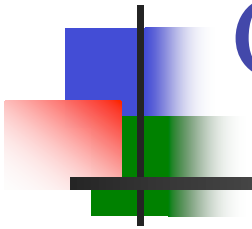


Security for Sensor Networks: Cryptography and Beyond

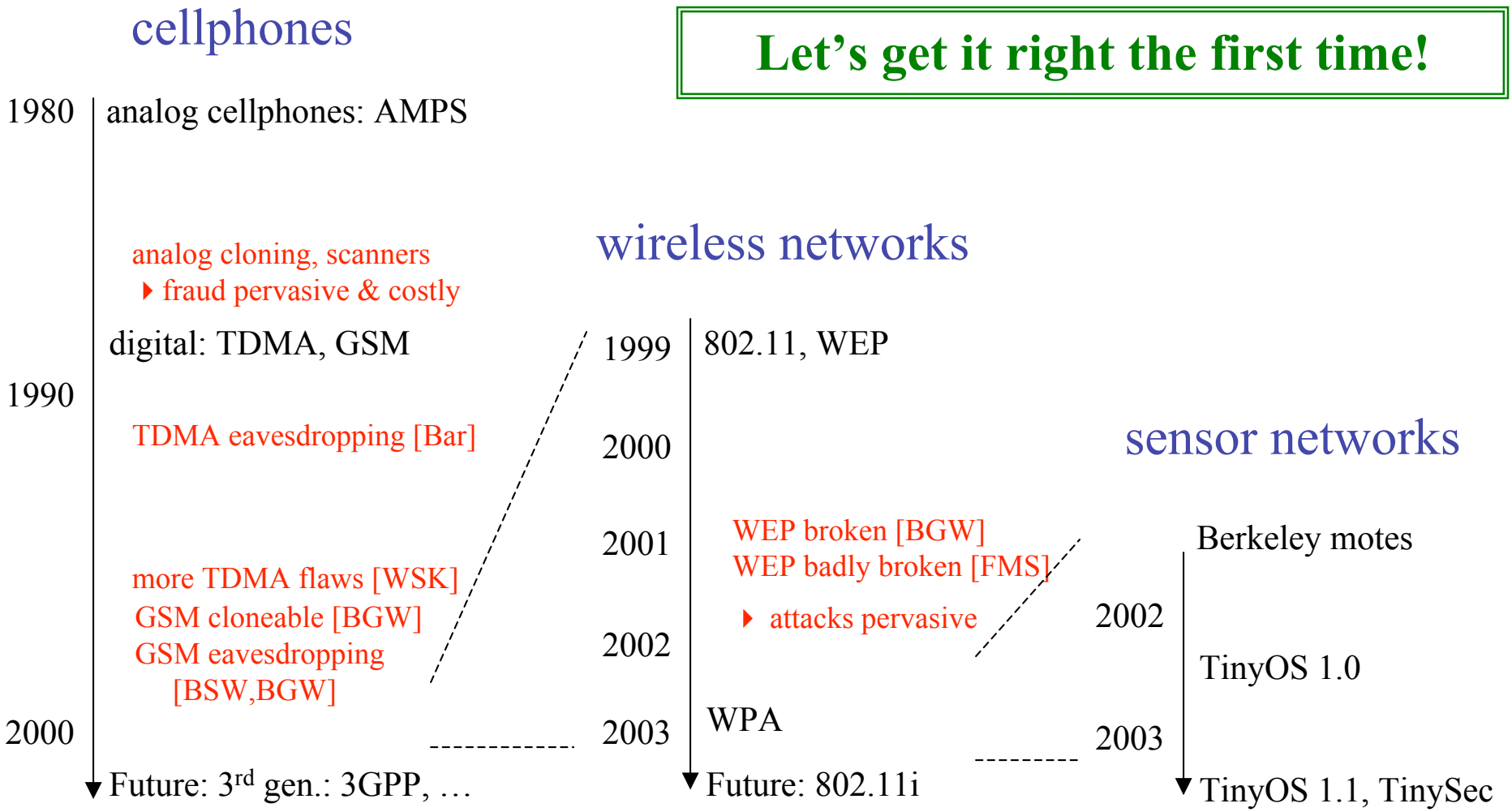
David Wagner
University of California at Berkeley

In collaboration with:

Chris Karlof, David Molnar,
Naveen Sastry, Umesh Shankar

Learn From History…

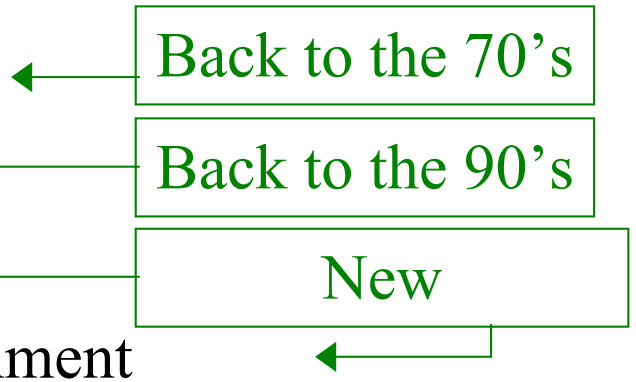
Let's get it right the first time!

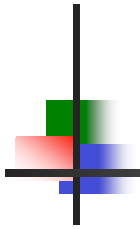


Sensor Nets: So What?

What's different about sensor nets?

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





Where Are We Going?

Some research challenges in sensor net security:

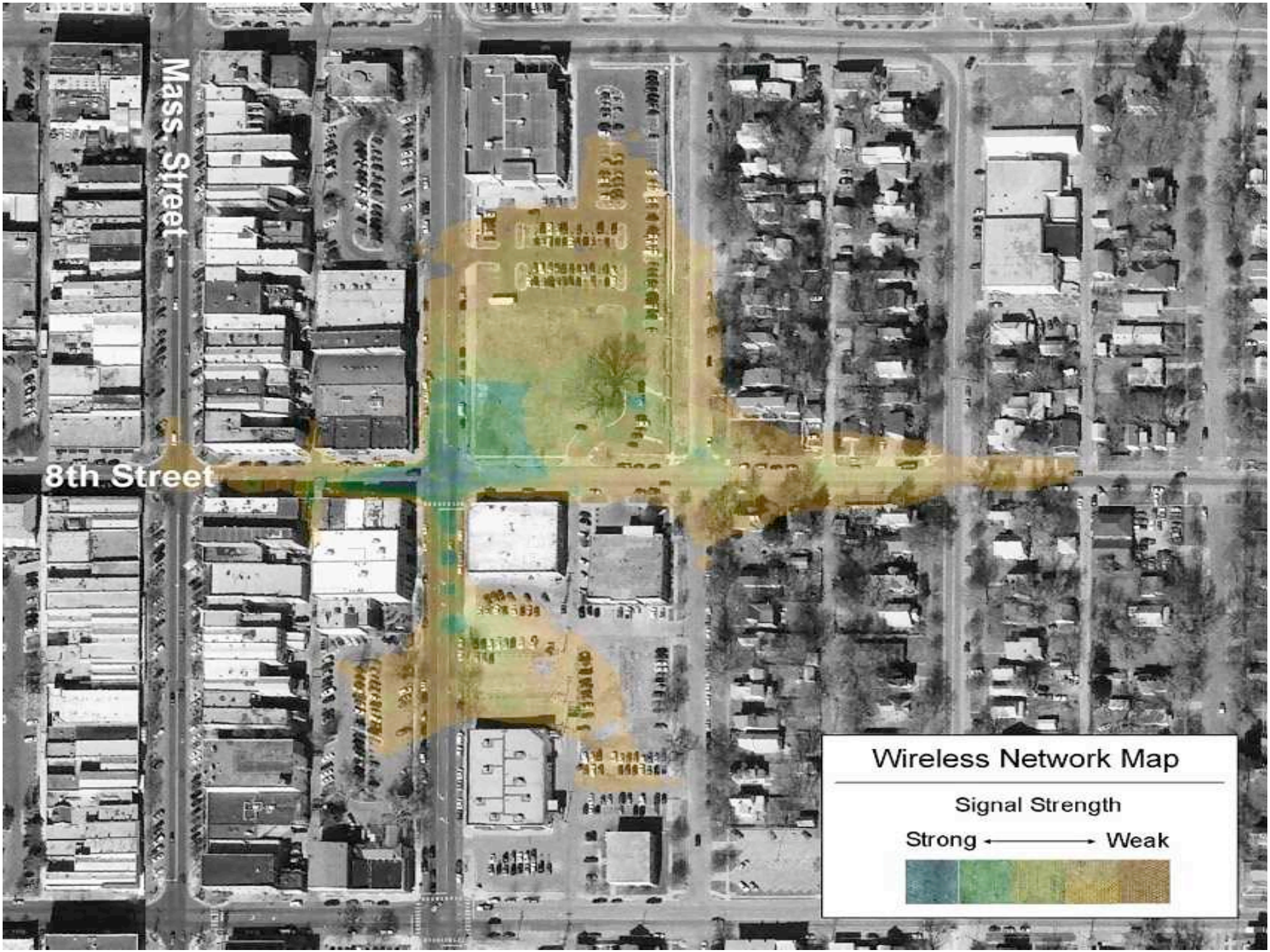
- Securing the communication link
- Securing distributed services
- Tolerating captured nodes

- **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

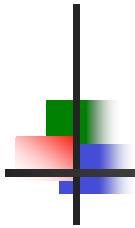

PRIMESTAR

Wardriving / Access Point Mapping

468 WEP

1,265 Clear

1,733 Total



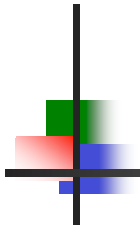
TinySec Design Philosophy

The lesson from 802.11:

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

TinySec Design Goals:

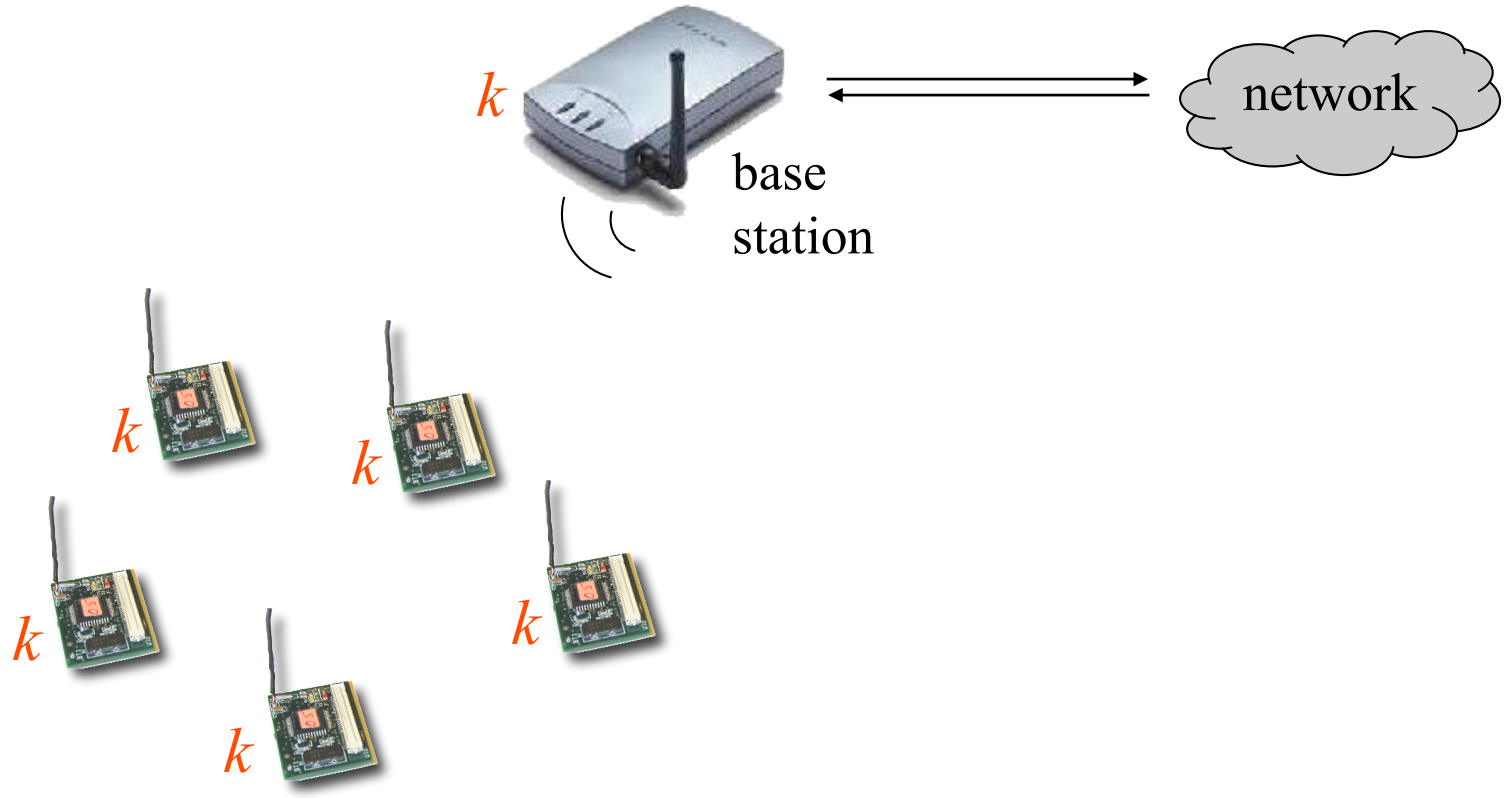
1. Encryption turned on by default
2. Encryption turned on by default
3. Encryption turned on by default
   - ⇒ Usage must be transparent and intuitive
   - ⇒ Performance must be reasonable
4. As much security as we can get, within these constraints



Challenges

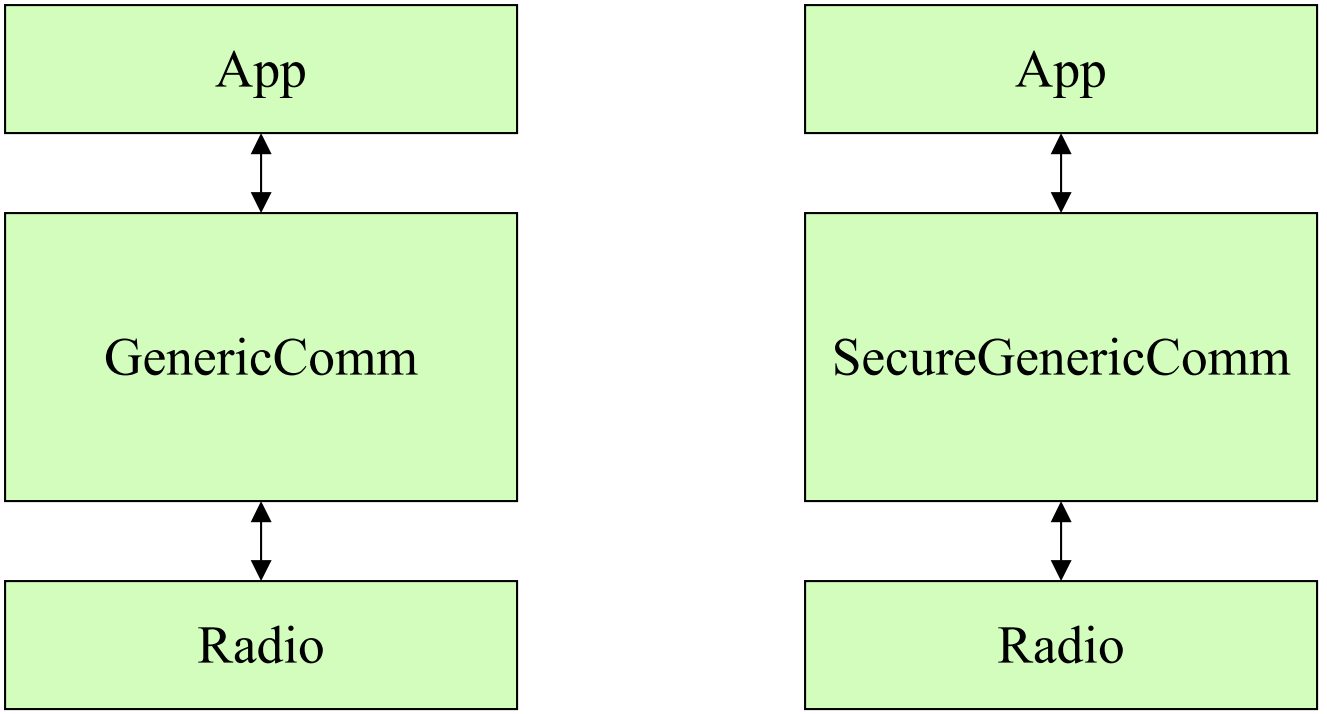
- Must avoid complex key management
  - TinySec must be super-easy to deploy
- Crypto must run on wimpy devices
  - We're not talking 2GHz P4's here!
  - Dinky CPU (1-4 MHz), little RAM (≤ 256 bytes), lousy battery
  - Public-key cryptography is right out
- Need to minimize packet overhead
  - Radio 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



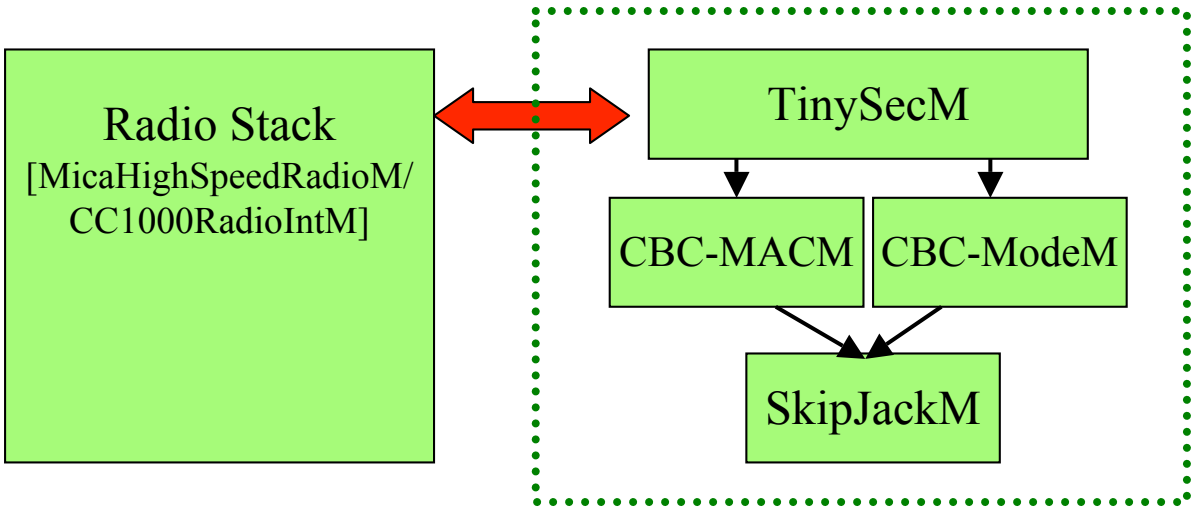
Making key management easy: global shared keys

Be Easy to Deploy



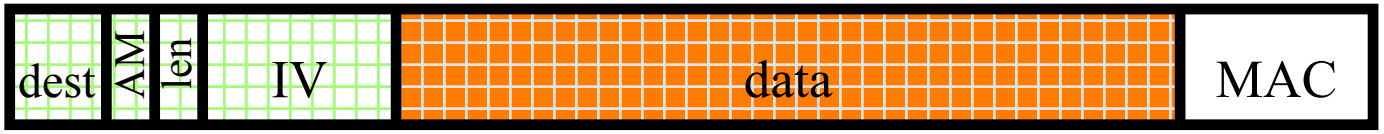
Making deployment easy:
plug-n-play crypto + link-layer security

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

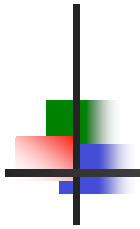
Encrypted

MAC'ed

Key Differences

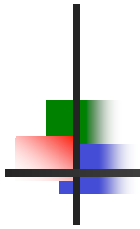
No CRC	-2 bytes
No group ID	-1 bytes
MAC	+4 bytes
IV	+4 bytes
Total:	+5 bytes

Minimize overhead: cannibalize, cheat, steal



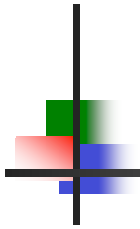
Tricks for Low Overhead

- 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



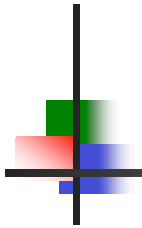
More Tricks & Features

- 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



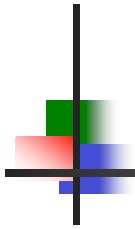
More Performance Tricks

- 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 ☹