CS 489 / 698: Software and Systems Security

Module 3: Operating System Security

malware and anti-malware techniques

Meng Xu *(University of Waterloo)*

Spring 2023

Outline

What is malware?

Definition: various forms of software written with malicious intent.

What is malware?

Definition: various forms of software written with malicious intent.

- Common characteristic of all types of malware: 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

Types of malware

- Virus
  - Malicious code that adds itself to benign programs/files
  - Code for spreading + code for actual attack
  - Usually activated by users
- Worms
  - Malicious code spreading with no or little user involvement
- Trojans
  - Malicious code hidden in seemingly innocent program that you download
- Logic Bombs
  - Malicious code hidden in programs already on your machine
- Ransomware / Spyware / Adware / Rootkits / Bot / . . .

Types of malware

Q: Are these malware types mutually exclusive?

Types of malware

Q: Are these malware types mutually exclusive?

A: I don't know... For all the textbooks I read, there seem to be sharp differences between them. But malware is usually layered by design which makes it hard to categorize them.

Consider the following hyperthetical scenario:

Alice downloaded a trojan that looks like the Chrome browser
    which released a worm (background process)
        which spread to Bob's computer and unpacks a logic bomb
            which exploded Saturday midnight and releases a virus
                which infected the web server and turns it into a bot
                    which received a rootkit, installed it, becomes root
                        which encrypted all Bob's files and asked for Bitcoin

Outline

What is a virus?

A virus is a specific type of malware that “infects” other files.

What is a virus?

A virus is a specific type of malware that “infects” other files.

- Traditionally, a virus could infect only executable programs
- Nowadays, many data document formats can contain executable code (such as macros in `.xlsx` or JavaScript in `.pdf`)

Typically, when the file is executed (or sometimes just opened), the virus activates, and tries to infect other files with copies of itself. In this way, the virus can spread between files in the local system, or even between computers.

Infection

Q: What does it mean to “infect” a file?

Infection

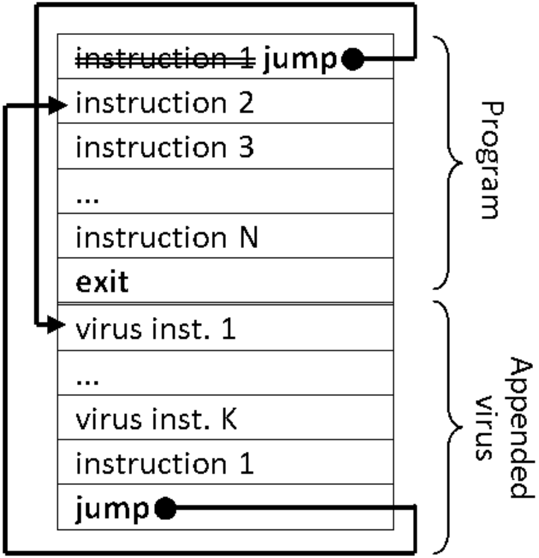
Q: What does it mean to "infect" a file?

A: The virus wants to modify an existing (non-malicious) program or document (the host) in such a way that executing or opening it will transfer control to the virus.

- The virus can do its "dirty work" first and then transfers control back to its host program or document.

- For executable programs:
  - Typically, the virus will modify other programs and copy itself to the beginning of the targets' program code
- For documents with macros:
  - The virus will edit other documents to add itself as a macro which starts automatically when the file is opened

Infection (one possibility)



Infection

In addition to infecting other files, Ia virus will often try to infect the computer itself. A typical goal is that every time the computer is booted, the virus is automatically activated.

- It might put itself in the boot sector of the hard disk
- It might add itself to the list of programs the OS runs at boot time or immediately after boot
- It might infect one or more of the programs the OS runs at boot time or immediately after boot
- It might try many of these strategies

Spreading

- How do viruses spread between computers?
- Usually, when the user sends infected files (hopefully not knowing they're infected!) or compromised website links to his friends
- A virus usually requires some kind of user action in order to spread to another machine
  - If it can spread on its own (via dictionary attack, for example), it's more likely to be a worm than a virus

Payload

In addition to trying to spread, what else might a virus try to do?

- Some viruses try to evade detection by disabling any active virus scanning software
- Most viruses have some sort of payload. At some point, the payload of an infected machine will activate, and do something (usually bad)
  - Erase your hard drive, or make your data inaccessible
  - Subtly corrupt some of your spreadsheets
  - Install a keystroke logger to capture your online banking password
  - Start attacking a particular target website

Outline

What is a worm?

A worm is a self-contained piece of code that can replicate with little or no user involvement.

What is a worm?

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, typically:

- A worm exploits a security flaw in some software on your computer, infecting it
- The worm immediately 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

The Morris worm

The Morris worm is 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 back door 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

The Morris worm has a few unique features:

- All three of these attacks were well known!
- First example of buffer overflow exploit in the wild
- Thousands of systems were offline for several days

The Slammer worm

The Slammer worm was launched in 2003, which performed denial-of-service attack.

- Exploited a buffer overflow in Microsoft's SQL Server (already had a patch available)
- A vulnerable server 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

Stuxnet

- Discovered in 2010
- Allegedly created by the US and Israeli intelligence agencies
- Allegedly targeted Iranian uranium enrichment program
- Targets Siemens Supervisory Control and Data Acquisition (SCADA) systems installed on Windows. One application is the operation of centrifuges
- It tries to be very specific and uses many criteria to select which systems to attack after infection

Stuxnet

- Very promiscuous: Used 4(!) different zero-day attacks to spread. Has to be installed manually (USB drive) for air-gapped systems.
- Very stealthy: Intercepts commands to SCADA system and hides its presence
- Very targeted: Detects if variable-frequency drives are installed, operating between 807–1210 Hz, and then subtly changes the frequencies so that distortion and vibrations occur resulting in broken centrifuges.

IoT Malware

- Internet-of-Things (IoT): connected devices for home or industry automation etc.
- Cheap commodity devices with Internet connectivity.
- Dismal security: lack of expertise, lack of resources (CPU, memory, etc.)
- e.g., Mirai (2016): Took out DNS provider Dyn, making many popular services unreachable.

Outline

Have you ever seen this?

Resident Shield alert

The horsey is infected

Threat detected!

File name:
C:\Turkey\Troad\Dardanus\Troy\WoodenHorse.exe

Threat name: Trojan Horse full of Soldiers
Detected outside of walls
More information about this threat ...

Remove threat as Cassandra

Process name: C:\Achaean\Mycenae\Agamemnon.exe
Process ID: 1184

Burn
Let Horse In
Ignore
Hide details

Retrieved from Internet

What is a trojan horse?

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

What is a trojan horse?

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

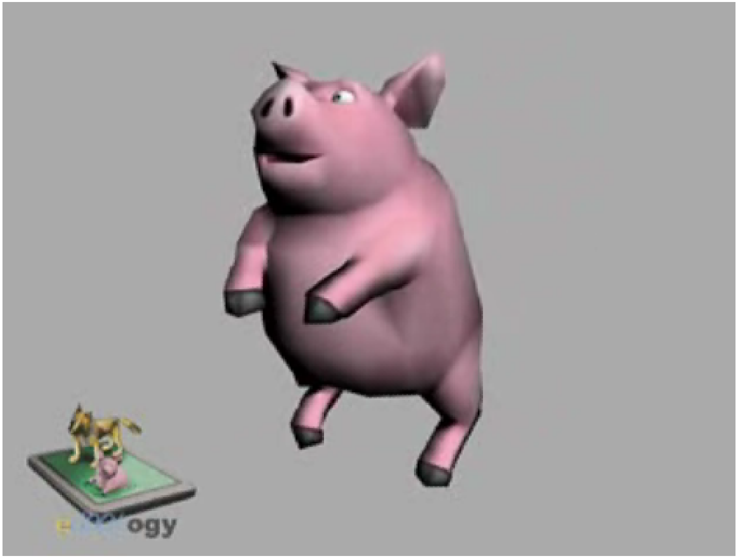
You're surfing the Web and you see a button on the Web site saying, "Click here to see the dancing pigs." And you click on the Web site and then this window comes up saying,

"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*

Dancing pig

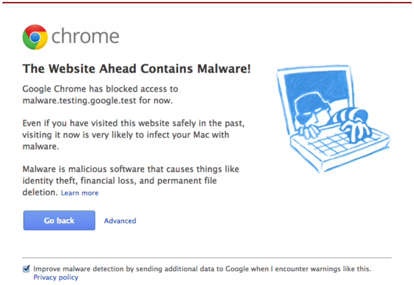


Available on YouTube

Better user-experience on security warnings?

Recent research has questioned the conventional wisdom regarding the dancing pig phenomenon. Based on a large scale study of browser warnings in Chrome and Firefox, the paper Alice in Warningland (USENIX Security 2013) shows that users do respond to well-designed warnings.

Malware / phishing warnings



chrome

The Website Ahead Contains Malware!

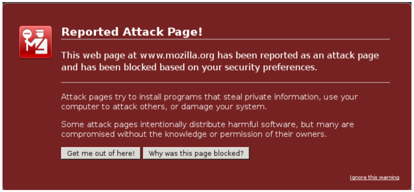
Google Chrome has blocked access to malware.testing.google.test for now.

Even if you have visited this website safely in the past, visiting it now is very likely to infect your Mac with malware.

Malware is malicious software that causes things like identity theft, financial loss, and permanent file deletion. Learn more

Go back Advanced

☑ Improve malware detection by sending additional data to Google when I encounter warnings like this. Privacy policy



Reported Attack Page!

This web page at www.mozilla.org has been reported as an attack page and has been blocked based on your security preferences.

Attack pages try to install programs that steal private information, use your computer to attack others, or damage your system.

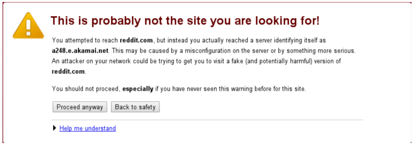
Some attack pages intentionally distribute harmful software, but many are compromised without the knowledge or permission of their owners.

Get me out of here! Why was this page blocked?

Ignore this warning

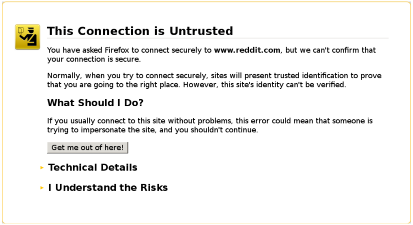
Click-through rates for the two browsers' malware and phishing warnings ranged from 9% to 23% (based on the user-study paper).

SSL warnings


This is probably not the site you are looking for!

You attempted to reach **reddit.com**, but instead you actually reached a server identifying itself as **a248.e.akamai.net**. This may be caused by a misconfiguration on the server or by something more serious. An attacker on your network could be trying to get you to visit a fake (and potentially harmful) version of **reddit.com**.

You should not proceed, **especially** if you have never seen this warning before for this site.


This Connection is Untrusted

You have asked Firefox to connect securely to **www.reddit.com**, but we can't confirm that your connection is secure.

Normally, when you try to connect securely, sites will present trusted identification to prove that you are going to the right place. However, this site's identity can't be verified.

What Should I Do?

If you usually connect to this site without problems, this error could mean that someone is trying to impersonate the site, and you shouldn't continue.

Click-through rates is 33% for Firefox, but 70% for Chrome (based on the user-study paper).

Outline

What is a 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).

What is a 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).

The payload of a logic bomb is usually pretty dire

- Erase your data
- Corrupt your data
- Encrypt your data, and charge you ransom for the decryption key

Spotting Trojan horses and logic bombs

Spotting Trojan horses and logic bombs is extremely tricky. One reason is that the user is intentionally running the code! I

- Trojan horses: the user explicitly clicked “yes, I want to see the dancing pigs”
- Logic bombs: the code is just (a hidden) part of the software already installed on the computer.

What is a back door?

A back door (also called a trapdoor) is a set of instructions designed to bypass the normal authentication mechanism and allow access to the system to anyone who knows the back door exists.

What is a back door?

A back door (also called a trapdoor) is a set of instructions designed to bypass the normal authentication mechanism and allow access to the system to anyone who knows the back door exists.

Example: several models of D-Link routers found to have backdoor:

- Set the user-agent in browser to "`xmlset_roodkcableoj28840ybtide`" and the admin web interface skips authentication
- Parsed backwards, that's "`edit by 04882 Joel backdoor teslmx`"

Sources of back doors

- Forget to remove them
- Intentionally leave them in for testing purposes
- Intentionally leave them in for maintenance purposes
- Intentionally leave them in for legal reasons
- Intentionally leave them in for malicious purposes
- Targeted planting

The Linux backdoor attempt of 2003

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

The Linux backdoor attempt of 2003

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

Q: What is wrong with this code?

The Linux backdoor attempt of 2003

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

Q: What is wrong with this code?

A: `current->uid = 0` vs `current->uid == 0`

The Linux backdoor attempt of 2003

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

Q: What is wrong with this code?

A: `current->uid = 0` vs `current->uid == 0`

For the full story, refer to this article (e.g., for hacking into the CVS server to plant this code change instead of going through a normal code review process).

Outline

Web bug

A web bug is an object (usually a 1x1 pixel transparent image) embedded in a web page, which is fetched from a different server from the one that served the web page itself.

Web bug

A web bug is an object (usually a 1x1 pixel transparent image) embedded in a web page, 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 (to link cookies across web sites)
- Any personal info the site has about you

Web bug example

This was previously on the `quicken.intuit.com` home page:

```
<IMG WIDTH="1" HEIGHT="1"
src="http://app.insightgrit.com/1/nat?
id=79152388778&
ref=http://www.eff.org/Privacy/Marketing/web_bug.html&
z=668951&
purl=http://quicken.intuit.com/">
```

Q: What information is sent out to where?

“Malicious code”?

- Why do we consider web bugs “malicious code”?
- 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
  - Much in the same way that a buffer overflow attack would instruct your browser to behave in a way contrary to the security policy

Rootkit

A rootkit is a method for gaining unauthorized root / administator privileges on a machine (either starting with a local unprivileged account, or possibly remotely).

- This method usually exploits some known flaw in the system that the owner has failed to correct
- It often leaves behind a back door so that the attacker can get back in later, even if the flaw is corrected

Rootkit

A rootkit is a method for gaining unauthorized root / administator privileges on a machine (either starting with a local unprivileged account, or possibly remotely).

- This method usually exploits some known flaw in the system that the owner has failed to correct
- It often leaves behind a back door so that the attacker can get back in later, even if the flaw is corrected

A unique features bout rootkits are their stealthiness.

Stealth capabilities

Q: How do rootkits hide their existence (a.k.a., cloaking)?

- Clean up any log messages that might have been created by the exploit
- Modify commands like `ls` and `ps` so that they don't report files and processes belonging to the rootkit
- Alternately, modify the kernel so that no user program will ever learn about those files and processes!

Example: Sony XCP

- Mark Russinovich was developing a rootkit scanner for Windows
- When he was testing it, he discovered his machine already had a rootkit on 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 into your computer, it contains an `autorun.exe` file which automatically executes
- `autorun.exe` installs the rootkit

Example: Sony XCP

- The “primary” purpose of the rootkit was to modify the CD driver in Windows so that any process that tried to read the contents of an XCP-protected CD into memory would get garbled output

Example: Sony XCP

- The "primary" purpose of the rootkit was to modify the CD driver in Windows so that any process that tried to read the contents of an XCP-protected CD into memory would get garbled output
- The "secondary" purpose was to make itself hard to find and uninstall
  - Hid all files and processe whose names started with `$sys$`
  - Piggy-backed by other malware to hide their existence as well!

Example: Sony XCP

- The "primary" purpose of the rootkit was to modify the CD driver in Windows so that any process that tried to read the contents of an XCP-protected CD into memory would get garbled output
- The "secondary" purpose was to make itself hard to find and uninstall
  - Hid all files and processe whose names started with `$sys$`
  - Piggy-backed by other malware to hide their existence as well!
- After people complained, Sony eventually released an uninstaller
  - But running the uninstaller left a back door on your system!
  - The software installs an ActiveX component which allows any Web site to run software on the user's computer without restriction

Keylogger

Almost all of the information flow from you (the user) to your computer (or beyond, to the Internet) is via the keyboard.

An attacker might install a keyboard logger on your computer to keep a record of:

- All email / instant messages you send
- All passwords you type
- All Google / GPT queries you make
- ...

This data can then be accessed locally, or it might be sent to a remote machine over the Internet

Kinds of keyboard loggers

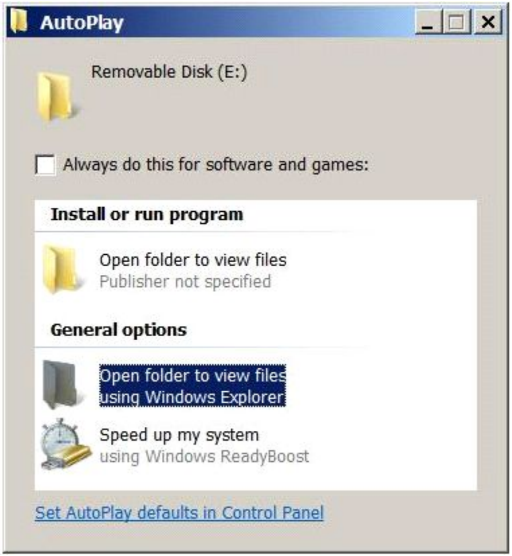
- Application-specific loggers:
  - Record only those keystrokes associated with a particular application, such as an IM client
- System keyboard loggers:
  - Record all keystrokes that are pressed (maybe only for one particular target user)
- Hardware keyboard loggers:
  - A small piece of hardware that sits between the keyboard and the computer
    - Works with any OS
    - Completely undetectable in software

Interface illusions

You use user interfaces to control your computer all the time.

- For example, you drag on a scroll bar to see offscreen portions of a document.
- But what if that scrollbar isn't really a scrollbar?
- What if dragging on that "scrollbar" really dragged a program (from a malicious website) into your "Startup" folder (in addition to scrolling the document)?

Interface illusion by the 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

Phishing

Phishing is an example of an interface illusion.

- E.g., paypal.com vs paypaI.com

Phishing

Phishing is an example of an interface illusion.

- E.g., paypal.com vs paypaI.com
- E.g., `paypal.com` vs `paypaI.com`
- It looks like you're visiting Paypal's website, but you're really not.
  - If you type in your password, you've just given it to an attacker

Phishing

Phishing is an example of an interface illusion.

- E.g., paypal.com vs paypaI.com
- E.g., `paypal.com` vs `paypaI.com`
- It looks like you're visiting Paypal's website, but you're really not.
  - If you type in your password, you've just given it to an attacker
- E.g., Cyrillic 'O' vs Alphabetic 'O' are distinct yet identical-looking unicode chars

Outline

Spotting malware

- When should we look for malware?
  - As files are added to your system
    - Via portable media
    - Via a network channel
  - From time to time, scan the entire state of the computer
    - To catch anything we might have missed on its way in
    - But of course, any damage the malware have done may not be reversible

Spotting malware

- When should we look for malware?
  - As files are added to your system
    - Via portable media
    - Via a network channel
  - From time to time, scan the entire state of the computer
    - To catch anything we might have missed on its way in
    - But of course, any damage the malware have done may not be reversible
- How do we look for malware?
  - Signature-based protection
  - Behaviour-based protection

Signature-based protection

- Keep a list of all known malware
- For each malware in the list, store some characteristic feature (the signature of the malware)
  - Most signature-based system use features of the malware code itself
    - The infection code
    - The payload code
  - Can also try to identify other characteristics of a particular malware
    - Where on the system it tries to hide itself
    - How it propagates from one place to another

Polymorphism

To try to evade signature-based malware scanners, some malware are polymorphic.

- This means that instead of making perfect copies of itself every time it infects a new file or host, it makes a modified copy instead.
- This is often done by having most of the malware code encrypted
  - The malware starts with a decryption routine which decrypts the rest of the payload, which is then executed
  - When the malware spreads, it encrypts the new copy with a newly chosen random key

Polymorphism

To try to evade signature-based malware scanners, some malware are polymorphic.

- This means that instead of making perfect copies of itself every time it infects a new file or host, it makes a modified copy instead.
- This is often done by having most of the malware code encrypted
    - The malware starts with a decryption routine which decrypts the rest of the payload, which is then executed
    - When the malware spreads, it encrypts the new copy with a newly chosen random key

Q: How would you scan for polymorphic malware?

Polymorphism

To try to evade signature-based malware scanners, some malware are polymorphic.

- This means that instead of making perfect copies of itself every time it infects a new file or host, it makes a modified copy instead.
- This is often done by having most of the malware code encrypted
  - The malware starts with a decryption routine which decrypts the rest of the payload, which is then executed
  - When the malware spreads, it encrypts the new copy with a newly chosen random key

Q: How would you scan for polymorphic malware?

A: A couple of heuristics

- Look for a part of the malware that doesn't change, e.g., the decryption routine.
- Substitute equivalent instructions instead: e.g., `x=0` ↔ `x=x-x`, `x=x*0`, `x=x mod x`, `x=x xor x`.

Behaviour-based protection

- Signature-based protection systems have a major limitation
  - You can only scan for viruses that are in the list!
  - But there are brand-new viruses identified every day
  - What can we do?

Behaviour-based protection

- Signature-based protection systems have a major limitation
  - You can only scan for viruses that are in the list!
  - But there are brand-new viruses identified every day
  - What can we do?
- Behaviour-based systems look for suspicious patterns of behaviour, rather than for specific code fragments
  - Some systems run suspicious code in a sandbox first

A collection of evasion techniques

Check this encyclopedia by Check Point Research.

$\langle$ **End** $\rangle$