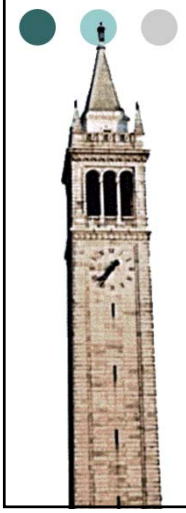


Introduction to Embedded Systems

Sanjit A. Seshia
UC Berkeley
EECS 149/249A
Fall 2015

Chapter 17: Security and Privacy

Security and Privacy: Definitions

No single definition in the literature, but broadly:

Security – the state of being protected from harm

Privacy – the state of being kept away from observation

Warning (adapted from CS 161)

This lecture may discuss vulnerabilities in embedded systems. This is *not* intended as an invitation to go exploit those vulnerabilities. It is important that we be able to discuss real-world experience candidly, and s*tudents are expected to behave responsibly*.

Berkeley policy is very clear: you may not break into machines that are not your own; you may not attempt to attack or subvert system security. Breaking into other people's systems is inappropriate, and the existence of a security hole is no excuse.

Unethical or inappropriate actions may result in failing the course and being referred for further disciplinary action.

About this Lecture

Security is increasingly a major concern for embedded systems designers

→ Automotive, avionics, medical devices, control systems, …

Need to know about the security pitfalls in design & implementation of embedded systems, and potential solutions

Take CS 161 to learn about computer security in general.

Properties and Threat Models

Secrecy/Confidentiality

Can secret data be leaked to an attacker?

Integrity

Can the system be modified by the attacker?

Authenticity

Who is the system communicating/interacting with?

Availability

Is the system always able to perform its function?

(Absence of “denial-of-service”)

Also important to think about Threat (attacker) Models

Example: Security Analysis of Pacemakers (ICDs) [Halperin et al., 2008]

Considered attacks on ICD security by three classes of attackers:

- Attacker possessing an ICD programmer
- Attacker who simply eavesdrops on communications between an ICD and the programmer, using commodity software radio
- Attacker who eavesdrops as well as generates arbitrary RF traffic to the ICD, possibly spoofing an ICD programmer.

Demonstrated that successful attacks are possible under all three classes!

Results of Experiments by Halperin et al.

	Commercial programmer	Software radio eavesdropper	Software radio programmer	Primary risk
Determine whether patient has an ICD	✔	✔	✔	Privacy
Determine what kind of ICD patient has	✔	✔	✔	Privacy
Determine ID (serial #) of ICD	✔	✔	✔	Privacy
Obtain private telemetry data from ICD	✔	✔	✔	Privacy
Obtain private information about patient history	✔	✔	✔	Privacy
Determine identity (name, etc.) of patient	✔	✔	✔	Privacy
Change device settings	✔		✔	Integrity
Change or disable therapies	✔		✔	Integrity
Deliver command shock	✔		✔	Integrity

TABLE I
RESULTS OF EXPERIMENTAL ATTACKS. A CHECK MARK INDICATES A SUCCESSFUL IN VITRO ATTACK.

Review of Cryptography (Sec. 17.1)

What's the difference between public-key cryptography and symmetric-key crypto?

What is a block cipher?

What is a one-way function?

Review of Cryptography (Sec. 17.1)

In the mathematical description of RSA, if the public key is the pair (n,e) and the message plaintext is M, then how do you compute the ciphertext C from these?

Name one similarity and one difference between digital signatures and message authentication codes.

Topics Covered in this Lecture

How to get keys in the first place

Software security

Information flow security

Other topics: sensor attacks, side channels, etc.

A *Key* Design Challenge: Key Exchange

Diffie-Hellman: Classic protocol (read the book)

Timed release of keys

Custom solutions for specific problem domains

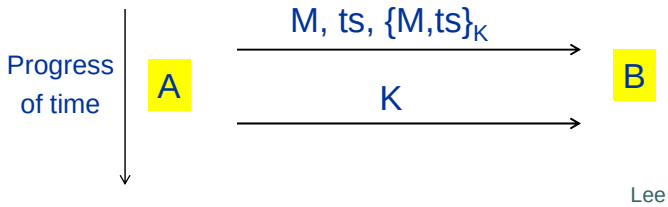
Timed Release of Keys (TESLA) [Perrig et al., ~2004]

Idea: Leverage timing properties of broadcast networks

- Time synchronization
- Scheduled transmission

Secure authentication can be achieved by

- Using a "chain" of keys
  - Each key generated from previous using a one-way function
  - Discard a key after use
- Releasing each key some time after its use



Version of TESLA for Automotive CAN bus

[Chung-Wei Lin Ph.D. Thesis, 2015, Chapter 5]

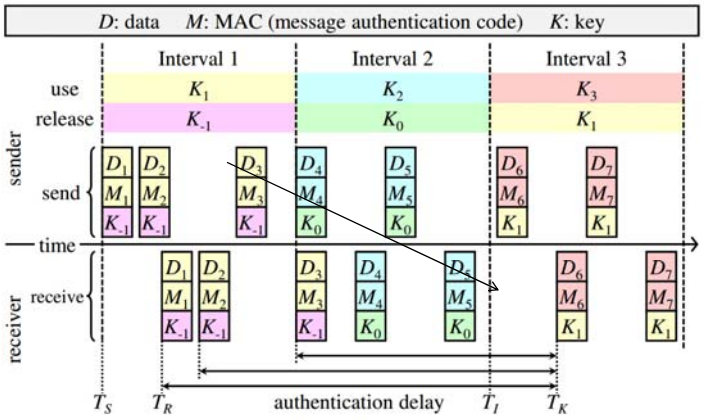


Figure 5.2: The time-delayed release of keys. T_S, T_R, and T_K are the sending time, the receiving time, and the key-receiving time of the packet (D_1, M_1, K_{-1}), respectively. T_I is the starting time of Interval 3.

Message authentic if $T_I > T_R$

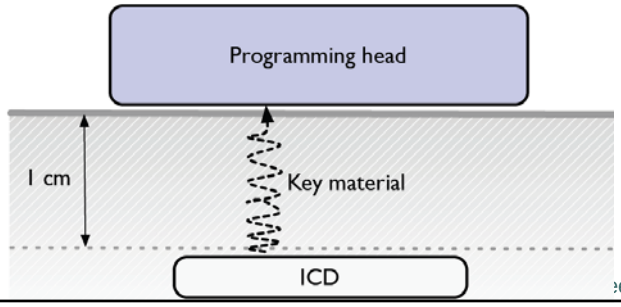
[Mechanism bootstrapped by using a "regular" signature scheme at the very beginning]

Custom Solution: Key Exchange for IMD

[Halperin et al., 2008]

Idea: Key exchange is possible only by close proximity

1. Programmer is placed very close to the skin of the patient (indicates patient awareness and consent?) and transmits an RF signal to IMD
2. IMD then responds with a random value to be used as K_m, transmitted as a modulated sound wave that cannot be sensed at appreciable distance (> 1 cm ?)



Software Security

Software design and implementation, if done wrong, can lead to malicious exploits (e.g. HeartBleed)

This is true of any system involving software, including embedded systems.

Effects may be worse for embedded systems, since you may not have an OS/hypervisor to mitigate the adverse effects!

See CS 161 material for a more in-depth coverage

Buffer Overruns

```
1  int sensor_flags[4];
2
3  void process_sensor_data() {
4    int i = 0;
5    char sensor_data[16];
6
7    // more_data returns 1 if there is more data,
8    // and 0 otherwise
9    while(more_data()) {
10     sensor_data[i] = get_next_byte();
11     i++;
12   }
13
14   // some code here that sets sensor_flags
15   // based on the values in sensor_data
16
17   return;
18 }
```

Information Flow Security

Many security properties concern the FLOW of information between different components/agents in a system.

Information Flow Security is the study of how such flows affect the security and privacy properties of a system.

Example 1: Illegal Information Flow?

```
1  int patient_id; // initialized to the
2                  // patient's unique identifier
3  void take_reading() {
4    float reading = read_from_sensor();
5
6    display(reading);
7
8    send(network_socket, hospital_server,
9         reading, patient_id);
10
11   return;
12 }
```

Example 2: Illegal Information Flow?

```
1  int patient_id; // initialized to the
2                  // patient's unique identifier
3  long cipher_text;
4
5  struct secret_key_s {
6    long key_part1; long key_part2;
7  }; // struct type storing 128-bit AES key
8
9  struct secret_key_s secret_key; // shared key
10
11 void take_reading() {
12   float reading = read_from_sensor();
13
14   display(reading);
15
16   enc_AES(&secret_key, reading, &cipher_text);
17
18   send_enc(network_socket, hospital_server,
19            cipher_text, patient_id);
20
21   return;
22 }
```


Example 3: Illegal Information Flow?

```
1  int patient_id; // initialized to the
2                  // patient's unique identifier
3  int patient_pwd; // stored patient password
4
5  float stored_readings[100];
6
7  void show_readings() {
8    int input_pwd = read_input(); // prompt user for
9                                  // password and read it
10   if (input_pwd == patient_pwd) // check password
11     display(&stored_readings);
12   else
13     display_error_mesg();
14
15   return;
16 }
```

Non-Interference

Any (security) property that specifies how actions taken by one or more principals can or cannot affect ("interfere with") actions taken by others.

Integrity: actions of an attacker cannot affect the values of certain trusted data or computations.

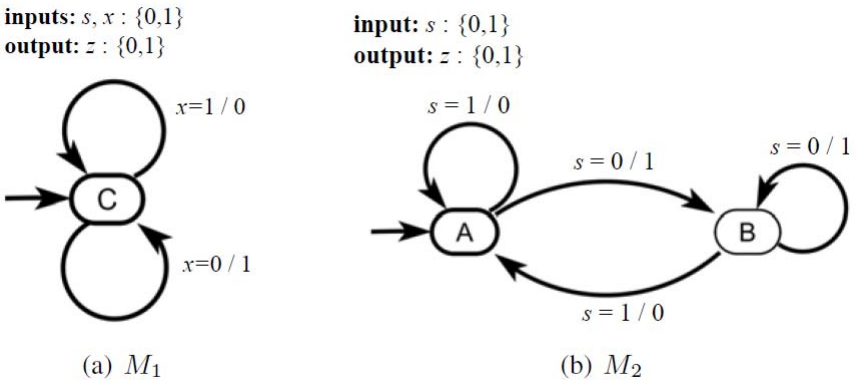
Confidentiality: actions taken by an attacker cannot depend on secret values (thus implying that the attacker has no information about those secrets).

Observational Determinism (OD)

States/inputs/outputs can be classified as HIGH (security) or LOW (security)

OD: If two traces of a system are initialized to be in states in which the low elements are identical, and they receive the same low inputs, then that implies that the low elements of all states and outputs in that trace must be identical.

Example



s – HIGH (secret)

x, z – LOW (public)

Misc. Topics: Sensor Attacks

Sensors can be manipulated to output wrong values, or at wrong times, or both

If the computation is not robust to such attacks, or fails to detect them and take corrective measures, bad things can happen!

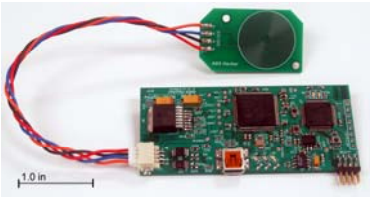
We'll see one representative example.

Warning: Do NOT try this "at home"!

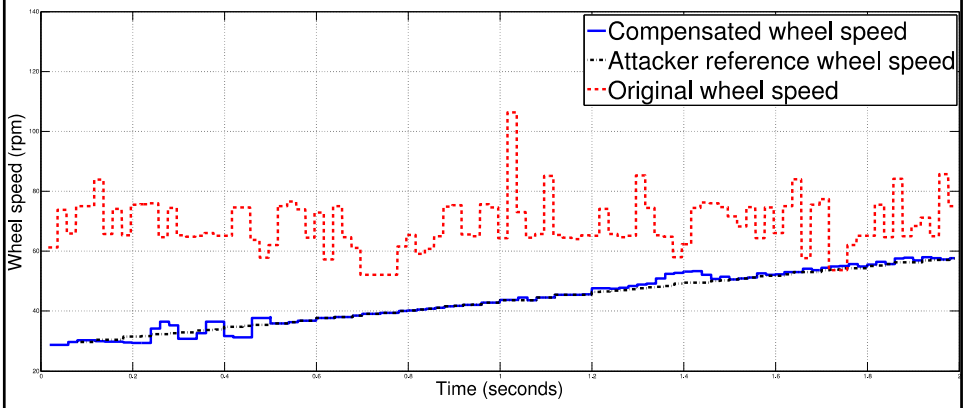
Noninvasive Spoofing of ABS Sensors

[Y. Shoukry et al.]

- ABS Sensors are accessible from outside the vehicle.
- An ABS sensor consists of two parts:
  - Permanent magnet (attached to the chassis)
  - Tone ring(attached to the wheel).
  - When the wheel rotates -> the tone ring rotates in front of the magnet -> magnetic field.
- Our "ABS Hacker":
  - Senses the magnetic field and cancels it.
  - Synthesize a new magnetic wave to spoof the sensor



Noninvasive Spoofing of ABS Sensors (Experiments using an ABS Testbed)



Red -> wheel speed in absence of the attack

Blue -> wheel speed in presence of attack

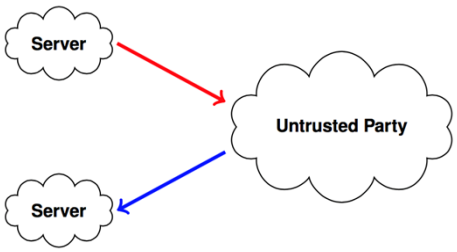
(Y Shoukry et al, "Noninvasive Spoofing Attacks for Anti-Lock Braking Systems," CHES 2013.)

Noninvasive Spoofing of ABS Sensors: Video (Simulations using Carsim)

Countermeasures

An authentication technique in cyber systems is done through "Challenge-Response Authentication"

- One party requires another party to prove their trustworthiness by issuing a *challenge* and checking the *response*.
- e.g. simple password query



- The same concept can be extended to "Physical Challenge Response Authentication"
  - Physics imposes **fundamental constraints** on how sensor signal can change.
  - Challenge comes in the form of a ***physical* stimulus** placed on the environment.
  - By checking the natural response of the environment, we can discriminate between the actual sensor measurements and synthetic attacks.

Y. Shoukry, et al. PyCRA: Physical Challenge-Response Authentication For Active Sensors Under Spoofing Attacks. CCS 2015

Summary of Topics (read Ch. 17 for those not covered in lecture)

Basics of Cryptography

Key Exchange

Software security

Information flow security

Other topics: sensor attacks, side channels, etc.