

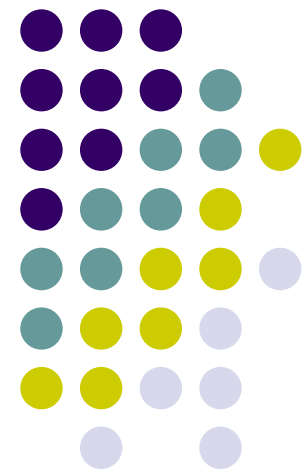
Authenticated Flooding in Large-scale Sensor Networks



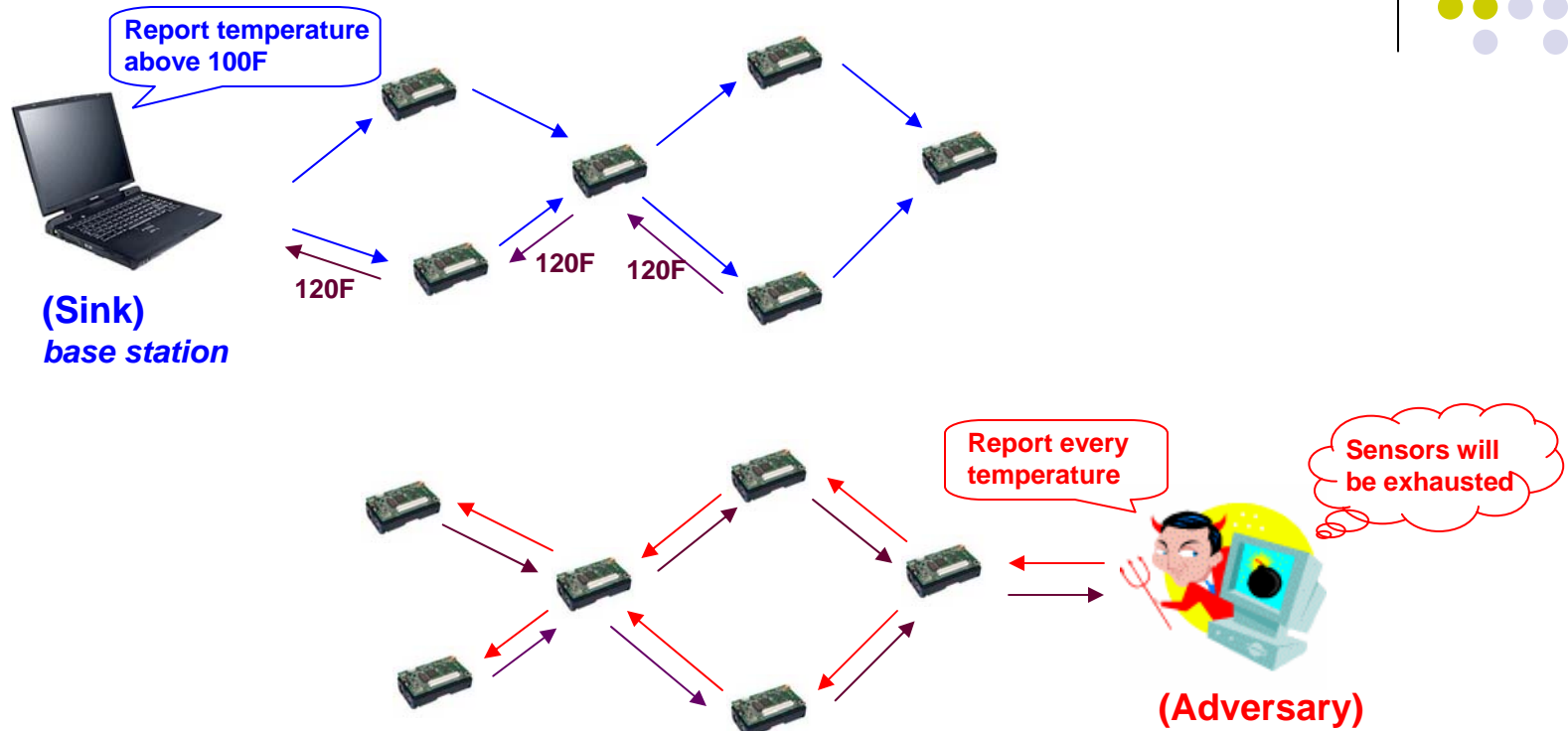
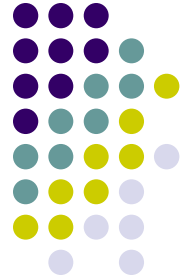
Ju-Hyung Son

with Haiyun Luo and Seung-Woo Seo

*Seoul National University
University of Illinois at Urbana-Champaign*



Motivation



- Every sensors should ***Authenticate*** sink's broadcast message before execution!

Introduction



- Problem Definition
 - Adversaries can activate sensors by flooding false control messages
 - Sensors will waste their energies by responding the false messages
 - We need **authentication of sink's control message**
- Goal
 - Efficient and Scalable authentication mechanism with
 - Low computation overhead
 - Low communication overhead

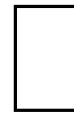
Previous Approach – 1



- Using “**Cryptographic Asymmetry**”
 - Sink attaches *Digital Signature* (signed by sink’s Private Key) for each message
 - Sensor verifies the *Digital Signature* using sink’s Public Key

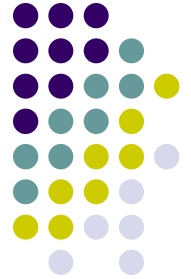


Pentium 4GHz CPU



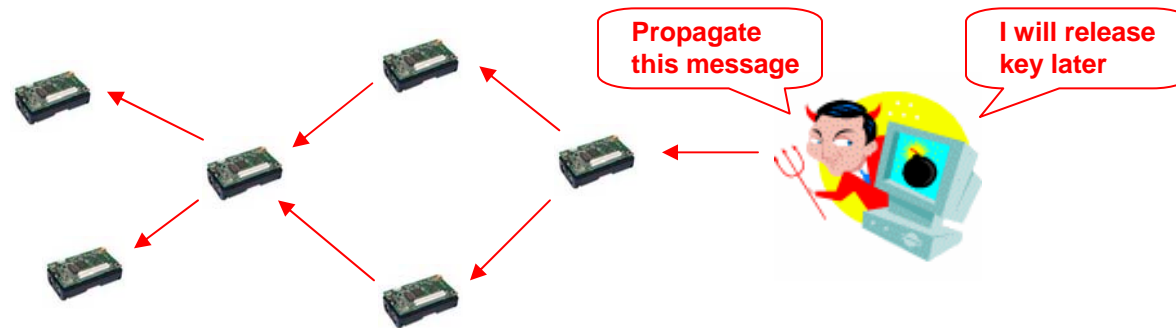
8 MHz MCU

- Signature verification is too heavy for sensors
- Continuous verification request of *false signature* can be used in Denial-of-Service attack



Previous Approach – 2

- Using “*Time Asymmetry*”
 - First, sink broadcast message attached with MAC (Message Authentication Code)
 - Later, sink releases the secret key used in the previous MAC



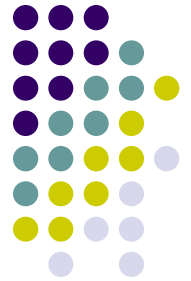
- Sensors cannot authenticate immediately
- Forwarding false message will waste much energies

Our Approach



- We propose
 - A. “Information Asymmetry” model**
- How to securely share information?
 - B. Random pre-deployment**
- How to efficiently represent multiple information?
 - C. Bloom Filter**
- Then we combine all these into
 - D. BMAP (Bloom-filter based Message Authentication Protocol)**

A. Information Asymmetry model

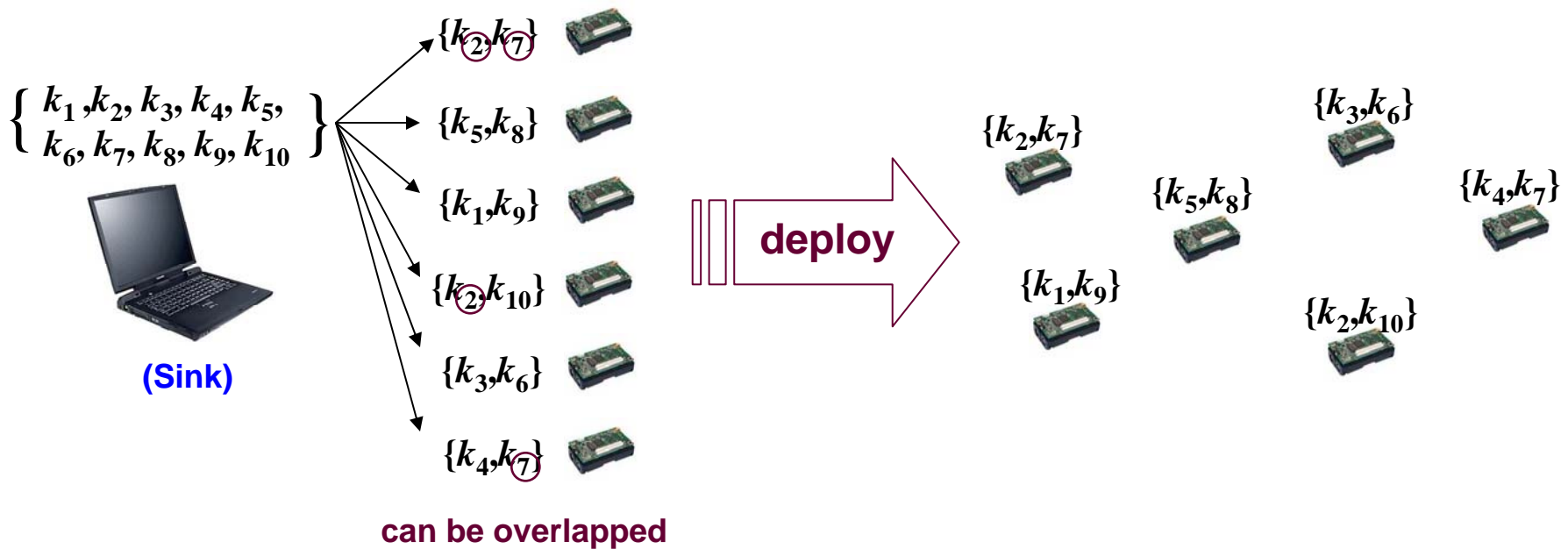


- Definition
 - Information \rightarrow (e.g. Secret Key)
 - Knowledge $\rightarrow f$ (Information + Message) (e.g. MAC)
- Model
 - n info. at sink
 - l info. at each sensors
 - n_c compromised info. at adversaries
 - Information Asymmetry: $n \gg n_c, l$
- Procedures
 - sink represents n knowledge on each message
 - sensors verify the message using l information

B. Random pre-deployment



- Information (=secret key) is randomly distributed before deployment

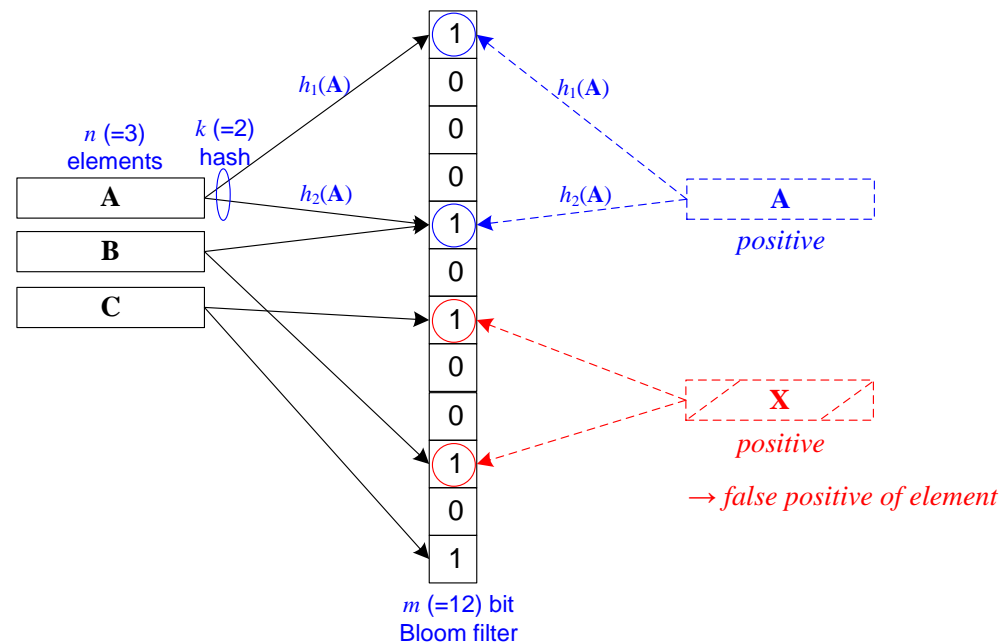


- Only sink knows whole information



C. Bloom Filter

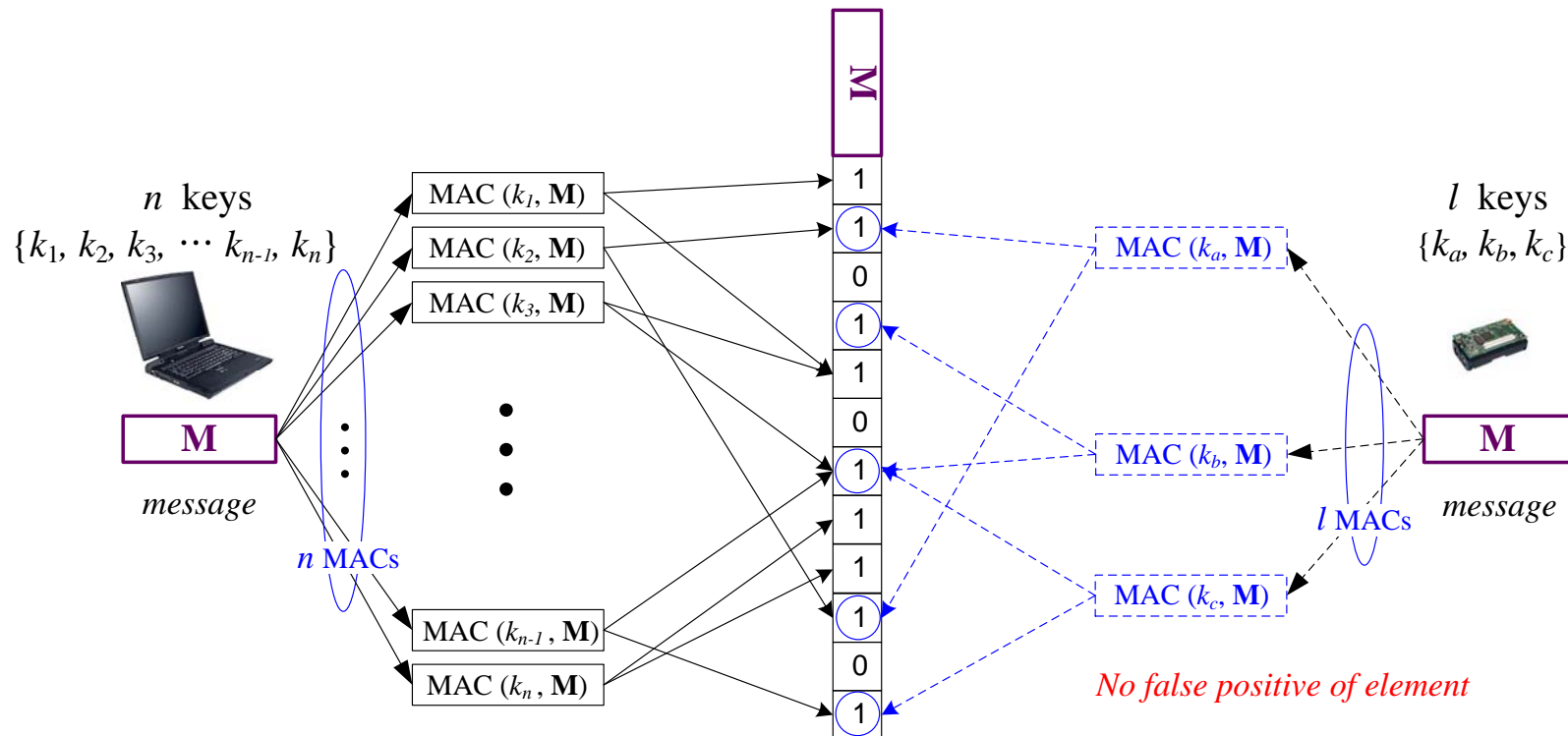
- Bloom Filter (by *Burton Bloom*, 1970)
 - Compression of multiple elements with *false positive rate*



- If m/n (=bits per element) increases, *false positive rate* decreases
 - Large Bloom filter \rightarrow Small *false positive rate*

D. BMAP

- Bloom-filter based **M**essage **A**uthentication **P**rotocol

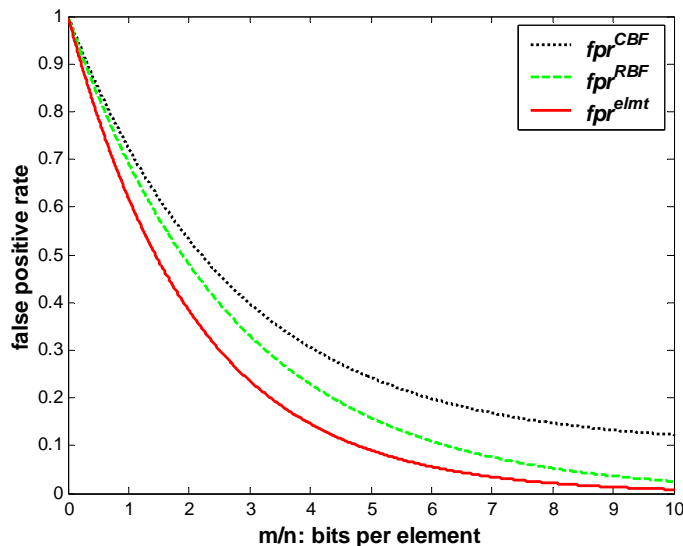


- Low computation overhead at sensors (several hash computations)
- Small message size due to Bloom filter compression
- Since keys are same between sink and sensors, there is no false elements



Analysis of Attack Scenarios

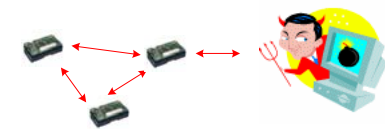
- Random Bloom Filter
 - Adversaries randomly generate false Bloom filter
- Compromise Bloom Filter
 - Adversaries use keys obtained from compromised sensors
- **false positive rate** of Bloom Filter
 - prob. of false Bloom filter succeeds in authentication of a certain sensor node



Large network,
Small message size



How can we further decrease
'false positive rate' ?

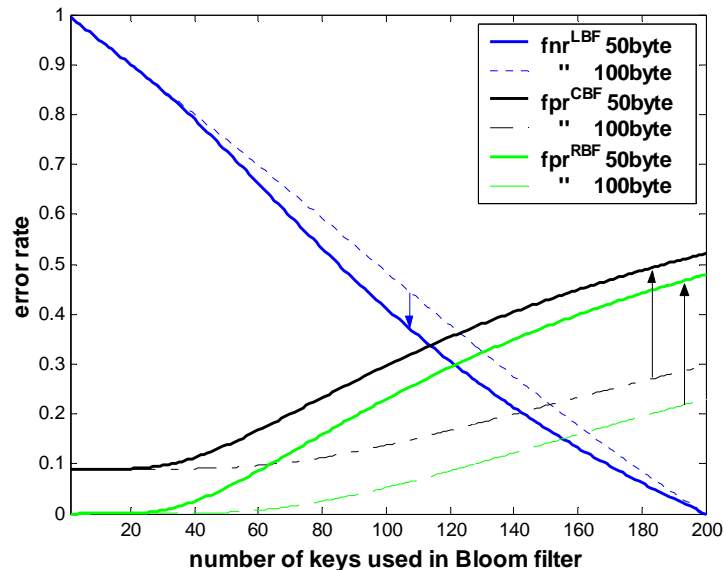


m : Bloom filter size,
 n : number of keys



Introducing False Negative Rate

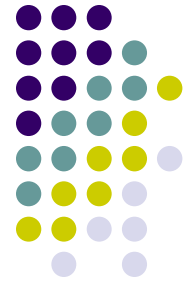
- False Negative?
 - sink only maps n' keys out of n keys
 - some sensors will fail to authenticate msg. from sink
- **false positive** : security degradation
- **false negative** : performance degradation
 - Lower **false positive rate** by relaxing **false negative rate**



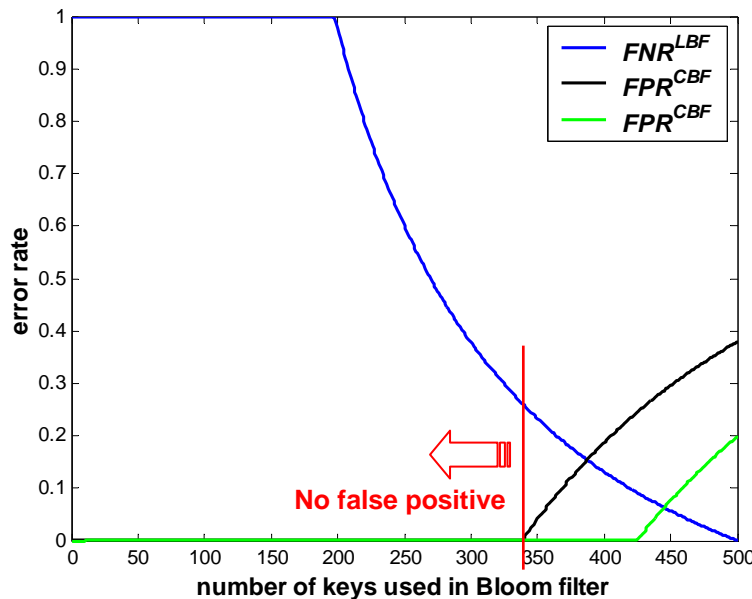
Trade-off between
false negative rate and
false positive rate

More trade-offs on
small Bloom filter

Combining with Flooding Model



- Flooding Model ?
 - Each sensor forward received message only once
 - Only if it succeeds authentication



Trade-off between
false positive rate and
false negative rate

BMAP Optimization Framework



- On each sensor network settings, how can we choose optimal parameters?

< Given constants >

- N : total number of sensors
- m : size of Bloom filter
- d : avg. number of neighbors for each sensor
- p_c : avg. percentage of compromised sensors

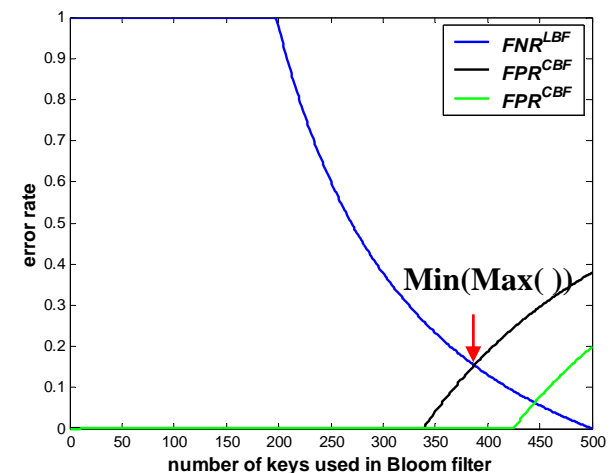
< Variables to optimize >

- n : total number of keys
- n' : number of keys actually used in Bloom filter
- l : number of keys for each sensors
- k : number of hash functions in Bloom filter

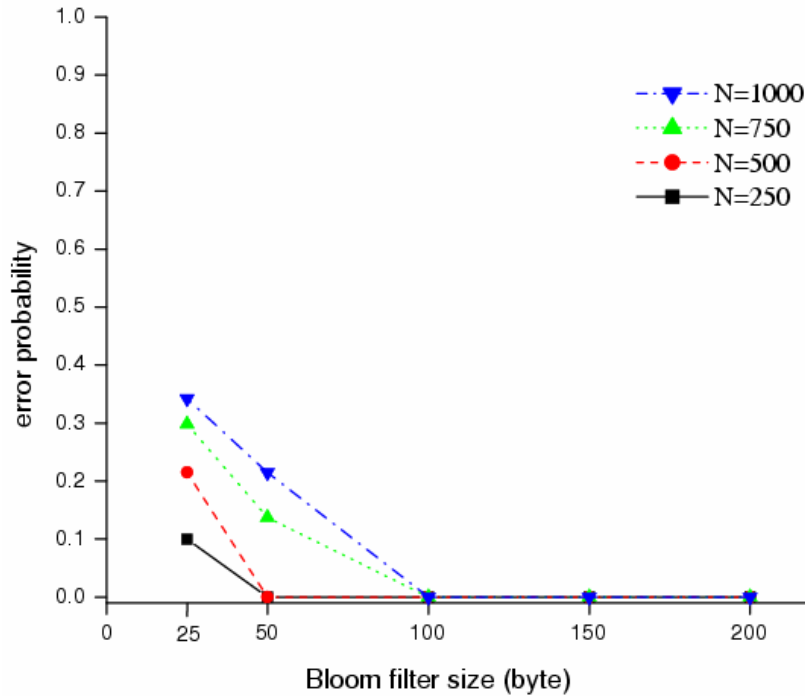
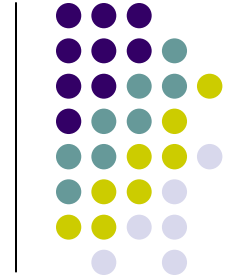
< Objective Function >

$$\text{Min} (\text{Max} (FNR^{LBF}, FPR^{RBF}, FPR^{CBF}))$$

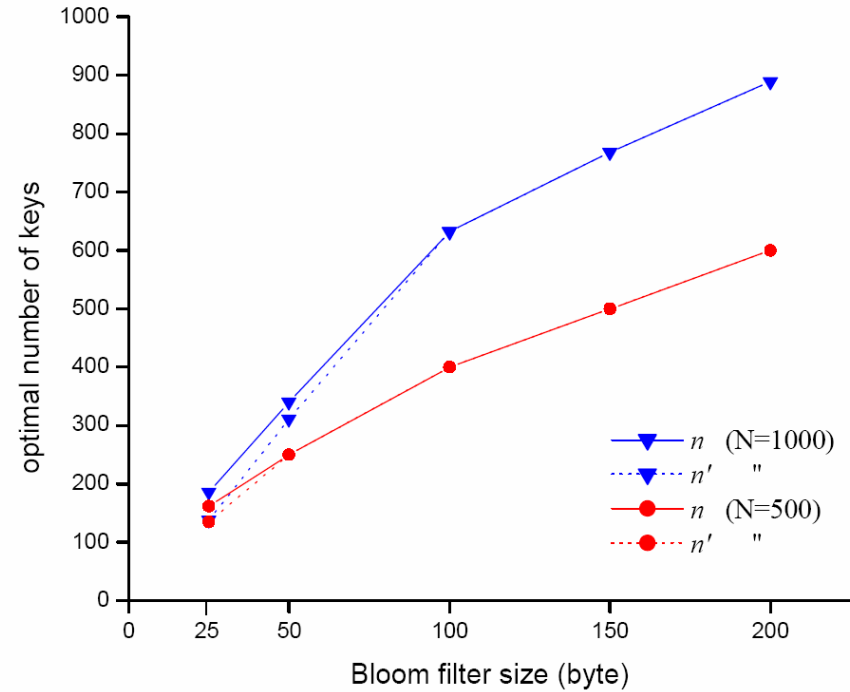
subject to $n' \leq n, n'k \leq m$



Optimization Results

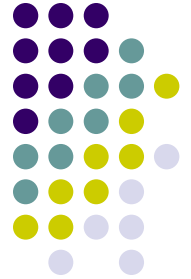


Results of Min (Max()),
 $p_c=5\%$



Optimal value of n and n',
 $p_c=5\%$

Discussions



- Size of Bloom filter
 - Sensor network requires smaller Bloom filter size
 - Bloom filter compression
 - One Bloom filter for multiple packets
- Effects of *false negative*
 - How much *false negative rate* is affordable in sensor networks?
 - Usually sensor network is redundant in topology and functionality
 - But it may depend on applications and scenarios

Conclusions



- BMAP provides broadcast message authentication with
 - Low computation overhead
 - Low communication overhead
- We analyzed two attack scenarios on our protocol, which incurs *false positive rate* of Bloom filter
- We decreased the *false positive rate* by introducing *false negative rate*
- In trade-off between two error rates, we minimized both of them by our optimization framework

Thank you !



Q & A

