6.857 Computer and Network Security November 14, 2002

Lecture Notes 20 : Smartcards, side channel attacks

Lecturer: Ron Rivest *Scribe: Giffin/Greenstadt/Plitwack/Tibbetts*

[These notes come from Fall 2001. These notes are neither sound nor complete. There is more material than is covered in lecture, and some is missing. Check your own notes for new topics brought up in 2002.]

Overview

Today's lecture is about losing secrets. Thus far we have assumed that cryptographic operations are isolated, with only inputs and outputs. Frequently, however, other kinds of information about the information being processed can be gained. Today we consider these other ways of loosing keys.

1. Smart cards and protocols
2. Power Analysis
3. Glitches
4. Tamper Resistance
5. Timing
6. Errors

1 Smart Cards and Protocols

Smart cards have all kinds of covert channels.

There are a number of applications of smart cards. Here are a few:

- Bank authorizing a person to an ATM.

  In this case the threat is that the ATM will get your secret key out. This is supposed to be hard though, because your smart card doesn't expose the secret key, it only does specific cryptographic operations with it.

- Pay TV boxes use smart cards for decryption algorithms. A basic box on the antenna decrypts the signal. It needs a smart card to allow the system and the keys to be changed easily.

  In this case the customer is the enemy. He wants to find out what is on the smart card so he can get free cable or sell boxes to offer other people free cable on the black market.

There are a number of successful attacks against smart cards.

2 Simple Power Analysis

Use what is observable. Measuring the power usage is a good first step. Surprisingly, this a devastating attack. Power usage correlates well with what is going on during the computation of a cipher.

For example, with DES there are 16 rounds, and a power trace from a device will show 16 bumps in the power usage. From this you can get secret key information out. Loading a 1 into a register bit is different (in terms of power usage) from loading a 0. Some implementations let you just read off the key. Others only give you information like how many 1s and 0s there are in a byte.

With RSA there is an even simpler attack. In an RSA decryption, we raise a user-supplied c to the power of d modulo n, where d is our secret. In an efficient implementation of modular exponentiation, the code cycles through d and either multiplies the total by c or squares the total based on the bits it finds. Multiplying and squaring use different amounts of power, in general. So the secret key can just be read from a power trace.

Q: Discounting efficiency, can you just draw constant power?

A: Yes, but silicon chips which draw constant power are hard to build. You can do things with capacitors to smooth the signal. Most attempts are partly effective. But variation in process can hurt you a lot, and it is hard to gage effectiveness.

3 Differential Power Analysis

This attack attempts to find correlations between power consumption, outputs and key bits by trying many inputs. By performing millions of trials on a card, one can find a statistical correlation. This attack is much like tempest.

Q: I have an RSA SecureID. It gives me a number every few seconds. What is it doing?

A: It has its own CPU and battery, with a secret and a clock, and is producing the numbers with a pseudo-random number generator.

Q: The clocks on the ID and the server are synced?

A: Yes, and the server knows the secret.

Q: Tamper resistant?

A: Maybe. :-)

4 Tamper Resistance

4.1 Glitches

Glitches are a form of an active attack. You can control the voltage and clock rate. What happens when you take the chip outside of its design parameters? Sometimes it does things that help you attack it.

Many chips have the following code:

```
1        b = answer address
2        a = length of answer
3  loop: if a = 0 goto end
4        transmit(*b)
5        b = b + 1
6        a = a - 1
7        goto loop
8   end:
```

The idea here is that you play with the voltage or clock length at lines 6 or 3, and you make sure that the loop never exits and `b` just keeps increasing. This way the chip will transmit its whole memory space to you.

Q: Why can't you read the memory yourself?

A: Typically a smart card does not expose the memory to you.

The tedium can be automated. Maybe it will not work (to find these glitches) in an ATM for 2 minutes, but the guy trying to steal cable is willing to let a computer chug along on the card for a few days.

4.2 Other Tampering

Sometimes the chips have their program and secrets in an EEPROM. EEPROM has a lock bit, supposed to keep the data from being read or written from outside the chip. By only lowering the voltage a little bit, or by using a UV light, the lock bit can be cleared.

Kuhn is really into this sort of thing. He can take apart almost anything with the right tools. He first analyzed tamper resistant devices in the lab. By fuming nitric acid, he is able remove the epoxy from a chip without damaging it. He also uses other chemistry. Hydrogen-floride removes silicon-dioxide. Ultrasonic vibrating probes can be used to burrow into the chip. Biology has developed laser tools attached to microscopes for cutting at the microscopic level. These laser-scopes are used to cut wires and other things. A lot of times it is possible to have a chip completely opened up and running. Electrical probes can be put right on the wires.

IBM has developed a process for taking a 3D image of the chip by etching away the chip one layer at a time. The metal, silicon and doped silicon is exposed. A layer of gold is applied, and an electron

microscope scans the area. The gold is removed and another layer of the chip is stripped away. It took 2 weeks and 6 chips to take an image of an Intel 386 (a fairly complicated chip) this way.

IBM has another similar technique for reverse engineering. They etch away the top layer and apply lithium-niobate to the parts that are of interest. Lithium-niobate changes its index of refraction with the electric field nearby. Using a laser, one can watch the 1s and 0s going around on the chip.

Sandia has a process for looking at chips without etching. At certain wavelengths in the infrared, silicon is transparent. Using infrared light, it is possible to look through the silicon and see the metallic traces.

Protecting against these sorts of attacks is very hard. The clipper chip tried to protect itself, but most people do not bother. According to Kuhn, "In conclusion, it is imprudent to assume that the design of silicon chips, or the information stored in them, can be kept from a capable motivated opponent."

One place where success in keeping electronics and secrets out of the hands of tamperers is nuclear weapon test detection equipment. The equipment itself has sensitive seismic equipment, and is programmed to self destruct if tampering is detected.

Another chip foiled by Kuhn is from Intel. Here a complex cryptographic system is used to separate the memory and CPU. Kuhn was able to tap the bus between the two subsystems and break the security.

5 Timing

The basic idea is to measure how long some encryption operation takes. This is an effective attack against smart cards and PCs.

A good example of this is an attack which works on RC5 implementations which do not use a barrel shifter. One can then measure the time taken by individual rotations and determine bits from the round keys.

Suppose you have $c^d \pmod n$ and you would like to factor n into p and q. One attack is to measure the time this decryption takes. If it is done using the Chinese Remainder Theorem, then c is divided into $c_1 \pmod p$ and $c_2 \pmod q$. These are used to create the expression $M = aM_1 - bM_2 \pmod n$. As c_1 is increased, it eventually becomes larger than p and thereby requires an extra operation to compute $\pmod p$. Someone watching the jump in this time can perform a binary search with a variety of c's and thereby discover p.

A variation of this idea is as follows. Suppose the c_1 side is processed and then the c_2 side is processed. First the answer is found mod p, then mod q and then they are linearly combined. If the adversary has the ability to play with the power supply, he can cause a glitch at an opportune moment. For instance suppose we did one decryption where the computation was good and then the second time, we allowed a good computation mod p but created a short clock pulse for mod q. Thus we would have M which was correct mod p and mod q and M' which is correct mod p but NOT correct mod q. Then $\gcd(M - M', n) = p$.

6 Errors

Another side channel is provided by error handling in some schemes. Bleichenbacher pointed out such a channel in SSL. SSL used a format for message transmission known as PKCS #1 which caused the message to be presented as [00][02]padding[00][message]. The server would generate and send an error message when it received a badly formatted message. This provided the adversary with an oracle to determine if it had formatted a message correctly. This turned out to be enough to decrypt Alice's key.

Q: What kind of time is required to perform this attack, how many messages?

A: For a 1024-bit modulus, about 2^{20} messages, not impossible. Surprisingly, a 1024-bit modulus is substantially harder than a 1025-bit modulus.

What is the fix for this? They moved away from PKCS #1, to OEAP, but it had the same sort of error handling bug. One way to approach it is to require the client to demonstrate knowledge of the message before an error message is sent back. With these sorts of schemes the devil is always in the details, and there are lots of details.