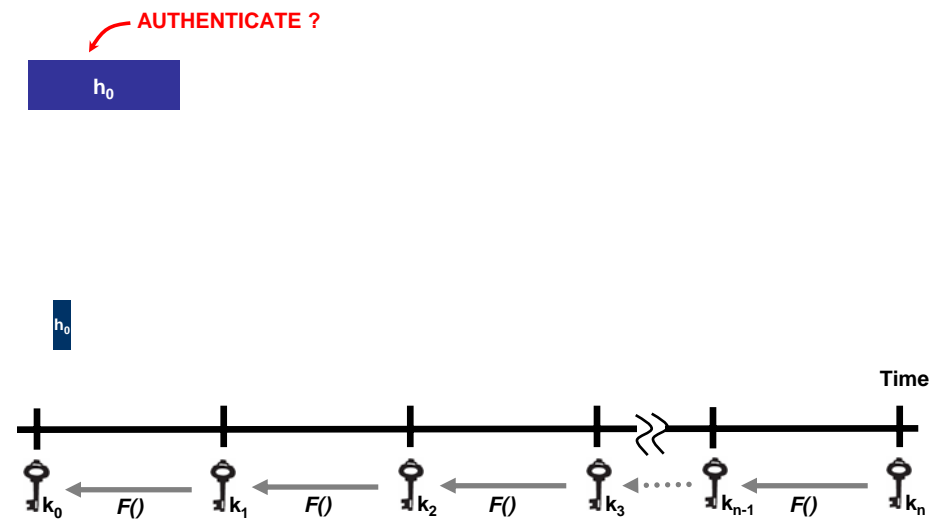


The problem of authenticating the entire update reduces to that of authenticating h_0

13

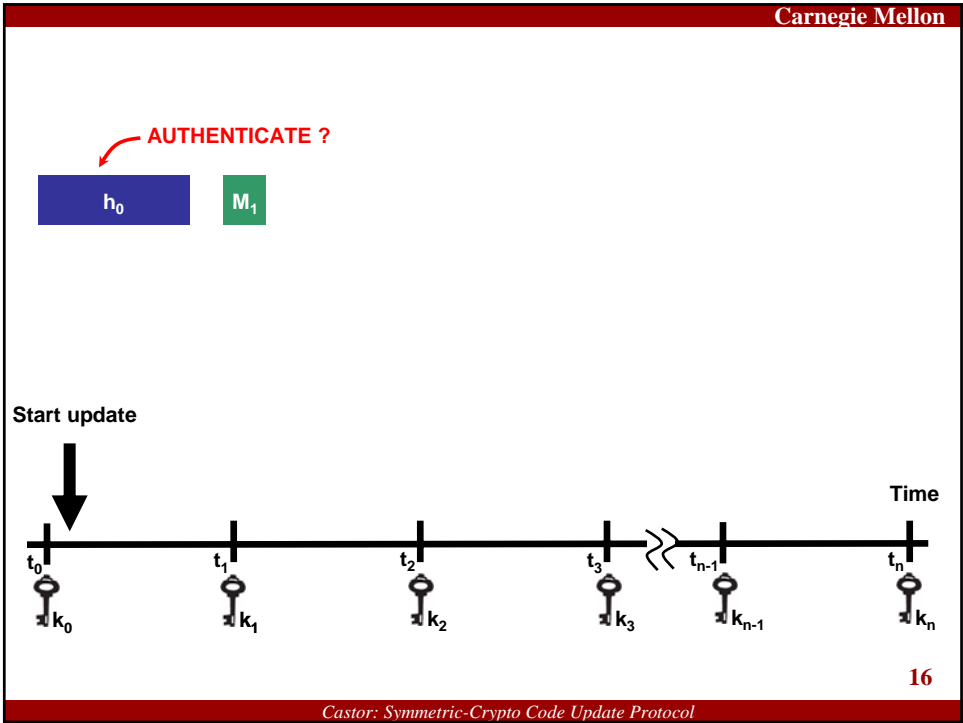
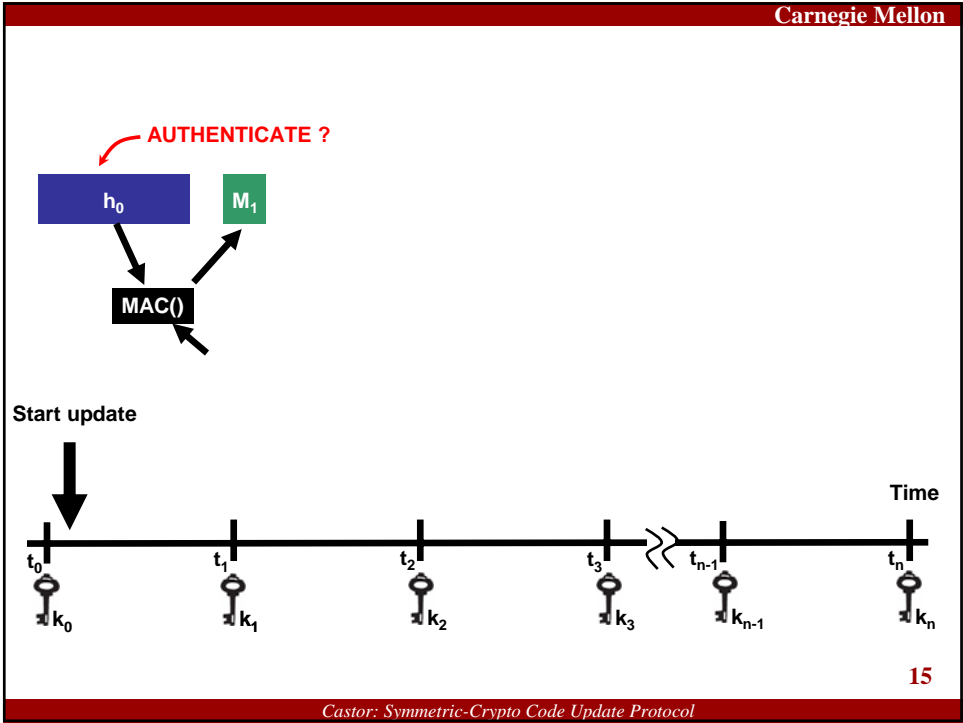
Castor: Symmetric-Crypto Code Update Protocol

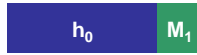
Use Message Authentication Codes to Verify h_0



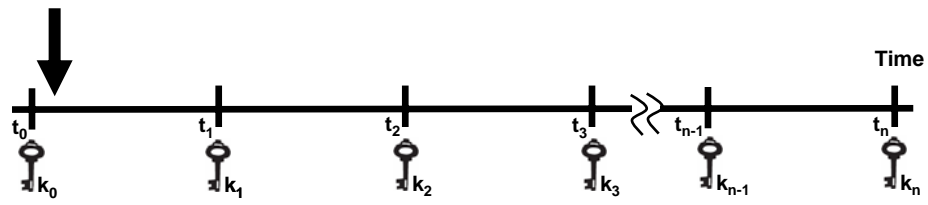
14

Castor: Symmetric-Crypto Code Update Protocol

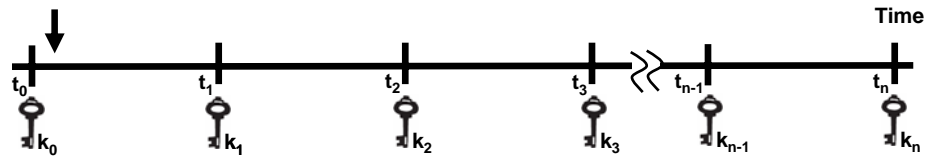




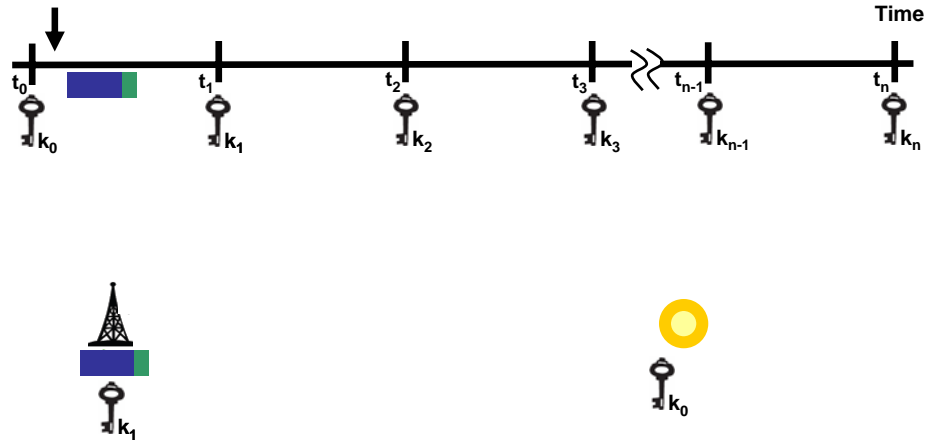
Start update



17



18



19

Castor: Symmetric-Crypto Code Update Protocol

Castor in a Nutshell

- Rest of Castor's mechanisms
 - ▼ m -element one-way verification key-chain, \mathbf{k}
 - ▼ s -element sequence, \mathbf{M} , of MACs of h_0 where each MAC is computed over h_0 using a distinct element of \mathbf{k}
 - ▼ One-way backbone key-chain, \mathbf{K} , to mitigate the computational costs associated with authenticating successive update-images,
- Delayed key-disclosure intervals
 - ▼ T seconds for \mathbf{k}
 - ▼ T' seconds for \mathbf{K}



20

Castor: Symmetric-Crypto Code Update Protocol

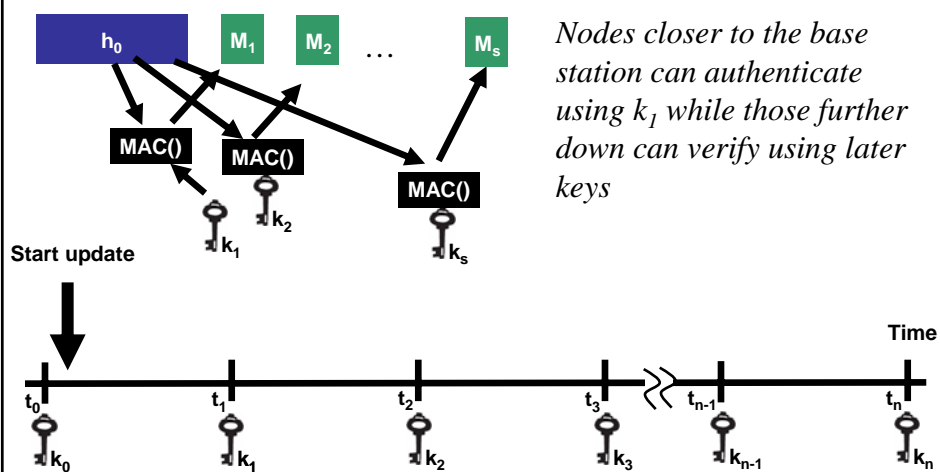
Challenges in Implementation

- Problem – how do we determine the key disclosure interval (\mathcal{T})?
- Deluge and other network reprogramming protocols do not place a bound on the propagation time of a packet/page
 - ▼ Choose a large value of \mathcal{T} so that all the nodes
- A large value of \mathcal{T} increases the end-to-end update latency
- A small value of \mathcal{T} will prevent nodes from verifying the authenticity of h_0
 - ▼ Because they receive the packet containing h_0 and its MAC after \mathcal{T} seconds

21

Castor: Symmetric-Crypto Code Update Protocol

Solution – Use Multiple MACs



22

Castor: Symmetric-Crypto Code Update Protocol

Castor in a Nutshell

- Rest of Castor's mechanisms
 - ▼ m -element one-way verification key-chain, \mathbf{k}
 - ▼ s -element sequence, \mathbf{M} , of MACs of h_0 where each MAC is computed over h_0 using a distinct element of \mathbf{k}
 - ▼ One-way backbone key-chain, \mathbf{K} , to mitigate the computational costs associated with authenticating successive update-images,
- Delayed key-disclosure intervals
 - ▼ T seconds for \mathbf{k}
 - ▼ T' seconds for \mathbf{K}



23

Castor: Symmetric-Crypto Code Update Protocol

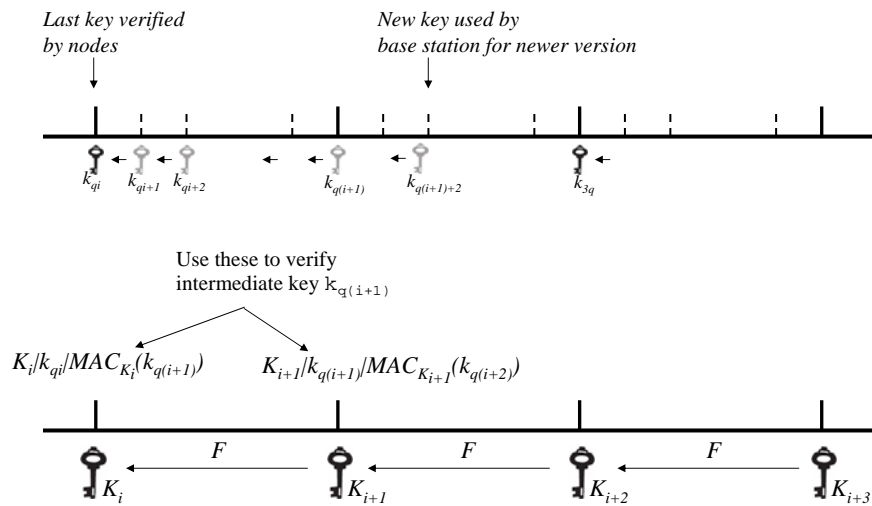
Multiple Updates/Versions

- If updates are infrequent, then verifying whether a key is part of a one-way key chain can become expensive
- Example – If version 2.0 of the code used \mathbf{k}_{100} and version 3.0 of the code used \mathbf{k}_{1000} then we need to perform 900 hash computations to verify that \mathbf{k}_{1000} is a part of the one-way key chain
- Solution – use another key chain \mathbf{K} (called backbone key chain in the paper) to verify intermediate keys of the key chain
 - ▼ The time disclosure interval T' of the keys in the backbone key chain is much larger than T (assumed to be a multiple of T i.e. $T' = \alpha T$)

24

Castor: Symmetric-Crypto Code Update Protocol

Multiple Updates/Versions



25

Castor: Symmetric-Crypto Code Update Protocol

Implementation Details

- Implemented Castor as an extension to Deluge in TinyOS
- Hash: 160-bit SHA1 digests
- MAC: 4-byte CBC-MAC based on RC5 block cipher
- Symmetric keys: 8-bytes

- Extended Deluge's advertisement packet to include the MACs and the keys

26

Castor: Symmetric-Crypto Code Update Protocol

Empirical Evaluation

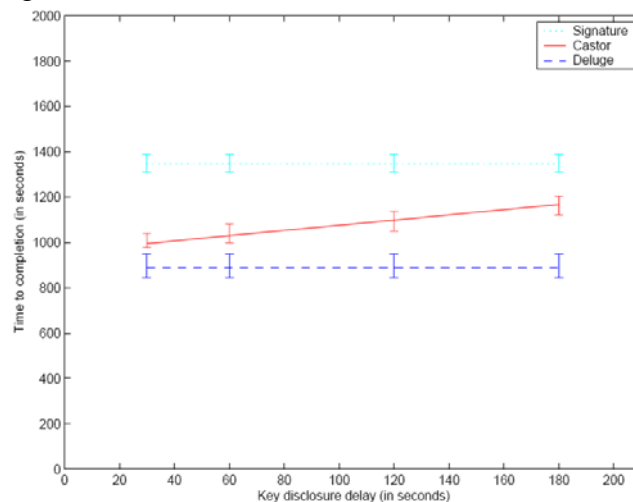
- Used TOSSIM to simulate Castor's secure dissemination of code updates
- Simulation topologies consisted of sensor nodes spaced 15 feet apart on an $N \times N$ grid (N varied from 10 to 20)
- Metrics of Interest
 - ▼ **Propagation performance** – end-to-end latency of disseminating the update
 - ▼ **Communication overhead** – communication cost of disseminating crypto-materials

27

Castor: Symmetric-Crypto Code Update Protocol

End-to-End Update Latency With a Single MAC

End-to-end latency of disseminating a 4-page update using a single MAC

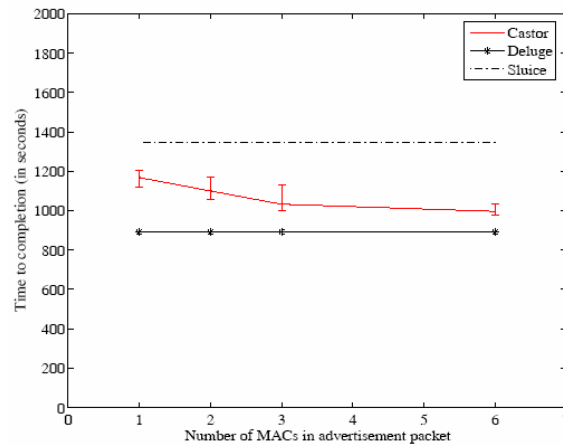


28

Castor: Symmetric-Crypto Code Update Protocol

End-to-End Update Latency With Multiple MACs

End-to-end latency of disseminating a 4-page update using multiple MACs with key disclosure interval of 30 seconds

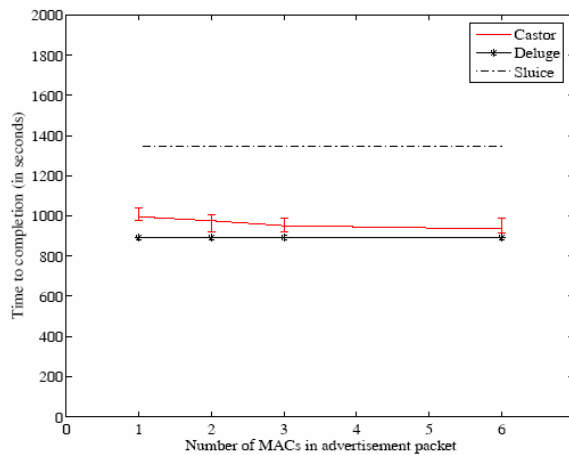


29

Castor: Symmetric-Crypto Code Update Protocol

End-to-End Update Latency With Multiple MACs

End-to-end latency of disseminating a 4-page update using multiple MACs with key disclosure interval of 5 seconds

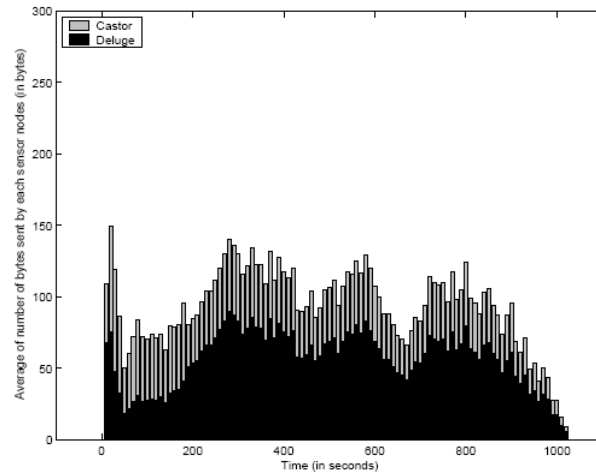


30

Castor: Symmetric-Crypto Code Update Protocol

Communication Overhead

Communication overhead of disseminating a 4-page update using 6 MACs with key disclosure interval of 30 seconds

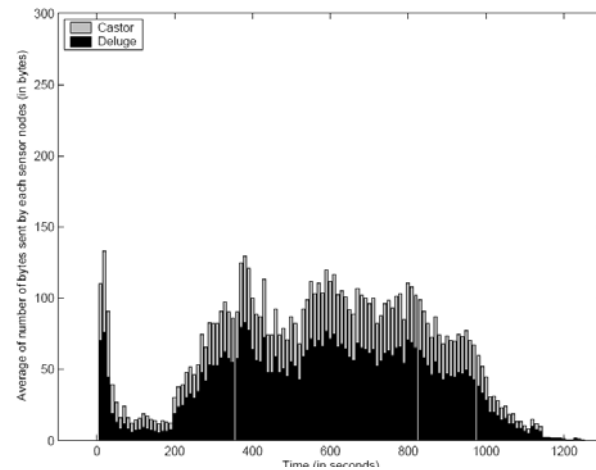


31

Castor: Symmetric-Crypto Code Update Protocol

Communication Overhead

Communication overhead of disseminating a 4-page update using 1 MAC with key disclosure interval of 180 seconds



32

Castor: Symmetric-Crypto Code Update Protocol

Related Work



- Castor – first symmetric-crypto update protocol
- Our previous work, Sluice [ICDCS 2006], uses a one-way hash chain and a single digital signature to verify the authenticity of an update
- Secure Deluge [IPSN 2006], like Sluice, uses a one-way hash chain and a single digital signature to verify the authenticity of an update
 - ▼ Unlike Castor, hash chain is computed over packets rather than pages
- Deng et. al. [IPSN 2006] use a one-way hash chain computed over packets and a single digital signature to verify the authenticity of an update
 - ▼ Use a hash tree of per-packet hashes to allow for out-of-order arrival of packets in a page

33

Castor: Symmetric-Crypto Code Update Protocol

Summary

- Secure code-update protocols for sensor networks
 - ▼ **Castor**: Provides source authentication for code updates
 - ▼ **Castor** : Uses a synergistic combination of one-way hash-chains, one-way key-chains and authenticated broadcast
 - ▼ **Castor**: Provides lower end-to-end update latency as compared to Sluice
- Potential future directions
 - ▼ Castor's main drawback is that nodes have to propagate the advertisement packet (containing the MAC) without authenticating the advertisement packet
 - ▼ Can lead to a potential DoS attack where an adversary sends advertisement packets with incorrect MACs
 - ▼ Currently looking at ways to minimize the number of nodes that forward an unauthenticated advertisement packet

34

Castor: Symmetric-Crypto Code Update Protocol

References

- Exploring Symmetric Cryptography for Secure Network Reprogramming, Donnie H. Kim, Rajeev Gandhi, and Priya Narasimhan, *Workshop on Wireless Ad-hoc and Sensor Networks (WWASN)*, Toronto, Canada (June 2007)
- Sluice: Secure Dissemination of Code Updates in Sensor Networks, Patrick E. Lanigan, Rajeev Gandhi, Priya Narasimhan, *International Conference on Distributed Computing Systems (ICDCS)*, Lisbon, Portugal (July 2006)
- Disseminating Code Updates in Sensor Networks: Survey of Protocols and Security Issues, Patrick E. Lanigan, Rajeev Gandhi, Priya Narasimhan, *Technical Report CMU-ISRI-05-122*, ISRI, Carnegie Mellon University, October 2005.

35

Castor: Symmetric-Crypto Code Update Protocol

For More Information

<http://www.ece.cmu.edu/~rgandhi>

36

Castor: Symmetric-Crypto Code Update Protocol

Thank You!

Questions

