

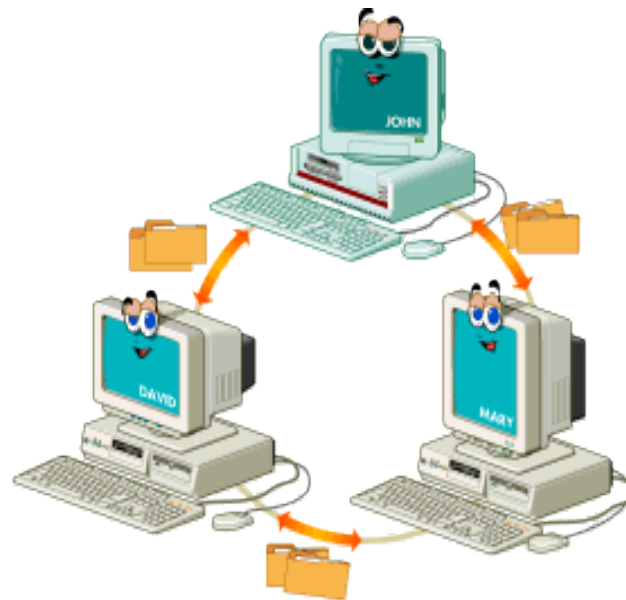
Massimiliano "Max" Pala <pala@cs.dartmouth.edu>  
Sean W. Smith <sws@cs.dartmouth.edu>

# PEACHES and Peers

... or how to globally distribute Pointers to PKI Resources



&



# Presentation Outline

---

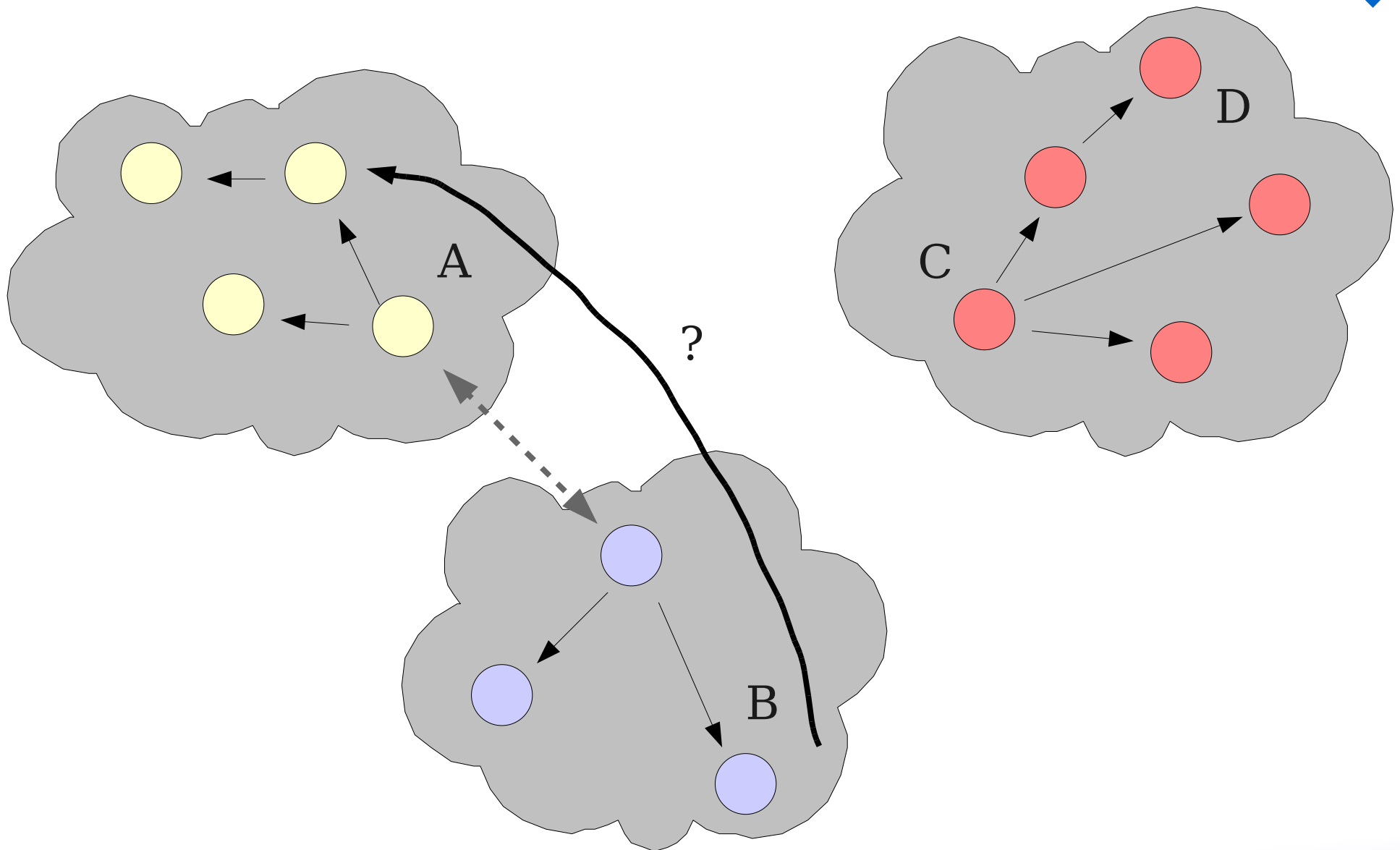
- **Motivations and Background**
  - PKI Resource Location
  - The PKI Resource Query Protocol
  - Distributed Systems & DHTs
- **The PEACH Protocol**
  - Certificate-Based node identifiers
  - Lookup & Join in PEACH
- **PEACH enabled PKI System**
  - Network Details
  - Integrating PEACH and PRQP
- **Performance Evaluation**
- **Conclusions and Future Work**

# Motivations

---

- Finding services and data is crucial for *PKI usability!*
- PKI resources are difficult to discover
  - **in particular for people alien to the issuing CA**
- User awareness about configuration options is way below any acceptable level
  - Configuration issues
- We need a way to *map services URLs to PKC*
  - *Usable*
  - *Distributed*

# Resource Discovery in PKIs



# PKI Resource Discovery

---

- Many Different Possibilities
  - Certificate Extensions
  - DNS Records
  - Webservices
  - Local Network Oriented Solutions
- Not all *needs* are currently met
  - Where is the http revocation system for my certificate ? Where is the CMS interface ?
  - Where can I renew my certificate ?

# Technical Trustworthiness

---

- We need to ensure **Technical Trustworthiness**, by providing an efficient method to find PKI resources:
  - Certificate and CRL repositories
  - Validation Services (OCSP)
  - Additional Services (e.g., Timestamping, DVCS, etc..)
- By automatically enabling the **effective use all the services provided by a CA**, the user:
  - Have not to worry about difficult (and often unknown) configuration options
  - Can take high-level trust decisions based on the level of security offered by available services or personal knowledge

# PKI Resource Query Protocol (PRQP)

---

- Allows a client to request services and repositories provided by a CA
- Provides “*discovery*” for any services (current and future)
  - CRL and Cert Repositories
  - Validation/Verification services (eg. OCSP, SCVP,..)
  - TimeStamping
  - Revocation/Subscription Service
  - Future (still not available) services

# Distributing Information

---

- ***Client/Server*** Model
  - Web Services
  - FTP
- Collaborative approaches: the ***Peer-to-peer*** Model
  - More scalable
  - Napster, Gnutella and Kazaa
- 2<sup>nd</sup> Generation Peer-to-peer Model: ***DHTs***
  - Chord
  - CAN
  - Tapestry, Kademilia and P-Grid

# The PEACH protocol

---

- *PKI Easy Auto-Discovery Collaborative Hash-Table*
  - Based on **Chord**
  - Use *Certificate-Based identifiers* for *join()* operation
  - Use a pointer table to optimize the number of hops during the *lookup()* operation
- Participating peers are **Resource Query Authorities**
  - Each RQA will provide responses for the CA they are associated with

# The PEACH protocol (cont.)

- Each node keeps a table of pointers
  - *Divides the network into progressively growing slices*
  - *If SHA-256 is used, the table has 256 entries*

Entry ID	Node ID	Network Addr	Port	Join_Data
$(ID_n + 2^i) \bmod 2^m$	$net\_id_p$	192.168.50.100	2561	[ PKCS#7 Data]

i-th entry

Network Identifier  
for node pointed by  
entry i-th (k)

# The PEACH protocol (cont.)

---

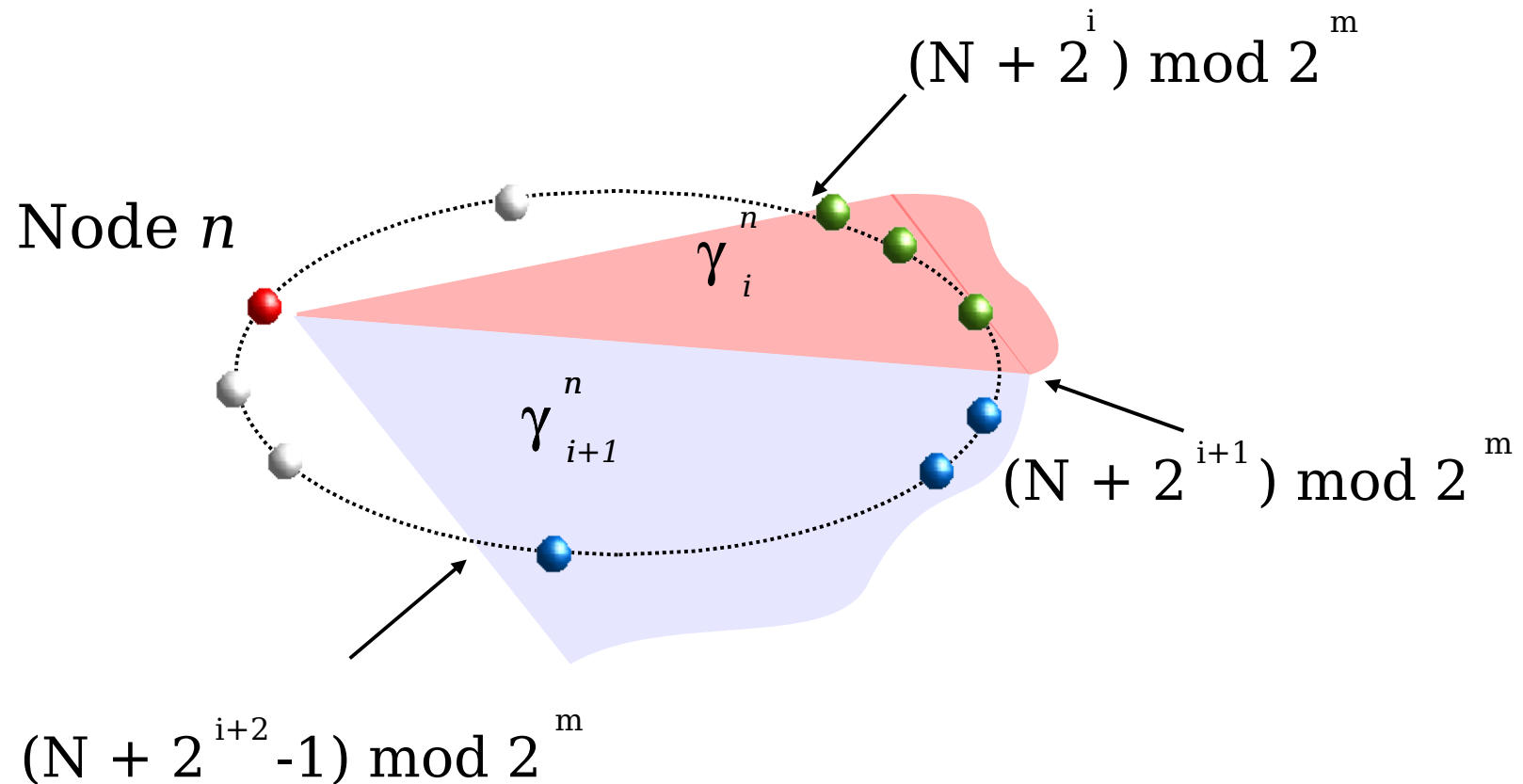
- Each Entry in the Pointers table identify a routing space for Node IDs
- *For Entry:*

$$x_i^n = (id_n + 2^i) \text{ mod } 2^m$$

- *The node ID space is:*

$$Y_i^n = [x_i^n, x_{i+1}^n)$$

# The PEACH Network



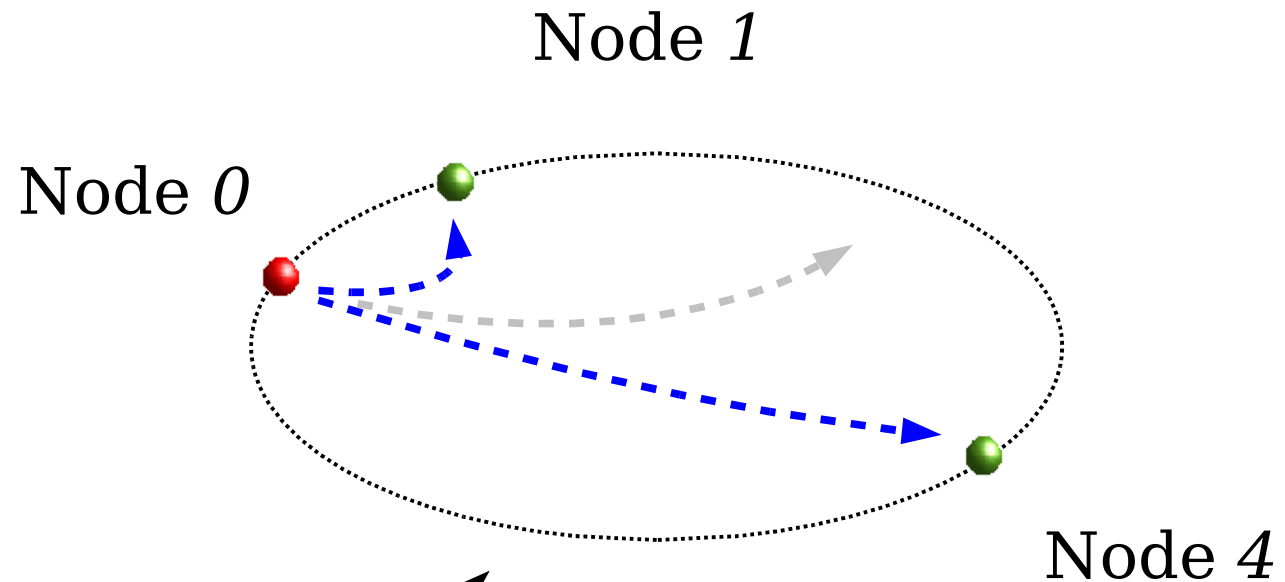
# Example

- Simple Network for 8 nodes (3 bits) - (0,1,4)
  - $ID=0$  (Node Identifier)
  - $M=3$  (bits)
  - $(ID_n + 2^0) \bmod 3$

Entry	Node ID	Network	Port	Join Data
1	1	192.168.50.100	2561	[...]
2	-	192.168.50.101	2561	[...]
4	4	192.168.50.102	2561	[...]

# The PEACH Network

---



Queries for Nodes ID  $> 4$  are routed to Node 4

# Joining the PEACH

---

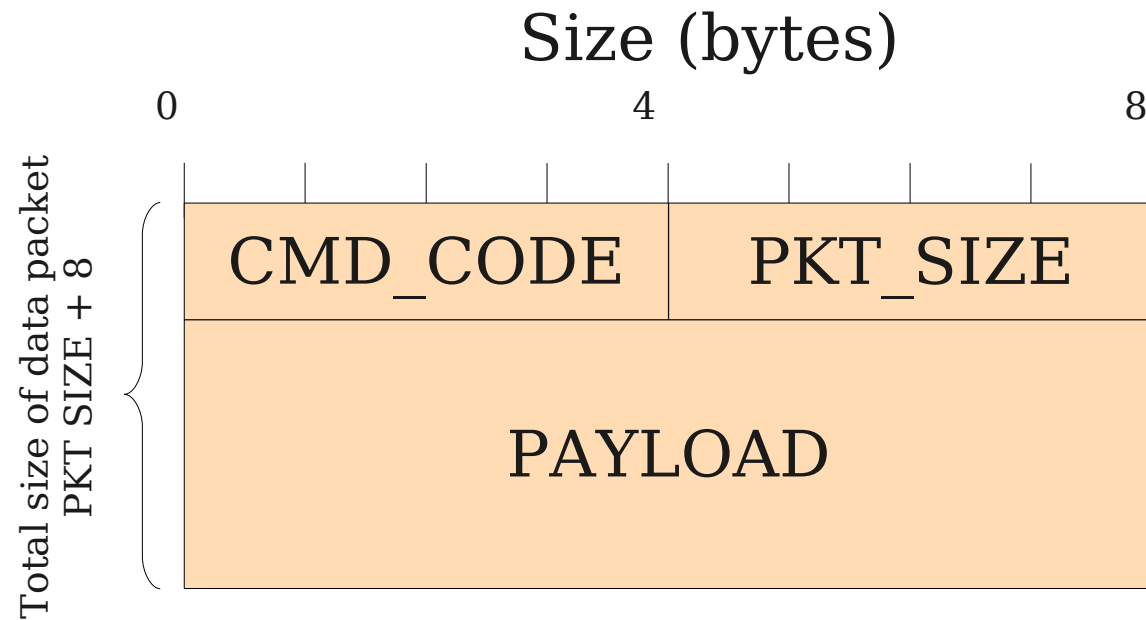
- Assumptions
  - Participating nodes are RQA
  - RQAs are issued a certificate by the CAs they are authoritative for
  - Node Identifiers
    - Build by calculating the fingerprint of the node's certificate
- To join the PEACH, a new node must:
  - Find its successor  $p$  (by performing a lookup for  $ID_{n+1}$ )
  - Send a request authorization to  $p$  (PKCS#7 signed object)
  - Inform  $p$ 's predecessor of the successful *join()*

# Joining the PEACH (cont.)

---

- Tasks for the  $p$  node
  - Verify the correct signature of the PKCS#7 data
  - Verify that the joining RQAs present a valid chain of certificates
  - Register the joining RQA with the ID calculated over the CA certificate (which is the CA the RQA is authorized to answer for)
  - Update the pointer to its *predecessor* node
- Validation (besides the revocation status of the certificate) is self-contained

# Network Communication

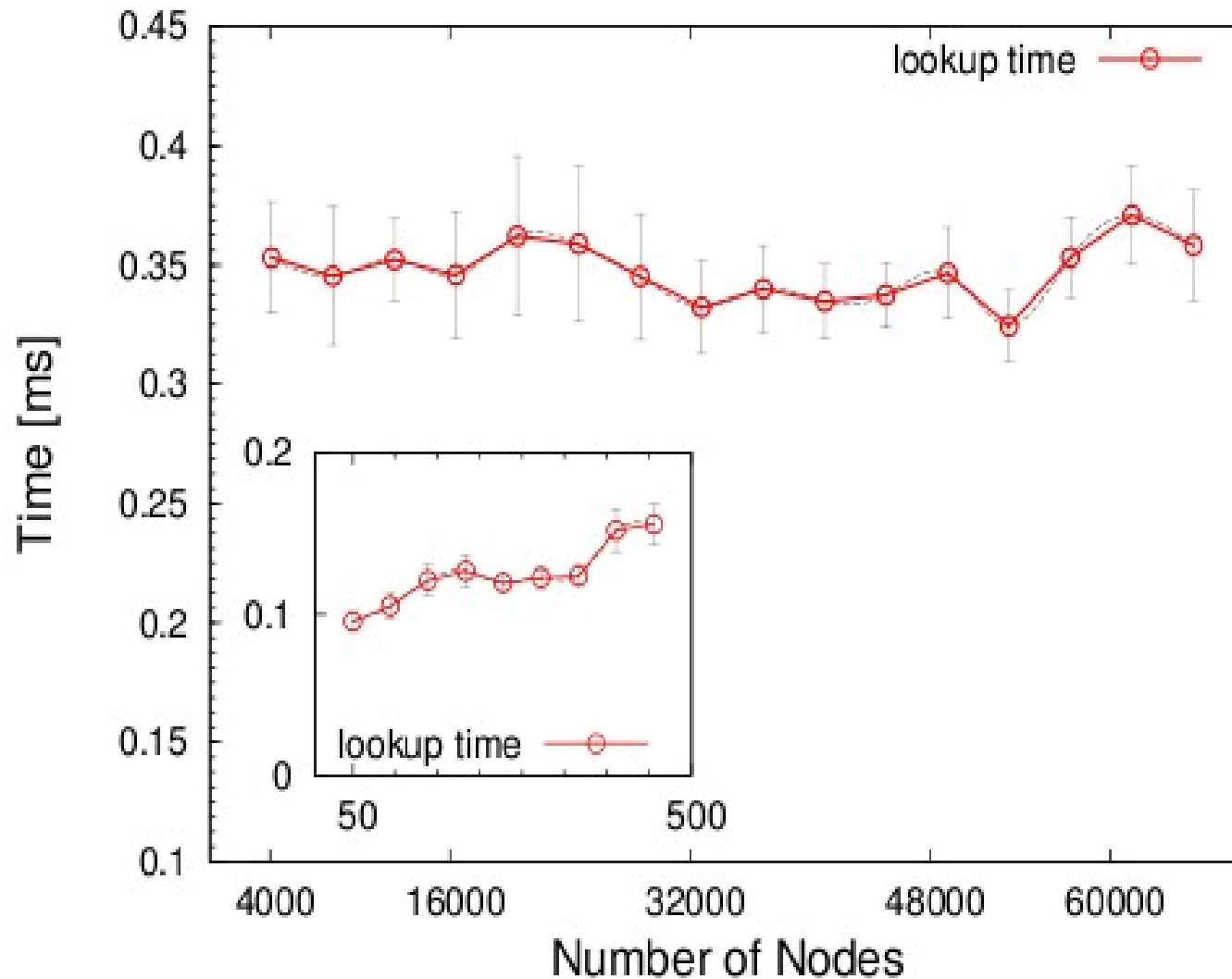


# PEACH Network Simulator

---

- Simulator Environment
  - 2.4Ghz Core-Duo Laptop (2Gb Memory)
  - Operating System is Linux 2.6.22
  - Programming language is PERL
- Simulate full PEACH protocol
  - Does not consider network-related delays
- Run *lookup()* tests on systems with nodes ranging from 50 up to 65535

# PEACH lookup() times



# Conclusions and Future Works

---

- Introduced a ***P2P PKI discovery infrastructure***
  - PKI Resource Query Protocol (PRQP)
  - Chord-based (modified) DHT
- Integrating PRQP and PEACH provides ***a mapping service*** similar to the one provided by DNS for IP addresses
- Promoting the ***standardization of PRQP*** at IETF
  - Soon on Experimental track within the PKIX WG
- Promotes ***Interoperability*** across PKIs
- Opens up ***new operational models*** (P2P) in X509 PKIs based on collaborative services

# Questions

---



## **DISCLAIMER:**

No answer are guaranteed to be either intelligent nor appropriate. If you can read this, you are probably too close to the screen or you are simply cheating. Have a nice day.

# Contacts

---

- Dartmouth College  
[pala@cs.dartmouth.edu](mailto:pala@cs.dartmouth.edu)
- OpenCA  
[madwolf@openca.org](mailto:madwolf@openca.org)
- Website  
<http://www.openca.org/projects/prqp>
- Online Web-Based demo  
<http://prqp.openca.org/prqp/>