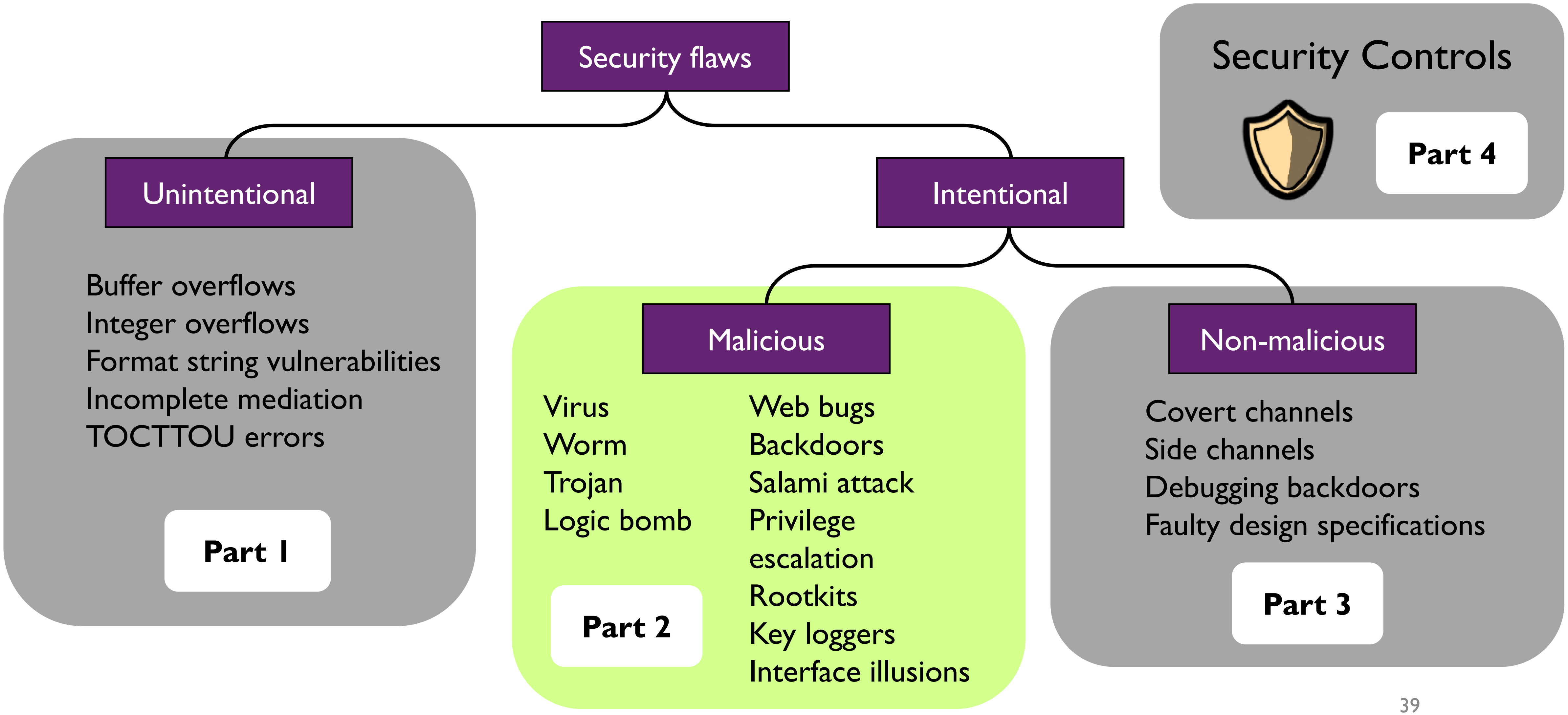
CPEN 442 – Introduction to Cybersecurity

Module 2 – Program Security

Part 2 – Malicious Flaws

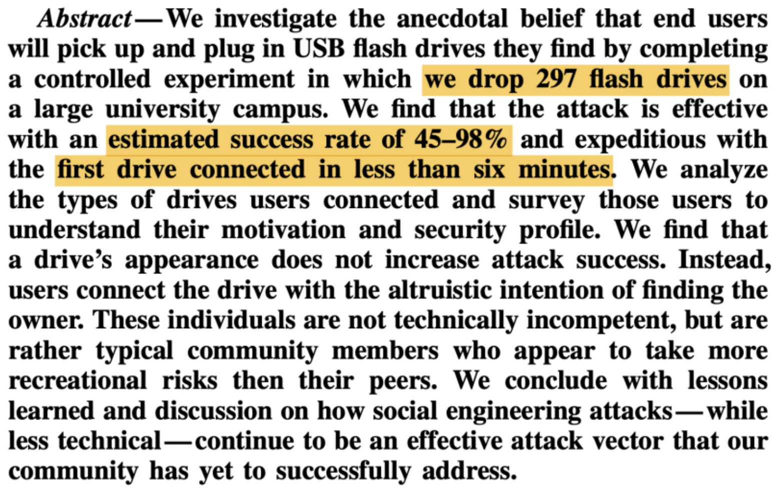
Module overview: Taxonomy of security flaws



Malware

- Malware is software written with malicious intent.
- Common characteristic of all types of malware: it needs to be executed in order to cause harm.
- How might malware get executed?
  - User action
    - Downloading and running malicious software
    - Viewing a web page containing malicious code
    - Opening an executable email attachment
    - Inserting a CD/DVD or USB flash drive
  - Exploiting an existing flaw in a system
    - Buffer overflows in network daemons
    - Buffer overflows in email clients or web browsers

Tischer et al. *"Users really do plug in USB drives they find". In 2016 IEEE Symposium on Security and Privacy (SP)*



***Abstract*—We investigate the anecdotal belief that end users will pick up and plug in USB flash drives they find by completing a controlled experiment in which we drop 297 flash drives on a large university campus. We find that the attack is effective with an estimated success rate of 45–98% and expeditious with the first drive connected in less than six minutes. We analyze the types of drives users connected and survey those users to understand their motivation and security profile. We find that a drive's appearance does not increase attack success. Instead, users connect the drive with the altruistic intention of finding the owner. These individuals are not technically incompetent, but are rather typical community members who appear to take more recreational risks then their peers. We conclude with lessons learned and discussion on how social engineering attacks—while less technical—continue to be an effective attack vector that our community has yet to successfully address.**

Types of Malware

- Virus
  - Malicious code that adds itself to benign program/files
  - It contains code for spreading, plus code for the actual attack
  - It is *usually* activated by users
- Worm
  - Malicious code spreading with no or little user involvement
- Trojans
  - Malicious code hidden in seemingly innocent programs that you download
- Logic bombs
  - Malicious code hidden in programs already on your machine

Later…
Web bugs
Backdoors
Salami attack
Privilege escalation
Rootkits
Key loggers
Interface illusions

Viruses (spreading)

- A virus is a kind of malware that infects other files (hosts)
  - Traditionally, it would infect executable programs
  - Nowadays, many data document formats can contain executable code (such as macros)
- Has a spreading part, and a payload (the actual attack)
- Spreading: when the file is executed (or just opened), the virus tries to spread by infecting other files, the computer itself, or even other computers.
  - For executable programs: usually it copies itself to the beginning of other (non-malicious) programs
  - For documents with macros: usually the virus will edit other documents to add itself as a macro
  - Sometimes it goes even further: copy itself to the boot sector to infect the computer itself, add itself as a program that runs at boot time
  - Between computers: when the user sends the files or compromised website links to their friends
- Big difference compared to worms: viruses spread at "user speed" (worms spread at "computer speed")

Viruses (payload)

- Besides spreading, what do viruses do?
- Some viruses try to evade detection by disabling any active virus scanning software
- Most viruses have some sort of payload (the attack), that does something, usually bad:
    - Erase your hard drive or make data inaccessible
    - Subtly corrupt some spreadsheets
    - Install a keystroke logger to capture your online banking password
    - Start attacking a particular target website

How do we defend against viruses?

- Look for them, when:
  - New files are added to computers (downloaded, or via portable network).
  - From time to time, scan the whole computer

- Two ways to look for viruses:
  - Signature-based protection
  - Behavior-based protection

Signature-based protection

- Keep a list of known viruses

- For each virus in the list, store some characteristic feature (its signature)
  - Most signature-based systems use features of the virus code itself:
    - the infection code
    - the payload code.
  - Identify where the virus usually tries to hide itself.
  - Identify how it propagates from one place to another.

- Major limitation: you can only scan for viruses in the list!
  - But there are new viruses every day
  - Some viruses are polymorphic (they make modified copies of themselves)
    - This can be done by encrypting the virus code.

What is the downside of this?

How would you detect a virus like this?

Behavior-based protection

- New viruses are identified every day! Signature-based protection cannot find these
- Instead of looking for a signature of a known virus, behavior-based protection looks for malicious behavioral patterns (usually in a sandbox):
  - Is it trying to disable security controls?
  - Is it trying to encrypt files?
  - Is it trying to delete files?
  - Is it trying to install something?

Think like an attacker! What would you do in this case to evade detection?

- This can help detect new viruses
- Machine learning models can be trained to decide whether a behavioral pattern is or not malicious.

Virus detection errors

- Any kind of test or scanner can have two types of errors:
  - False negatives: fail to identify a threat that is present
  - False positives: claim that a threat is present when it is not

Which error is worse?

Can I design a scanner without false negatives?

Signature-based vs behavioral-based
Which one usually has more false positives?

Base rate fallacy

- Base rate fallacy:
  - Suppose a virus scanner reports false positives in 0.5% of cases (FPR=0.5%), but never fails to detect a true threat (FNR=0%). This is a pretty good test, right?
  - Suppose that 1 in every 10,000 files is infected (the base rate).

If this scanner reports that a threat has been found, what is the probability that the threat is truly present?

What would be the effect of this for this user? (e.g., in the virus case)

Worms

- A worm is a self-contained piece of code that can replicate with little or no user involvement.
- Worms often use security flaws in widely deployed software as a path to infection.
  - The worm exploits a security flaw in some software on your computer, infecting it.
  - The worm starts searching for other computers (on your local network, or on the Internet generally) to infect.
  - There may or may not be a payload that activates at a certain time, or by another trigger

Example: the Morris worm

- The first Internet worm, launched by a graduate student at Cornell in 1988
- Once infected, a machine would try to infect other machines in three ways:
  - Exploit a buffer overflow in the "finger" daemon
  - Use a backdoor left in the "sendmail" mail daemon
  - Try a "dictionary attack" against local users' passwords. If successful, log in as them, and spread to other machines they can access without requiring a password
- First example of a buffer overflow exploit in the wild
- Thousands of systems were offline for several days

Robert Morris

Example: the Code Red worm

- Launched in 2001
- Exploited a *buffer overflow* in Microsoft's IIS web server (for which a patch had been available for a month)
- An infected machine would:
  - Deface its home page
  - Launch attacks on other web servers (IIS or not)
  - Launch a denial-of-service attack on a handful of web sites, including www.whitehouse.gov
  - Installed a backdoor to deter disinfection (which would stay even if you installed the patch!)
- Infected 250,000 systems in nine hours

The Slammer worm

- Launched in 2003, performed a denial-of-service attack
- First example of a “Warhol worm”
  - A worm which can infect nearly all vulnerable machines in just 15 minutes
- Exploited a buffer overflow in Microsoft’s SQL Server (also having a patch available)
- A vulnerable machine could be infected with a single UDP packet!!
  - This enabled the worm to spread extremely quickly
  - Exponential growth, doubling every 8.5 seconds
  - 90% of vulnerable hosts infected in 10 minutes

Conficker Worm

- First detected in November 2008
- There were multiple variants
- It had a command-and-control style botnet
- Security experts had to generate a sinkhole and sinkhole C&C domains
- Number of infected hosts:
    - in 2009: 9–15 million
    - in 2011: 1.7 million
    - in 2015: 400,000

Stuxnet

- Discovered in 2010
- Allegedly created by the US and Israeli intelligence
- Allegedly targeted Iranian uranium enrichment facilities
- Targeted Siemens SCADA systems installed on Windows. One application is the operation of centrifuges

- Very promiscuous: used four zero-day exploits
- Very stealthy: it only triggers in particular computers connected to particular centrifuges
- Very targeted: detects a particular frequency of the centrifuge, and subtly changes it so that the centrifuge breaks

IoT Malware

- Internet-of-Things (IoT): connected home, industry automation, etc.
- Cheap devices with Internet connectivity
- Dismal security: lack of expertise, lack of resources (CPU, memory, etc.)
- Perfect target for worms!

- Example: Mirai botnet:
  - Targeted cameras and routers
  - In 2016 it took out the DNS provider Dyn, making many popular services unreachable

Trojans

Trojans

- Trojan horses are programs which claim to do something innocuous (and usually do), but which also hide malicious behavior.

You're surfing the Web and you see a button on the Web site saying, "Click here to see the dancing pigs." And you click on the Web site and then this window comes up saying, "Warning: this is an untrusted Java applet. It might damage your system. Do you want to continue? Yes/No." Well, the average computer user is going to ***pick dancing pigs over security*** *any day. And we can't expect them not to.*

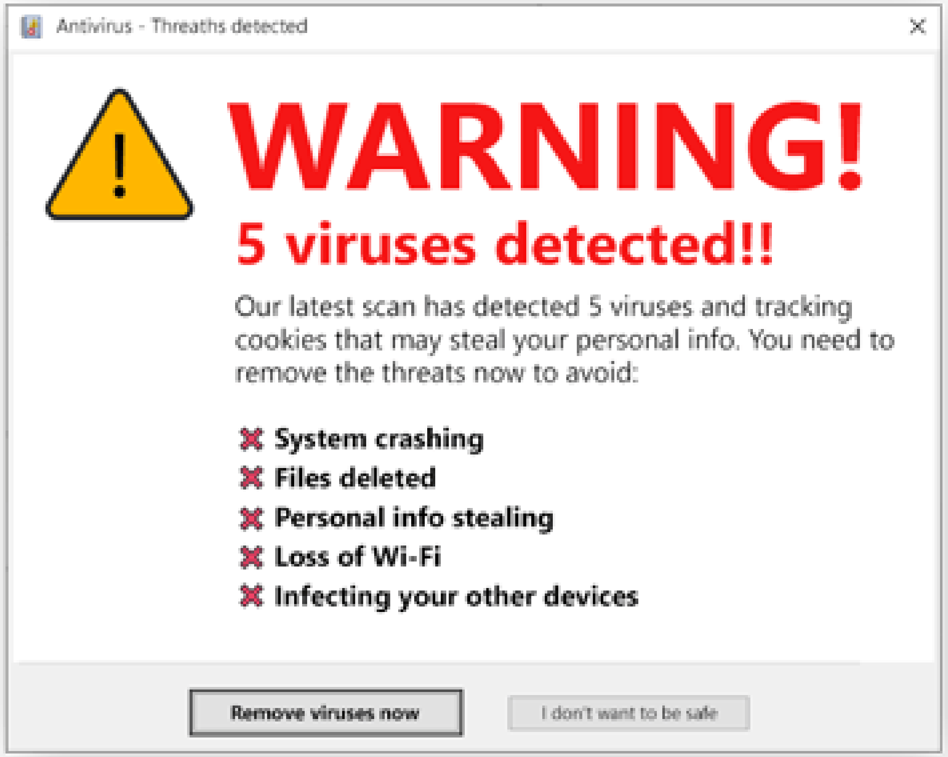
— Bruce Schneier

Trojan horses

- They gain control by getting the user to run code of the attacker's choice, usually by also providing some code that the user wants to run:
  - "PUP" (potentially unwanted programs) are an example.
  - For scareware, the user might even pay the attacker to run the code.
- The payload can be anything; sometimes the payload of the Trojan horse is itself a virus.
- They usually do not themselves spread between computers, but rely on multiple users executing the "trojaned" software.

Example: Scareware

It scares the user into installing an unwanted software, usually by making the user believe their computer is infected with a virus


Antivirus - Threaths detected

WARNING!

5 viruses detected!!

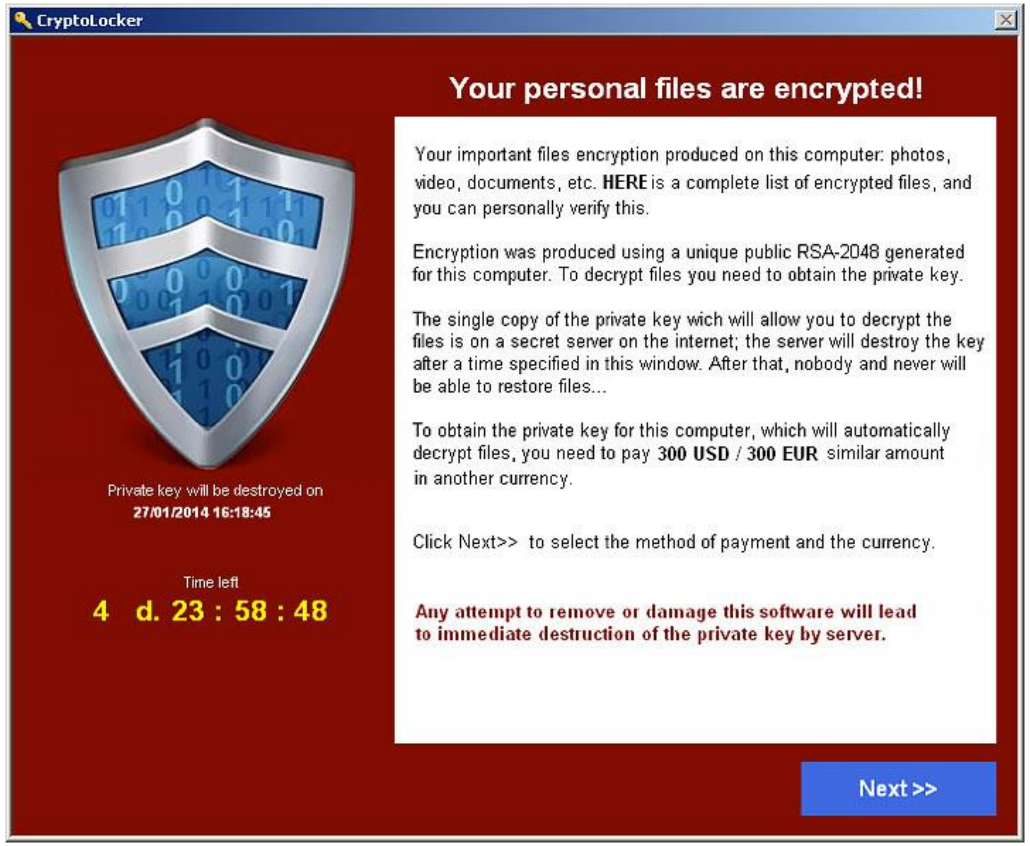
Our latest scan has detected 5 viruses and tracking cookies that may steal your personal info. You need to remove the threats now to avoid:

- ✖ System crashing
- ✖ Files deleted
- ✖ Personal info stealing
- ✖ Loss of Wi-Fi
- ✖ Infecting your other devices

Ransomware

- Holds some valuable resource from the victim hostage
  - For example, by encrypting the victim's hard drive
- Asks for a ransom to return the hostages
  - E.g., ransom for the decryption key

- Cryptolocker in 2013:
  - Spread with spoofed email attachments
  - Taken down in 2014, estimated ransom collected $3–$30 million
- Extremely common now!



CryptoLocker

Your personal files are encrypted!

Your important files encryption produced on this computer: photos, video, documents, etc. **HERE** is a complete list of encrypted files, and you can personally verify this.

Encryption was produced using a unique public RSA-2048 generated for this computer. To decrypt files you need to obtain the private key.

The single copy of the private key wich will allow you to decrypt the files is on a secret server on the internet; the server will destroy the key after a time specified in this window. After that, nobody and never will be able to restore files...

To obtain the private key for this computer, which will automatically decrypt files, you need to pay **300 USD / 300 EUR** similar amount in another currency.

Click Next>> to select the method of payment and the currency.

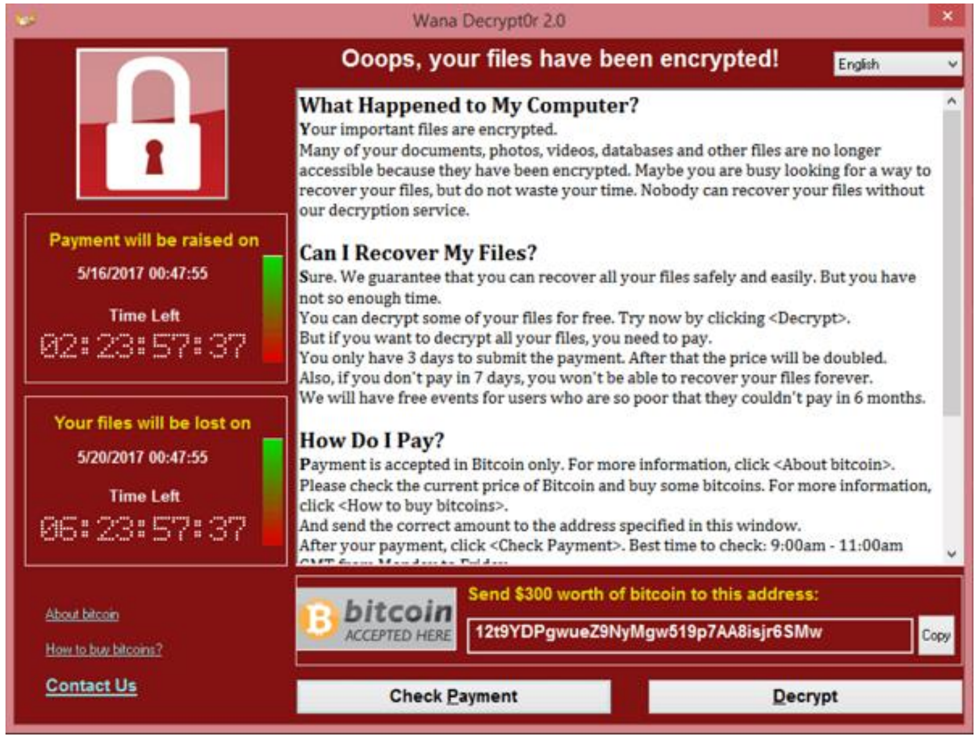
Any attempt to remove or damage this software will lead to immediate destruction of the private key by server.

Private key will be destroyed on
27/01/2014 16:18:45

Time left
4 d. 23 : 58 : 48

WannaCry

- Launched in May 2017, ransomware
- Infected 230,000 computers, including many of the British National Health Service
- The NSA had discovered a vulnerability on Windows (a zero-day)
- They kept it secret, to exploit it themselves
- The NSA was itself hacked by the "Shadow Brokers", who leaked this exploit code in April 2017
- Microsoft had released a patch after being alerted by the NSA but many systems remained unpatched
- Emergency patch for Windows XP and 8 in May 2017



Ooops, your files have been encrypted!

Payment will be raised on
5/16/2017 00:47:55
Time Left
02:23:57:37

Your files will be lost on
5/20/2017 00:47:55
Time Left
06:23:57:37

What Happened to My Computer?
Your important files are encrypted.
Many of your documents, photos, videos, databases and other files are no longer accessible because they have been encrypted. Maybe you are busy looking for a way to recover your files, but do not waste your time. Nobody can recover your files without our decryption service.

Can I Recover My Files?
Sure. We guarantee that you can recover all your files safely and easily. But you have not so enough time.
You can decrypt some of your files for free. Try now by clicking <Decrypt>.
But if you want to decrypt all your files, you need to pay.
You only have 3 days to submit the payment. After that the price will be doubled.
Also, if you don't pay in 7 days, you won't be able to recover your files forever.
We will have free events for users who are so poor that they couldn't pay in 6 months.

How Do I Pay?
Payment is accepted in Bitcoin only. For more information, click <About bitcoin>.
Please check the current price of Bitcoin and buy some bitcoins. For more information, click <How to buy bitcoins>.
And send the correct amount to the address specified in this window.
After your payment, click <Check Payment>. Best time to check: 9:00am - 11:00am

Send $300 worth of bitcoin to this address:
12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw

Whose fault was this?

Las Vegas MGM hack

- This just appeared in the news yesterday (Sep 13, 2023), so *none of this information is truly confirmed*
- A group of hackers known as the "Scattered Spider"
- The hackers use social engineering to lure users into giving up their login information or one-time-password (OTP) to bypass authentication
- Casinos seem to be the target of ransomware attacks

BET SPORTS HERE
BETMGM
I'm having trouble communicating with the system. When the link is restored, your session will resume.
BET SPORTS HERE

Logic bomb

- A logic bomb is malicious code hiding in the software already on your computer, waiting for a certain trigger to "go off" (execute its payload).
- Usually written by "insiders" and meant to trigger in the future.
- The trigger is usually the insider can affect once they're no longer an insider (a number combination on a keypad, triggers at a certain time in the future...)
- The payload is usually dire: erase your data, corrupt your data, encrypt your data and ask for a ransom…

Example: 2013 South Korea cyberattack:

Affected banks and broadcasting companies, wiping off their machines. The program had a string indicating the date and time the attack would trigger.

Spotting Trojan horses and logic bombs

- It is extremely tricky, since the user is intentionally running the code.
    - Trojans: the user clicked "yes" to see the dancing pigs
    - Logic bombs: the code is a hidden part of the software already installed on the computer.

- Do not run code from untrusted sources?

- Even better: prevent the payload from doing bad things?

Next: we'll see these at a high level

- ~~Virus~~
- ~~Worms~~
- ~~Trojan~~
- ~~Logic bombs~~

- Web bugs (beacon)
- Back doors
- Salami attacks
- Privilege escalation
- Rootkits
- Keystroke logging
- Interface illusions

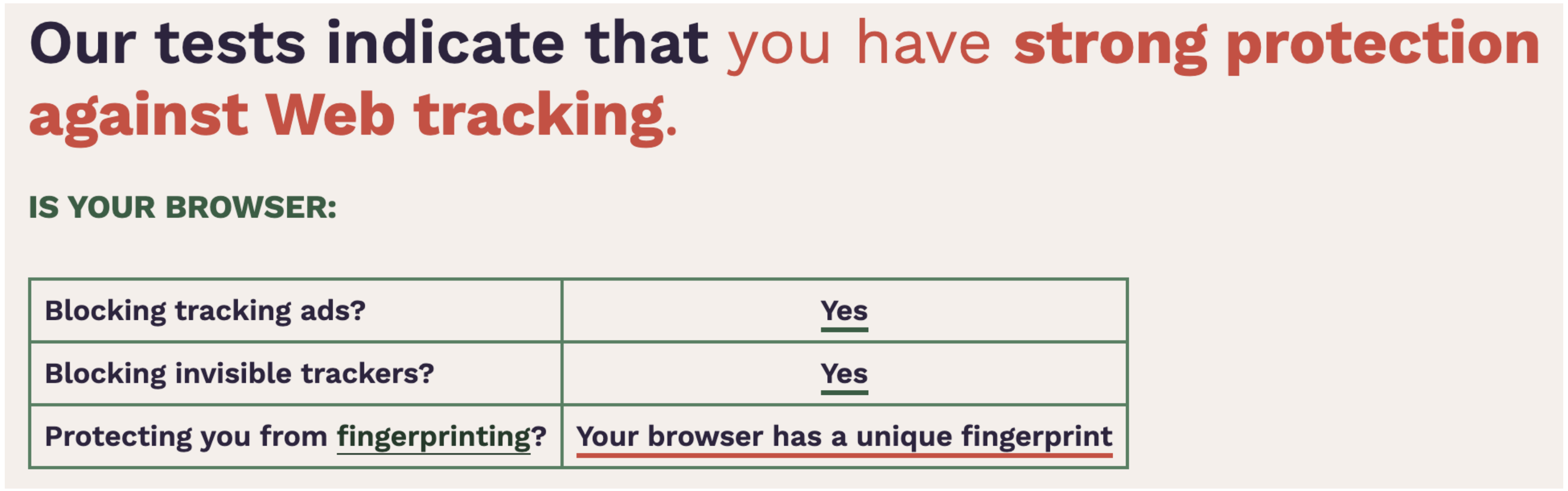
Web bugs

*From Wikipedia: also called web beacon, tracking bug, tag, web tag, page tag, tracking pixel, pixel tag, 1×1 GIF, or clear GIF.

- A web bug* is an object (usually a 1x1 pixel transparent image) embedded in a web page or email, which is fetched from a different server from the one that served the web page itself.
- Information about you can be sent to third parties (often advertisers) without your knowledge or consent:
  - IP address
  - Contents of cookies
  - Personal info the site has about you
- This is an issue of privacy more than of security:
  - The web bug instructs your browser to behave in a way contrary to the principle of informational self-determination

Example

- An example, from an article published in 2022: the Meta Pixel https://themarkup.org/pixel-hunt/2022/04/28/applied-for-student-aid-online-facebook-saw-you
- Want to check how well-protected you are vs. web tracking? Click here: https://coveryourtracks.eff.org/
- These are my results:



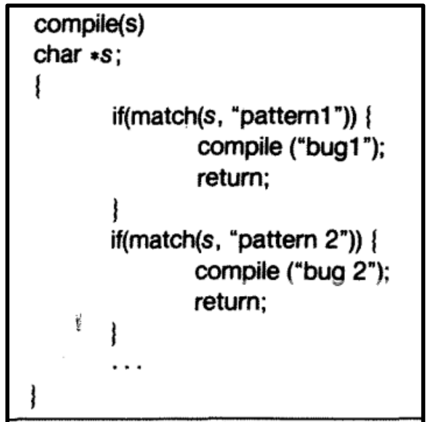
Our tests indicate that you have **strong protection against Web tracking.**

IS YOUR BROWSER:

Blocking tracking ads?	Yes
Blocking invisible trackers?	Yes
Protecting you from fingerprinting?	Your browser has a unique fingerprint

Backdoors

- A backdoor (or trapdoor) is a set of instructions designed to bypass the normal authentication mechanism and allow access to the system to anyone who knows that the backdoor exists.
  - Sometimes these are useful for debugging the system, but don't forget to take them out before you ship them!
- Sources of backdoors:
  - Someone forgets to remove them
  - Intentionally left for testing or maintenance purposes
  - Intentionally left them for legal reasons ("lawful access")
  - Intentionally left for malicious purposes
- Reflections on trusting trust: *(very cool reading, recommended!)*
  - If you put the backdoor in the source code, it might be obvious…
  - But you can put the backdoor into the binary (and remove it from the source code later)
  - Even worse: you can put a backdoor in the compiler (the C compiler is written in C)
  - Even worse: you can put the backdoor in the compiler binary!



```
compile(s)
char *s;
{
        if(match(s, "pattern1")) {
                compile ("bug1");
                return;
        }
        if(match(s, "pattern 2")) {
                compile ("bug 2");
                return;
        }
        . . .
}
```

FIGURE 3.3.

Backdoor examples

- Debugging back door left in sendmail: exploited by the Morris Worm
- Backdoor planted by Code Red worm
- Port knocking (a "more stealthy" backdoor):
  - The system listens for connection attempts to a certain pattern of (closed) ports. All those connection attempts will fail, but if the right pattern is there, the system will open, for example, a port with a root shell attached to it.
- Attempted hack to Linux kernel source code

```
if ((options == (__WCLONE|_WALL)) && (current->uid = 0))
retval = -EINVAL;
```

Salami attack

- A salami attack is an attack that is made up of many smaller, often considered inconsequential, attacks.
- Classic example: send the fractions of cents of round-off error from many accounts to a single account owned by the attacker.
- More commonly:
  - Credit card thieves make very small charges to many cards
  - Clerks slightly overcharge customers for merchandise
  - Gas pumps misreport the amount of gas dispensed

Privilege escalation

- A privilege escalation attack is an attack which raises the privilege level of the attacker beyond that to which they would ordinarily be entitled.
- Usually occurs when part of the system that legitimately runs with higher privilege can be tricked into executing commands with that privilege on behalf of the attacker
  - Buffer overflows in *setuid* programs or network daemons
  - Component substitution
- The attacker might also trick the system into thinking they are in fact a legitimate (higher-privileged) user:
  - Problems with authentication systems
  - Obtain session id/cookie from another user to access their bank account

Rootkits

- A rootkit is a tool often by "script kiddies", which has two main parts:
  - A method for gaining unauthorized root/admin privileges on a machine
    - Usually exploiting some flaw in the system
    - Leaving a backdoor so that the attacker can be get back even if the flaw is corrected
  - A way to hide its own existence
    - Cleaning up any log messages that might have been created by the exploit
    - Modifying commands like ls and ps so that they don't report files belonging to the rootkit
    - Modifying the kernel so that no user program will ever learn about those files and processes

Rootkit example: Sony XCP

- Mark Russinovich was developing a rootkit scanner for Windows
- When he was testing it, he though his scanner had a bug, cause it was detecting a rootkit in his computer
    - But it was correct, it had a rootkit in it!

- The source of the rootkit turned out to be Sony audio CDs equipped with XCP "copy protection"
- When you insert such an audio CD, it contains an autorun.exe file which automatically executes, installing the rootkit
- The "primary" purpose of the rootkit was to modify the CD driver so that trying to read a protected CD would not work
- The "secondary" purpose was to make itself hard to find and uninstall
    - Hid any files whose names started with sys
    - Eventually, other attackers started using this strategy, and Sony released an uninstaller…
        - …which left a backdoor on your system

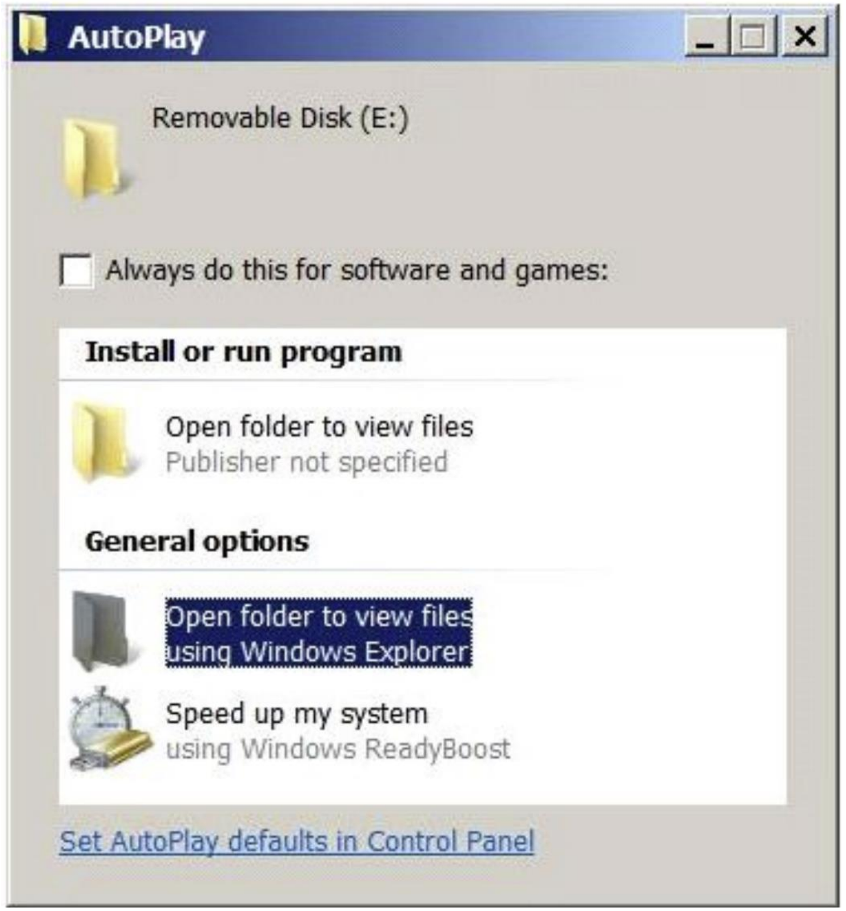
Keylogger

- A keyboard logger is a piece of software or hardware that captures information from the user to the computer via the keyboard
  - This allows capturing passwords, emails, etc.
  - This information can be sent to a remote attacker
- Some keyboard loggers can be installed by malware
- Sometimes installed by family members to spy on another
- Sometimes they only record keystrokes associated with a particular application
- Sometimes they are a small piece of hardware that sits between the keyboard and the computer
  - This is completely undetectable in software.

Interface illusions

- Malicious code can make nonstandard user interface elements in order to trick us.
  - What if the scrollbar isn't really a scrollbar? What if dragging on a "scrollbar" is actually dragging a malicious program to your "startup" folder?

Example: Conficker worm →


AutoPlay

Removable Disk (E:)

Always do this for software and games:

Install or run program

Open folder to view files
Publisher not specified

General options

Open folder to view files using Windows Explorer

Speed up my system
using Windows ReadyBoost

Set AutoPlay defaults in Control Panel

Interface Illusions: Phishing

- Phishing is an example of an interface illusion:
  - e.g., it looks like you're vising Paypal's website, but you're really not. If you type in your password, you've just given it to an attacker
- Advanced phishers can make websites that look every bit like the real thing, even the address bar or the SSL certificate!
- How to detect phishing?
  - Unusual email/URL
  - Attachments with uncommon names
  - No https

Example: www.paypa1.com

Man-in-the-middle attacks

- Keyboard logging, interface illusions, and phishing are examples of man-in-the-middle (MitM) attacks.
- The website/program/system you're communicating with isn't the one you think you're communicating with.
- The user thinks nothing is wrong, since the MitM still forwards the information to the intended other party.
- The MitM not only can record everything we do, but can hijack the session to insert malicious commands