ECEN 426: Security of Embedded Systems **Fall 2021**

Lecture 2: Hardware Trojans

Lecturer: JV Rajendran *Scribe: Sarah Wauson*

Note: *LaTeX template courtesy of UC Berkeley EECS dept.*

2.1 What is a Hardware Trojan?

A Hardware Trojan is a hidden change or addition to an integrated circuit that causes harm to the original intent of the design. Often, attackers use Hardware Trojans in order to discretely cause unwanted effects for their opponent.

Hardware Trojans:

1. cause integrated circuits to malfunction or work against the designer.
2. are typically placed in the integrated circuit during the design or fabrication phases.
3. can be difficult to detect and find.

Due to the wide use of integrated circuits, it is crucial for Hardware Trojans to be detected. Hardware Trojans can have catastrophic effects and can potentially cost lives. This is why it is important to understand the different types of Hardware Trojans and where they can be placed.

2.2 Where Can Hardware Trojans Be Inserted?

Understanding where Hardware Trojans can be placed and how they are activated is important for prevention and locating them. Figure 2.1 highlights where Hardware Trojans can be placed, activated, and what effects they can have.

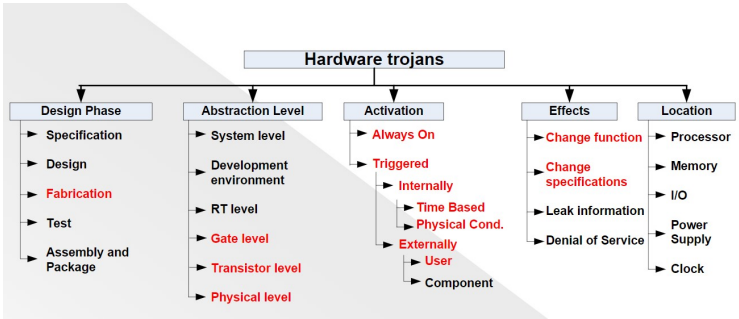

Figure 2.1: Hardware Trojans

Hardware Trojans can be inserted:

1. in the Fabrication Stage.
    (a) In the Fabrication Stage, a Hardware Trojan can be placed by the company fabricating the integrated circuit, or by an individual with malicious intent.
2. in the Design Stage.
    (a) Gate Level
    (b) Transistor Level
    (c) Physical Level

Insertion is one of the most important parts in the creative process of developing a Hardware Trojan. If it is not well hidden, the defense can quickly identify it and produce a quick counter-attack. The key is placing a stealth Trojan where it would least likely be found.

Understanding where Hardware Trojans can be inserted is important for those on the defense as well. It is important to know where your design is vulnerable to an attacker so that a Trojan may be prevented or found quickly.

2.3 How Can Hardware Trojans Be Activated?

Hardware Trojans can be:

1. Always on
2. Triggered
    (a) Internally
        i. Time-Based
        ii. Physical Condition
    (b) Externally
        i. User

2.4 The Location of a Hardware Trojan Can Be:

Hardware Trojans can be in the:

1. Processor
2. Memory
3. I/O
4. Power Supply
5. Clock

2.5 Examples of Trojans:

1. Leak Encryption Key

   (a) This type of Trojan is an input-triggered Trojan. The Trojan activates after the key phrase is input into the system. An example of this can be seen in Figure 2.2. In this example, the Trojan leaks information to the attacker after the key phase is inputted.

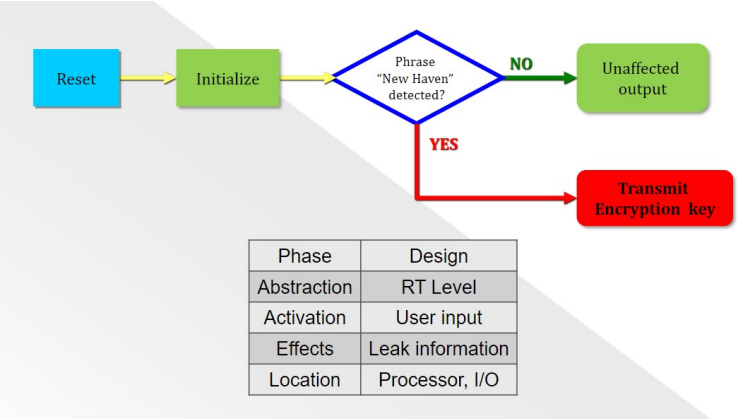

Figure 2.2: Leak Encryption Key

   (b) The con for this Trojan is that the attacker must be physically present in order to instantiate the Trojan.

2. Denial-of-Service Under Special Input

   (a) This Trojan is triggered by a key on the keyboard that does not have much use, such as F12 used in Figure 2.3. this technique is slightly stealthier than the Leak Encryption Key method. This Trojan will entirely disarm the program once the key is pressed.

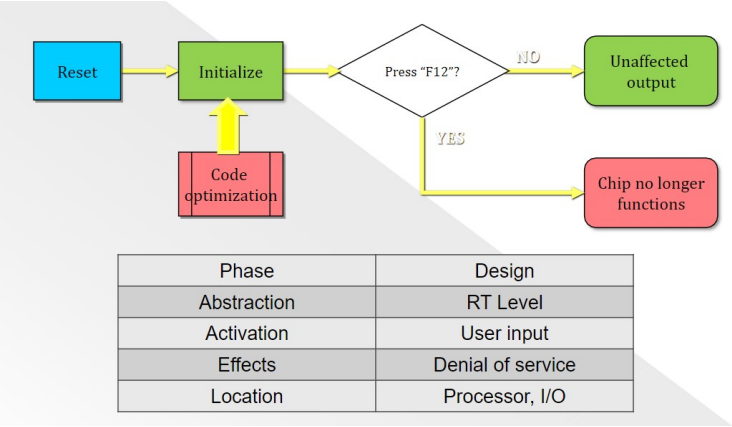

Figure 2.3: Denial-of-Service Under Special Input

   (b) Since a key such as F12 is not widely used, the user may not find this Trojan as quickly. The key is also discrete enough that an attacker may be able to use F12 while being watched by other

individuals that don't know the attacker's identity. However, the Trojan may be rather obvious if accidentally instantiated by someone who is not the attacker. They may notice that once the F12 key is pressed that the program immediately messes up. This can be rather obvious.

3. Faked Output

   (a) This type of Trojan detects certain words or phrases and changes them to the attacker's desired word or phrase.

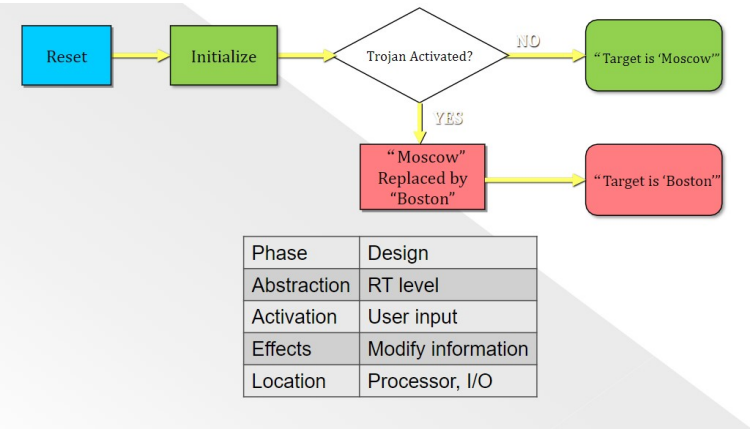

Figure 2.4: Faked Output

   (b) The con for this Trojan is that the attacker must be physically present in order to instantiate the Trojan. It may also be apparent that there has been an attack if the defense believes there to be an expected output.

4. Buffer Overflow

   (a) This type of Trojan will be activated if the user input is larger than a specified size. When it is activated, it will leak information to the attacker.

   (b) The con for this Trojan is that the user input may need to be very large. This may cause an inconvenience. However, if the program generally takes large user inputs, the original user may catch the Trojan if they enter in a large input.

5. Attacking the Transmission Protocol

   (a) This type of Trojan will always be activated. In order to be used, the attacker must know where to look for the hidden bits. They can be found by filtering by a chosen baud rate. After checking the chosen baud rate, leaked information can be found.

   (b) This method may be very effective due to the wide range of values where the data may be hidden. It may take awhile before the defense ever knows where the Trojan is hidden.

6. Time Bomb

   (a) This type of Trojan will be activated after the chosen amount of time is up. After the time has ran out, the Trojan will cause a change in the function of the chip.

   (b) The con for this Trojan is that it may be very obvious to the defense that a Trojan is present when the chip suddenly stops functioning correctly.

7. Controlling the Device

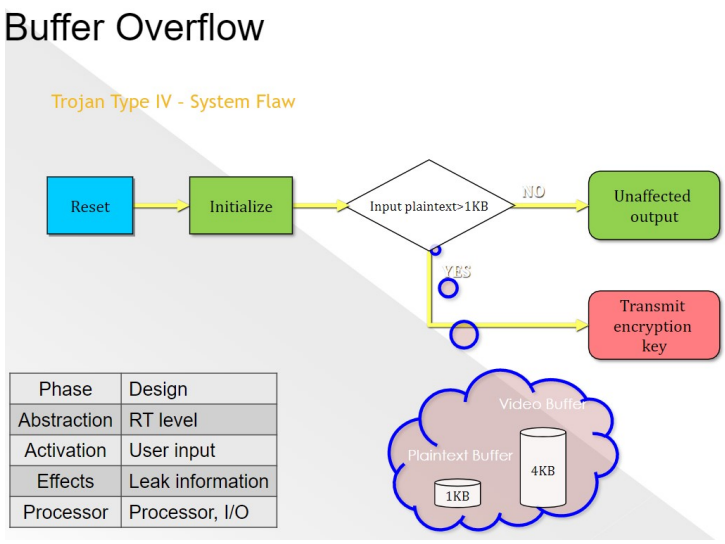

Figure 2.5: Buffer Overflow

(a) After receiving user input, the Trojan will not allow any user to use the chip and will leak information to the attacker.

8. Stealing Data from a Keyboard

(a) This Trojan is activated through user input and sends the attacker the information that is being typed into the keyboard.

(b) The con of this Trojan is that the attacker may have a hard time discerning what the information being typed into the keyboard is pertaining to. Without any information from the monitor, there may be missing information.

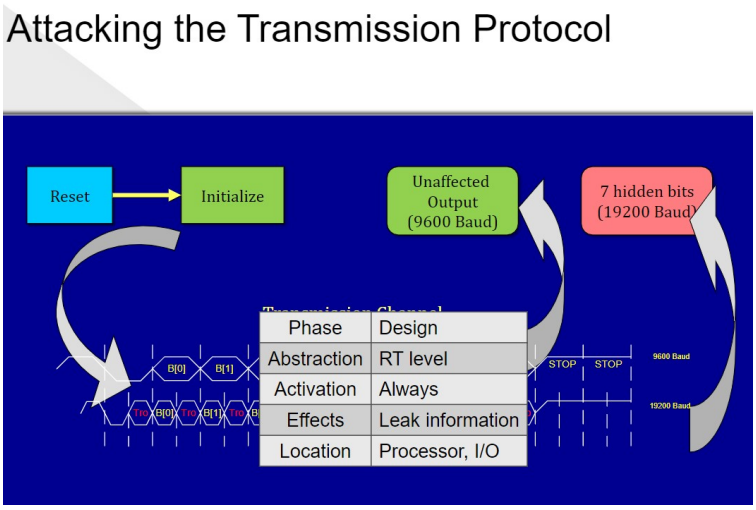

Figure 2.6: Attacking the Transmission Protocol

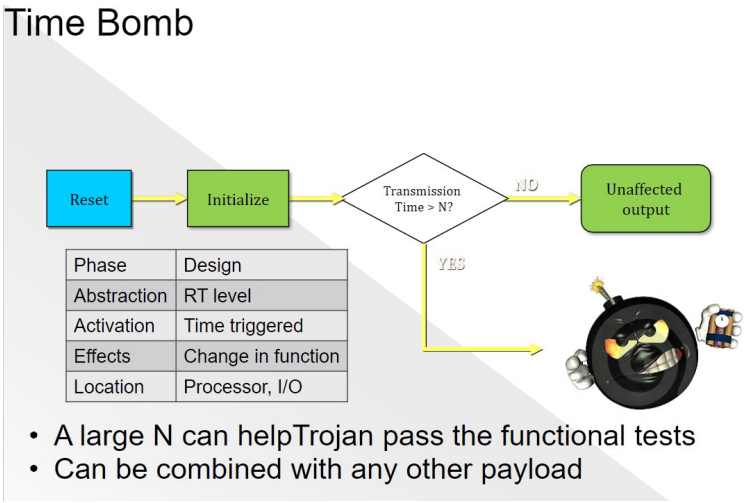

Figure 2.7: Time Bomb

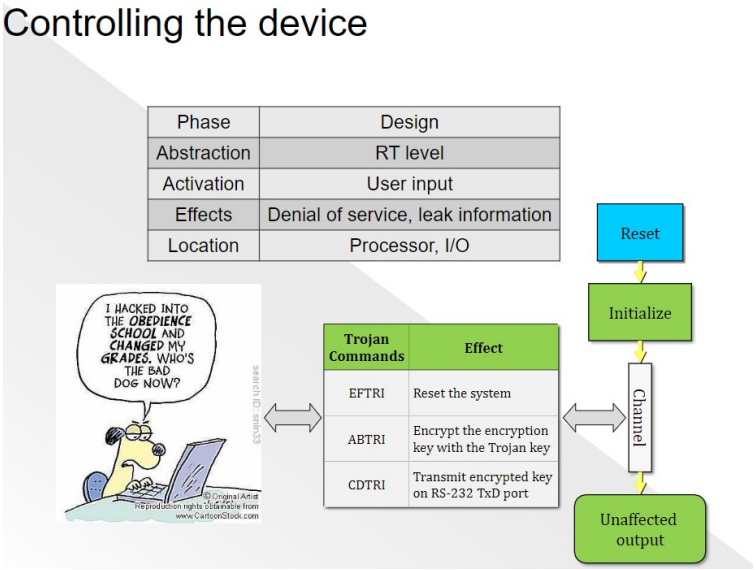

Figure 2.8: Controlling the Device

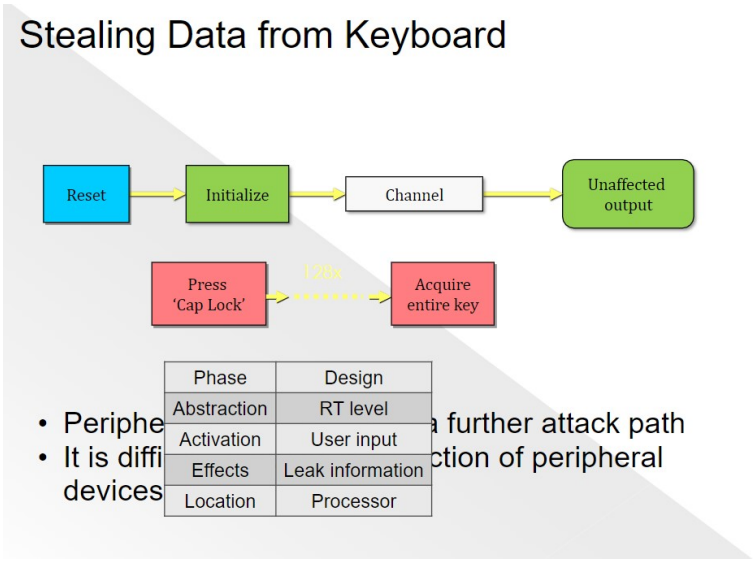

Figure 2.9: Stealing Data from a Keyboard