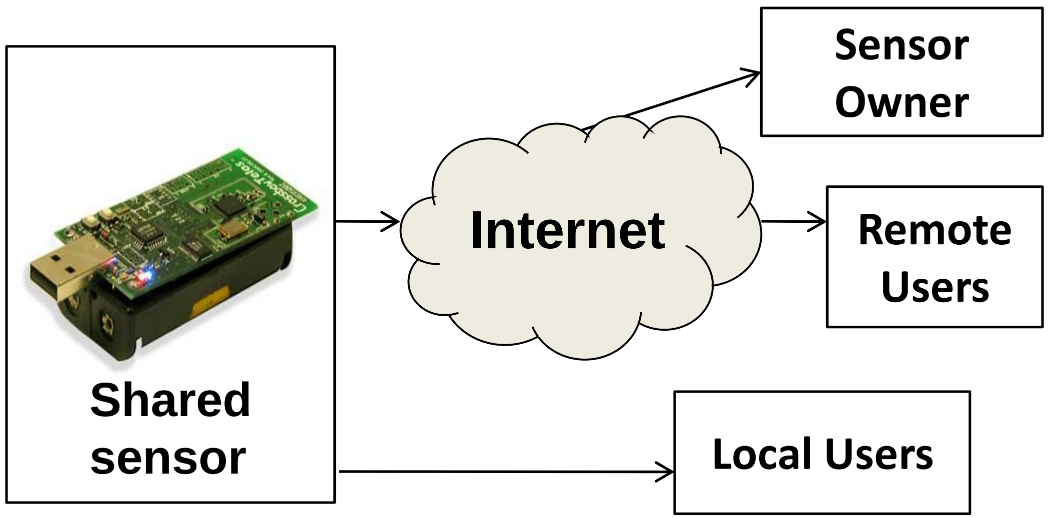
Secure-TWS: Authenticating Node to Multi-user Communication in Shared Sensor Networks

Leonardo B. Oliveira (Unicamp/Brazil), **Aman Kansal**, Bodhi Priyantha, Michel Goraczko, and Feng Zhao

Microsoft Research

Shared Sensors, Directly Connected to the Internet

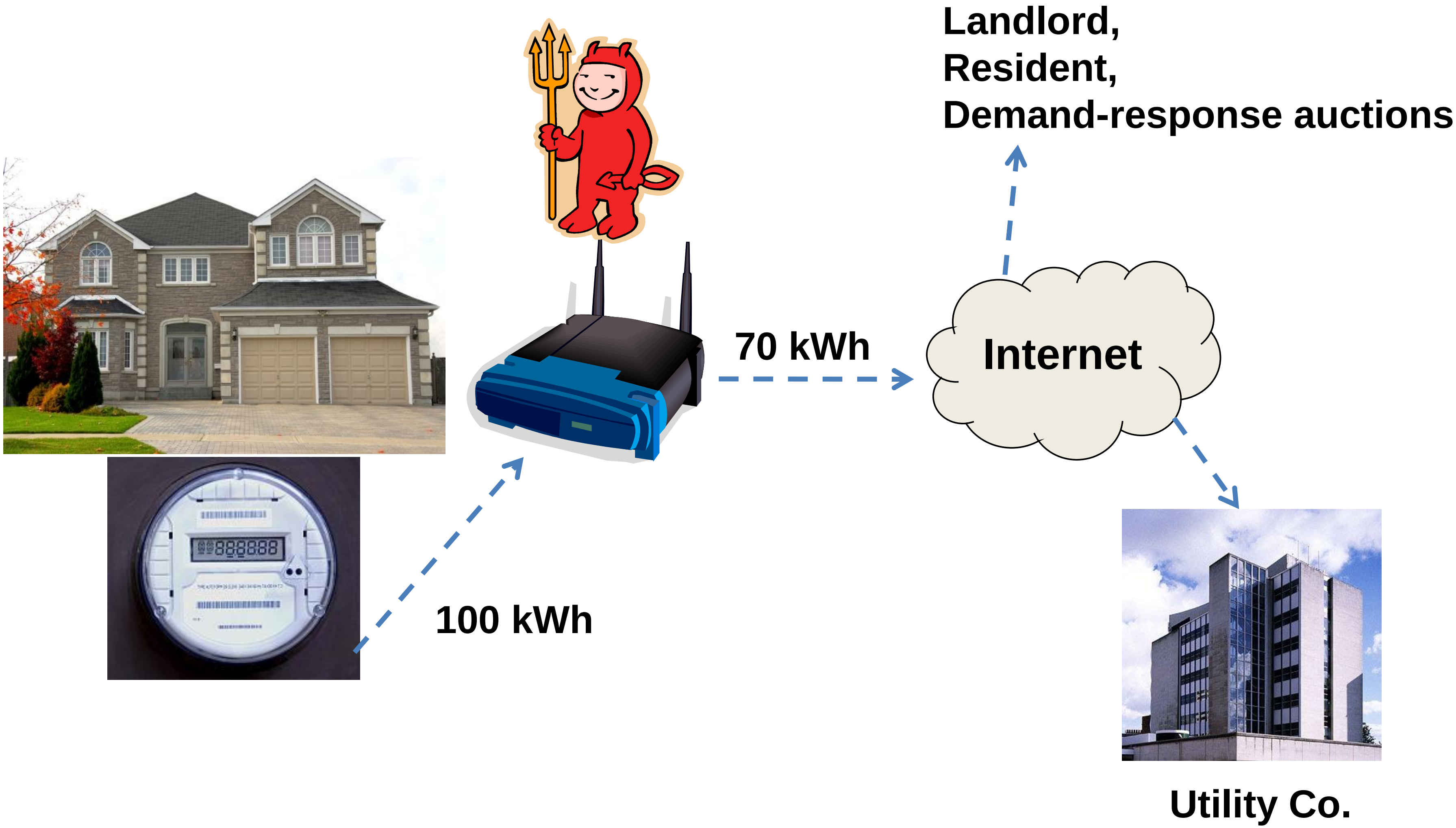
Multiple sensors from many vendors:
Power meter by Utility,
Fire Sensor,
Motion Sensor by Security Co,
Air quality sensor by City,
Soil sensor by Landscaping Co.,
Medical sensor,



Multiple users:
Home owner,
Utility Co,
Fire Station,
Healthcare SP,
Travelling user,
Area Visitors,
Landlord, etc.

- IP Layer: [Dunkels, Mobisys'03], [Hui & Culler, Sensys'08]
- App Service: [Priyantha et al, Sensys'08]

Example Attack



Problem: Data Authentication

- Authenticate data received from sensor
  - Ensure data not modified by gateway, network
  - Assumption: Users trust sensors but not network and gateway

Challenges

- Sensors are resource constrained
  - Traditional web authentication methods may not directly be ported
- Multiple users, sporadic users
  - Cannot establish secure keys with each user

Secure-TWS

Design a node-to-multiuser authentication solution

Compare two signature schemes for communication and computation overhead

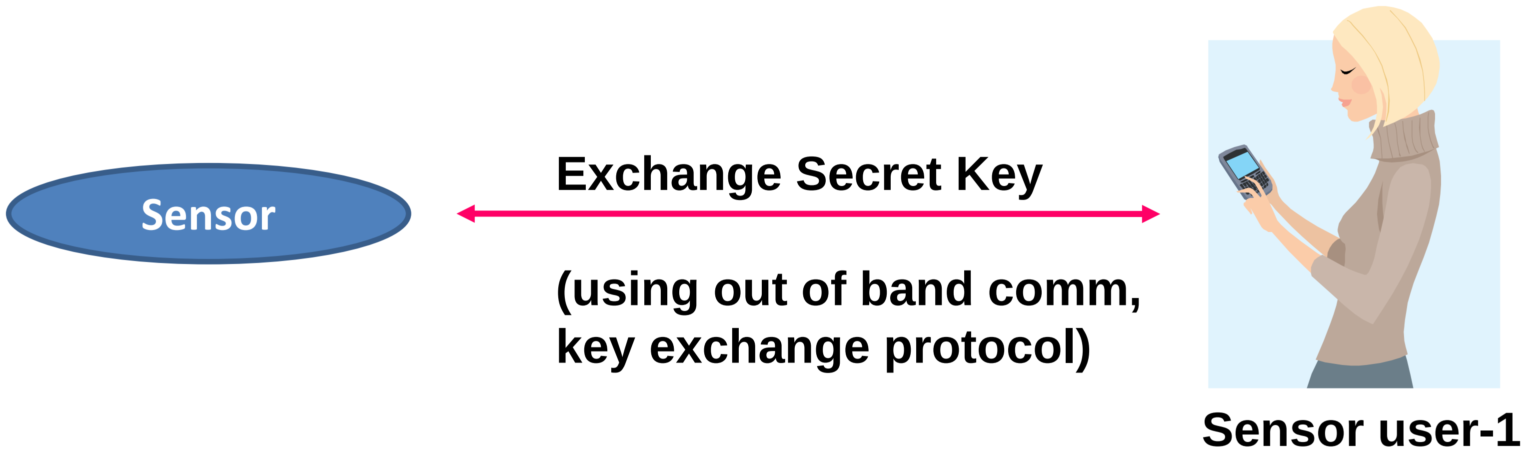
Authentication protocol stack implementation: over Tiny Web Services

Solution Design

Authentication

- Authentication implies:
  - Data authentication: data is not-modified
  - Source authentication: data originated from claimed sensor
  - Non-Repudiation: cannot dispute own data
- Two methods to authenticate:
  - Message authentication codes
  - Digital signatures

Message Authentication Codes

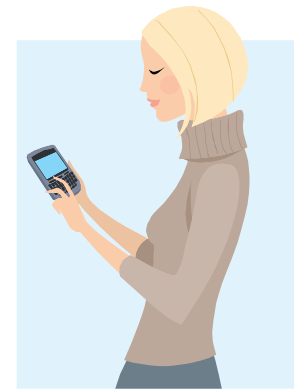


Message Authentication Codes

Sensor

Shared-Secret-1

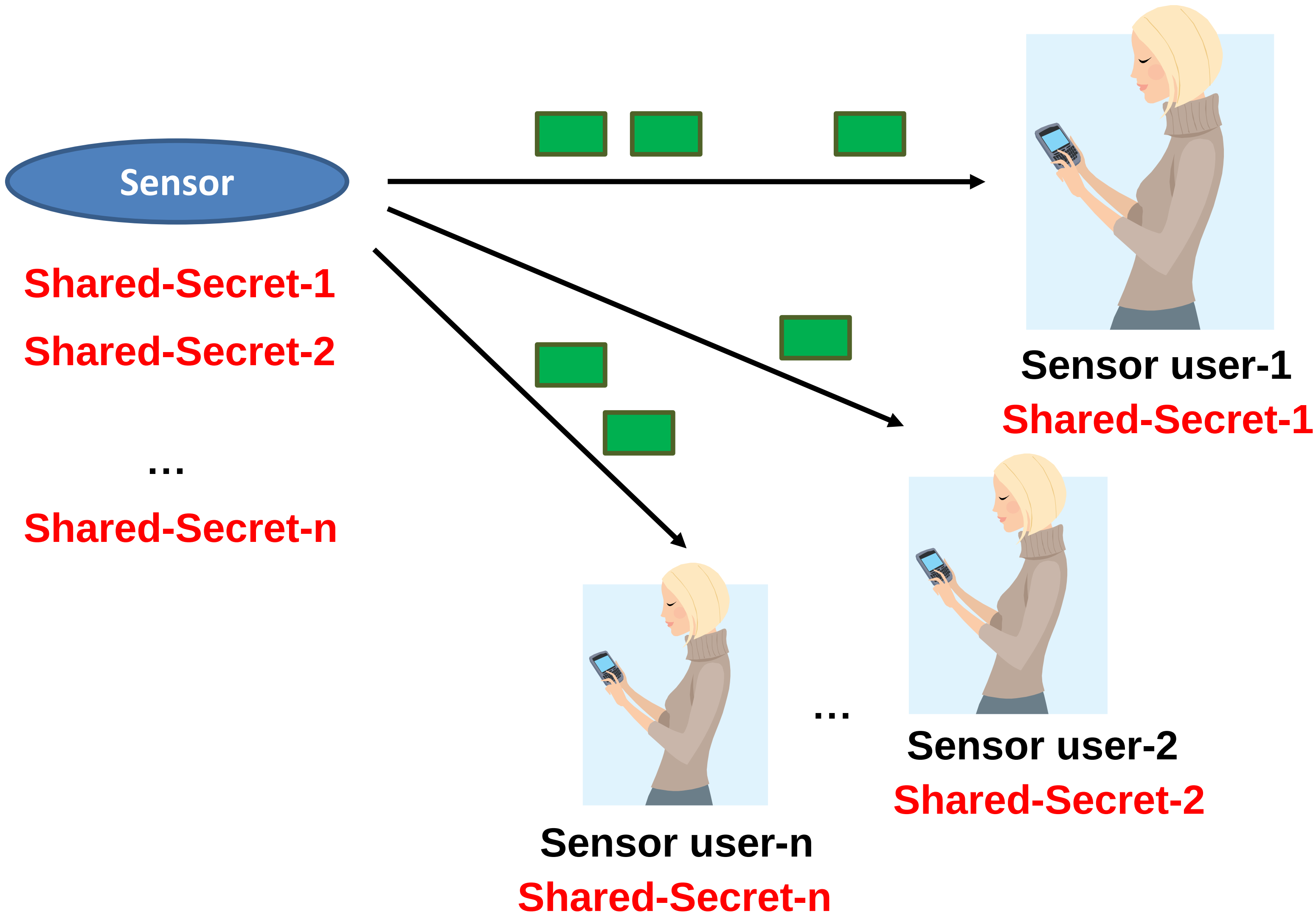
Data packets with auth. code computed using shared secret



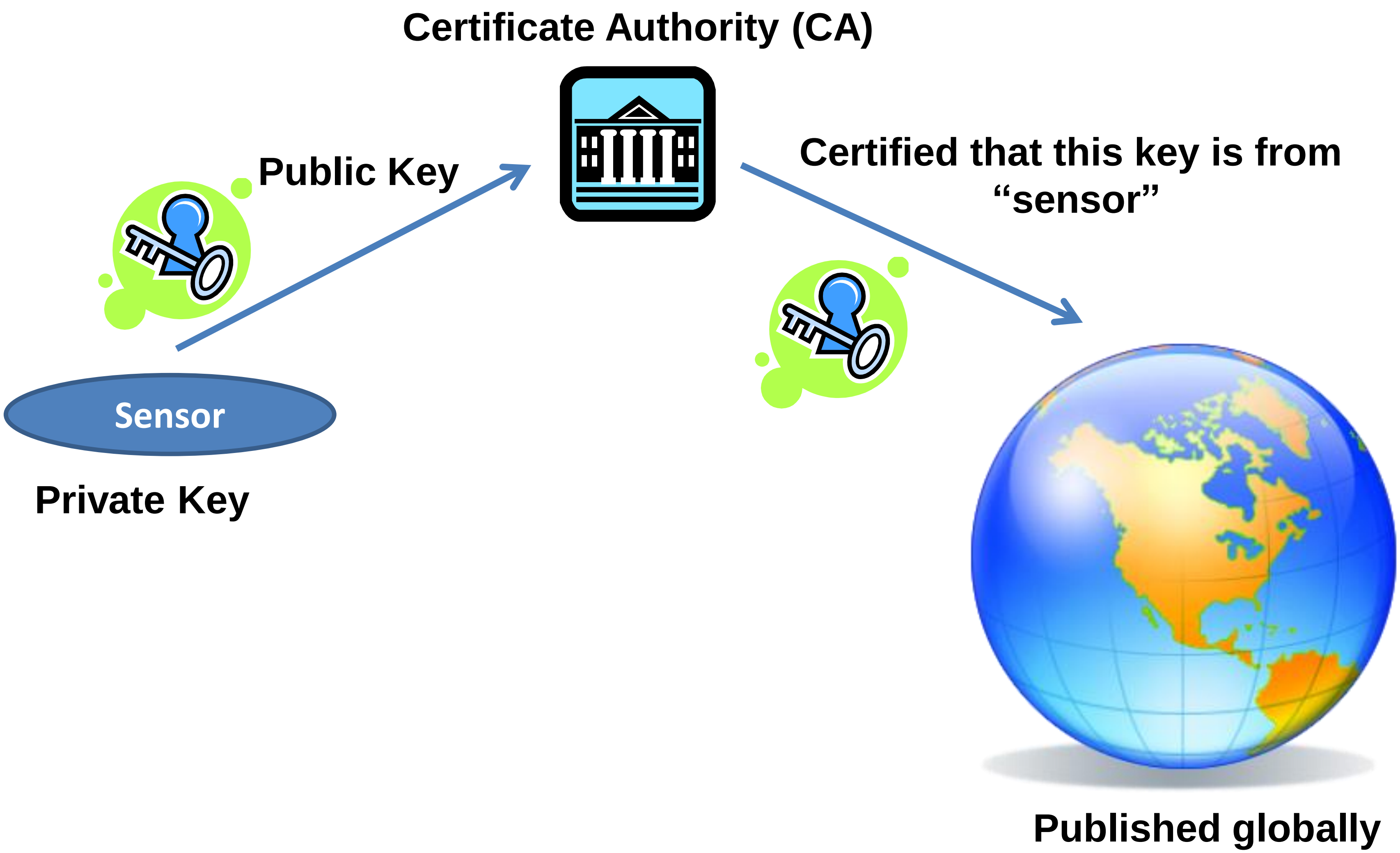
Sensor user-1

Shared-Secret-1

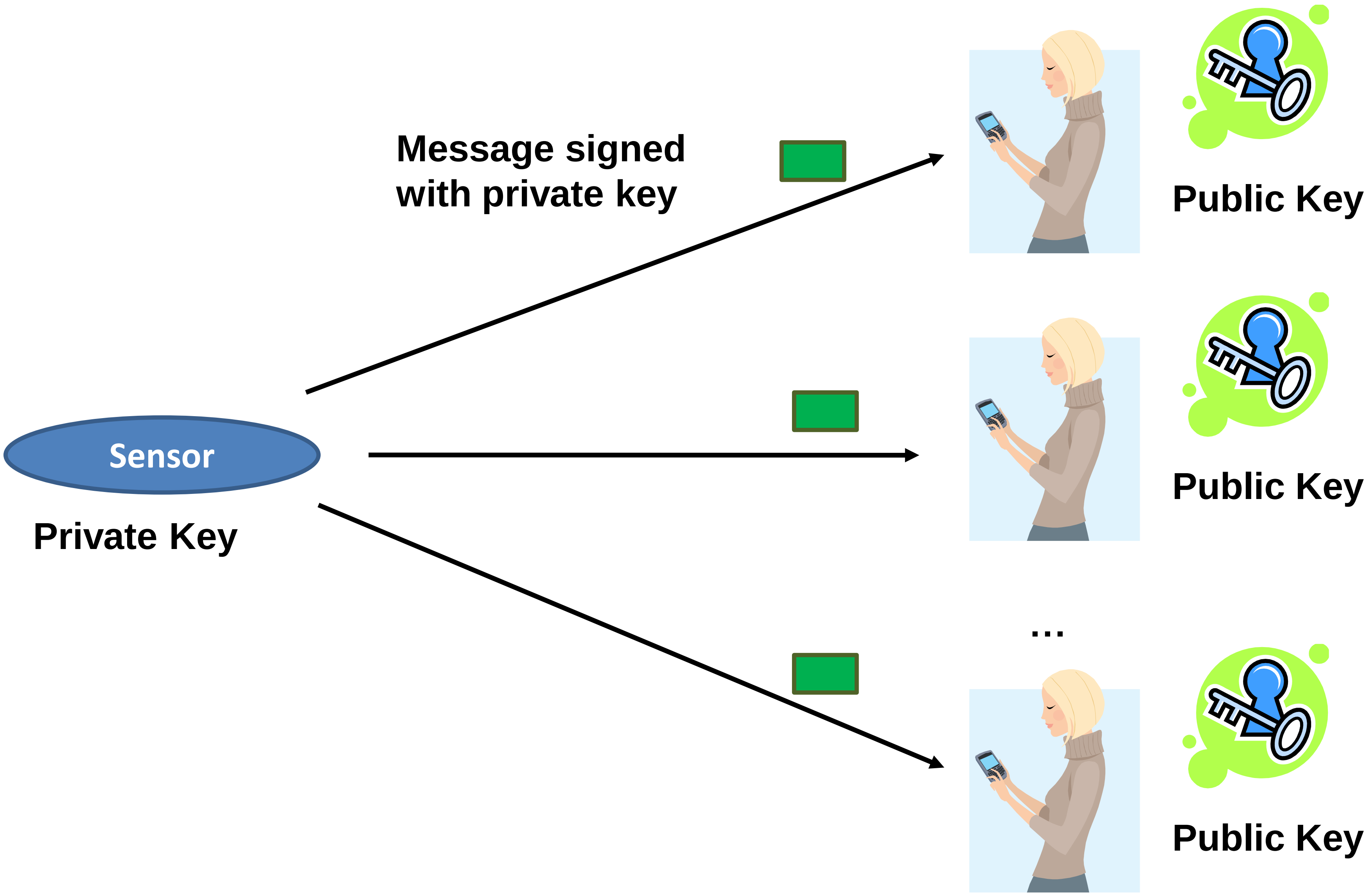
Message Authentication Codes



Digital Signatures



Digital Signatures



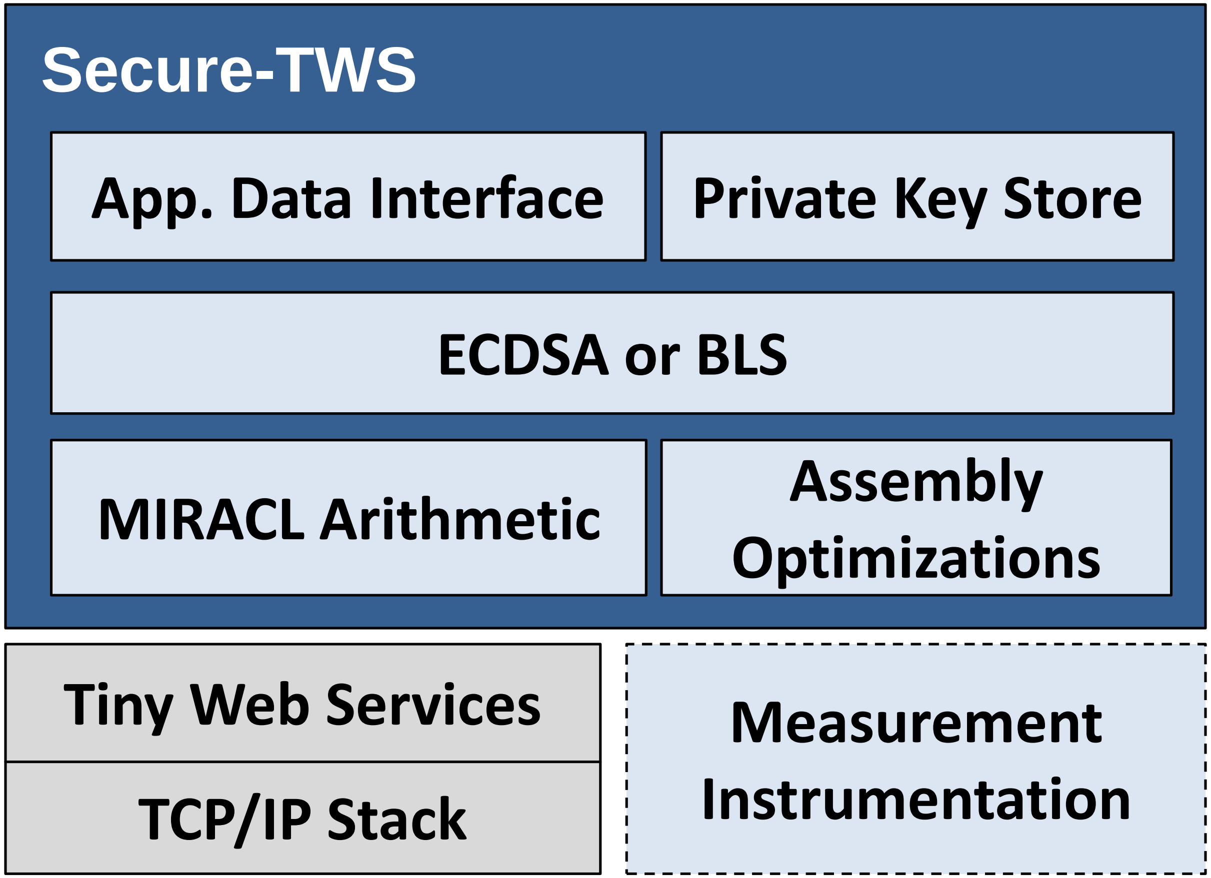
Digital Signature Overheads

Signature Scheme	Computation		Communication (Bytes)
	Generation	Verification	
ECDSA	1 pt multiply	2 point mult	40
BLS	1 pt multiply	2 pairings	20
DSA	1 exp	2 exp	40
Identity Based	2 pt multiply	1 pt multiply + 1 pairing	40

Choose between ECDSA and BLS.

Implementation and Evaluation

Platform and Software



Platforms:

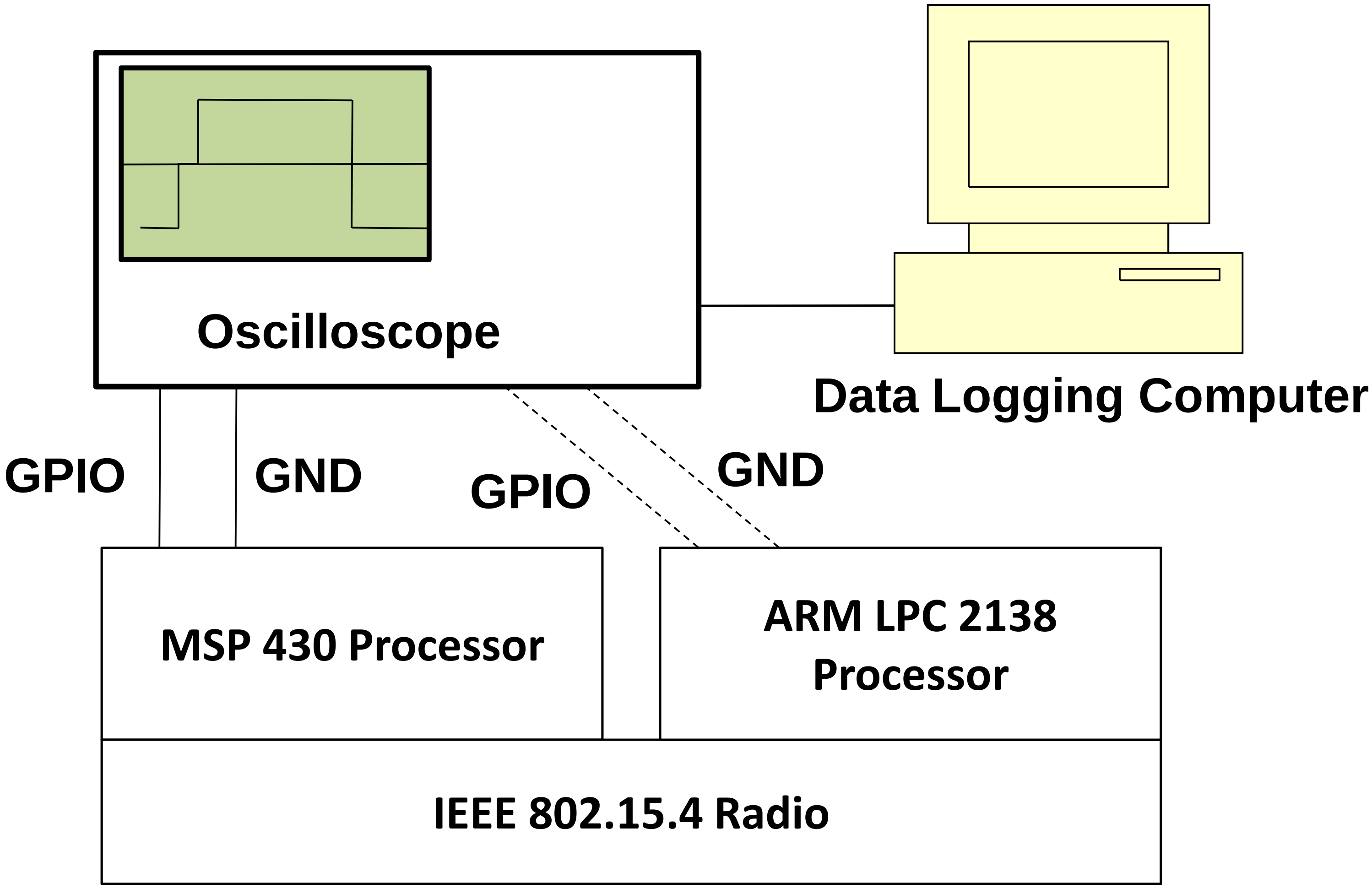
1. MSP-430: 16bit, 16MHz, 8kB RAM, 116kB ROM

2. ARM: 32 bit, 60MHz, 32kB RAM, 512kB ROM (no floating pt unit)

Implementation Parameters

- RSA-1024 equivalent security
  - Geared for Internet use, higher security than used in many WSN solutions
- ECDSA: use *Mersenne* prime (form 2^p-1): reduces computation overhead
  - elliptic curve $y^2 = x^3 - 3x + 157$ with the prime based on $p = 2^{160} - 2^{112} + 2^{64} + 1$
  - Pre-compute parts independent of data
- BLS: No special prime, no precomputation

Measurement Setup



MSP: Storage

RAM

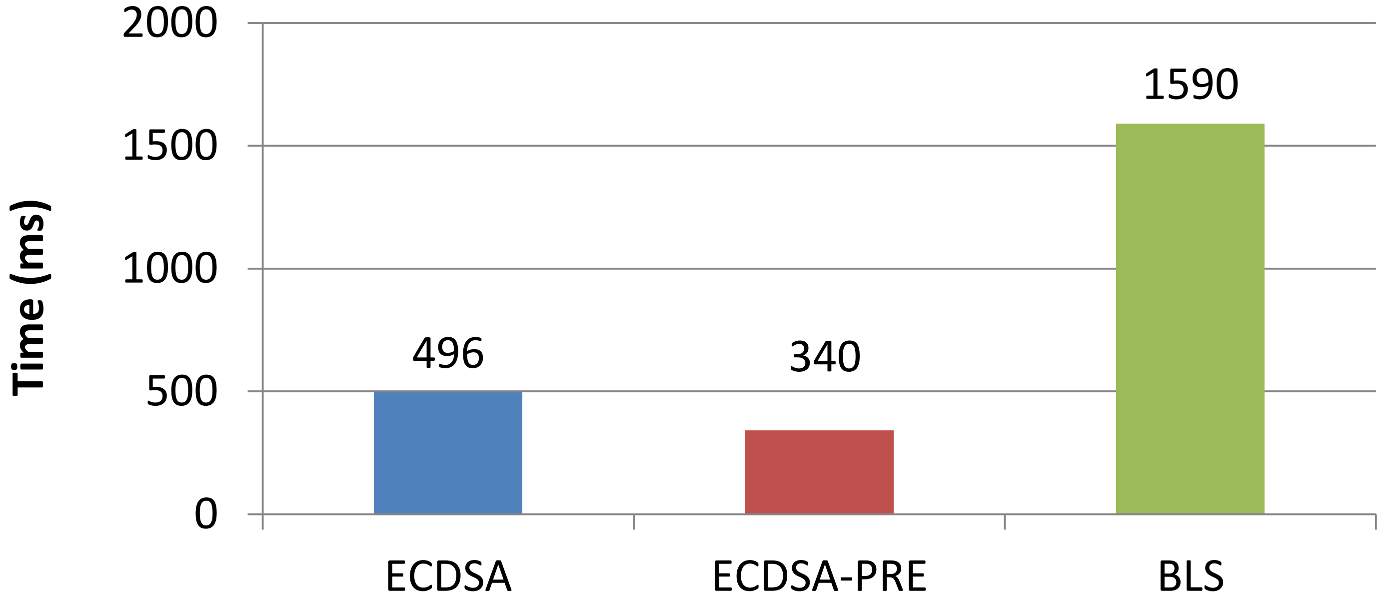
Algorithm	Memory Used
BLS	2.9 kB
ECDSA	2.9 kB
ECDSA+Precomp.	2.9 kB

ROM

Algorithm	Code Space
BLS	47.0 kB
ECDSA	31.3 kB
ECDSA+Precomp.	31.9 kB

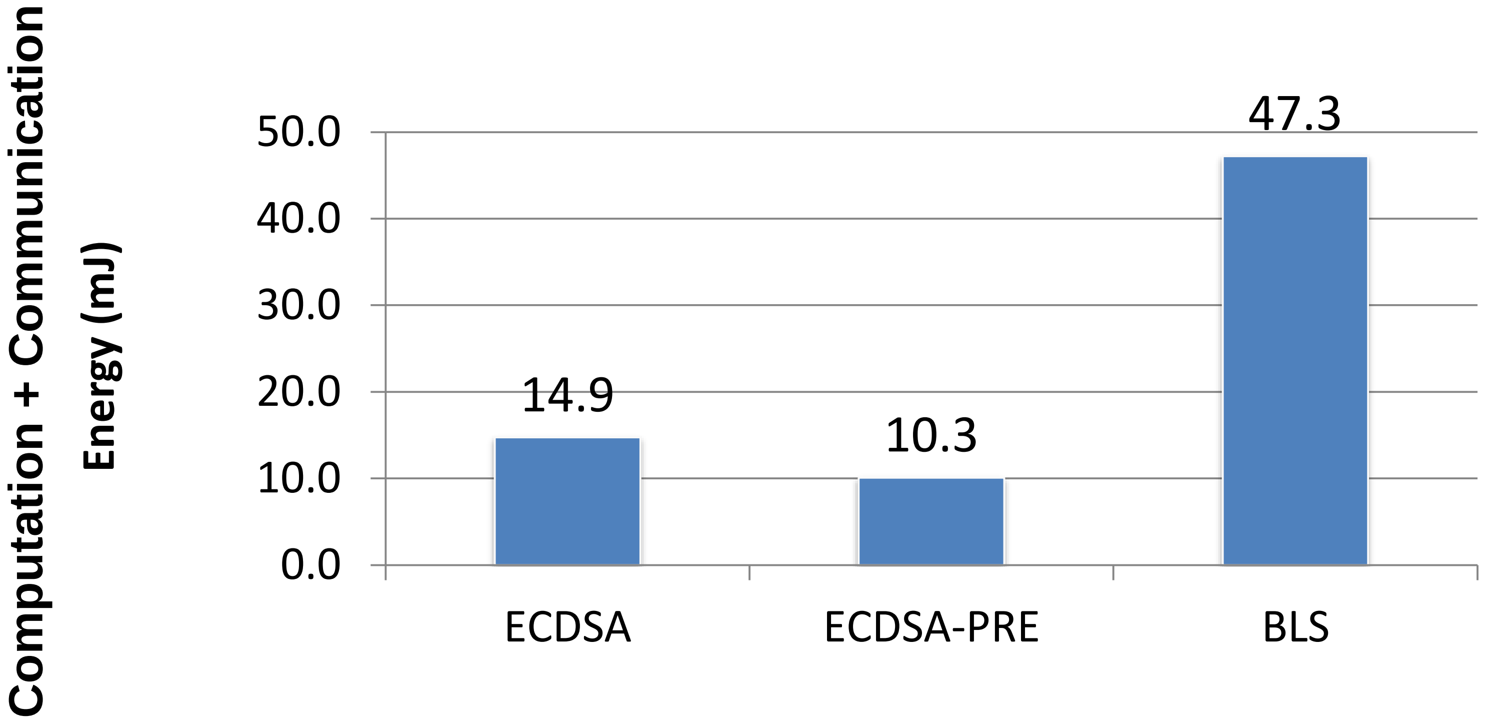
With Tiny Web Service stack: 47.6kB for ECDSA with precomputation, 62.7k for BLS

MSP: Computation



- ECDSA-Precomp is 4.7x faster than BLS
  - Precomputation makes ECDSA 31.4% faster

MSP: Total Energy



- Communication is
  - ECDSA: 0.23mJ
  - BLS: 0.15mJ

ARM: Storage

RAM

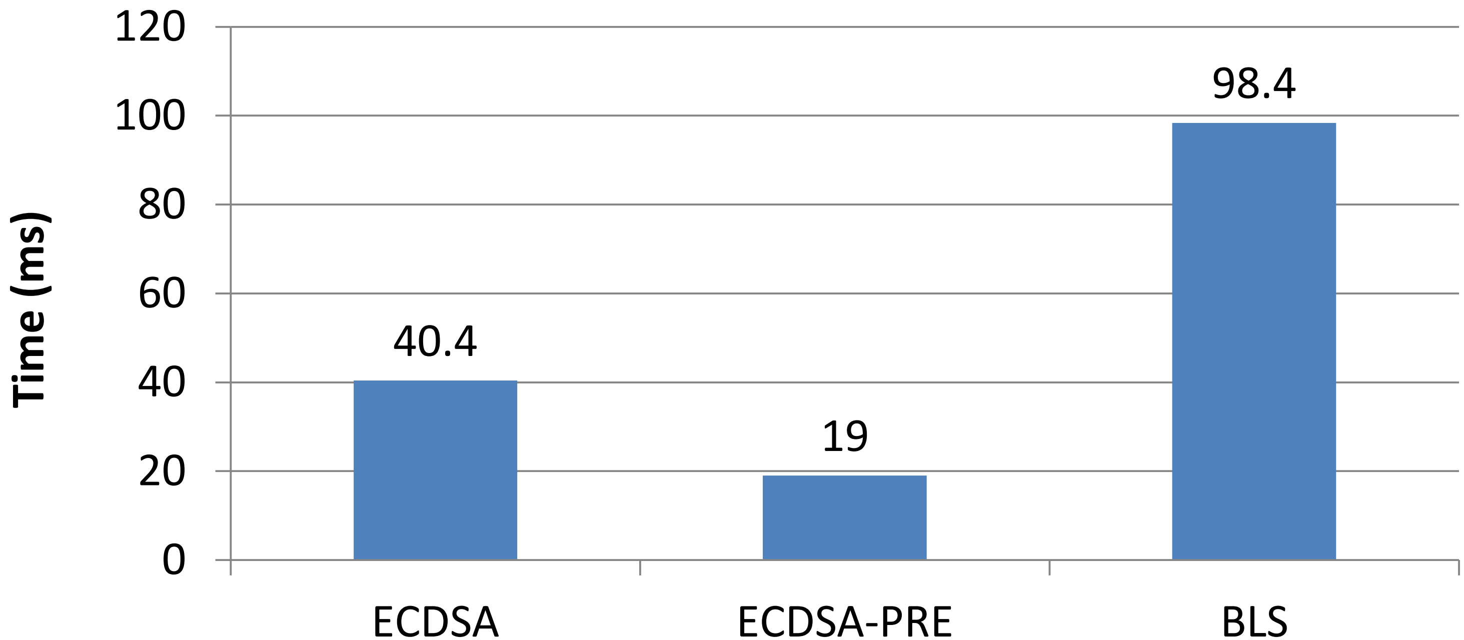
Algorithm	Memory Used
BLS	9.22 kB
ECDSA	9.22 kB
ECDSA+Precomp.	9.22 kB

ROM

Algorithm	Code Space
BLS	61.63 kB
ECDSA	48.59 kB
ECDSA+Precomp.	49.23 kB

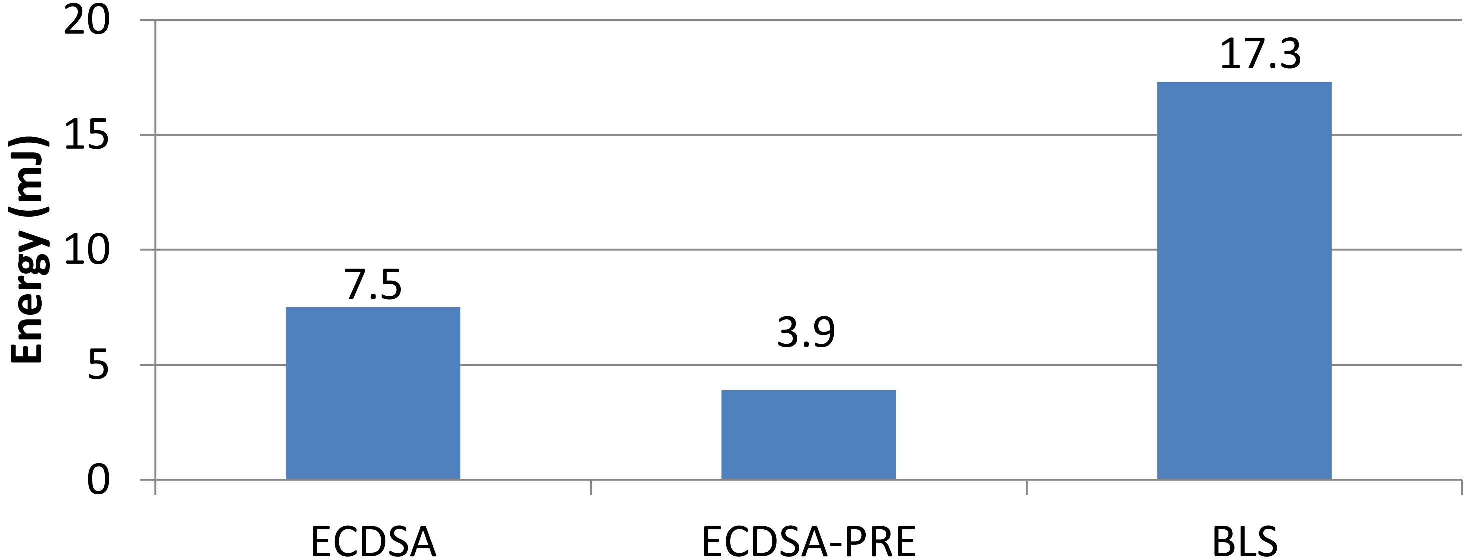
- Without assembly optimization, use 5.9kB more ROM.
- ROM footprints different than MSP: commercial compiler used for MSP, different assembly opt.

ARM: Computation



- 16x faster than MSP

ARM: Total Energy



- Communication (same radio as with MSP)
  - ECDSA: 0.59 mJ
  - BLS: 0.40 mJ

Related problems

- If gateway is trusted and resource rich
  - Shared key between sensor and gateway, digital signature from gateway
- Node to node or node to one user (few users)
  - Message Authentication Codes (MACs), symmetric crypto
- Node to multiple nodes
  - LEAP (Zhu 2003)
- User to multiple nodes
  - uTESLA (Perrig 2001)

Conclusions

- Digital signatures preferred for authentication with shared sensors
- Even though BLS has lower communication overhead and the same underlying operation, ECDSA has lower total energy
  - Same level of security
  - Implementation optimizations matter

Background

Elliptic Curve Cryptography

- Security of cryptosystems rely on hard problems
- Traditional cryptosystems (RSA/DSA) rely on subexponential problems (e.g. DLP)
- ECC relies on fully exponential problems
- Parameters in ECC are then much smaller than RSA/DSA
- The most important underlying operation in point multiplication is ModMult

ECDSA

- Signature generation
  - One point multiplication
- Signature verification
  - Two point multiplications
- Signature length
  - 40 bytes
- Cons
  - ECDSA’s signature is as long as DSA’s
  - Certificate-based scheme

Boneh-Lynn-Shacham Scheme

- Signature generation
  - One point multiplication
- Signature verification
  - Two pairings
- Signature length
  - Around 20 bytes
- Cons
  - Pairings are expensive
  - Certificate-based scheme

Chosen schemes

- ECDSA and BLS
- They both are based on certificates
  - Certificates solve the key authentication problem
  - A public key of B held by A does in fact belong to B
- Due to the direction of the communication (node-to-users) that is not a problem here

Chosen schemes' costs summary

Signature Scheme	Computation		Communication
	Generation	Verification	
ECDSA	1 point mult	2 point mult	320 bits
BLS	1 point mult	2 pairings	around 170 bits

- Verification in BLS may be very expensive

Chosen schemes’ costs summary

Signature Scheme	Computation		Communication
	Generation	Verification	
ECDSA	1.2s	2.4s	320 bits
BLS	1.2s	**16.8s**	around 170 bits

- And in fact it is!
- In our scenario, only users are required to verify signatures

Outline

- Introduction
- Goal
- Solution
- Results
- Conclusion

What

Authentication

- Source authentication
  - Ensures a receiver that the message originates from the claimed sender
- Data origin authentication
  - Ensures a receiver that the msg from the sender is “fresh” and its content was unchanged (integrity)

Why

Security in WSNs

- Due to the nature of WSNs, attackers can easily
  - Take part in the network activities
  - Inject bogus data
  - Alter the content of genuine messages
  - Impersonate other nodes
- Authentication the most important security property in WSN communication

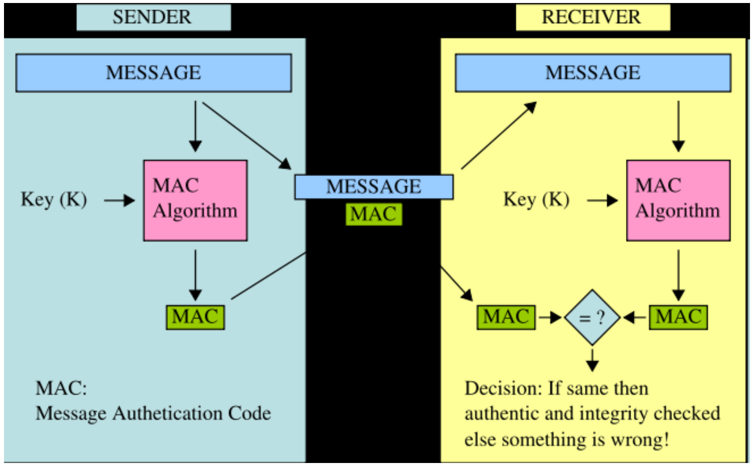
How

Authentication in WSNs

- Node-to-Node/Node-to-User
  - Message Authentication Codes (MACs)
- Node-to-Nodes
  - LEAP (Zhu 2003)
- User-to-nodes
  - uTESLA (Perrig 2001)

MACs

- Symmetric scheme
- Computation is negligible even for motes
- Communication overhead is about 16-20 bytes



Authentication in WSNs

- Node-to-Node/Node-to-User
  - MACs
- Node-to-Nodes
  - LEAP (Zhu 2003)
- User-to-nodes
  - uTESLA (Perrig 2001)
- Node-to-Users
  - ?

Why node2multi-user?

- Nodes may want to send the same information to multiple users
- A WSN w/ multiple base stations
- A tiny web server reporting data to multiple users over the web
- And so on

Outline

- Introduction
- Goal
- Solution
- Results
- Conclusion

Goal

- Authenticate node to multi-user communication in WSNs

Authentication in WSNs

- Node-to-Node/Node-to-User
  - MACs
- Node-to-Nodes
  - LEAP (Zhu 2003)
- User-to-nodes
  - uTESLA (Perrig 2001)
- Node-to-Users
  - Multiple unicasts using MACs? Hum…

Multiple Authenticated Unicasts

- A node generates and sends multiple MACs using different keys
  - Each key is shared w/ a different user
- Each user checks the legitimacy of a msg as it does for any other msg
  - Generates a MAC and check if it matches the received one

Problems

- Each pair (user,node) needs to agree in a common key
  - Key predistribution
    - Users need to be known a priori
  - Key agreement protocol
    - Probably demands PKI/certificates
- Nodes need to store multiple keys
- Not efficient for a large number of users

Why many users?

Fire Alarm Application

- Fire alarm system is based on sensor nodes
- Fires are rare and comm. between nodes and users might not even take place
- It is one way communication
  - Users do not query sensors about fires
- But in case of a fire, every user wants to aware (and be sure that is serious!)

Outline

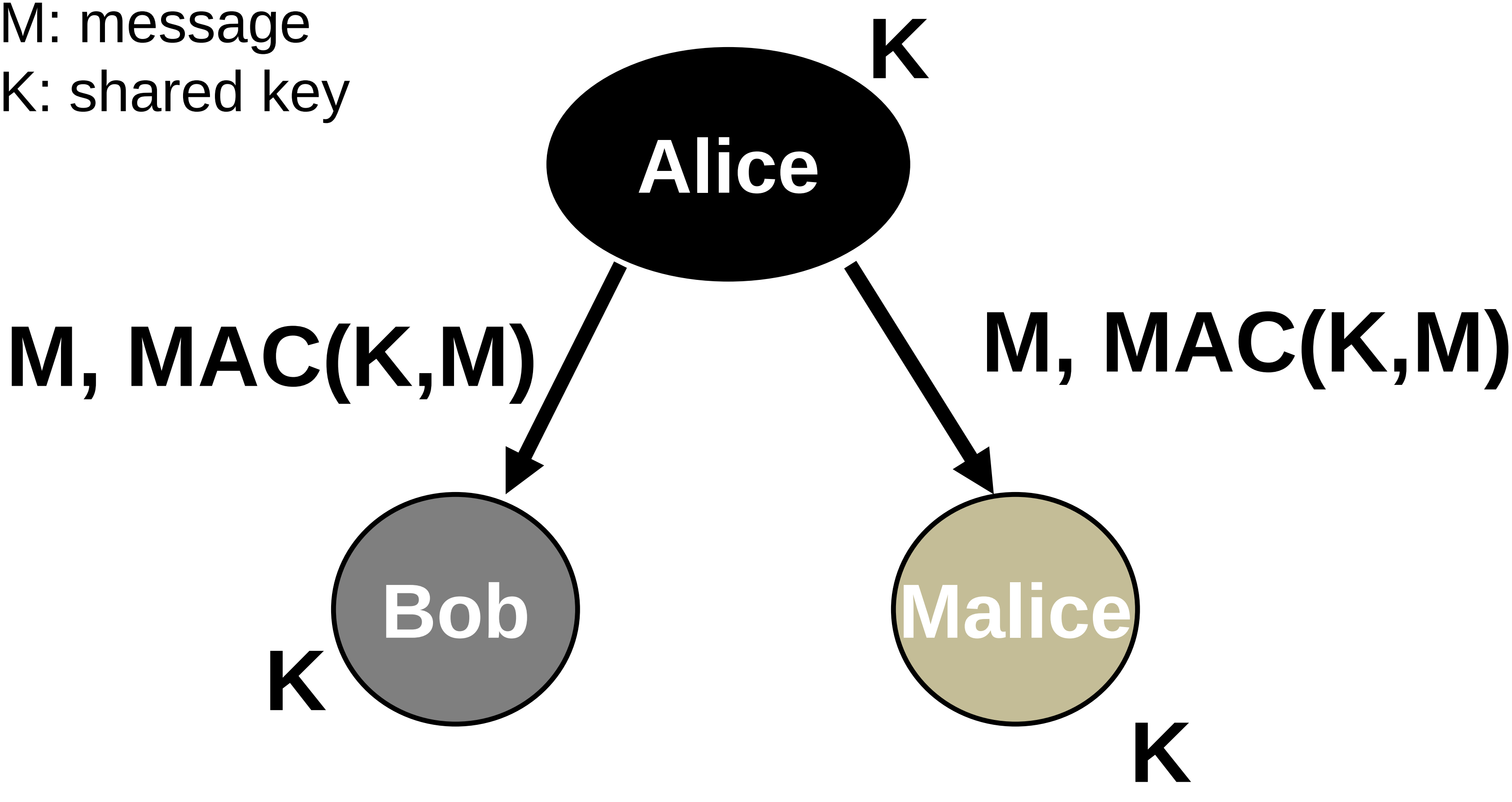
- Introduction
- Goal
- Solution
- Results
- Conclusion

Authenticated Broadcast

- Group secret key
  - Symmetric scheme (Cheap!)
  - All principals of the communication share the same key
- Digital signature
  - Sender uses its pvt key to sign msgs
  - Receivers verify the signature using sender’s pub key

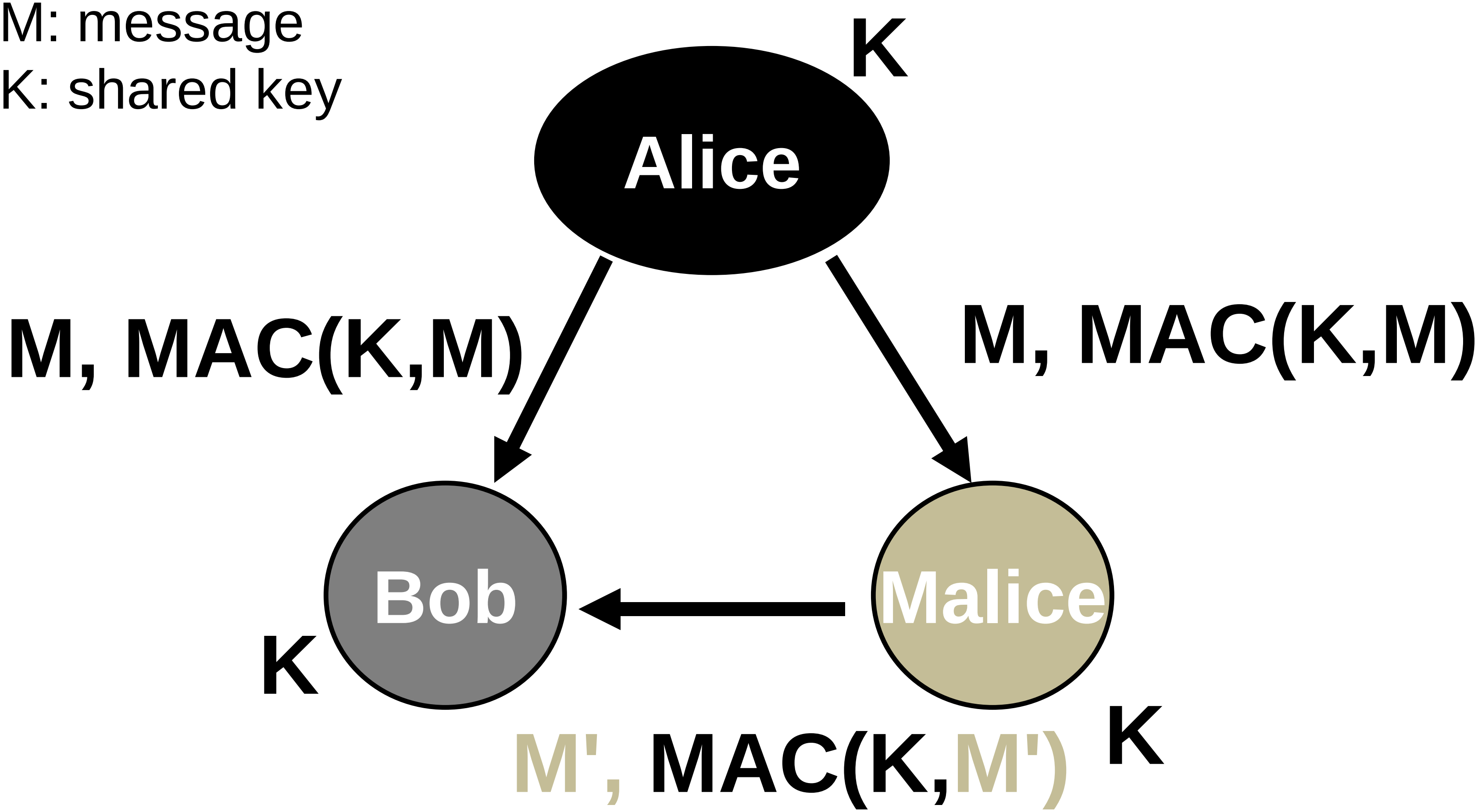
Group key-Based Authenticated Broadcast

- M: message
- K: shared key



Group key-Based Authenticated Broadcast

- M: message
- K: shared key



Digital signatures

- Asymmetric scheme
  - Each principal has a pvt/pub pair of keys
- Computation and communication overhead are usually high
- Properties
  - Source authentication
  - Data origin Authentication
  - Non-repudiation

Digital Signature Schemes

- DSA
- ECDSA
- ID-based
- Certificateless
- BLS

DSA

- Traditional public key cryptosystem
- The standard signature scheme
- Parameters are large
  - Expensive computation, communication, keys and storage
- Problem
  - Costs
  - Known to be inadequate for WSNs

ECDSA

- Signature generation
  - One point multiplication
- Signature verification
  - Two point multiplications
- One-slide overview of ECC

Elliptic Curve Cryptography

- Security of cryptosystems rely on hard problems
- Traditional cryptosystems (RSA/DSA) rely on subexponential problems (e.g. DLP)
- ECC relies on fully exponential problems
- Parameters in ECC are then much smaller than RSA/DSA
- The most important underlying operation in point multiplication is ModMult

ECDSA

- Signature generation
  – One point multiplication
- Signature verification
  – Two point multiplications
- Signature length
  – 40 bytes
- Cons
  – ECDSA’s signature is as long as DSA’s
  – Certificate-based scheme

Identity-Based Signatures

- Shamir 1984
- No need of certificates
- Costs varies depending on the scheme
  – Today schemes rely mostly on pairings
- One-slide overview of PBC

Pairings

- A bilinear map of groups $\mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$
- First used in the context of cryptanalysis
  - Map the ECDLP into the DLP
- It has the property of bilinearity
  - $e(aP,bQ)= e(P,Q)^{ab}$
  - Consider sID as pvt keys, where s is a mater key
  - $e(sAlice,Bob) = e(Alice, Bob)^{s} = e(Alice, sBob)$

Identity-Based Signatures

- Shamir 1984
- No need of certificates
- Costs varies depending on the scheme
  - Today schemes rely mostly on pairings
- Cons
  - Pairings are expensive
- Problem
  - Requires a Trusted Authority

Certificatless

- Al-Riyami and Paterson 2003
- Neither certificates nor TAs
- Signature generation
  - One pairing and two point multiplications
- Signature verification
  - Four pairings
- Four pairings is a problem for resource constrained devices

Boneh-Lynn-Shacham Scheme

- Signature generation
  - One point multiplication
- Signature verification
  - Two pairings
- Signature length
  - Around 20 bytes
- Cons
  - Pairings are expensive
  - Certificate-based scheme

Chosen schemes

- ECDSA and BLS
- They both are based on certificates
  - Certificates solve the key authentication problem
  - A public key of B held by A does in fact belong to B
- Due to the direction of the communication (node-to-users) that is not a problem here

Chosen schemes' costs summary

Signature Scheme	Computation		Communication
	Generation	Verification	
ECDSA	1 point mult	2 point mult	320 bits
BLS	1 point mult	2 pairings	around 170 bits

- Verification in BLS may be very expensive

Chosen schemes' costs summary

Signature Scheme	Computation		Communication
	Generation	Verification	
ECDSA	1.2s	2.4s	320 bits
BLS	1.2s	**16.8s**	around 170 bits

- And in fact it is!
- In our scenario, only users are required to verify signatures

Figures?

Implementation

- Based on the Multiprecision Integer and Rational Arithmetic C Library (MIRACL)
  - Scott, MIRACL's author, is coauthor of the etat pairing (Barreto et al. 2006)
    - ESI hot paper (0.1% most cited paper in category)
  - Support for various node processors
    - AVR, ARM, and MSP430
- Prime fields
- RSA 1024-bit security level

Evaluating Platform

- M-Platform
  - MSP430F2418
    - 16-bit 16MHz processor, 8KB of RAM, 116 of ROM
  - ARM LPC2138
    - 32-bit processor, 32kB RAM and 512kB ROM
- Radio
  - CC2420 (802.15.4)

Outline

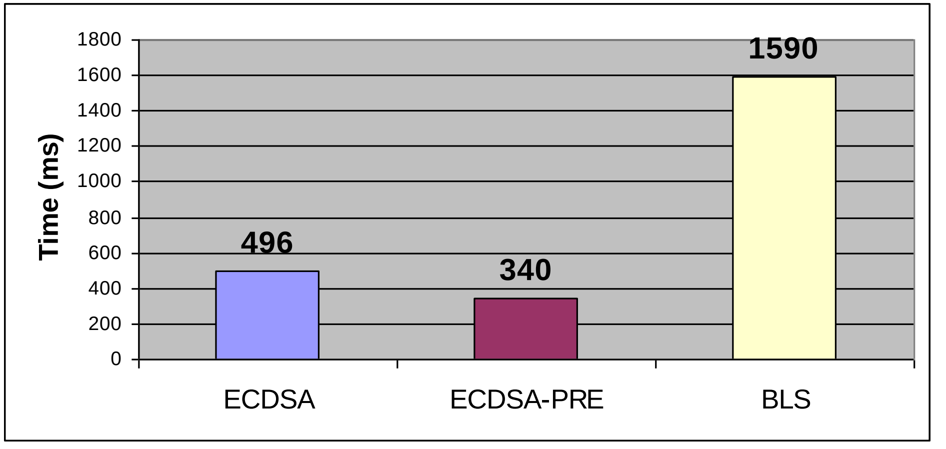
- Introduction
- Goal
- Solution
- Results
- Conclusion

MSP430

Storage

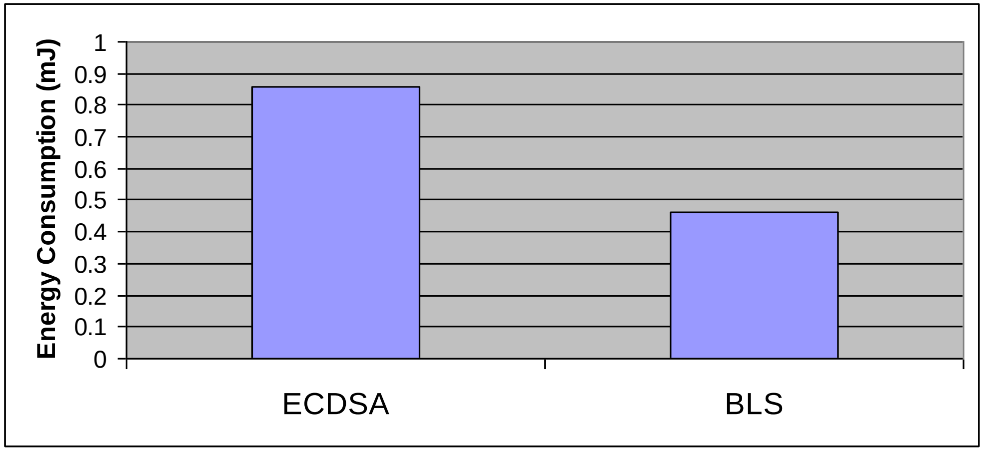
- RAM
  - Around 3KB, but most from the stack
  - After signature generation, virtually all memory is available for applications
- ROM
  - Around 40KB
  - In ECDSA, if precomputation is used, then it takes more ~½ KB

Computation Time



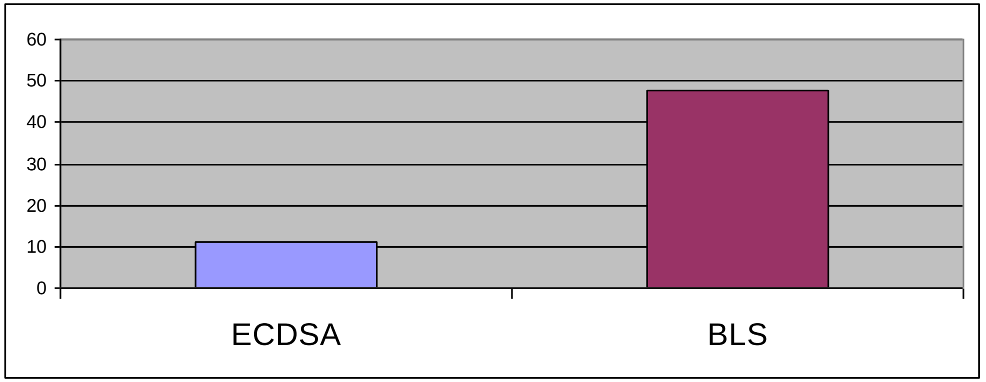
- ECDSA can make use of a special prime which speeds up the reduction (*x mod p*)
- ECDSA also allows precomputation
- BLS needs to hash and map msgs into a point of the curve (a bit expensive, too)
- ECDSA is around 4.7X faster than BLS

Communication



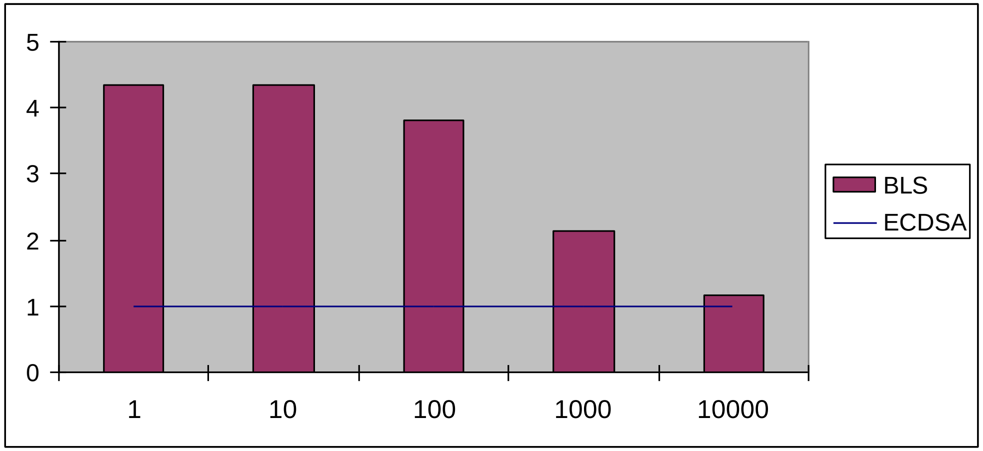
- Energy cost to transmit a signature
- In our implementation, BLS's signature has exactly half the length of ECDSA's
- Here we also assume radio start-up

Overall Energy Consumption



- The overall energy consumption is dominated by the computation
  - The energy cost to generate a signature in BLS is about 100x higher than the cost to transmit that
  - While it takes 1.5s to generate a signature it is sent in only 1.7ms
- ECDSA turned out to be more than 4x cheaper

Overall Energy Consumption



- Overall consumption as function of the payload size
- The security costs are fixed whatever the payload size
  - (To tell the truth, the hash function cost varies a bit)
- For a large payload, cost is dominated by its transmission

Discussion

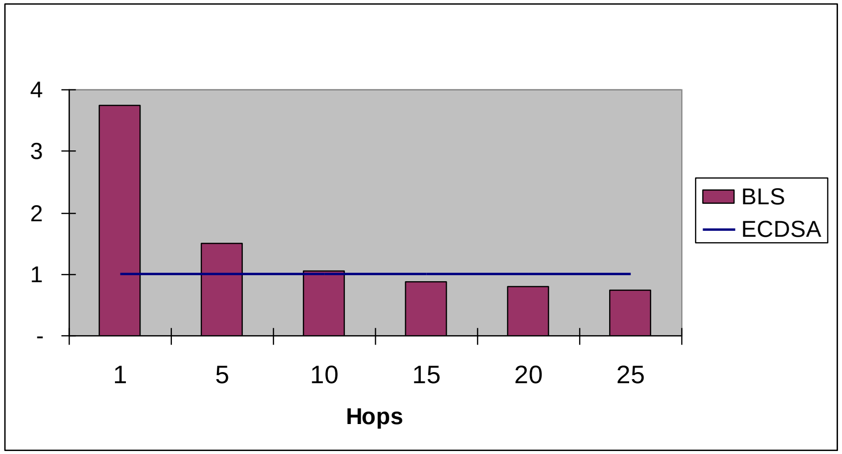
- User needs rather than security costs may be the most important criteria
- ECDSA can compute part of a signature generation offline
  - Response time will be even faster
- BLS provides signature aggregation
  - Sig1 + Sig2 = Sig1,2
  - Users may store signatures aggregated thus saving storage

ARM

Computation

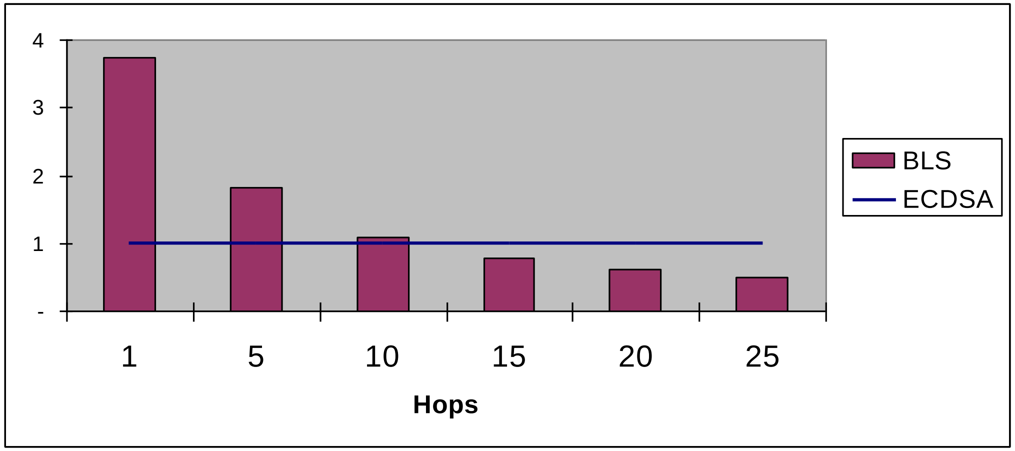
- Estimates based on liu et al. 2007
- ARM is able to compute ECDSA faster
  - 11.8ms
- We assumed the same ratio to estimate BLS
  - ECDSA signature generation 4.7x faster than BLS's
- Signature generation is now 20x more expensive than its transmission
  - Remember signatures need to be received as well

Multi-hop/Single Signature



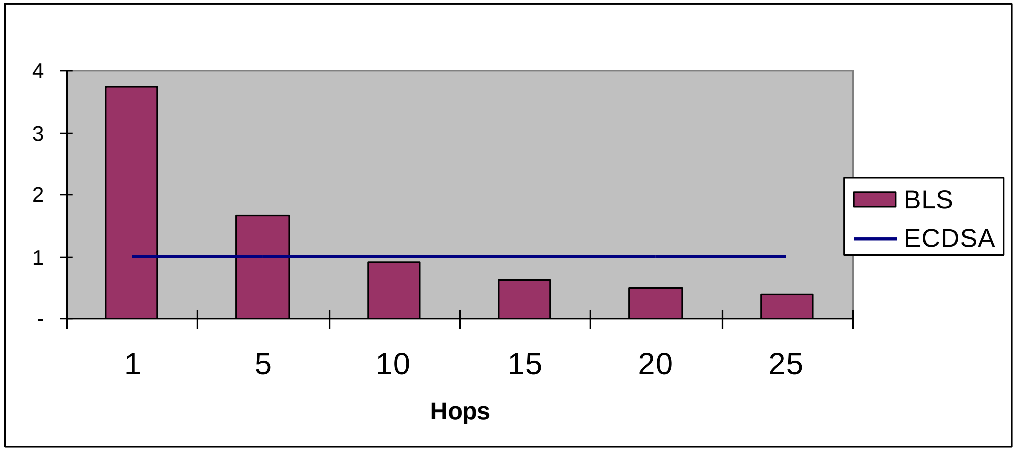
- Overall energy consumption for generating a signature and sending it over multiple hops
- BLS aggregates signatures
- Computation is carried only once, signatures are (re)sent over multiple hops
- Break even point is at the 12th hop

Multi-hop/Multiple Signatures



- Now every node over the path also generates and send a signature
- BLS makes use of signature aggregation
- Break even point is the 12th
- Beyond that point BLS scales far better

Routing tree/Multiple Signatures



- Now assuming a (ternary) tree routing protocol
- Break even point is at the 10th hop
- Again, beyond that point, BLS scales much better

Outline

- Introduction
- Goal
- Solution
- Results
- Conclusion

Conclusion

- We compared BLS and ECDSA schemes for authenticating node to multi-user communication
- First figures for BLS in resource-constrained nodes
- Figures for signature communication overhead
- Signature computation dominates the costs in signature schemes
  - As opposed to symmetric schemes, where communication is the “bottleneck”
- Powerful processors are more energy efficient when generating signatures

Conclusion

- Schemes should be chosen based on the network idiosyncrasies
- ECDSA allows optimizations that makes it more efficient for small-scale WSNs
- BLS aggregate signature feature seem to be a useful feature in multi-hop WSNs
- For a large amount of data, the security costs may become negligible
  - The chosen scheme should be driven by the user side needs

Future Directions

- Implement critical times routines in assembly
  - Computation costs as a whole should decrease
  - The difference between ECDSA's and BLS's costs probably will decrease, too
- Other BLS implementation
  - Under $GF(3^m)$

Thank you

leob@ic.unicamp.br