Security for Sensor Networks: Cryptography and Beyond

David Wagner
University of California at Berkeley

In collaboration with:

Chris Karlof, David Molnar,
Naveen Sastry, Umesh Shankar

Learn From History…

Learn From History…

cellphones

1980 analog cellphones: AMPS

analog cloning, scanners
- fraud pervasive & costly

digital: TDMA, GSM

1990

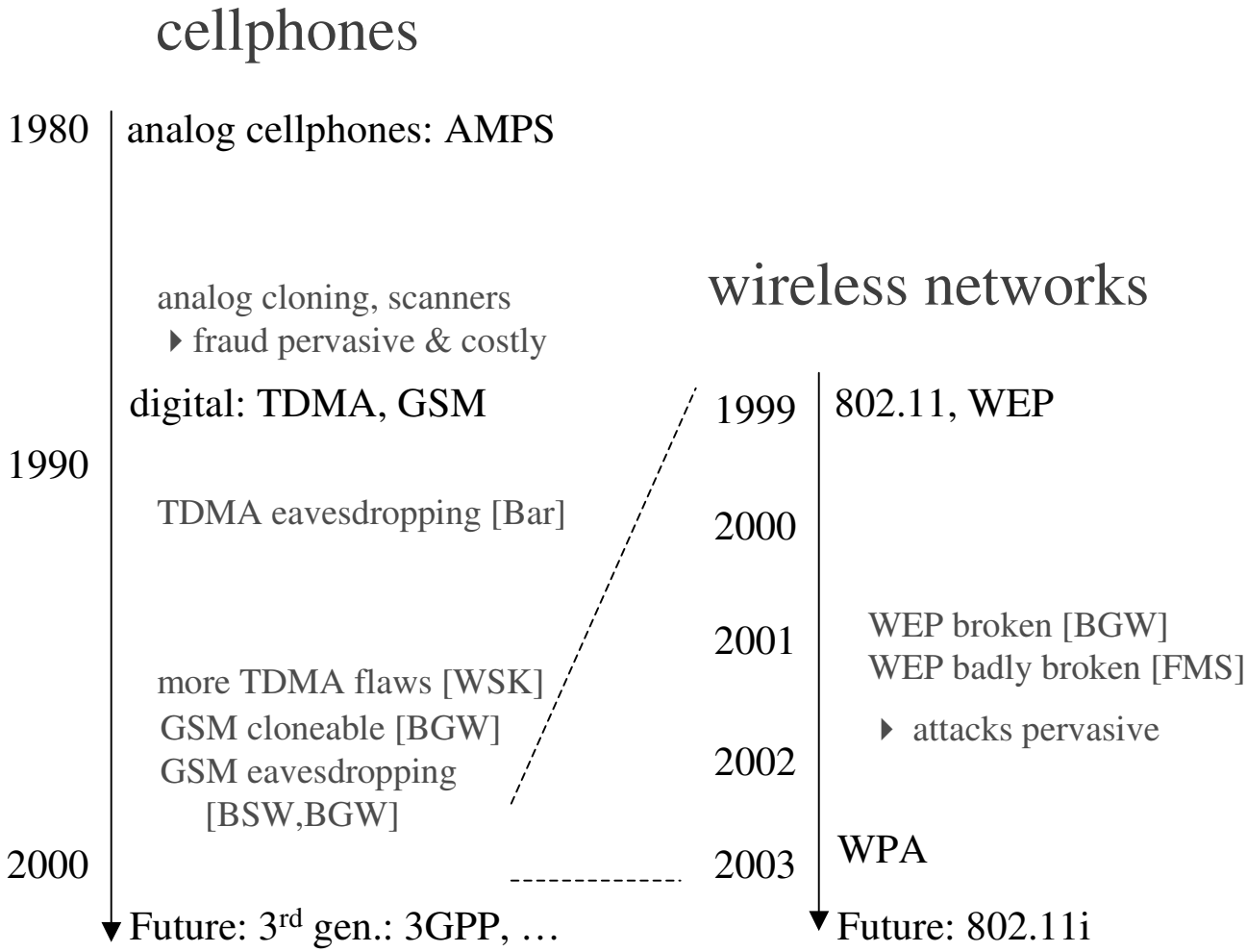
TDMA eavesdropping [Bar]

more TDMA flaws [WSK]
GSM cloneable [BGW]
GSM eavesdropping [BSW,BGW]

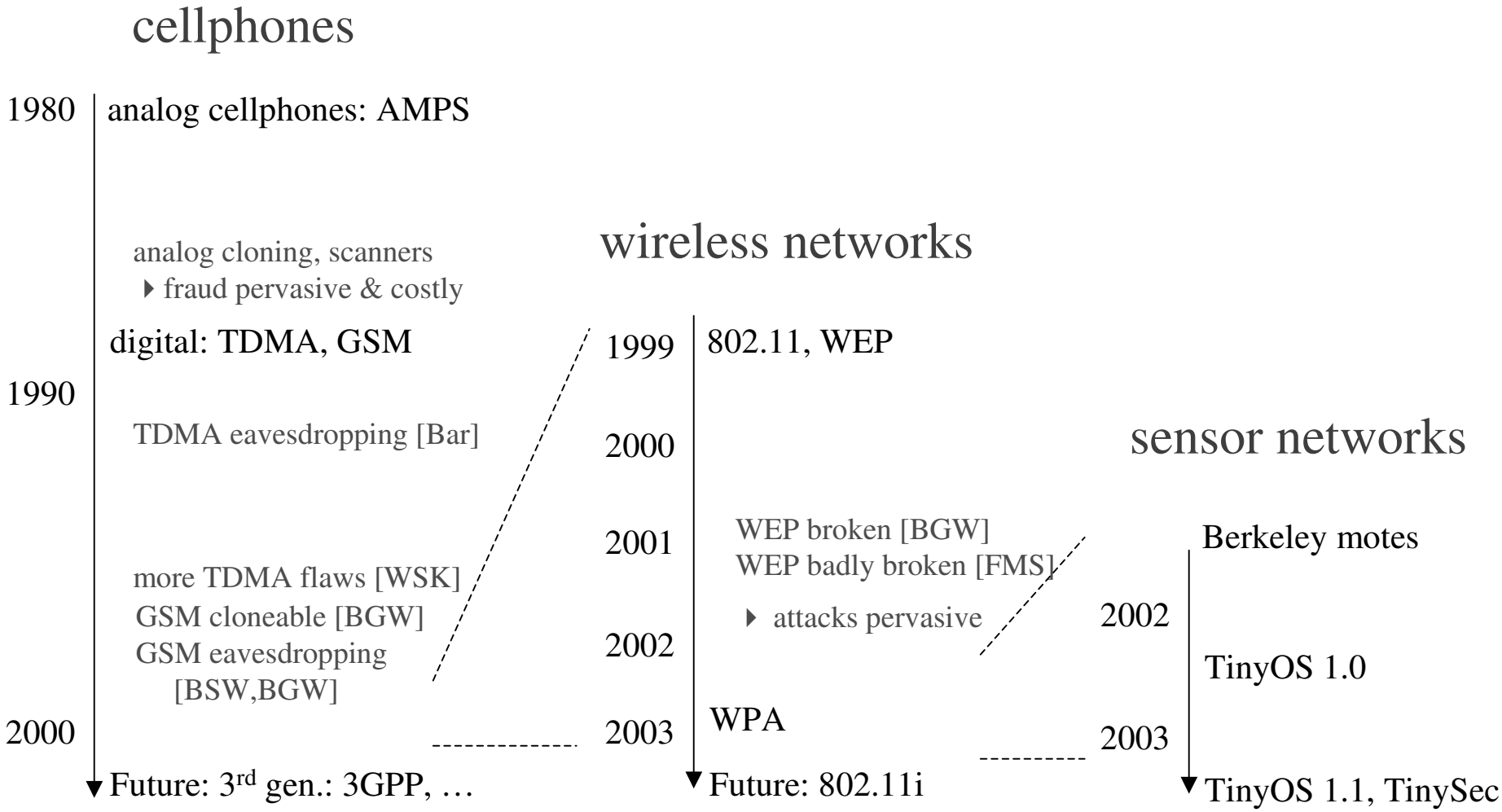
2000

Future: 3rd gen.: 3GPP, …

Learn From History…

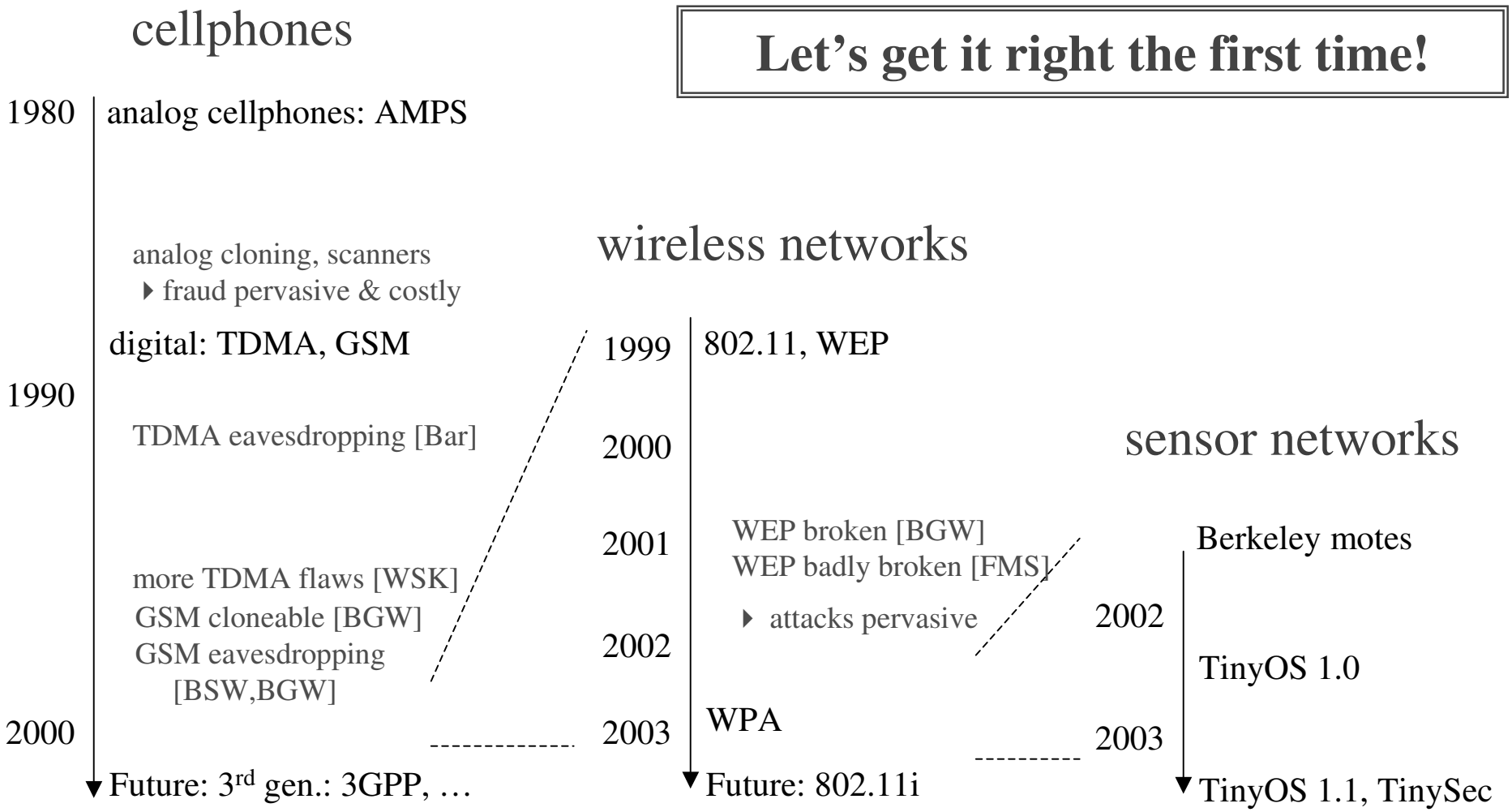


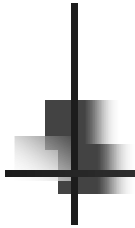
Learn From History…



Learn From History…

Let's get it right the first time!





Sensor Nets: So What?

What's different about sensor nets?

- Stringent resource constraints
- Insecure wireless networks
- No physical security
- Interaction with the physical environment

Sensor Nets: So What?

What's different about sensor nets?

- Stringent resource constraints
- Insecure wireless networks
- No physical security
- Interaction with the physical environment

Sensor Nets: So What?

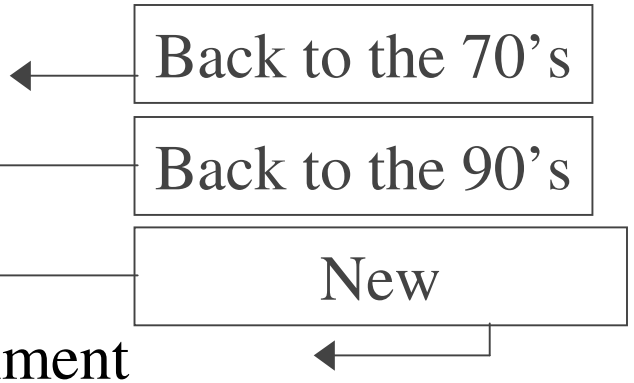
What's different about sensor nets?

- Stringent resource constraints ← Back to the 70's
- Insecure wireless networks ← Back to the 90's
- No physical security
- Interaction with the physical environment

Sensor Nets: So What?

What's different about sensor nets?

- Stringent resource constraints
- Insecure wireless networks
- No physical security
- Interaction with the physical environment



Where Are We Going?

Some research challenges in sensor net security:

- Securing the communication link
- Securing distributed services
- Tolerating captured nodes

Where Are We Going?

Some research challenges in sensor net security:

- Securing the communication link
- Securing distributed services
- Tolerating captured nodes

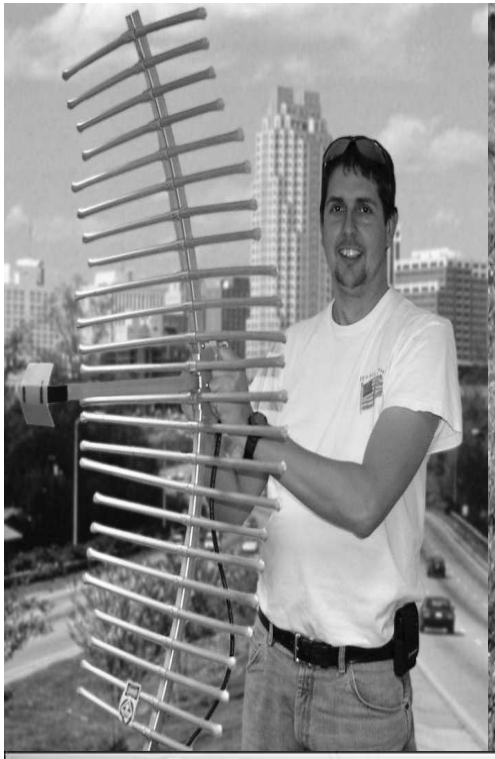
- **Cryptography and beyond**

In this talk: Techniques and thoughts on these problems.

I. Communications Security: The TinySec Architecture

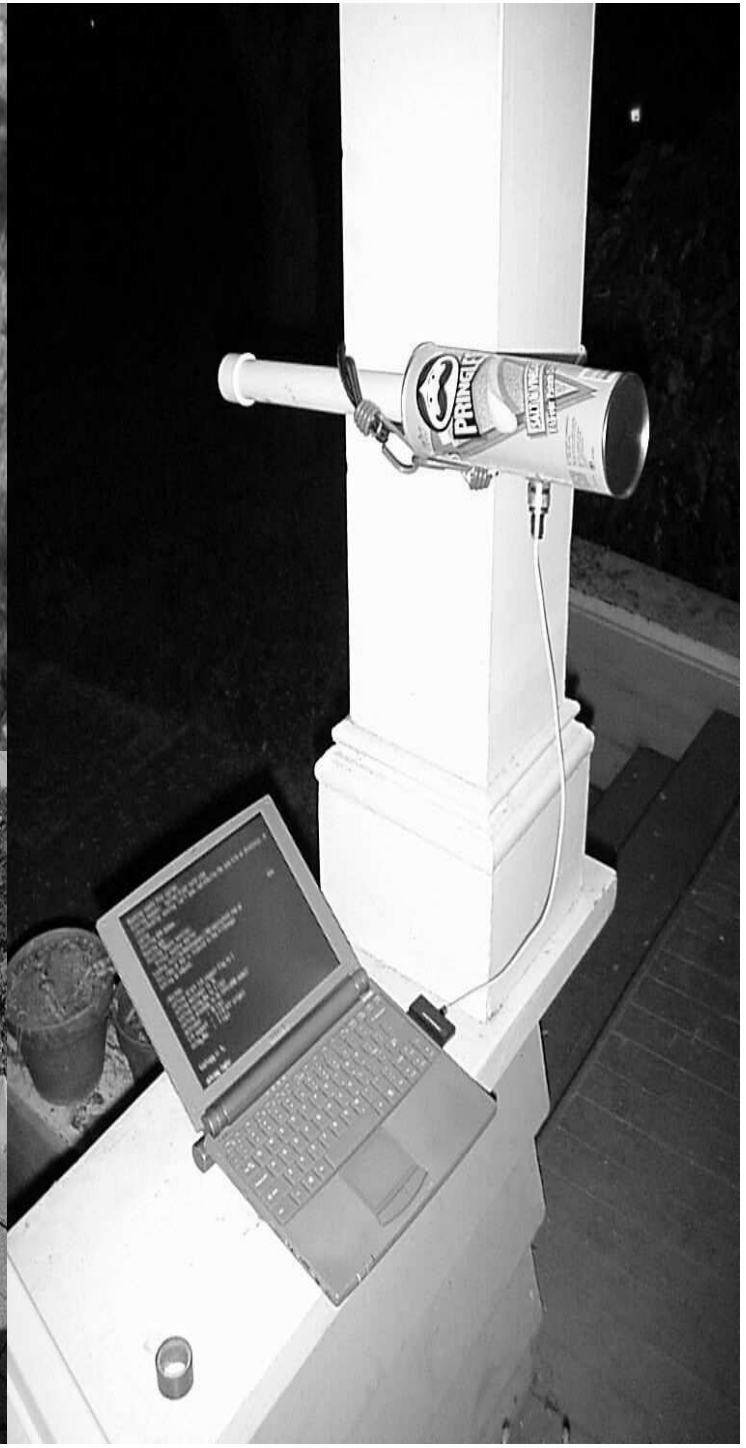
"It doesn't matter how good your crypto is if it is never used."

Mass. Street
8th Street
Wireless Network Map
Signal Strength
Strong
Weak



RINGLES
SPICY CAJUN
Potato Crisps

PRIMESTAR

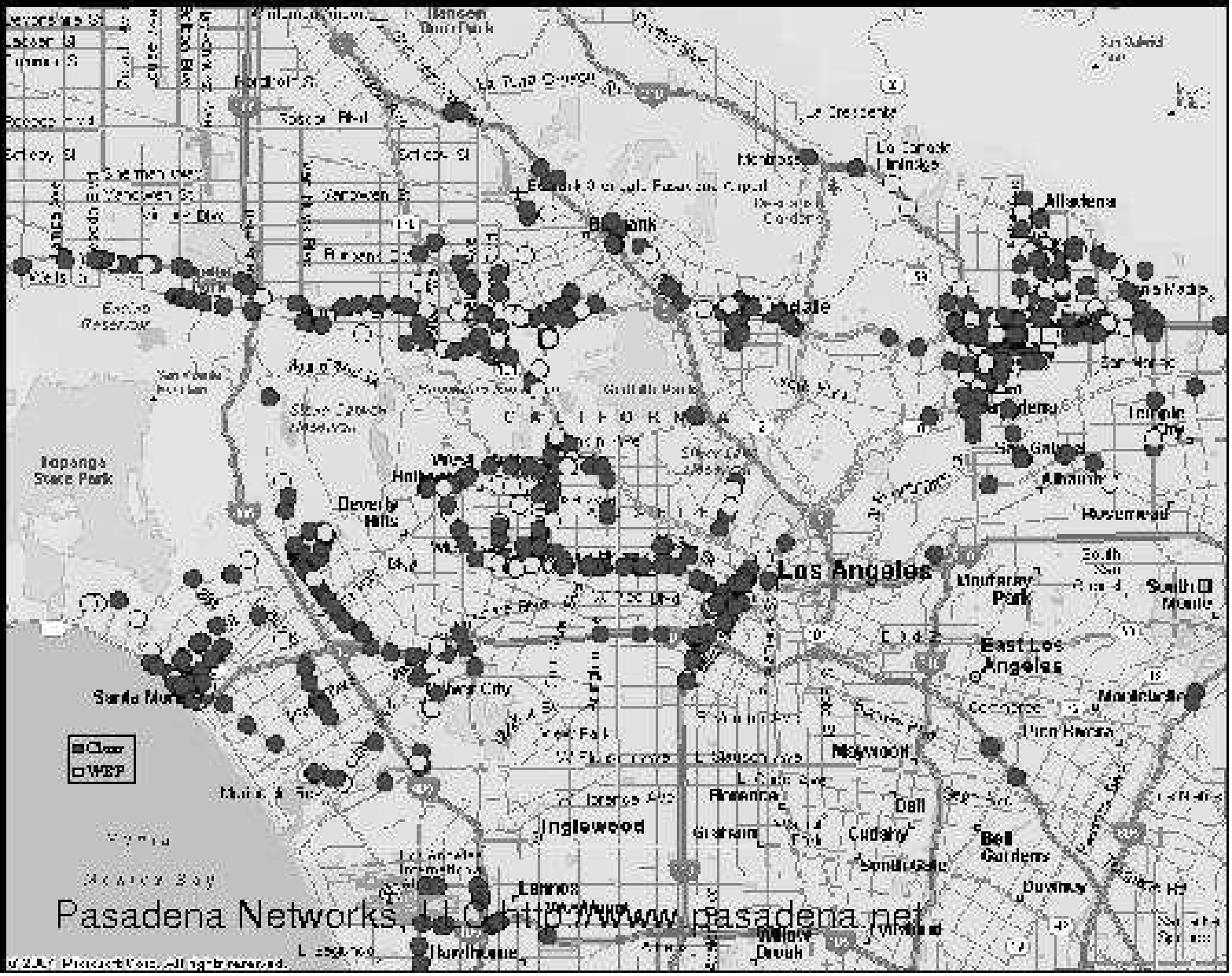

PRINGLES

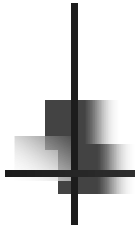
Wardriving / Access Point Mapping

468 WEP

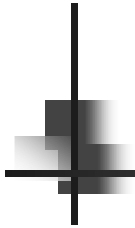
1,265 Clear

1,733 Total



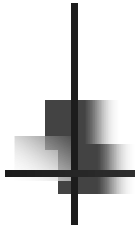


TinySec Design Philosophy



TinySec Design Philosophy

The lesson from 802.11:



TinySec Design Philosophy

The lesson from 802.11:

- Build crypto-security in, and turn it on by default!

TinySec Design Philosophy

The lesson from 802.11:

- Build crypto-security in, and turn it on by default!

TinySec Design Goals:

TinySec Design Philosophy

The lesson from 802.11:

- Build crypto-security in, and turn it on by default!

TinySec Design Goals:

1. Encryption turned on by default

TinySec Design Philosophy

The lesson from 802.11:

- Build crypto-security in, and turn it on by default!

TinySec Design Goals:

1. Encryption turned on by default
2. Encryption turned on by default

TinySec Design Philosophy

The lesson from 802.11:

- Build crypto-security in, and turn it on by default!

TinySec Design Goals:

1. Encryption turned on by default
2. Encryption turned on by default
3. Encryption turned on by default

TinySec Design Philosophy

The lesson from 802.11:

- Build crypto-security in, and turn it on by default!

TinySec Design Goals:

1. Encryption turned on by default
2. Encryption turned on by default
3. Encryption turned on by default

$\Rightarrow$ Usage must be transparent and intuitive

TinySec Design Philosophy

The lesson from 802.11:

- Build crypto-security in, and turn it on by default!

TinySec Design Goals:

1. Encryption turned on by default
2. Encryption turned on by default
3. Encryption turned on by default

$\Rightarrow$ Usage must be transparent and intuitive

$\Rightarrow$ Performance must be reasonable

TinySec Design Philosophy

The lesson from 802.11:

- Build crypto-security in, and turn it on by default!

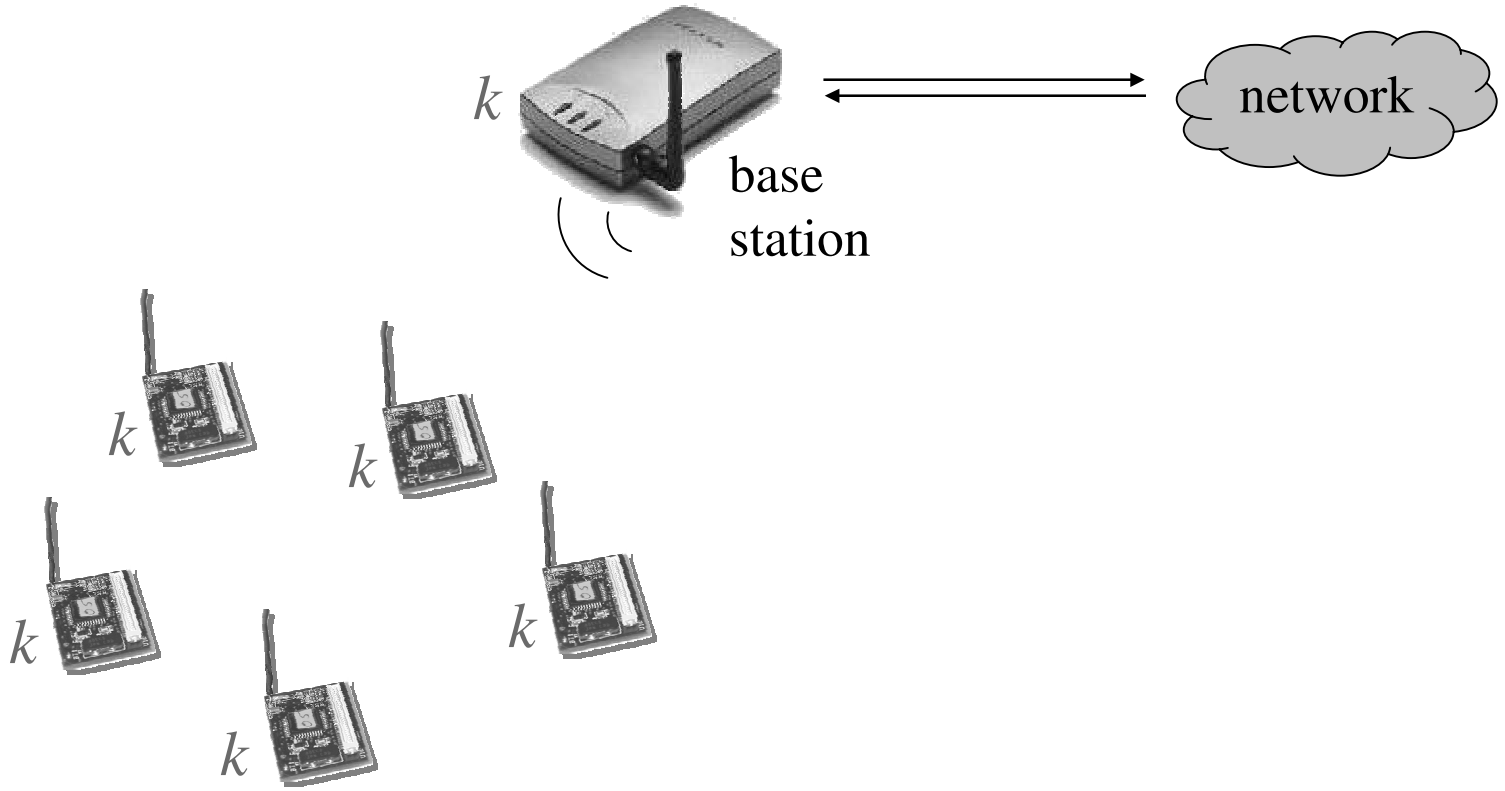
TinySec Design Goals:

1. Encryption turned on by default
2. Encryption turned on by default
3. Encryption turned on by default
   - $\Rightarrow$ Usage must be transparent and intuitive
   - $\Rightarrow$ Performance must be reasonable
4. As much security as we can get, within these constraints

Challenges

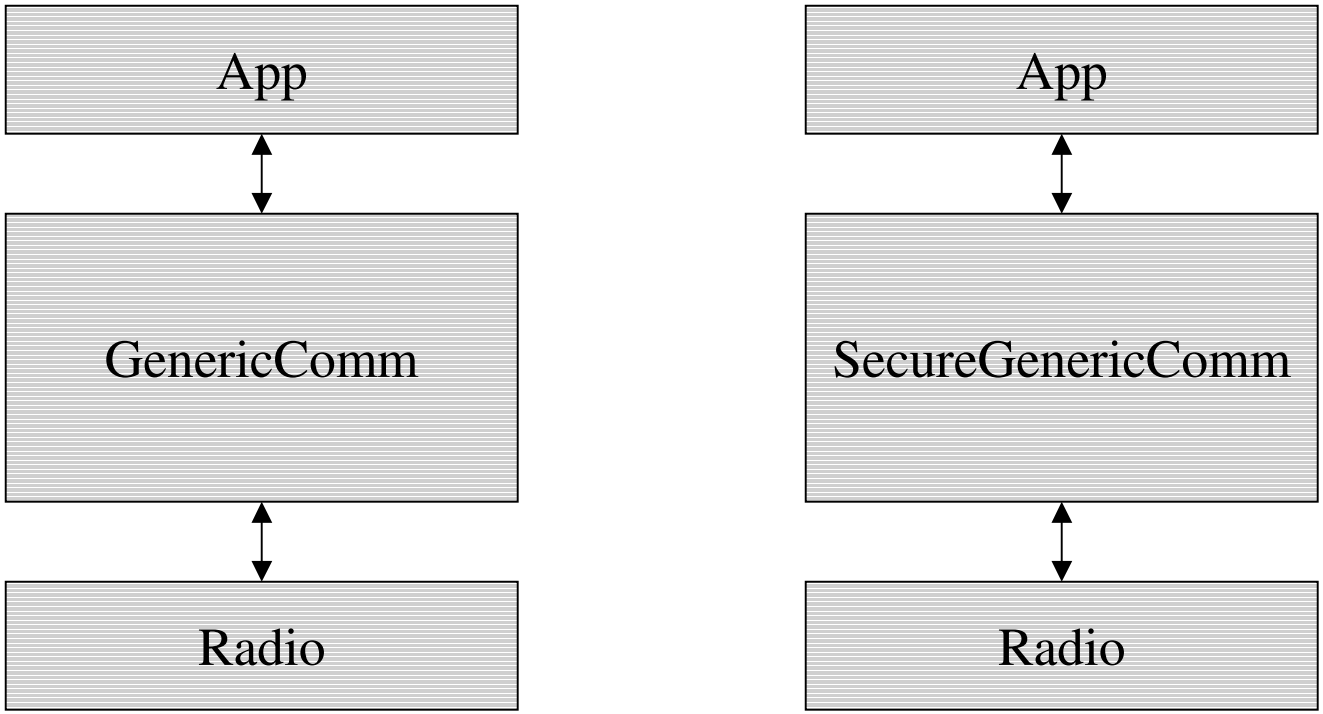
- Must avoid complex key management
  - TinySec must be super-easy to deploy
- Crypto must run on wimpy devices
  - We're not talking 2GHz P4's here!
  - Dinky CPU (1-4 MHz), little RAM (≤ 256 bytes), lousy battery
  - Public-key cryptography is right out
- Need to minimize packet overhead
  - Radio is very power-intensive:
    1 bit transmitted ≈ 1000 CPU ops
  - TinyOS packets are ≤ 28 bytes long
  - Can't afford to throw around an 128-bit IV here, a 128-bit MAC there

Easy Key Management



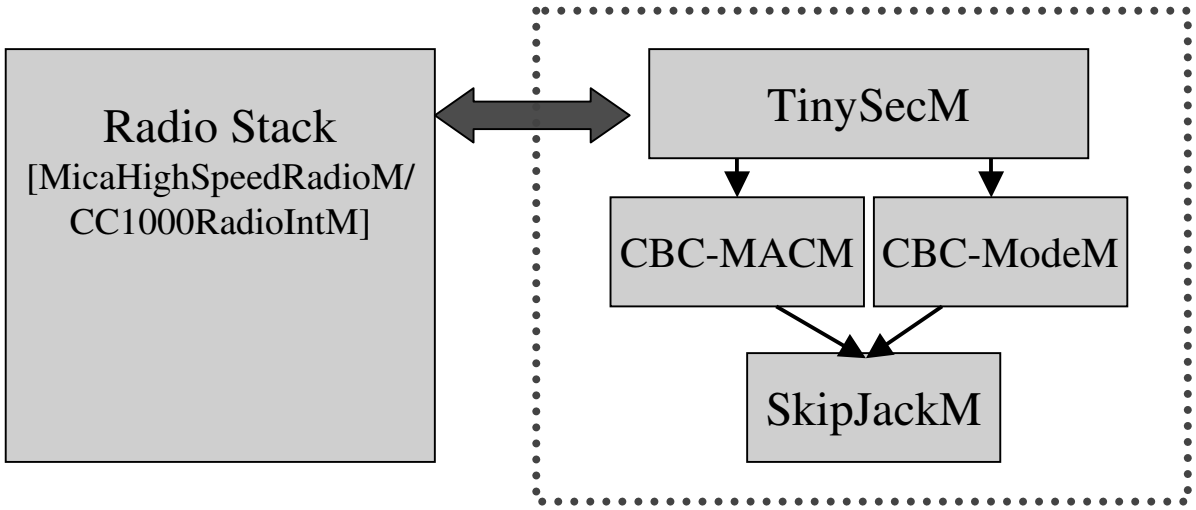
Making key management easy: global shared keys

Be Easy to Deploy



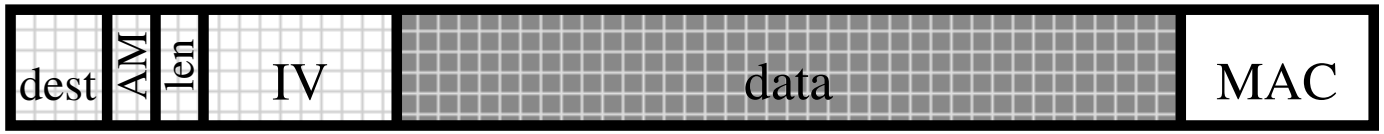
Making deployment easy:
plug-n-play crypto + link-layer security

Perform Well on Tiny Devices



- Use a block cipher for both encryption & authentication
- Skipjack is good for 8-bit devices; low RAM overhead

Minimize Packet Overhead



2 1 1 4 4

Encrypted

MAC'ed

Key Differences	
No CRC	-2 bytes
No group ID	-1 bytes
MAC	+4 bytes
IV	+4 bytes
Total:	+5 bytes

Minimize overhead: cannibalize, cheat, steal

Tricks for Low Overhead

- CBC mode encryption, with encrypted IV
  - Allows flexible IV formatting:
    4 byte counter, + cleartext hdr fields (dest, AM type, length);
    gets the most bang for your birthday buck
  - IV robustness: Even if IV repeats, plaintext variability may provide an extra layer of defense
  - Ciphertext stealing avoids overhead on variable-length packets
- CBC-MAC, modified for variable-length packets
  - Small 4-byte MAC trades off security for performance; the good news is that low-bandwidth radio limits chosen-ciphertext attacks
  - Can replace the application CRC checksum; saves overhead
- On-the-fly crypto: overlap computation with I/O

More Tricks & Features

- Early rejection for packets destined elsewhere
  - Stop listening & decrypting once we see dst addr ≠ us
- Support for mixed-mode networks
  - Interoperable packet format with unencrypted packets, so network can carry both encrypted + unencrypted traffic
  - Crypto only where needed ⇒ better performance
  - Length field hack: steal 2 bits to distinguish between modes
- Support fine-grained mixed-mode usage of TinySec
  - Add 3 settings: no crypto, integrity only, integrity+secrecy
  - These come with performance tradeoffs
  - Select between settings on per-application or per-packet basis

More Performance Tricks

- App-level API for end-to-end encryption
  - TinySec focuses mainly on link-layer crypto, but end-to-end crypto also has value
  - End-to-end secrecy enables performance optimizations (don't decrypt & re-encrypt at every hop), enables more sophisticated per-node keying, but incompatible with in-network transformation and aggregation; thus, not always appropriate
  - End-to-end integrity less clear-cut, due to DoS attacks

TinySec: Current Status

- Design + implementation stable
- Released in TinyOS 1.1
  - Integration with RFM & Chipcon radio stacks; supports nesC 1.1
  - Simple key management; should be transparent
- Several external users
  - Including: SRI, BBN, Bosch

TinySec Evaluation

Wins:

- Performance is ok
- Integration seems truly easy

Neutral:

- Out of scope: per-node keying, re-keying, sophisticated key mgmt; PKI; secure link-layer ACKs
- No security against insider attacks;
  What if a node is captured, stolen, or compromised?

Losses:

- Not turned on by default in TinyOS yet ☹

II. Communications Security: What Crypto Can't Do

"If it's provably secure, it's probably not."
-- Lars Knudsen

Limitations of Crypto

- Can't prevent traffic analysis
- Can't prevent re-transmitted packets
- Can't prevent replayed packets
- Can't prevent delayed packets
- Can't prevent packets from being jammed
- Can't prevent malicious insiders, captured nodes

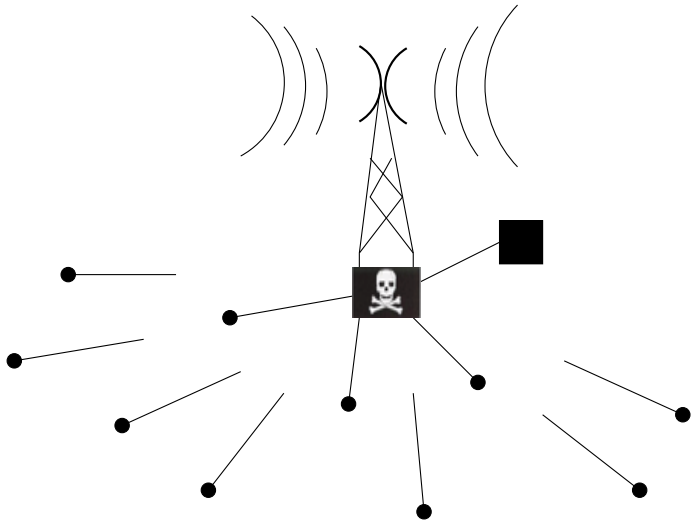
- Crypto is not magic fairy dust;
  It won't magically make insecure services secure.

Isn't Crypto All We Need?

Crypto doesn't automatically make X secure, where:

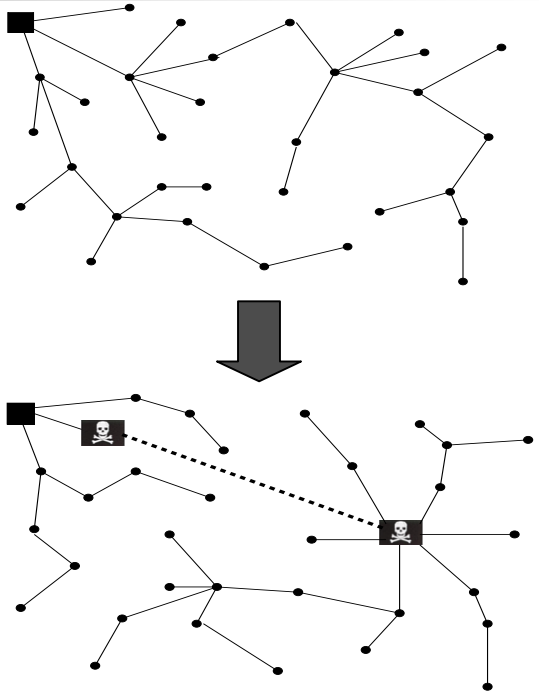
- X = network programming
  - Attacker could replay old programs
- X = time synchronization
  - Attacker could delay beacon packets, propagating wrong timing
- X = routing
  - Some attacks on next slide
- X = localization
  - Attack in three slides
- X = aggregation
  - Attacks after a few more slides

Example: Attacks on Routing



Hello flood attack:

- Broadcast really loudly; then everyone will think you are near them.



Wormhole attack:

- Tunnel packets from one part of the network and replay them in a different part.

Protocols analyzed in [KW03]

Protocol	Relevant attacks
TinyOS beaconing	Bogus routing information, selective forwarding, sinkholes, Sybil, wormholes, HELLO floods
Directed diffusion and multipath variant	Bogus routing information, selective forwarding, sinkholes, Sybil, wormholes, HELLO floods
Geographic routing (GPSR,GEAR)	Bogus routing information, selective forwarding, Sybil
Minimum cost forwarding	Bogus routing information, selective forwarding, sinkholes, wormholes, HELLO floods
Clustering based protocols (LEACH,TEEN,PEGASIS)	Selective forwarding, HELLO floods
Rumor routing	Bogus routing information, selective forwarding, sinkholes, Sybil, wormholes
Energy conserving topology maintenance	Bogus routing information, Sybil, HELLO floods

All are insecure

III. Interacting With The Environment: Location Verification

"Where ever you go, there you are."

Lightning

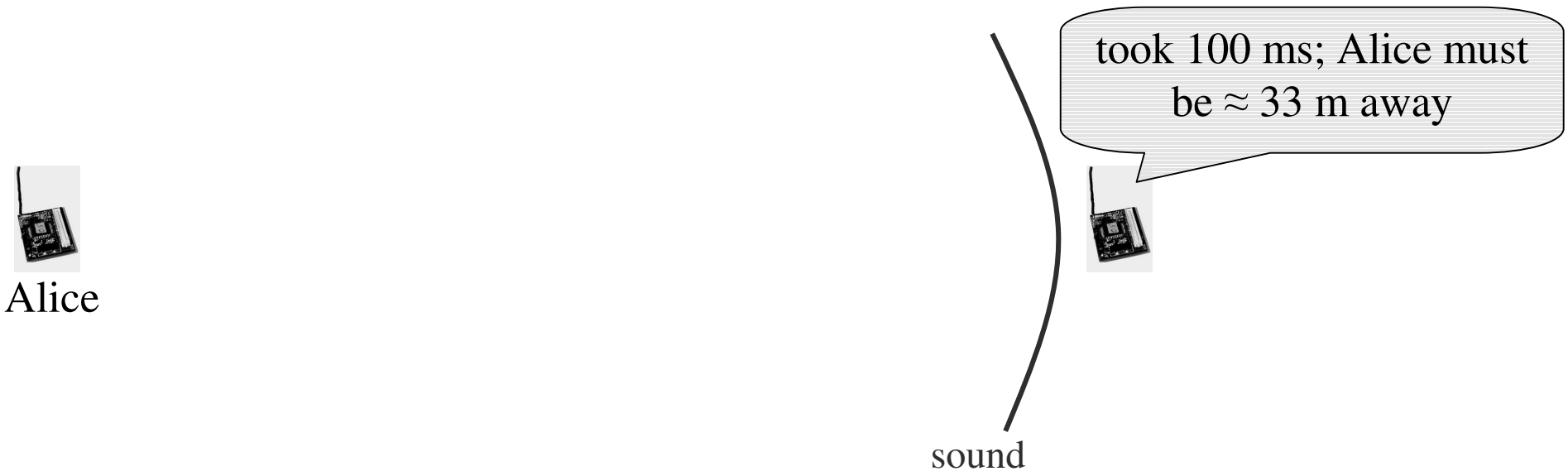
- How far away was the lightning?

Lightning

- How far away was the lightning?
- Count, then divide by 5…

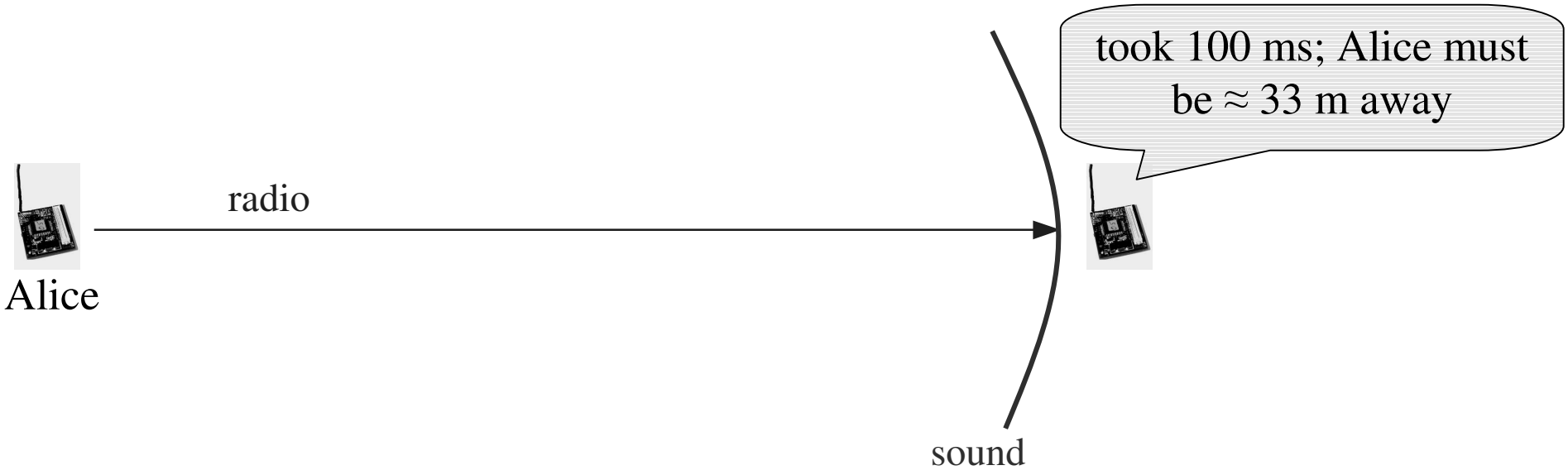
Location Determination

- How far away is Alice?
  - Have her transmit & chirp; measure elapsed time



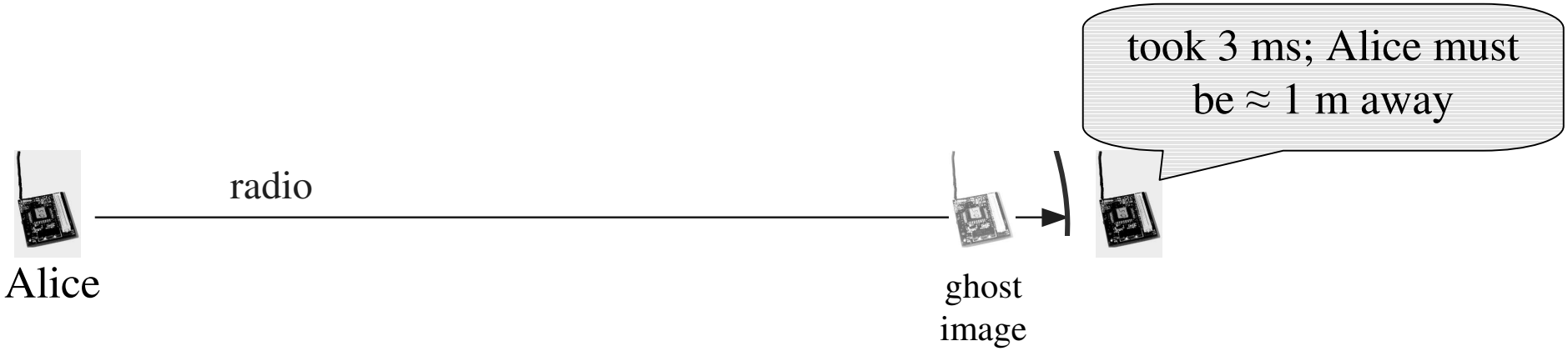
Location Determination

- How far away is Alice?
  - Have her transmit & chirp; measure elapsed time



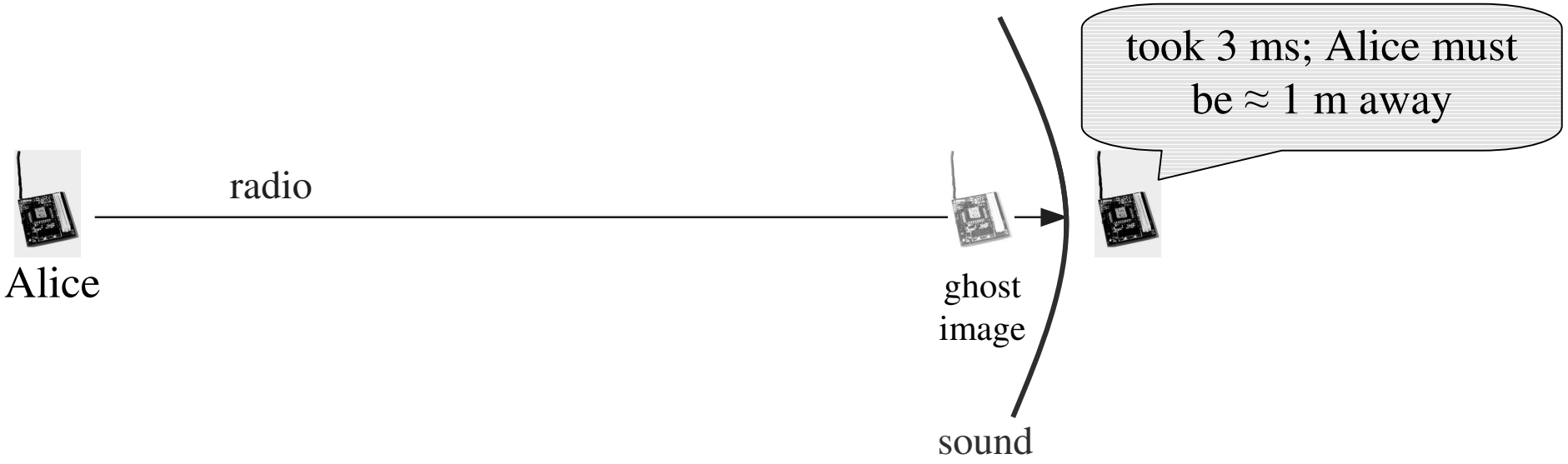
The Ventriloquist Attack

- Alice is malicious; she wants to seem nearby
  - Attack: Chirp in advance, wait a little, then transmit



The Ventriloquist Attack

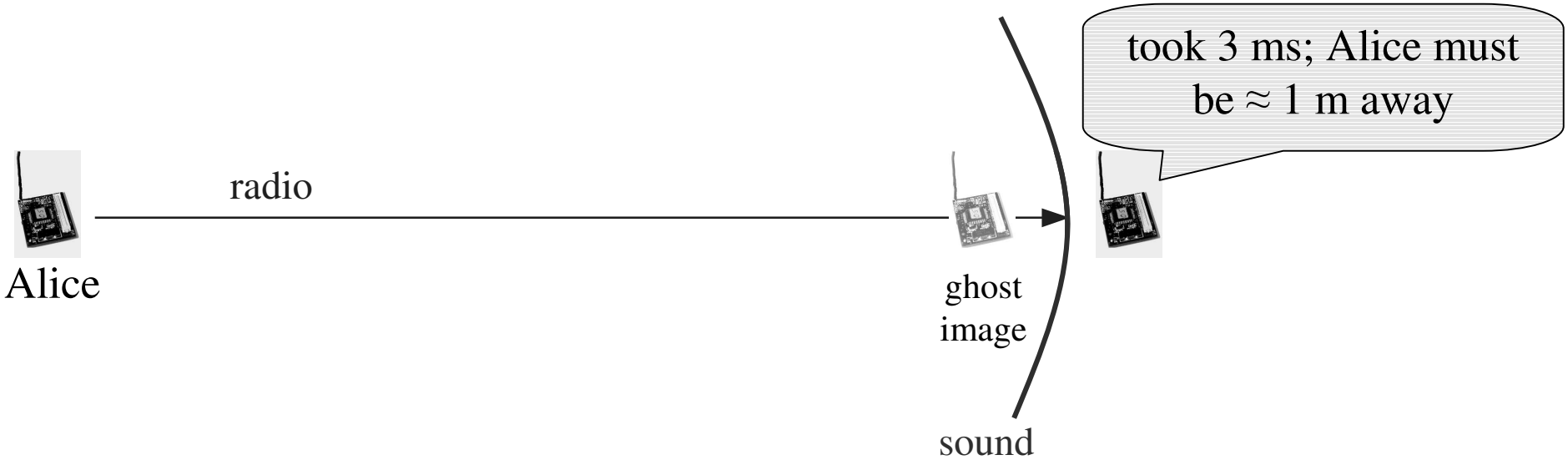
- Alice is malicious; she wants to seem nearby
  - Attack: Chirp in advance, wait a little, then transmit



The Ventriloquist Attack

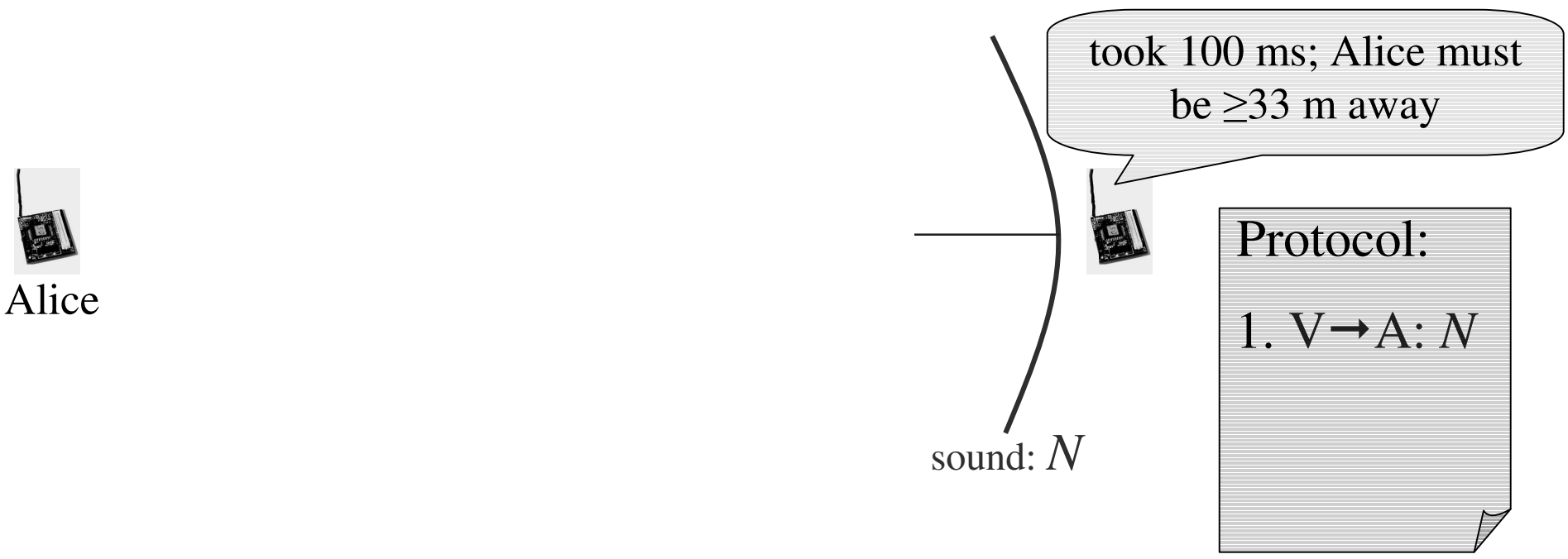
- Alice is malicious; she wants to seem nearby
  - Attack: Chirp in advance, wait a little, then transmit

Effect: Alice is able to lie about her location.



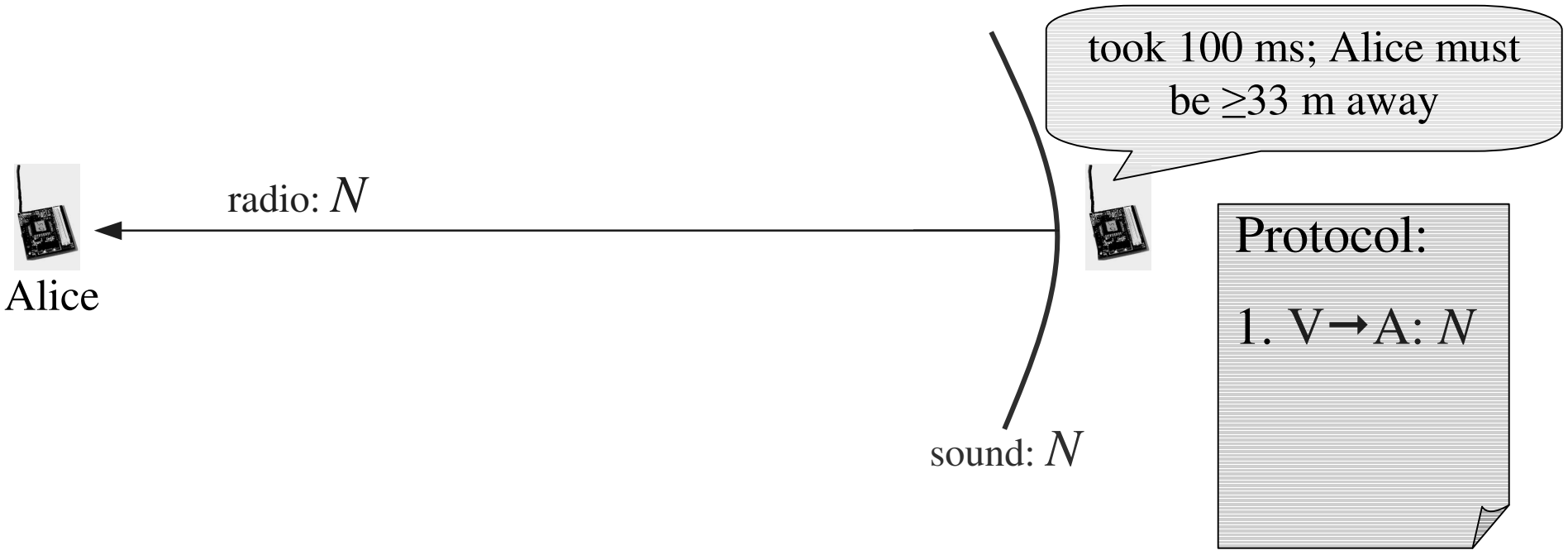
The Echo Protocol

- Secure location verification
  - Add a challenge-response, and Alice can't chirp early



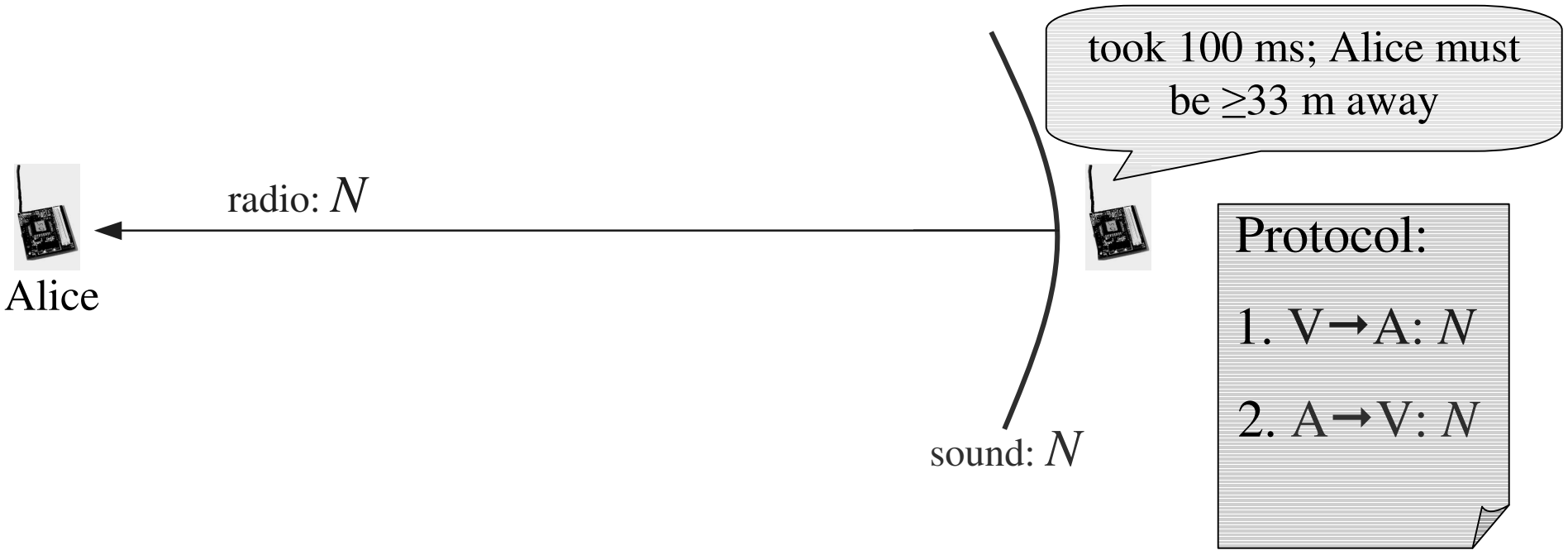
The Echo Protocol

- Secure location verification
  - Add a challenge-response, and Alice can't chirp early



The Echo Protocol

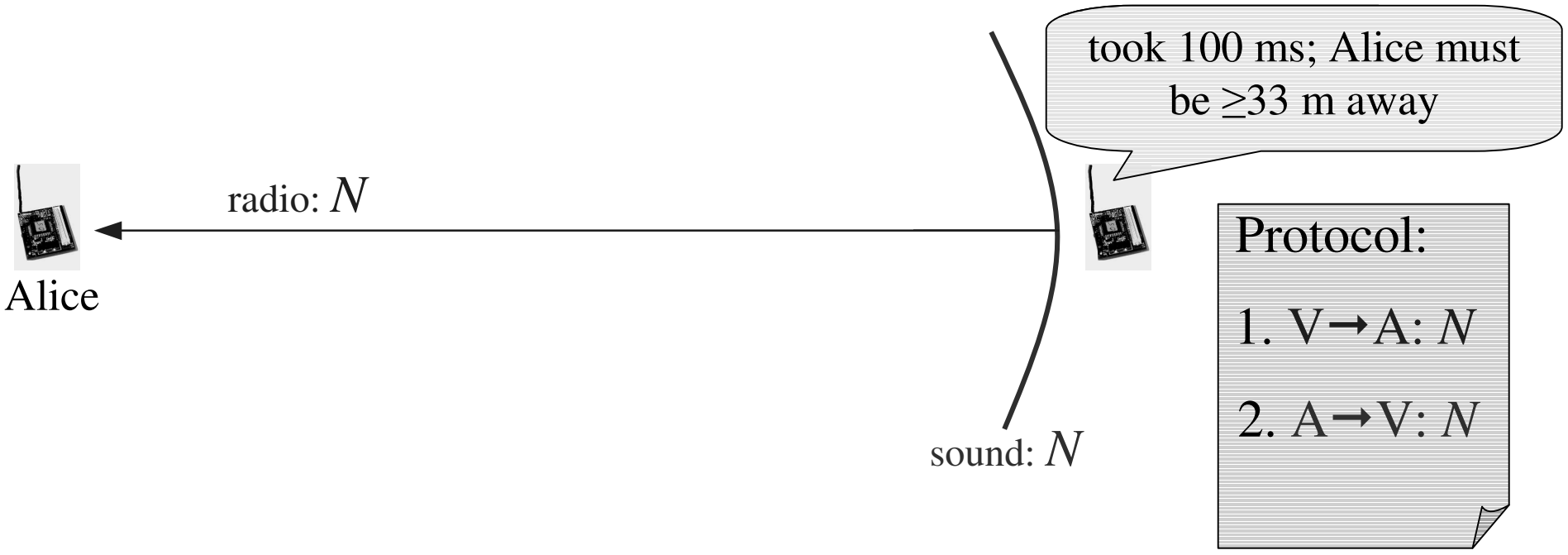
- Secure location verification
  - Add a challenge-response, and Alice can't chirp early



The Echo Protocol

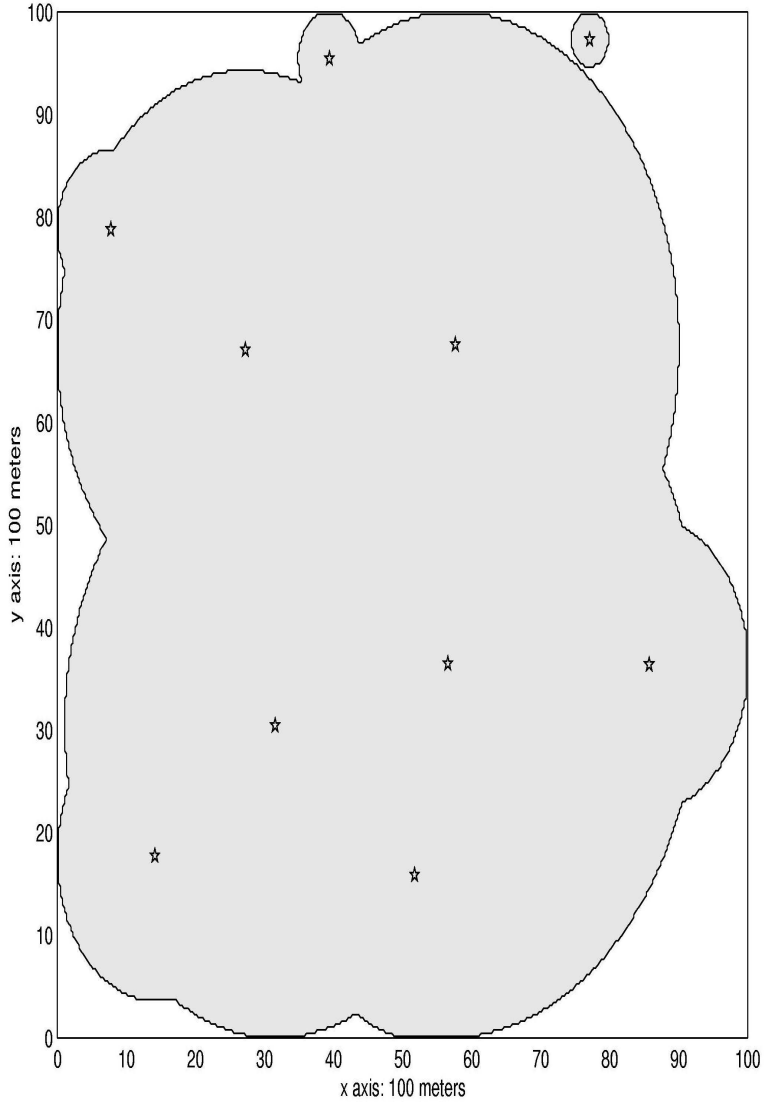
- Secure location verification
  - Add a challenge-response, and Alice can't chirp early

Result: Alice can no longer lie about her location.



Secure Location Services

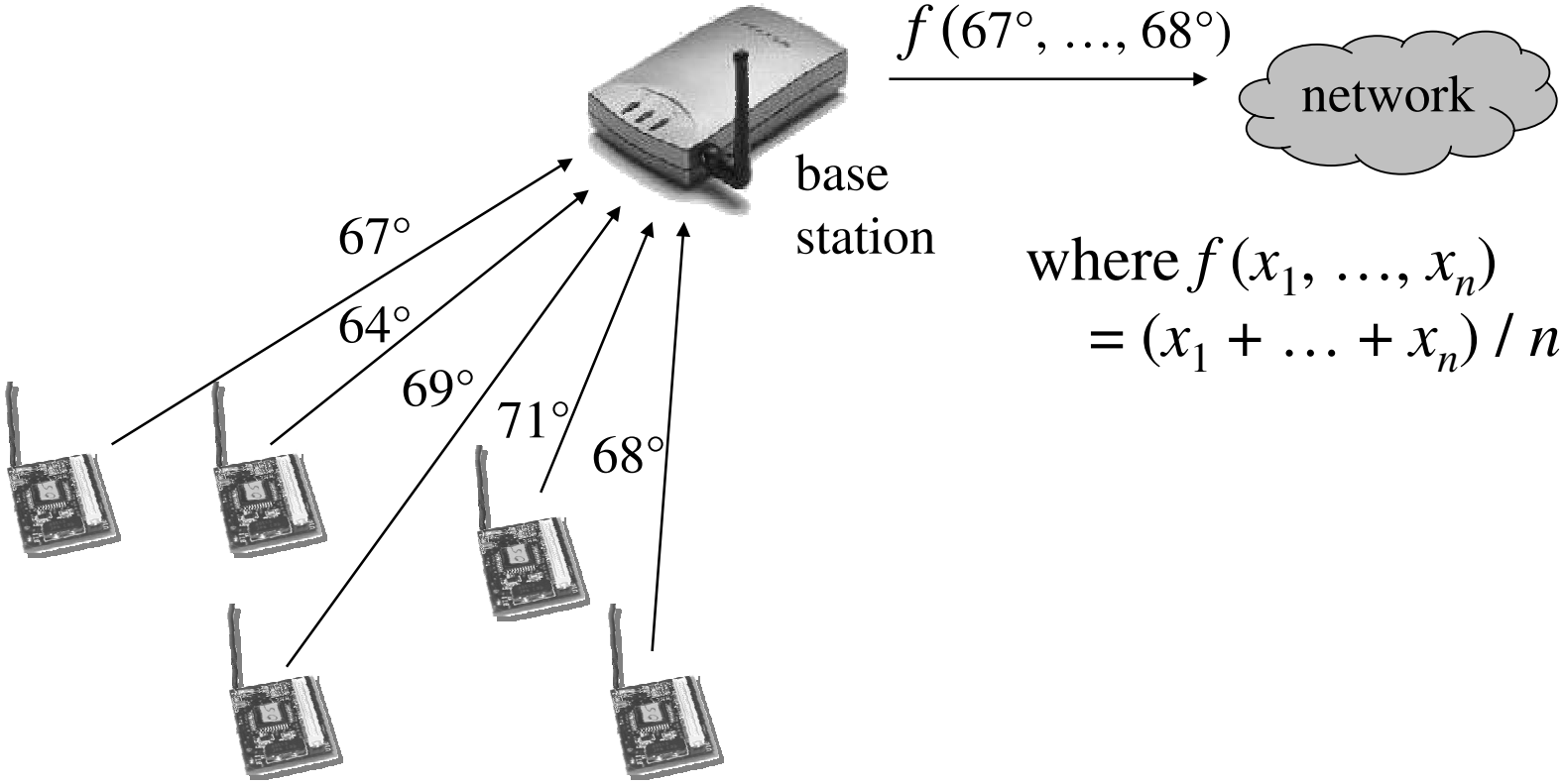
- For more details: see the Echo protocol [SSW03], a secure protocol for location verification
- Applications: location-based access control



IV. Tolerating Malicious Data: Resilient Aggregation

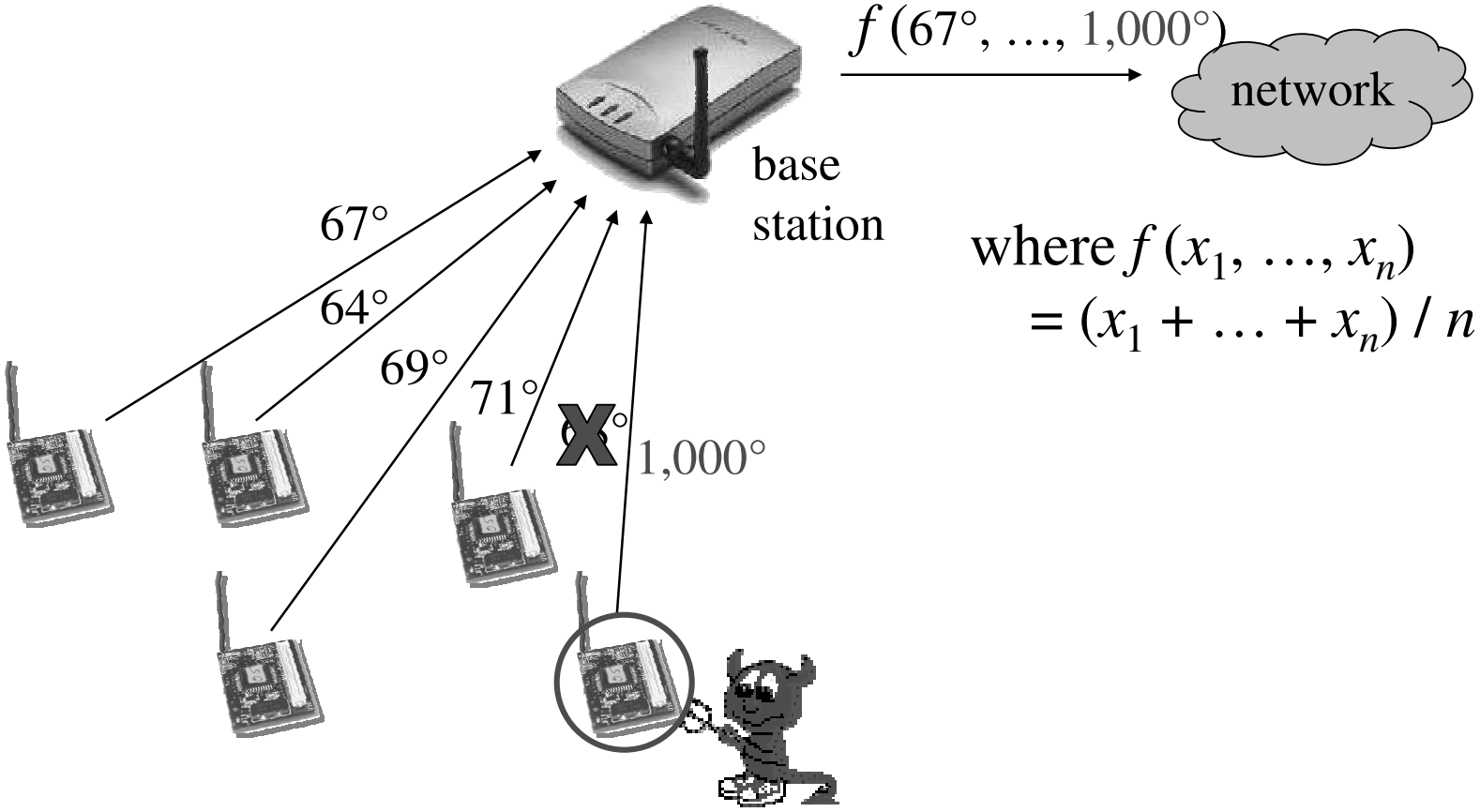
"If you believe that, I have a bridge to sell."

An Example



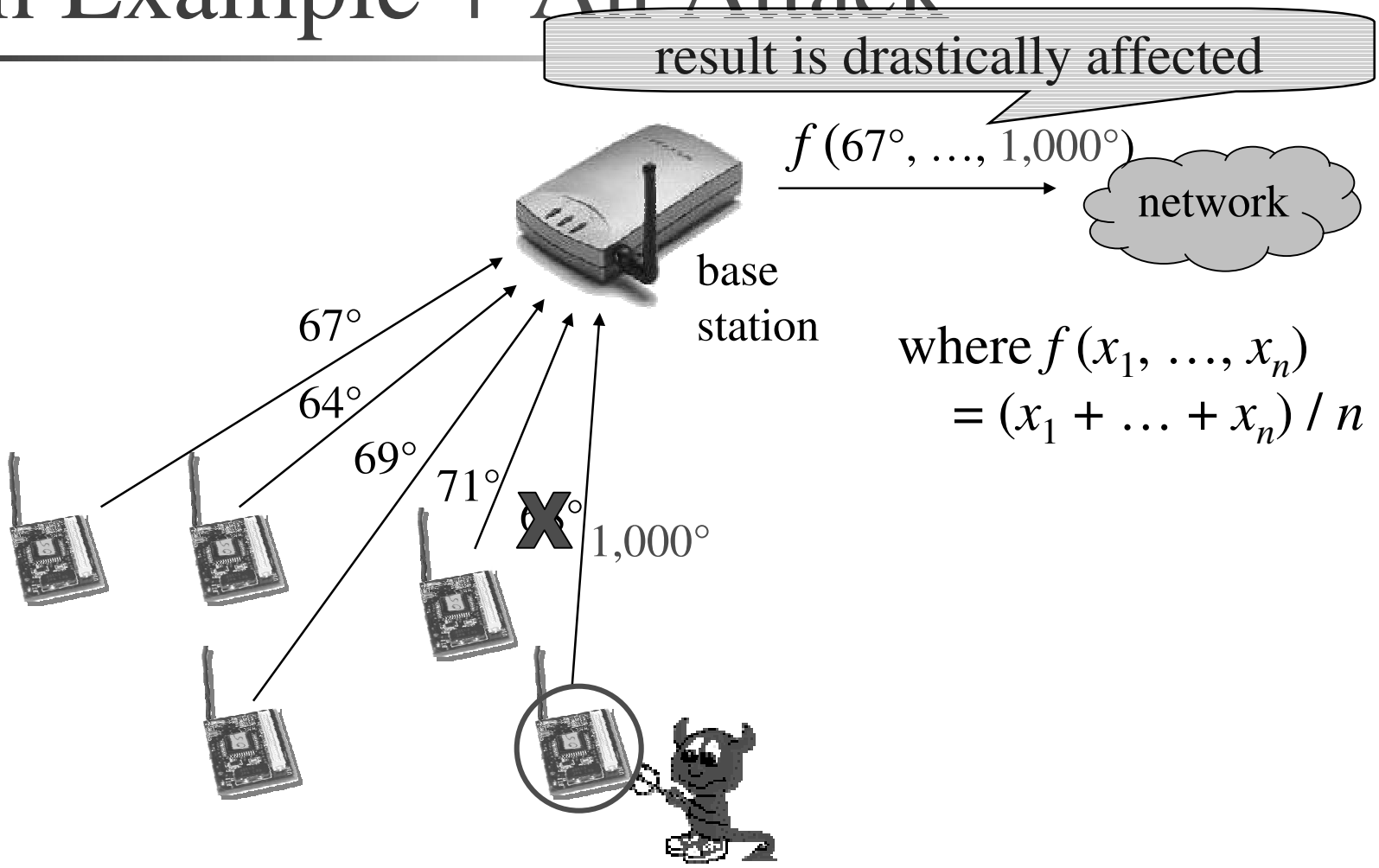
Computing the average temperature

An Example + An Attack



Computing the average temperature

An Example + An Attack



where $f(x_1, \ldots, x_n) = (x_1 + \ldots + x_n) / n$

Computing the average temperature

Statistical Theory

- First, some background:
  - Let $D(\Theta)$ be a parametrized distribution on $\Re$ (Θ = param), $X = (X_1, \ldots, X_n)$ denotes n samples from $D(\Theta)$
  - $f : \Re^n \rightarrow \Re$ is an *estimator* if $\Theta' = f(X)$ is an estimate of Θ
  - The *root mean square error* of an estimator f is
    $\text{rms}(0) = \text{E}[(\Theta' - \Theta)^2]^{1/2}$
- Next, a novel defense: resilient aggregation
  - A k-node attacker A is a function $A: \Re^n \rightarrow \Re^n$ that changes only k of its inputs. Let $\Theta^* = f(A(X))$, $\text{rms}(k) = \max_A \text{E}[(\Theta^* - \Theta)^2]^{1/2}$
  - Definition: f is (k, α)-resilient if $\text{rms}(k) \leq \alpha \times \text{rms}(0)$
  - E.g.: the "average" is an estimator, but it is not $(1, \alpha)$-resilient for any constant α

Relevance of Resilience

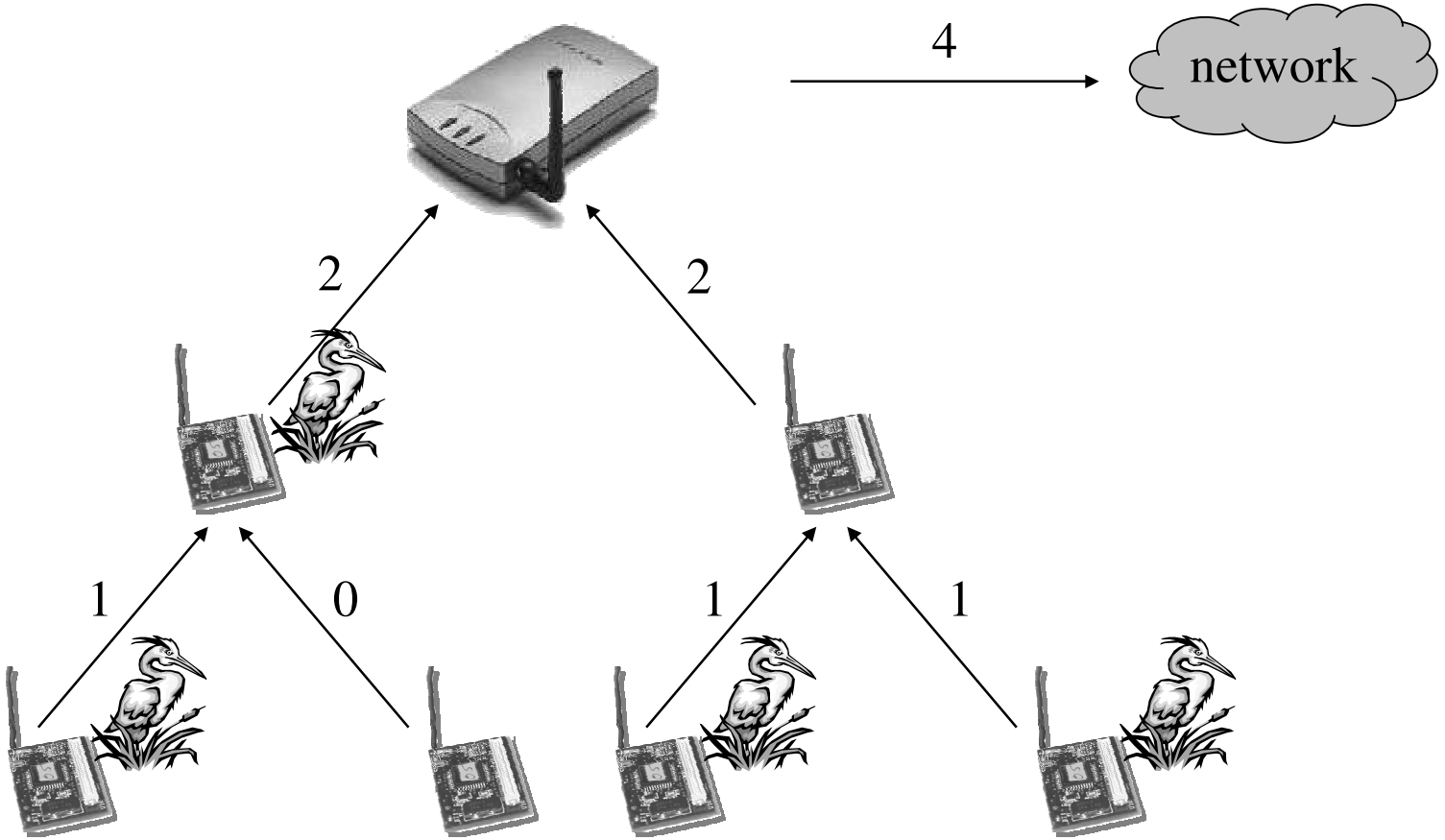
- Intuition
  - The (k, α)-resilient functions are exactly the ones that can be meaningfully and securely computed in the presence of k malicious insiders.
- Formalism
  - (see paper)

Results (excerpts)

f	… is (k, α)-resilient, where
minimum	$\alpha = \infty$
maximum	$\alpha = \infty$
sum	$\alpha = \infty$
average	$\alpha = \infty$
average, discarding 5% outliers	$\alpha \approx 6.28\ k/n$ for $k < 0.05\ n$ $\alpha = \infty$ for $k > 0.05\ n$
median	$\alpha \approx 0.32\ k$ for $k < 0.5\ n$
max	$\alpha = \infty$
count	$\alpha \leq 0.25\ k/n$

(assuming n independent Gaussian/Bernoulli distributions)

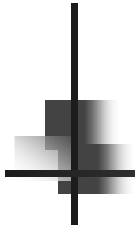
Hard: In-Network Resilient Agg.



In-network aggregation introduces new security challenges

Hard Problems

- Communication security
  - Defeating traffic analysis; spread spectrum for real?
- A library of secure distributed services & protocols
- Security against node compromise/capture
  - e.g., routing that can tolerate just one malicious insider?
  - Byzantine attack tolerance, on the cheap?
- Privacy



Summary

- Crypto helps, but isn't a total solution
  - Be aware of the systems tradeoffs
- Seek robustness against insider attack
  - Resilience gives a way to think about malicious/captured nodes
  - The law of large numbers is your friend
- Feedback?