Secure Vehicular Communication System: Design & Implementation of VPKI

(Providing Credential Management in a Secure VANET)

MSc Thesis:
Mohammad Khodaei

Supervisor:
Prof. Panos Papadimitratos

LCN – KTH

October, 2012

Outline

- ***Introduction***
  - Problem Statement
  - Contribution
  - Key Concepts
  - Security Requirements
  - Adversary Model
- ***Protocol Design***
- ***Performance Evaluation***
- ***Conclusion***
  - Future Direction

Outline

- ***Introduction***
  - Problem Statement
  - Contribution
  - Key Concepts
  - Security Requirements
  - Adversary Model
- ***Protocol Design***
- ***Performance Evaluation***
- ***Conclusion***
  - Future Direction

Introduction

- The life cycle of vehicles is pretty long
- Security has to be put in place
- Many attacks which could jeopardize the system performance from security point of view
- Mitigating unknown threats and upcoming attacks

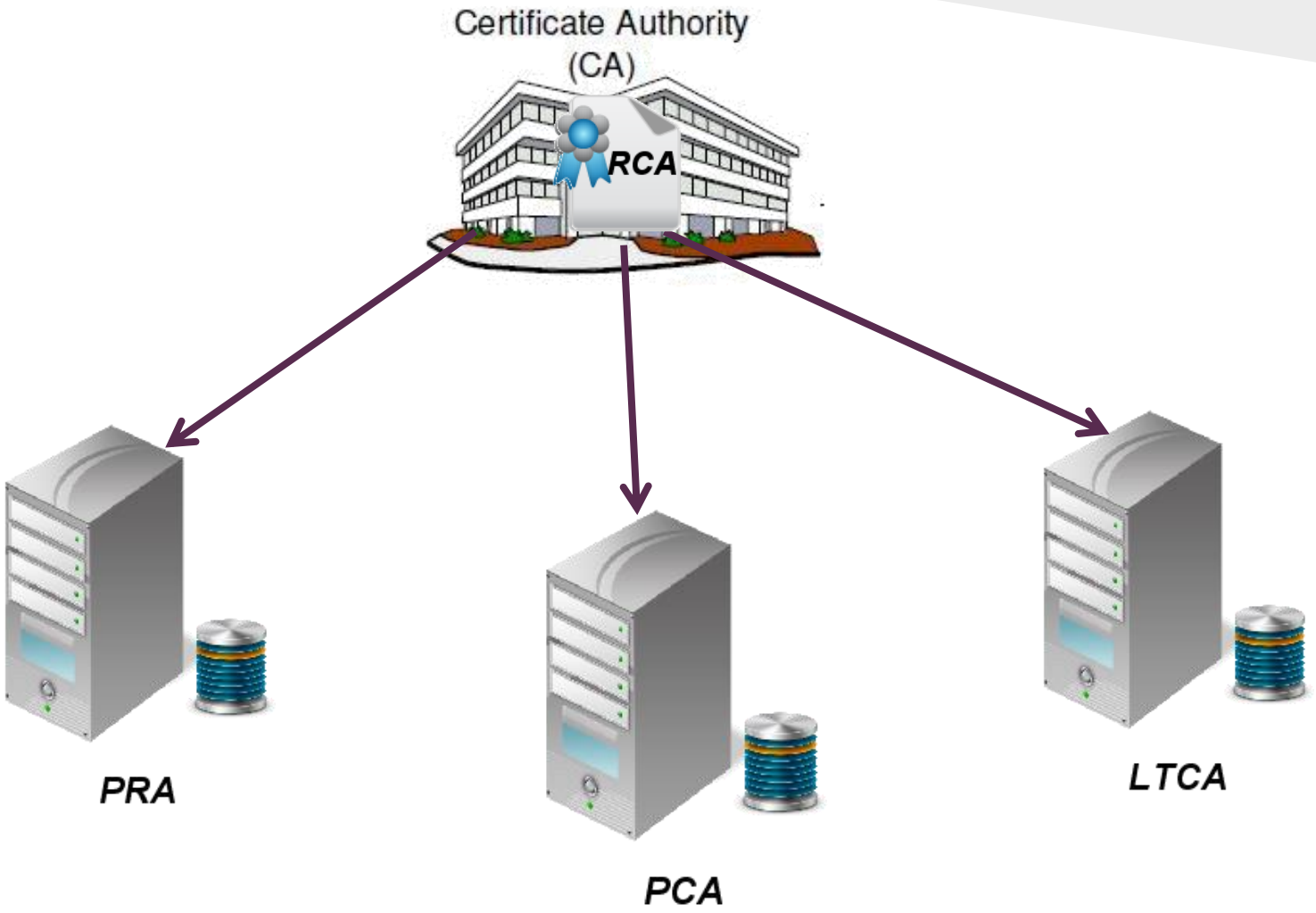
Problem

- The lack of an infrastructure
- Exposed to different threats and attacks
- Staging attacks to jeopardize users' privacy and disclose confidential information
- Exploiting the vulnerabilities
- Violating the VC system security policy
- What to do to thwart the threats and make the system operations secure?

Contribution

- **Research Purpose**
  - Design and Implementation of VPKI for the secure VC system
  - An infrastructure called VPKI, to enable entities communicate securely
  - Providing Credential Management in a Secure VC system
  - PKI is considered as an essential requirement to provide security services
- **Goal**
  - Build an artifact, using the currently available open-source PKI, OpenCA, equiped with extra protocols for VANET
- **Methodology**
  - Designing and Implementation of extra protocols for VANET
  - Using Open-Source OpenCA

Key Concepts



Key Algorithms and Size

Entities	***Algorithm***
PCA	**RSA, key size: 1024-bit** ***ECDSA, key size: 256-bit***
LTCA	**RSA, key size: 1024-bit** ***ECDSA, key size: 256-bit***
PRA	**RSA, key size: 1024-bit** ***ECDSA, key size: 256-bit***
Police	**RSA, key size: 1024-bit** ***ECDSA, key size: 256-bit***
Vehicle	**RSA, key size: 1024-bit** ***ECDSA, key size: 256-bit***

Why not normal PKI?

- Pseudonymity
- Unlinkability
- Unobservability
- User's Privacy

Security Requirements

- Message Authentication and Integrity
- Message Non-Repudiation
- Privacy
  - Anonymity
  - Unlinkability and Unobservability
- Pseudonym Resolution
  - Liability Identification, Forensics Investigation
- Message Confidentiality
- Availability, Fault-Tolerant and Robustness
- Scalability and Performance

Adversary Model

- Localized and Selective Denial of Communication
- Internal Active Adversaries
  a. Modification and Tampering
  b. Forgery
  c. Recollecting Past Messages
  d. Multiple Adversarial Nodes
- Bounded Adversarial Presence
- Input-Controlling Adversary
- Other Adversary Models (Byzantine, Dolev-Yao (DY))

Related Work

- V-Tokens for Conditional Pseudonymity in VANETs
  - Resolution information is embedded in pseudonyms
  - Vehicle signs using its current valid pseudonym
  - Pseudonym information is encrypted with PK_PR
  - Uses separation of duties
  - Cooperation of a subset of RAs is required to perform pseudonym resolution

Outline

- Introduction
  - Problem Statement
  - Contribution
  - Key Concepts
  - Security Requirements
  - Adversary Model
- ***Protocols Design***
- Performance Evaluation
- Conclusion
  - Future Direction

Protocol Design

- How to Request for Pseudonymous Certificates
- How to Request the Latest Pseudonym CRL
- How to Perform Pseudonym Resolution

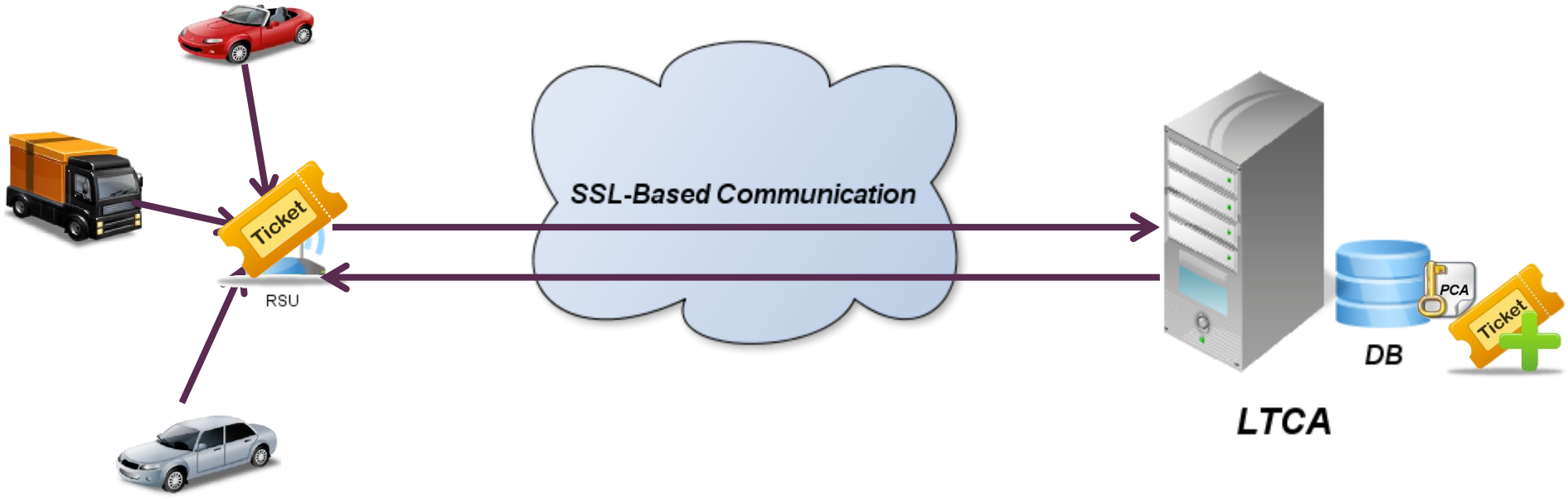
Obtaining Pseudonym Cert.

Two Steps:

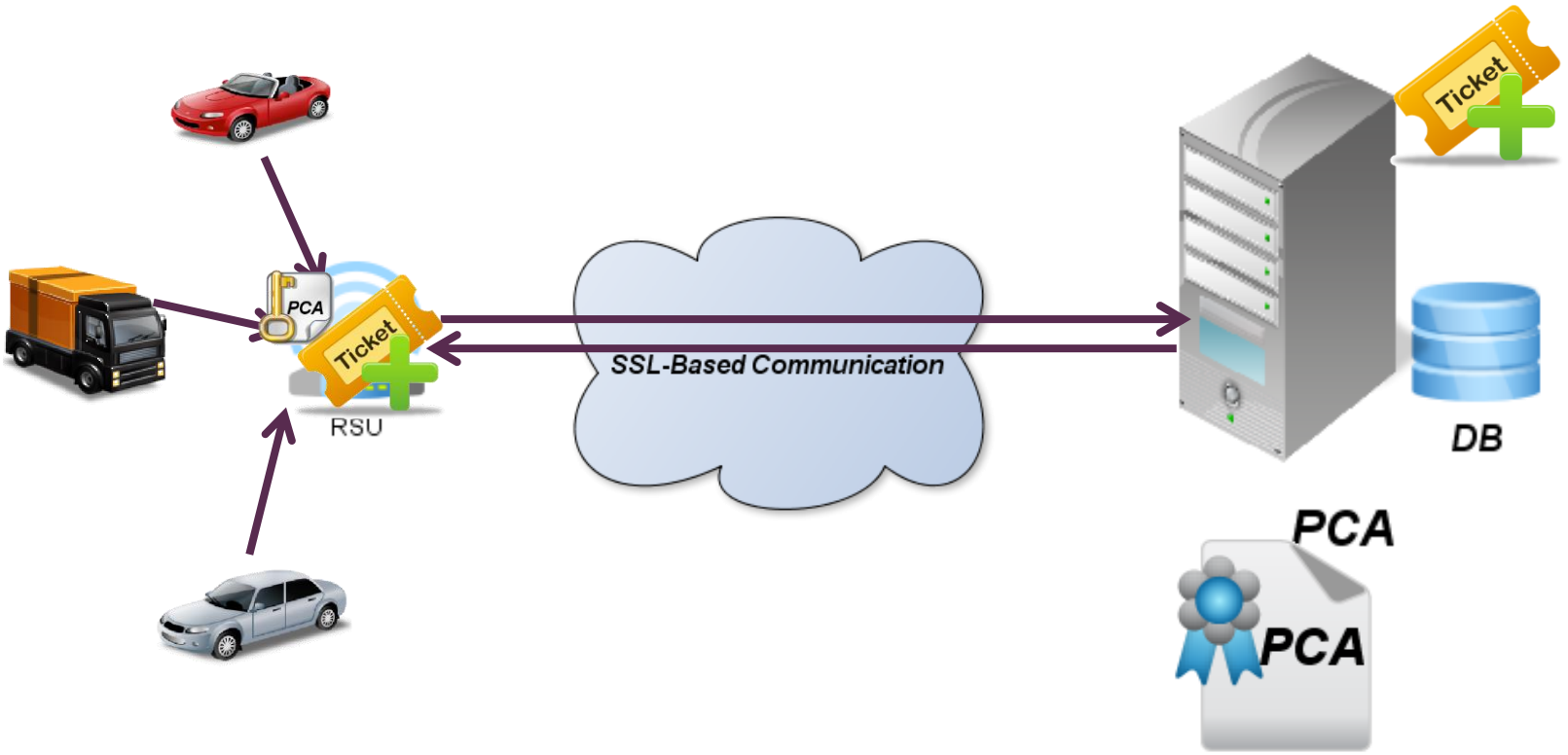
a. **Obtain a Token**
   i. *Vehicle queries LTCA*
   ii. *LTCA issues an encrypted Token with PCA's Public key, if it is a legitimate vehicle*
   iii. *Vehicle stores the Token for the second step*

b. **Obtain Pseudonymous Certificates**
   i. *Vehicle sends the Token to PCA*
   ii. *PCA verified the Token locally*
   iii. *PCA issues short-term certificate*

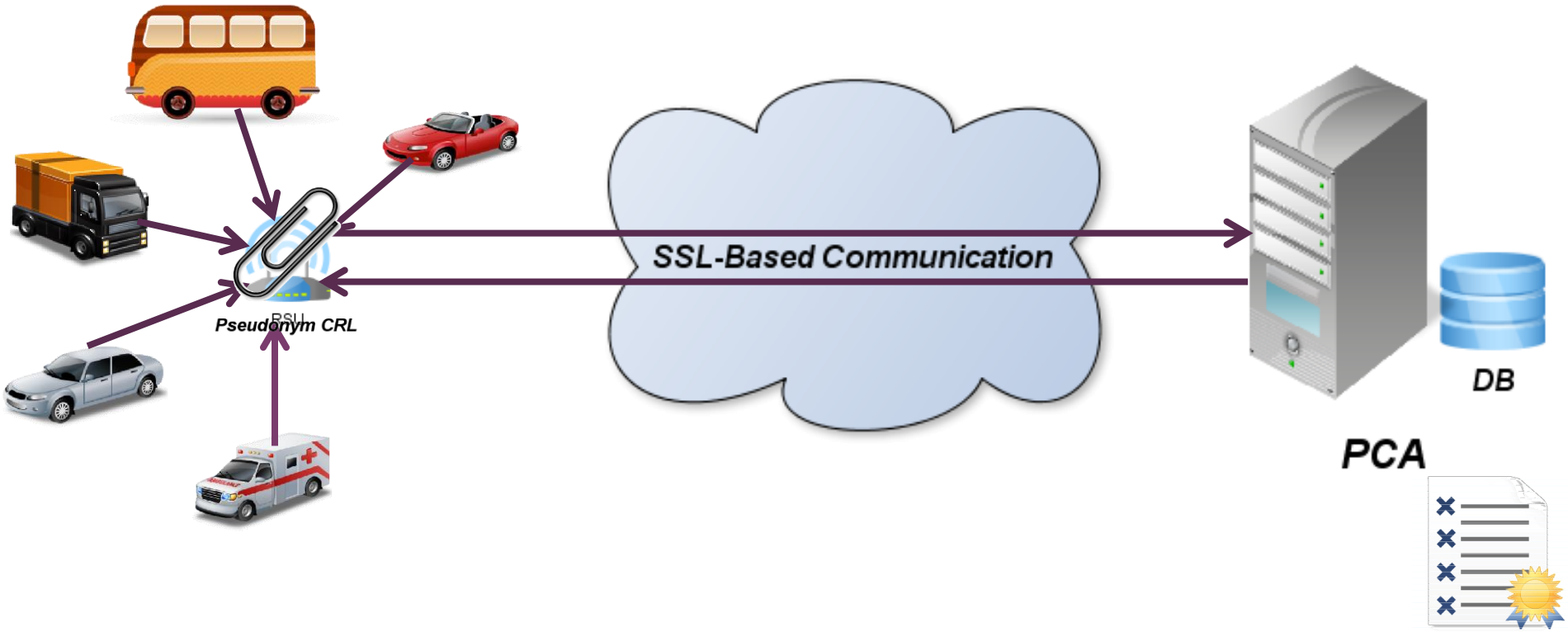
Obtaining a Token



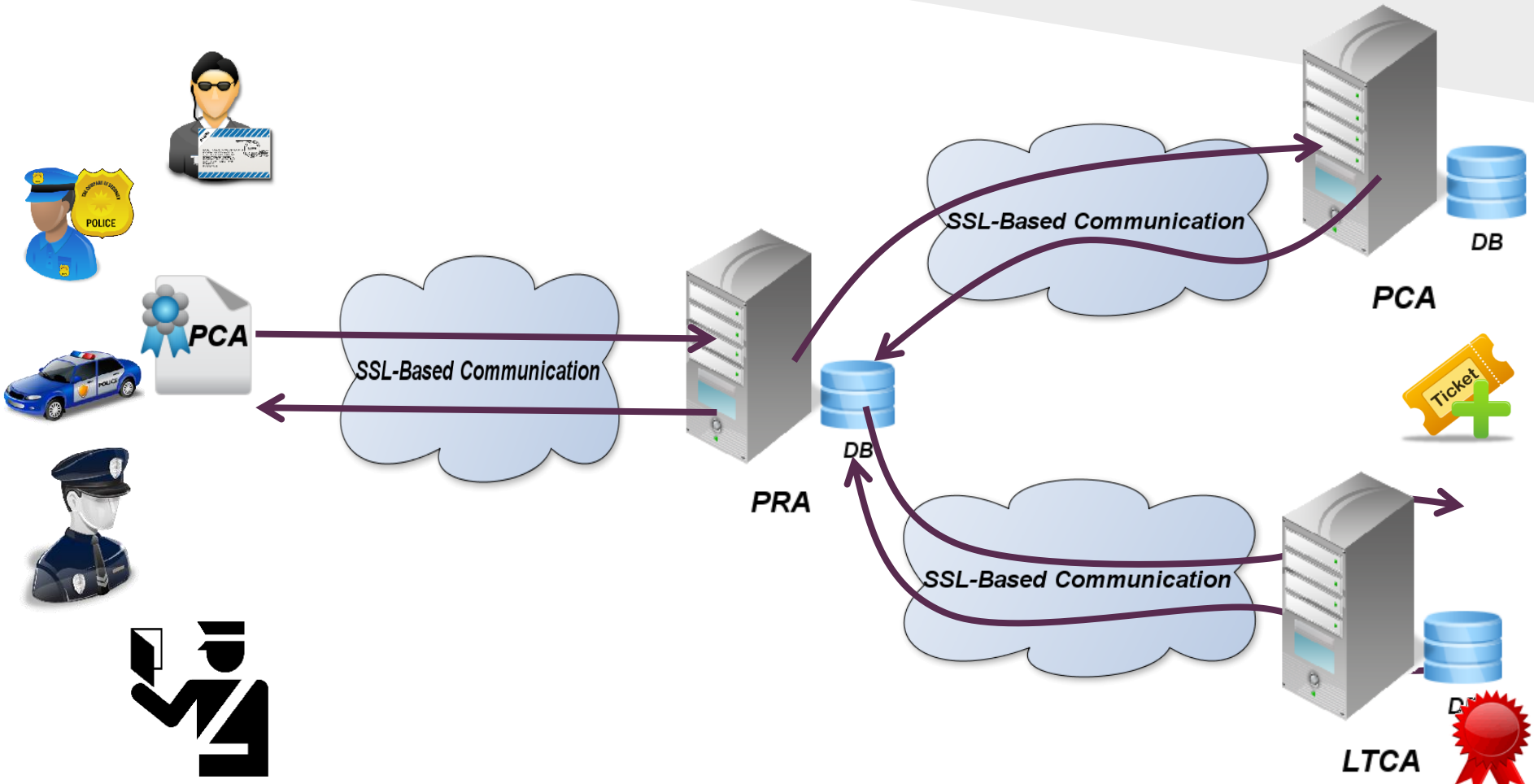
Obtaining Pseudonym Cert.



Obtaining Pseudonym CRL



Pseudonym Resolution



Token & Pseudonym Format

Token Format
Token-Type
Token-Serial No.
Token-Identifiable-Key
LTCA-Id, PCA-Id
Maximum Number of Pseudonym Certificates
Token Start-Time *Token Expiry-Time*
Pseudonym Start-Time *Pseudonym Expiry-Time*
Signature

Pseudonym Cert. Format
Serial No.
Pseudonym Cert. Identifiable Key
Signer-ID
Valid-From *Valid-To*
EC Public key
Signature

Pseudonym CRL Format

Pseudonym CRL Format
Pseudonym-CRL Serial No.
CRL Version
PCA-Id
Revoked Pseudonym-Cert. No.
Revoked Pseudonym-Cert. Serial No.
Time-Stamp
Signature

Binding Token to Pseudo- Cert.

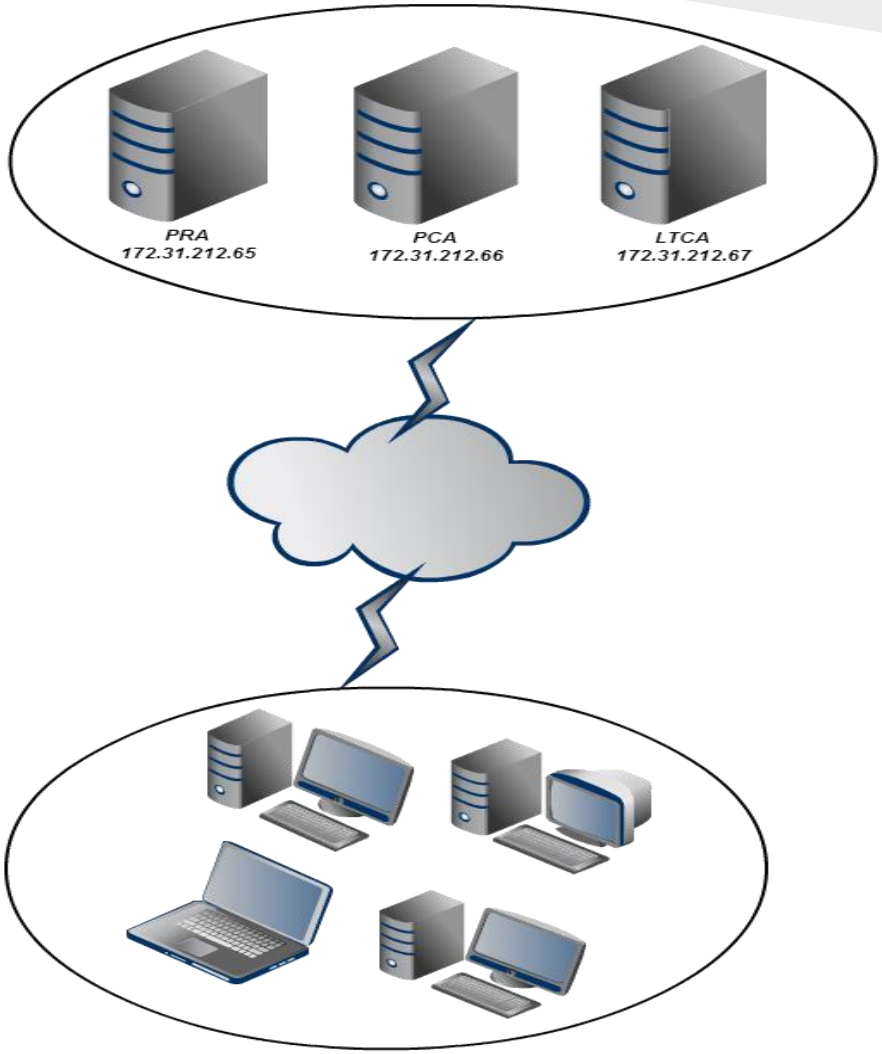
- LTCA:
  - ***Token-Identifiable-Key*** = hash(Vehicle Long-Term Certificate Serial No. || Time-Stamp || Nonce)

- PCA:
  - PseuCertIdentifiableKey = hash(***Token-Identifiable-Key*** || Pseudo-Public Key || Time-Stamp || Nonce)

Outline

- Introduction
  - Problem Statement
  - Contribution
  - Key Concepts
  - Security Requirements
  - Adversary Model
- Protocols Design
- ***Performance Evaluation***
- Conclusion
  - Future Direction

Network Topology



Servers & Client Spec.

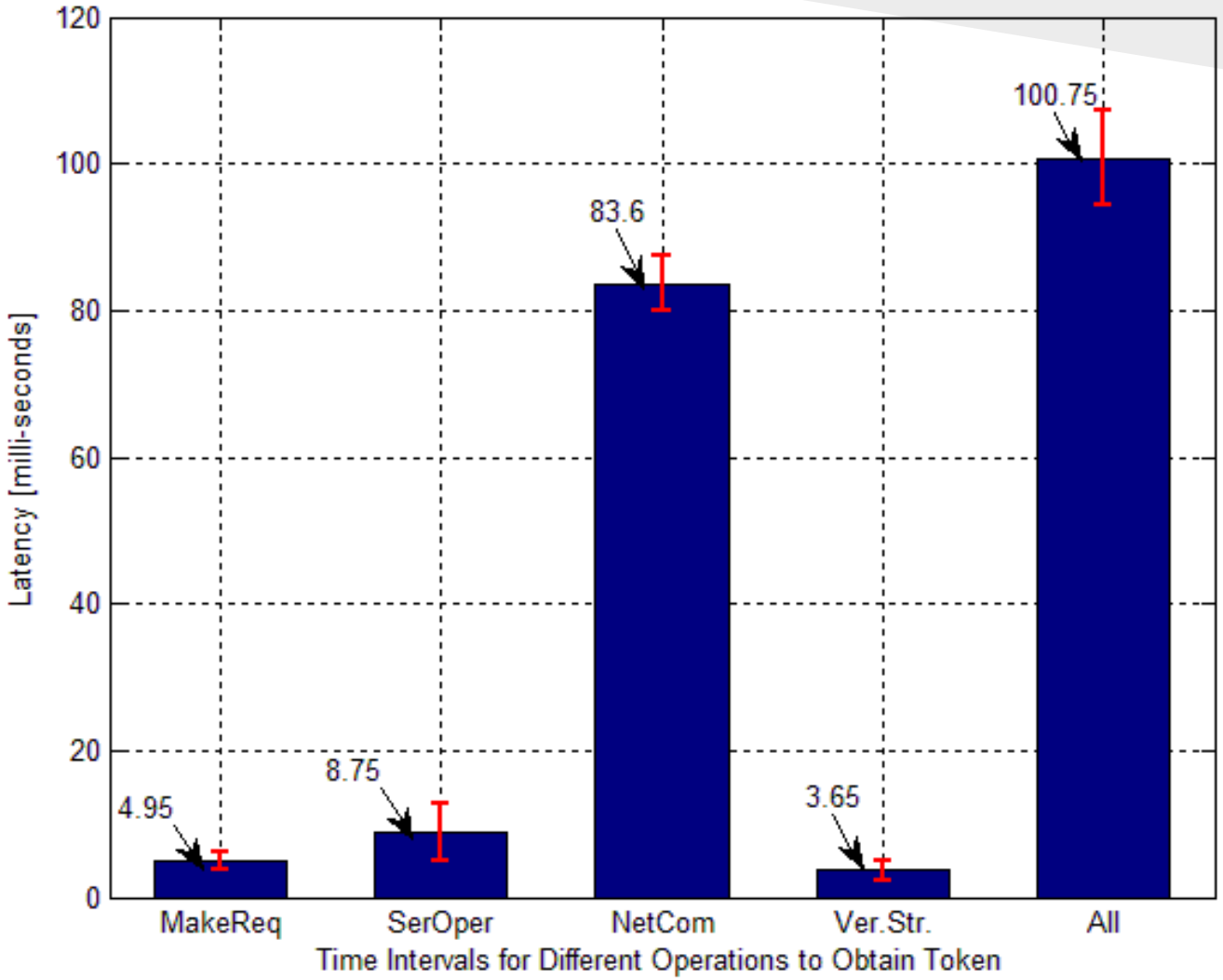
- Servers:

Processor Model Name	***Intel(R), Dual-Core, Xeon(TM), CPU 3.40GHz***
Bogomips	***6782.71***
RAM	***8 GB***

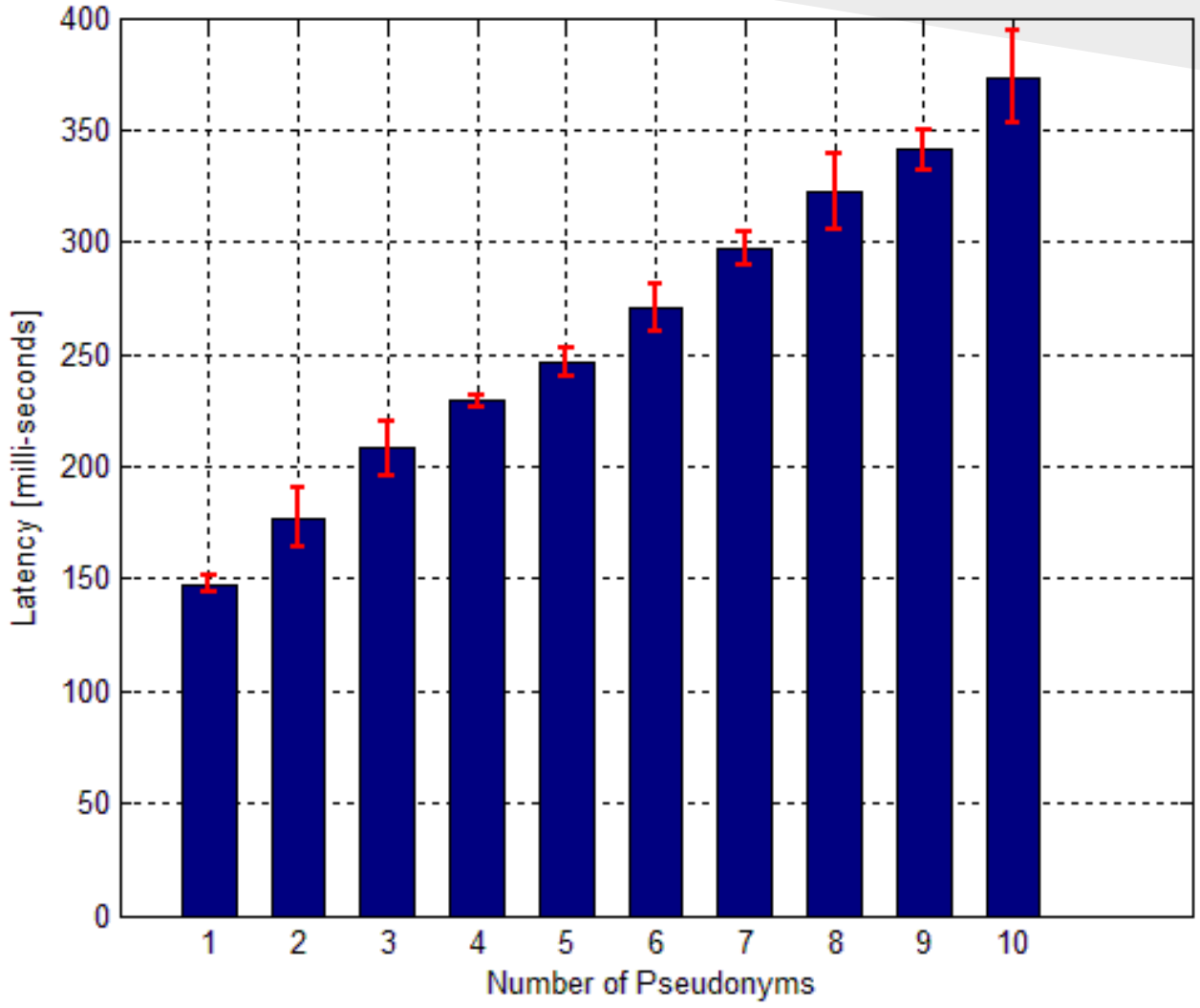
- Client:

Processor Model Name	***Intel(R), Dual-Core(TM), CPU 3.00 GHz***
Bogomips	***5960.58***
RAM	***2 GB***

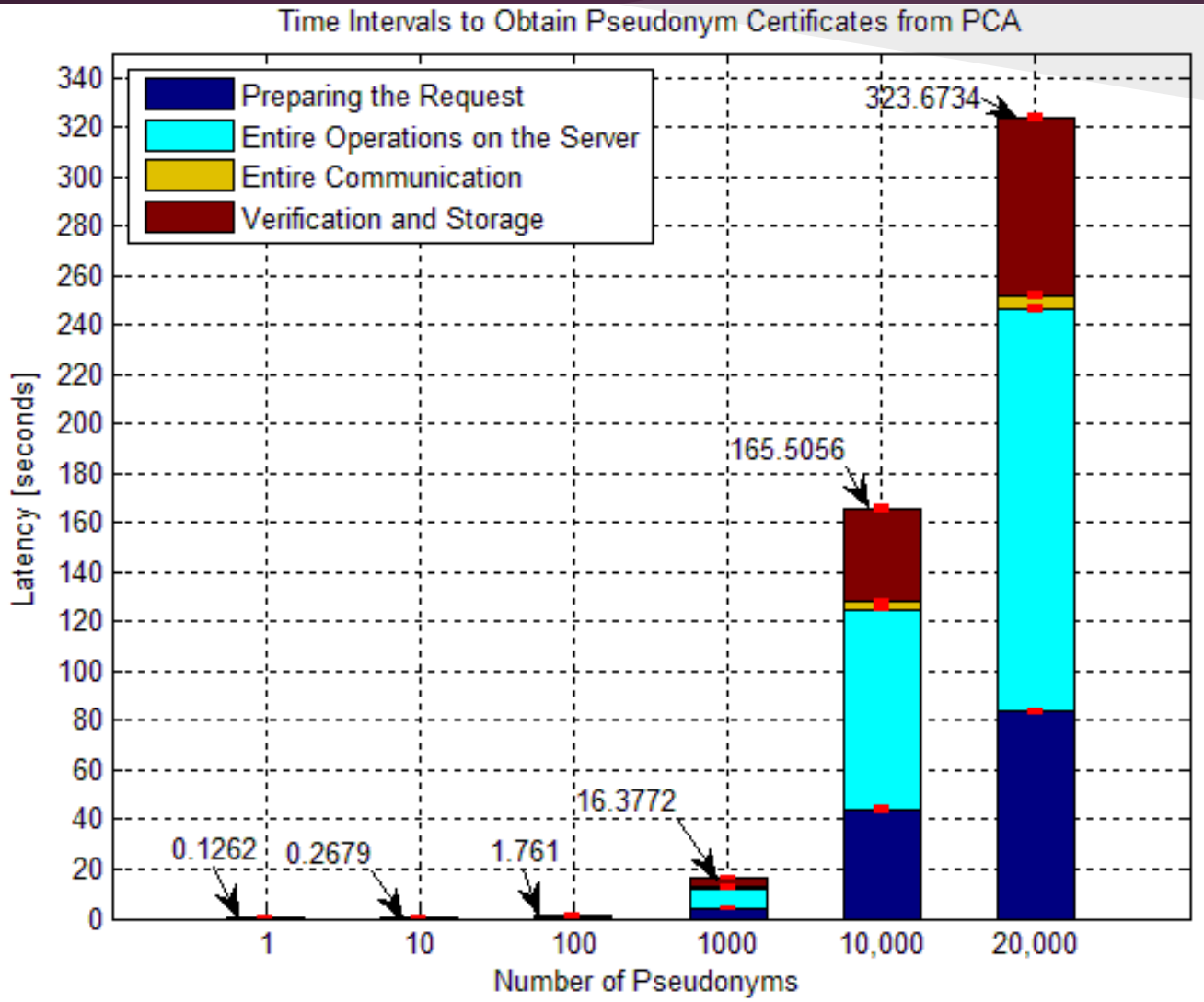
Obtaining Token from LTCA



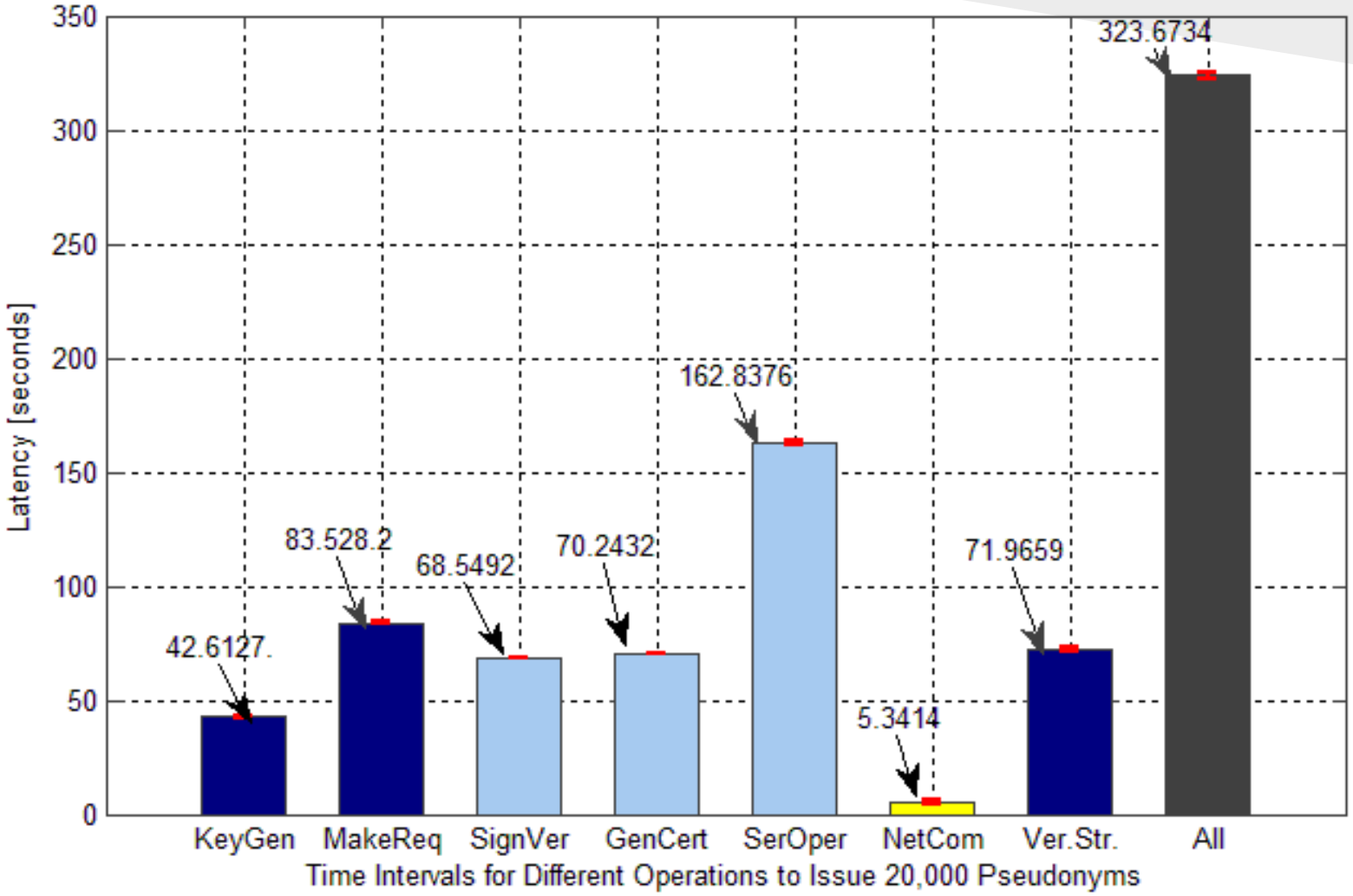
Time Interval to Obtain 10 Pseudonyms



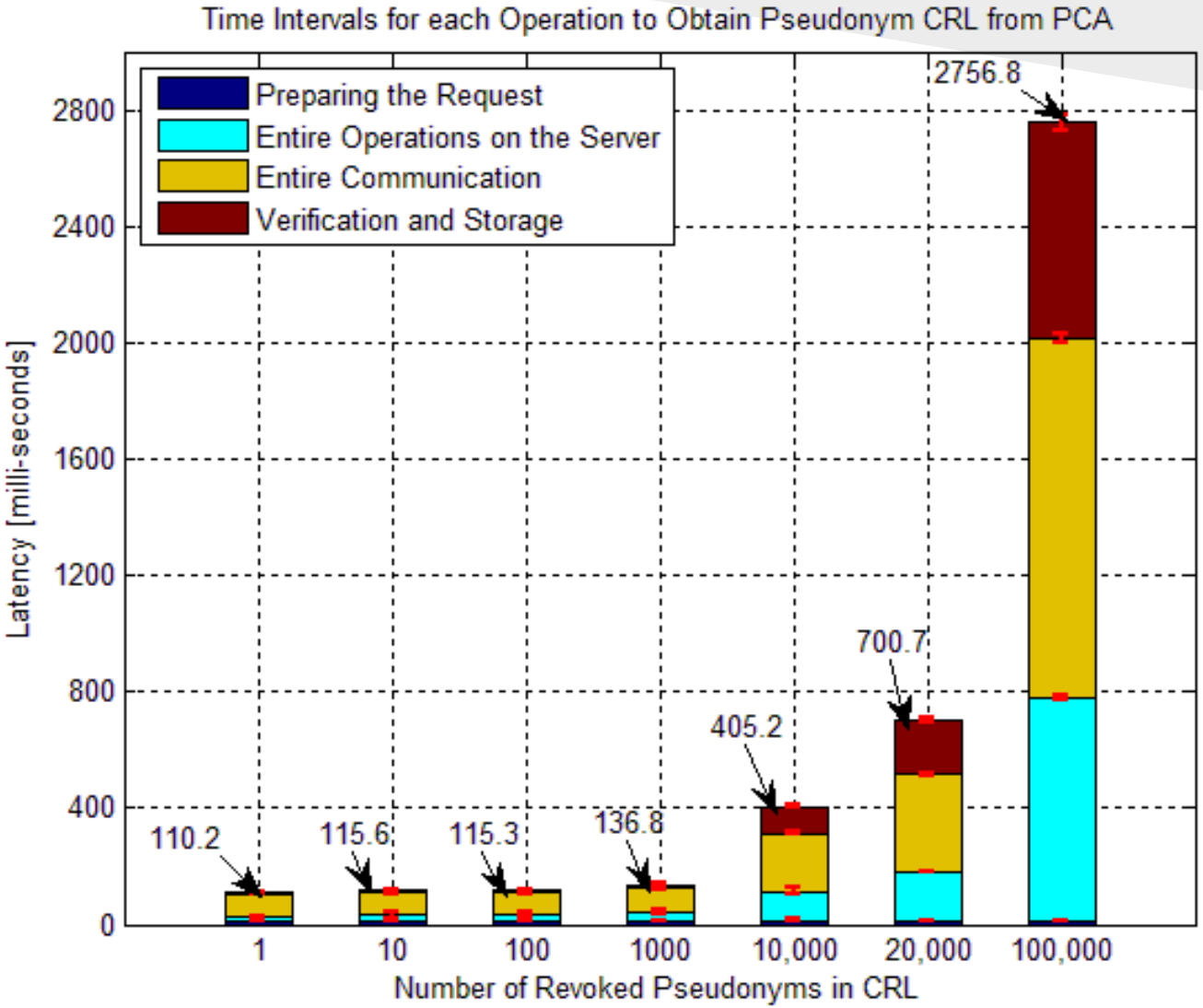
Time Intervals for Different Operations to Obtain Pseudonym Certificates



Time Interval to Obtain 20,000 Pseudonyms from PCA



Time Intervals for Different Operations to Obtain Pseudonym CRL



Pseudonym CRL File Size

No. of Revoked Pseudonyms in CRL	*Size in bytes*
1	*778 bytes (778 bytes)*
10	*1.36 KB (1,398 bytes)*
100	*7.33 KB (7,507 bytes)*
1000	*67.1 KB (68,723 bytes)*
10,000	*664 KB (680,718 bytes)*
20,000	*1.29 MB (1,360,714 bytes)*
100,000	*6.48 MB (6,800,715 bytes)*

Outline

- Introduction
  - Problem Statement
  - Contribution
  - Key Concepts
  - Security Requirements
  - Adversary Model
- Protocols Design
- Performance Evaluation
- ***Conclusion***
  - ***Future Direction***

Conclusion

- Three protocols are integrated into OpenCA to provide security functionality for VANETs
- Improvement in compare with similar projects
  - Linkability
  - Privacy
  - Pseudonym Resolution
- Performance evaluation shows reasonable time to obtain pseudonyms, CRL and pseudonym resolution
- Experiments should be done on a vehicle for a more precise result

Future Direction

- Providing a PKI Trust Model in VANETs
  - Introducing a new PCA, LTCA and PRA
  - Foreign Pseudonym Certificates
  - Integrating Short-Term CRLs from Different PCAs
- Token Should be Used Only Once
- Mitigate the Threat of Sybil Attack
  - resource testing techniques, social networking approaches, radio testing, trusted certification

Future Direction Cont.

- Token Verification by any PCA to Enhance Privacy
- Performing Reverse Pseudonym Resolution
- Resolving Multiple Pseudonyms in a Request
- Using FastCGI instead of CGI
- Performance and Efficiency for VANETs

Acknowledgement

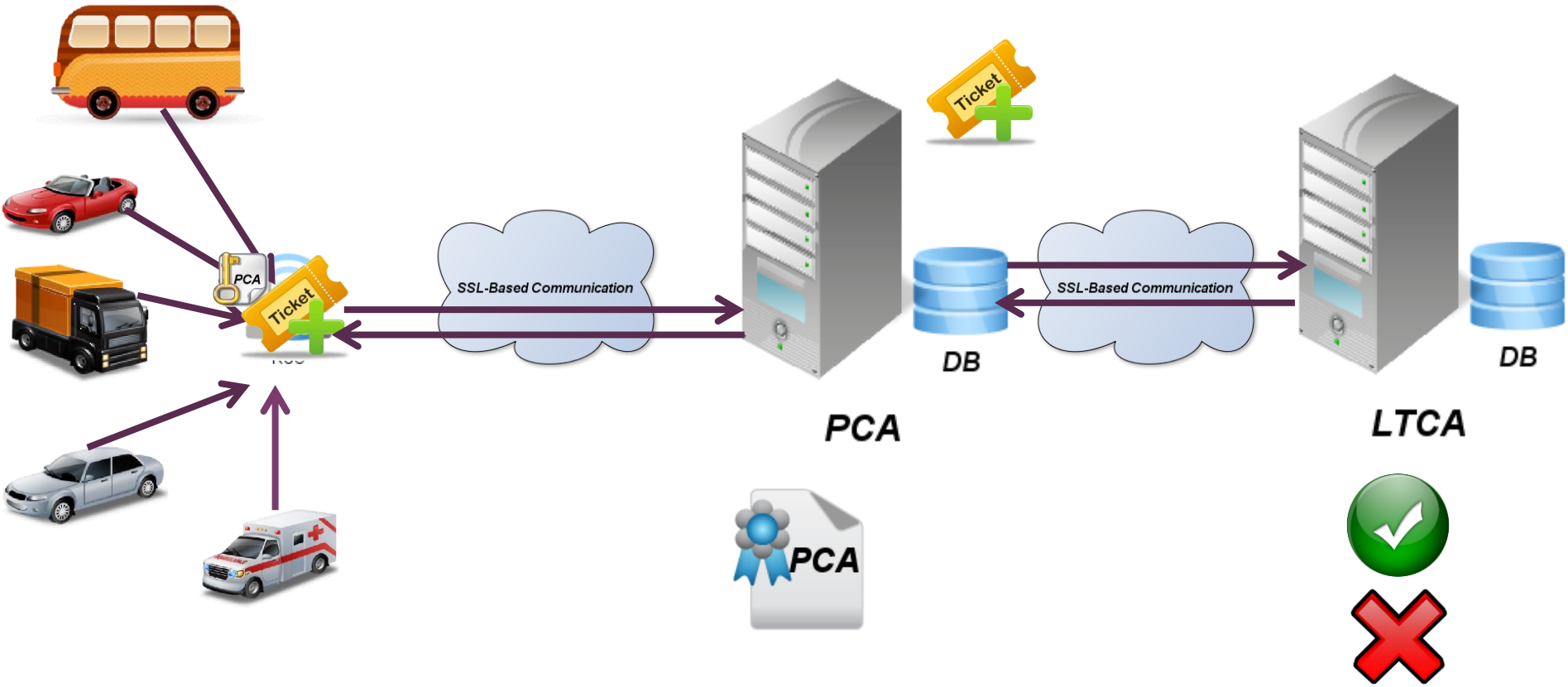
References

- *Secure Vehicular Communication Systems: Design and Architecture*
- *Sevecom - Secure Vehicle Communication*
- *Efficient and Robust Pseudonymous Authentication in VANET*
- *Securing Vehicular Communications - Assumptions, Requirements, and Principles*
- *V-Tokens for Conditional Pseudonymity in VANETs*
- *Intelligent Transport Systems (ITS), Security, Stage 3 mapping for IEEE 1609.2. V0.0.6*
- *"On the Road" - Reflections on the Security of Vehicular Communication Systems*
- *Secure Vehicular Communication Systems: Implementation, Performance, and Research Challenges*

Questions

Thanks for your attention!

Obtaining Pseudonym Cert.



OpenCA

- Written in C
- Two packages:
  - openca-base
  - openca-tools
- Uses Open-SSL Libraries
- Support Open-LDAP
- Web-based Interface
- With an Apache-style license

Token Req-Res Format

Token Request	***Token Response***
Req. Type *X509 VLTC Length* *X509 VLTC* *Pseudonym Cert. No. Request* *LTCA-Id* *PCA-Id* *Nonce* *Time-Stamp* *Signature*	*Req. Type* *Token Size* *Token* *Max No. Pseudonym Cert.* *LTCA-Id* *PCA-Id* *Nonce* *Time-Stamp* *Error-Info* *Signature*

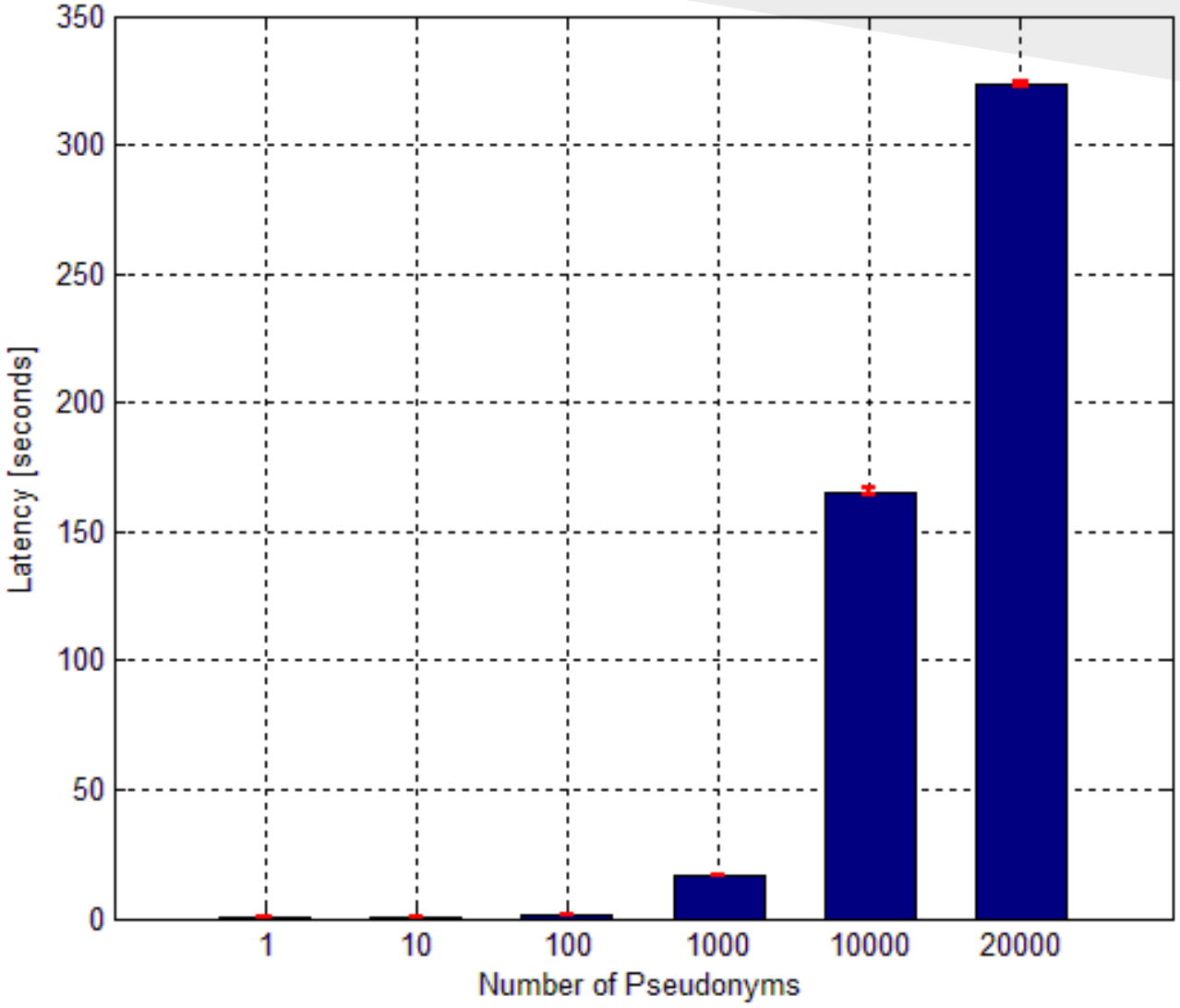
Pseudonym Req-Res Format

Pseudonym Request	***Pseudonym Response***
Req. Type *Token Size* *Token* *LTCA-Id* *PCA-Id* *Location* *Pseudonym Cert. No* *Pseudonym Public-Key(s)* *Nonce* *Time-Stamp*	*Req. Type* *Req. Identification* *LTCA-Id* *PCA-Id* *Pseudonym Cert No* *Pseudonym Cert.* *Nonce* *Time-Stamp* *Error-Info* *Signature*

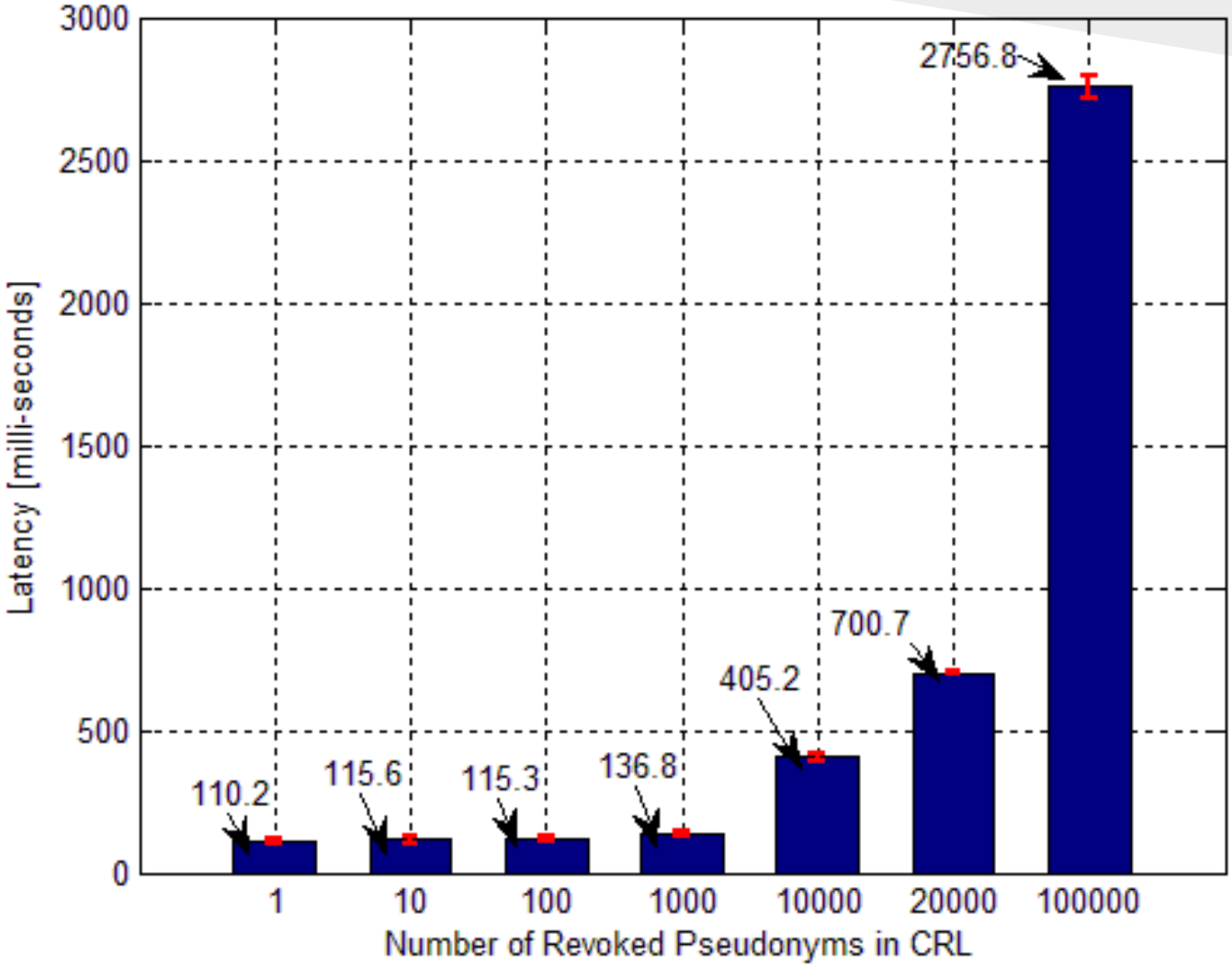
Pseudonym CRL Res-Res Format

Pseudonym CRL Request	***Pseudonym CRL Response***
Req. Type *Current CRL Version* *PCA-Id* *Region-Id* *Pseudonym Cert. Length* *Pseudonym Cert.* *Nonce* *Time-Stamp* *Signature*	*Req. Type* *PCA-Id* *CRL Size* *CRL* *Nonce* *Time-Stamp* *Error-Info* *Signature*

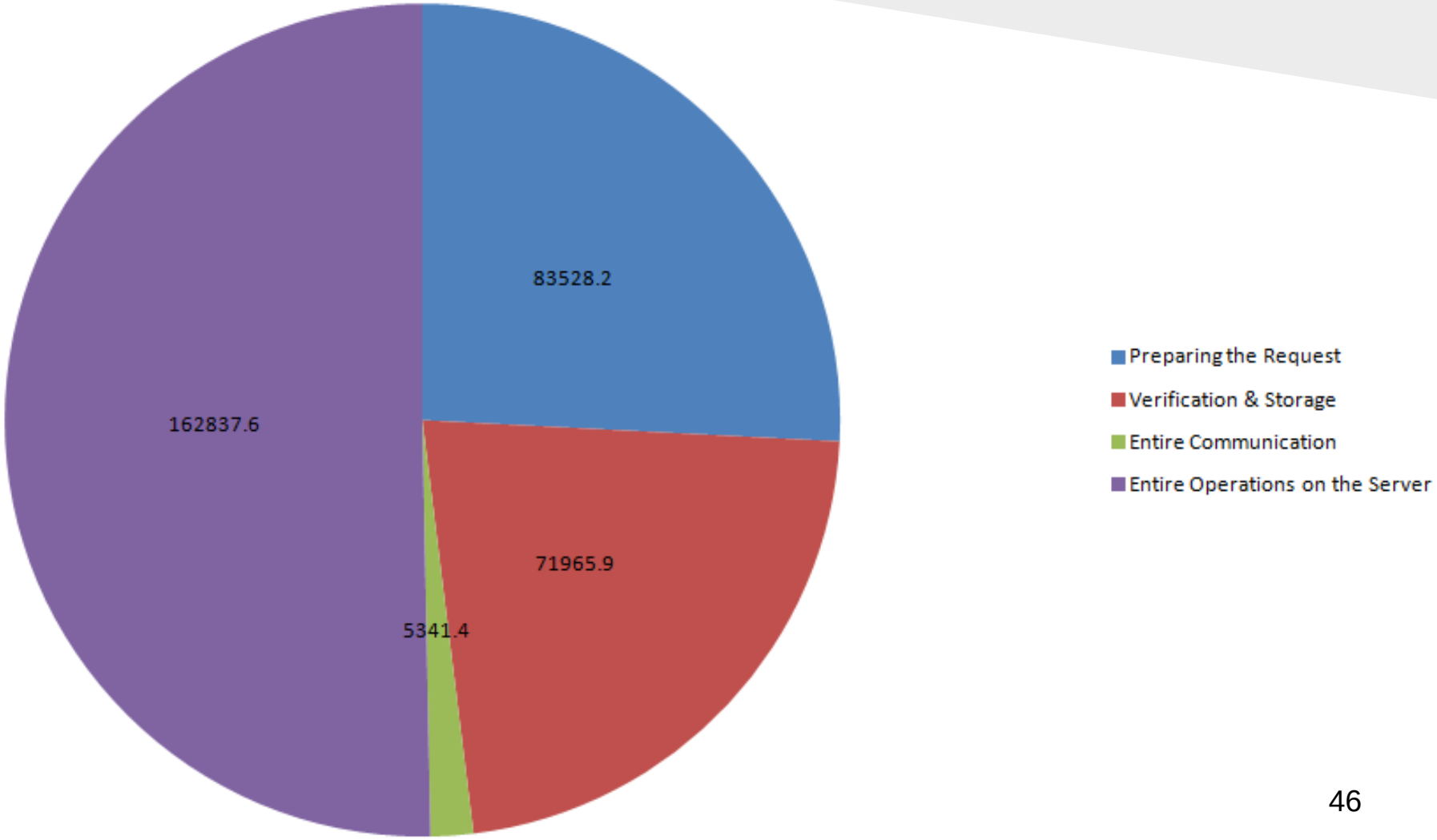
Obtaining Pseudonyms from PCA



Obtaining Pseudonym CRL



Percentage of Different Operations to Obtain 20000 Pseudonyms



Implementation

- C++
- OpenCA as the base implementation
- Installed and configured PCA , LTCA and PRA on Different Servers
- Libraries:
  - OpenSSL
  - Xmlrpc
  - MySQL
  - Boost-Serialization

Time Intervals to Obtain a Token from LTCA

Operations	***Latency in ms***
Preparing Token Request	*4.95 ms*
Issuing the Token (Server Side)	*8.75 ms*
Entire Communication	*83.6 ms*
Verification and Storage of the Token	*3.65 ms*
Entire Operations	*100.75 ms*

Token Size	*477 bytes*

Pseudonym Certificate Size	*2.0 KB (2078 bytes)*
Pseudonym Private-Key File Size	*5.0 KB (5153 bytes)*