Computer and Network Security

Lecture 13:
Malware and Common Attacks pt2

COMP-5370/6370
Fall 2025


IF YOU DON'T WANT MALWARE
DON'T INSTALL MALWARE
imgflip.com

Malware Distribution

Malware is distributed via almost every imaginable technique and vector.

- Installed via Exploitation

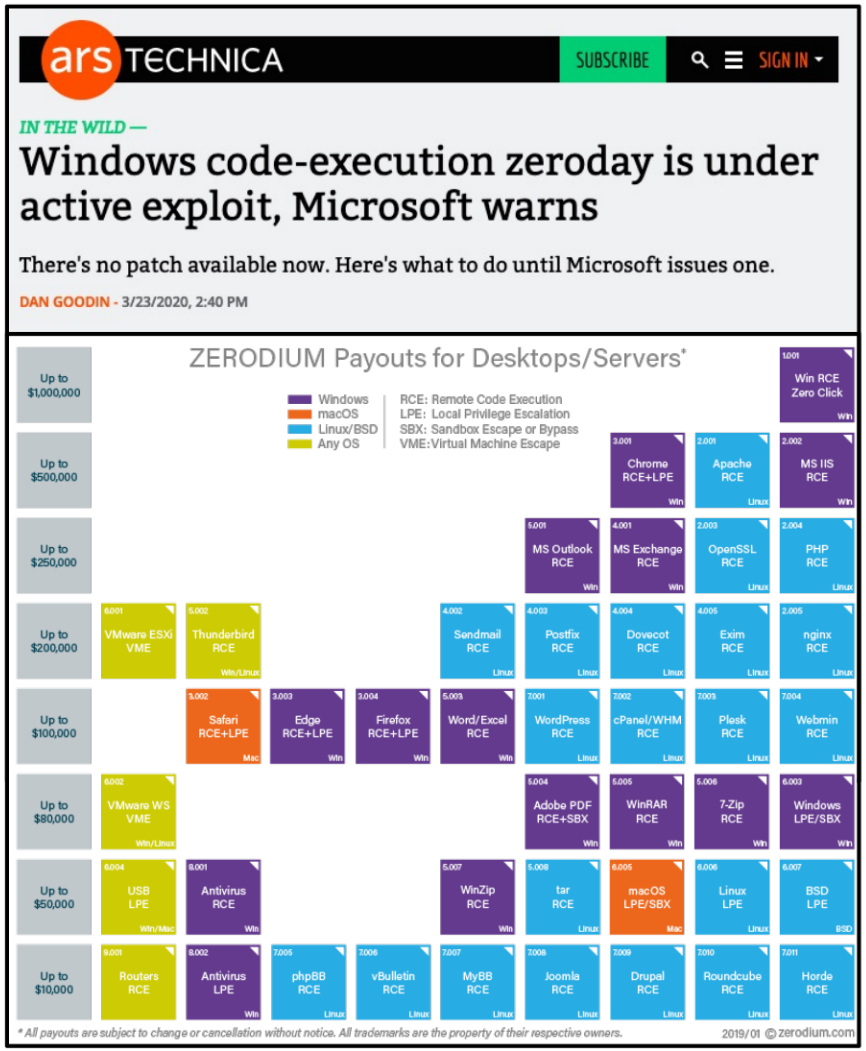
Control Flow Hijacking

Control flow hijacking is when the attack gains the ability to maliciously influence the program's execution path.

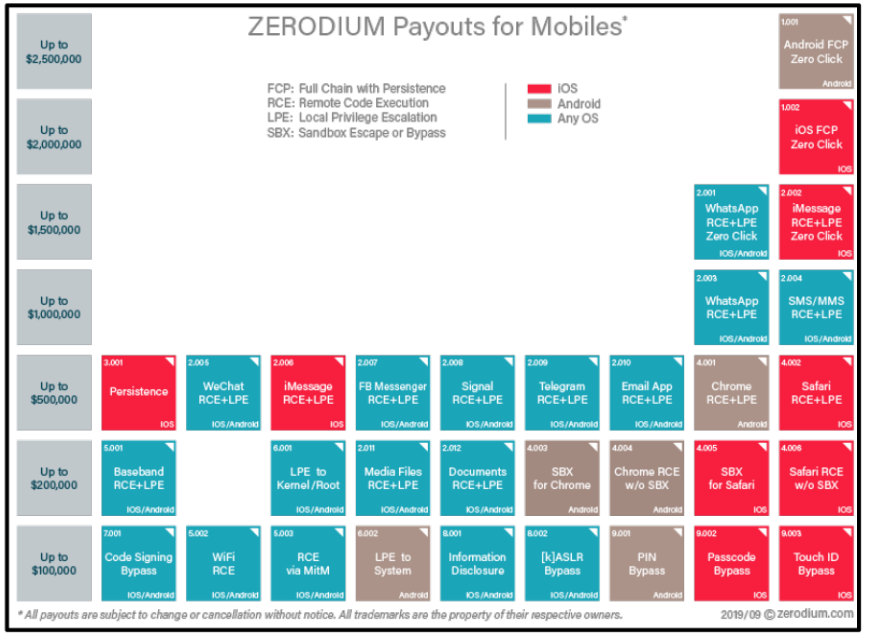
- End-goal of most binary exploitation attacks and technique

If you control EIP, you control the world.

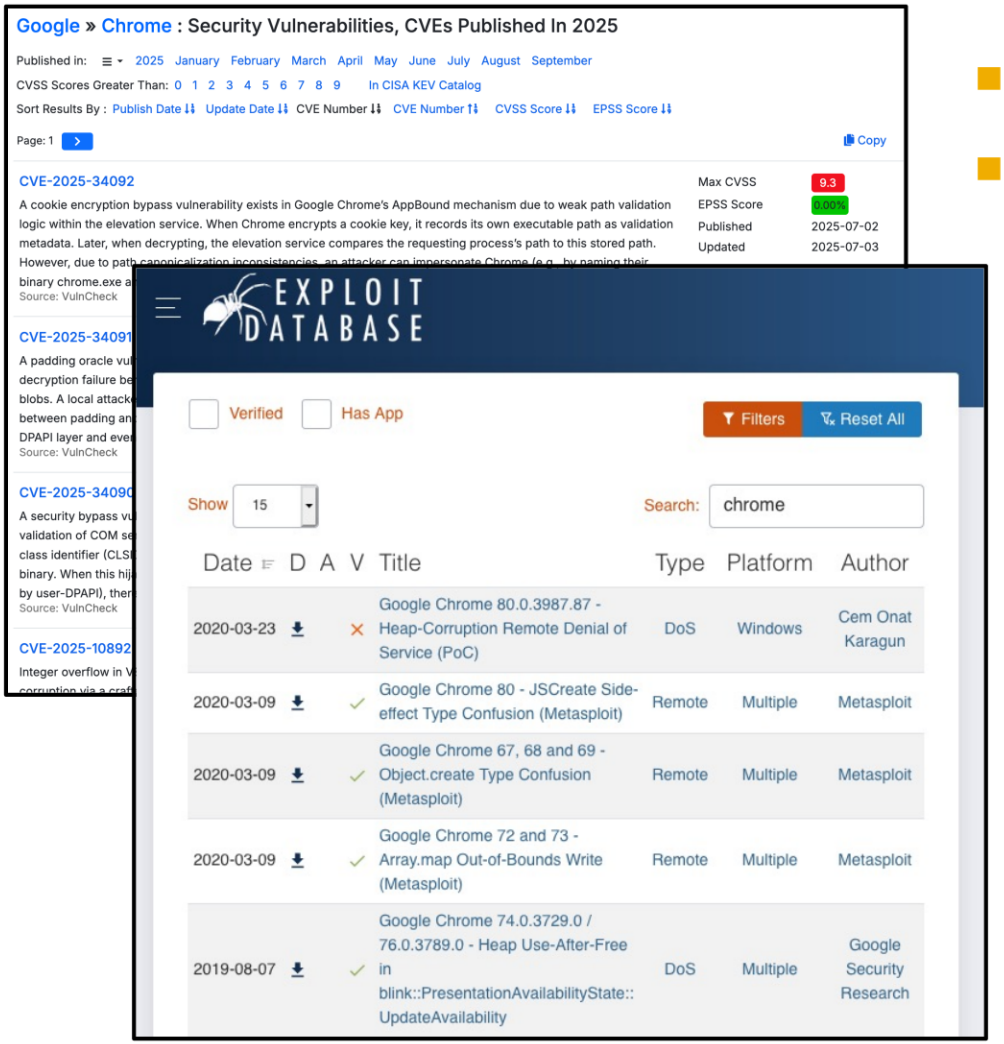
Install via Exploitation



- 0-Day Vulnerability
  - Brand-new to vendor, defenders, and users
  - Find, exploit, & install

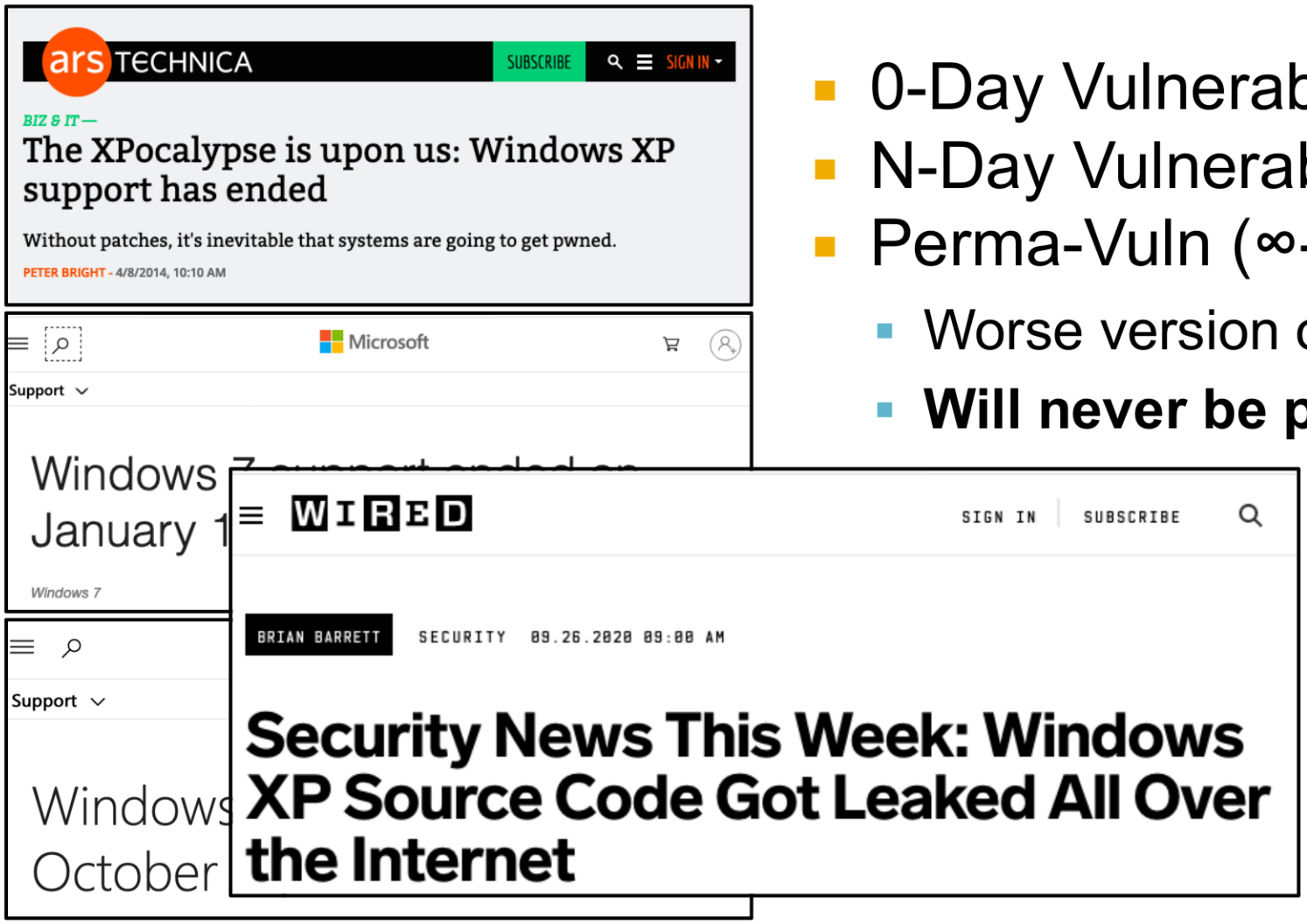


Install via Exploitation



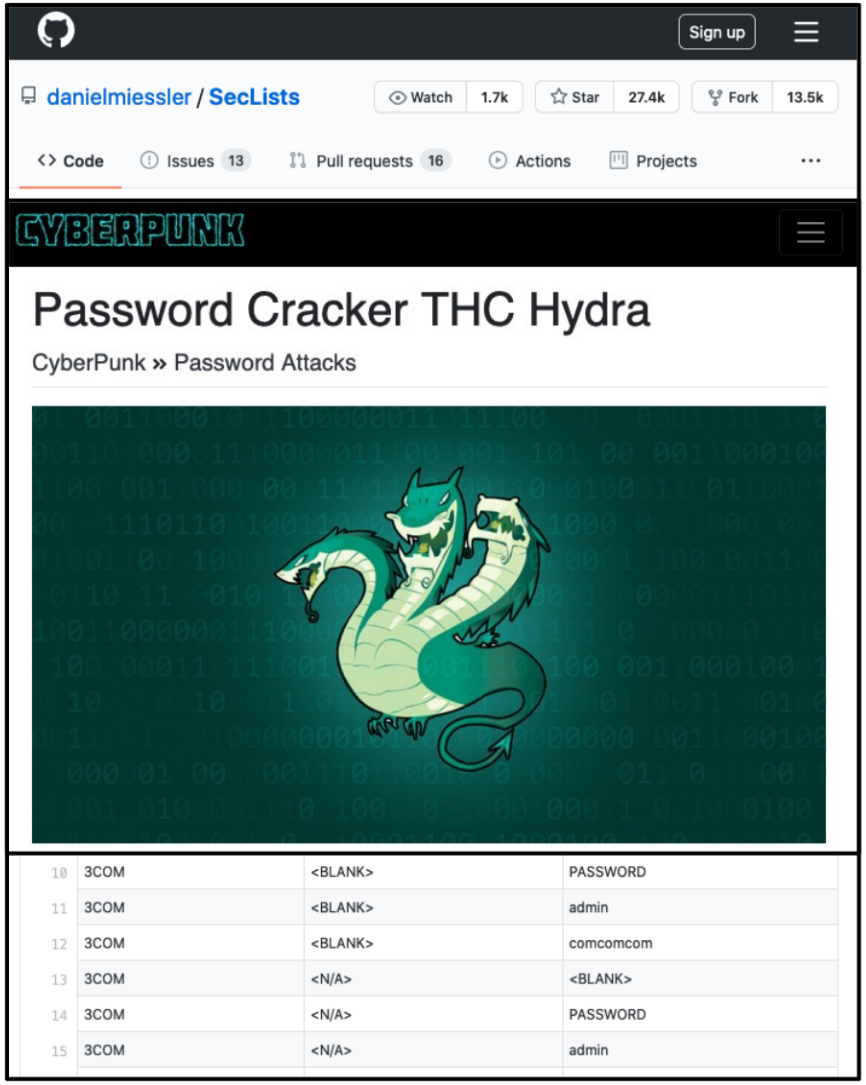
- 0-Day Vulnerability
- N-Day Vulnerability
  - Patch exists but is not applied to host
  - Old =!= Ineffective
  - Google, exploit, & install

Install via Exploitation



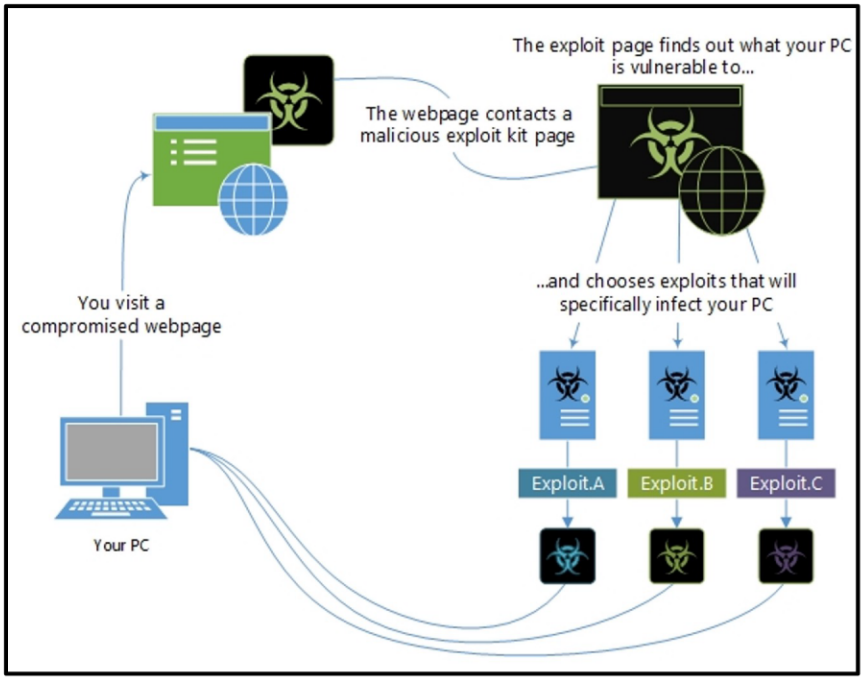
- 0-Day Vulnerability
- N-Day Vulnerability
- Perma-Vuln (∞-day)
  - Worse version of n-day
  - **Will never be patched**

Install via Exploitation



- 0-Day Vulnerability
- N-Day Vulnerability
- Perma-Vuln (∞-day)
- Password Guessing
  - **Default creds are bad**
  - Repeatedly try until successful or blocked
  - Gain access & install

Install via Exploitation



- 0-Day Vulnerability
- N-Day Vulnerability
- Perma-Vuln (∞-day)
- Password Guessing
- Drive-by-Download
  - Clients are largely not arbitrarily accessible
  - Get client to interact w/ attacker & hijack
  - Get interaction, profile, select, & install

Malware Distribution

Malware is distributed via almost every imaginable technique and vector.

- Installed via Exploitation
- Installed via Third-Party

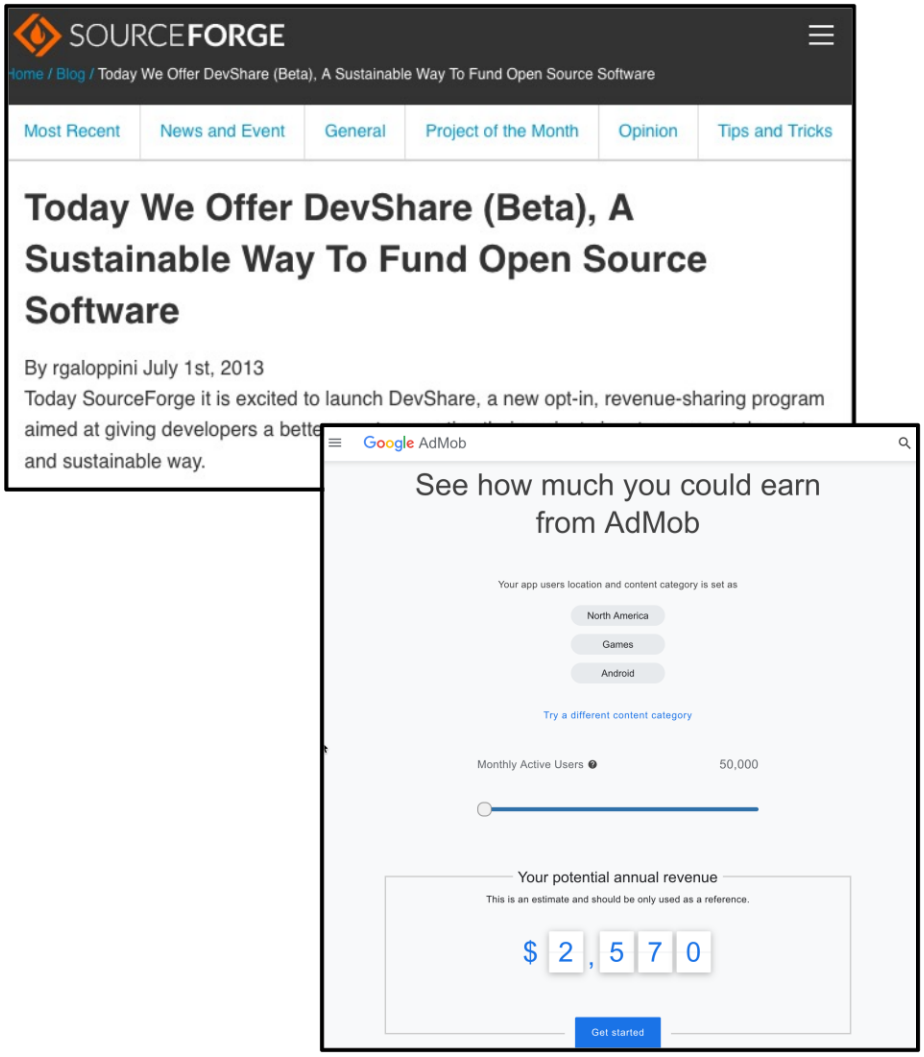
Installed via Third-Party

- Added to benign SW by developer
  - Leverage existing userbase & trust

Installed via Third-Party

- Added to benign SW by developer
  - Leverage existing userbase & trust
  - Change of control

Installed via Third-Party

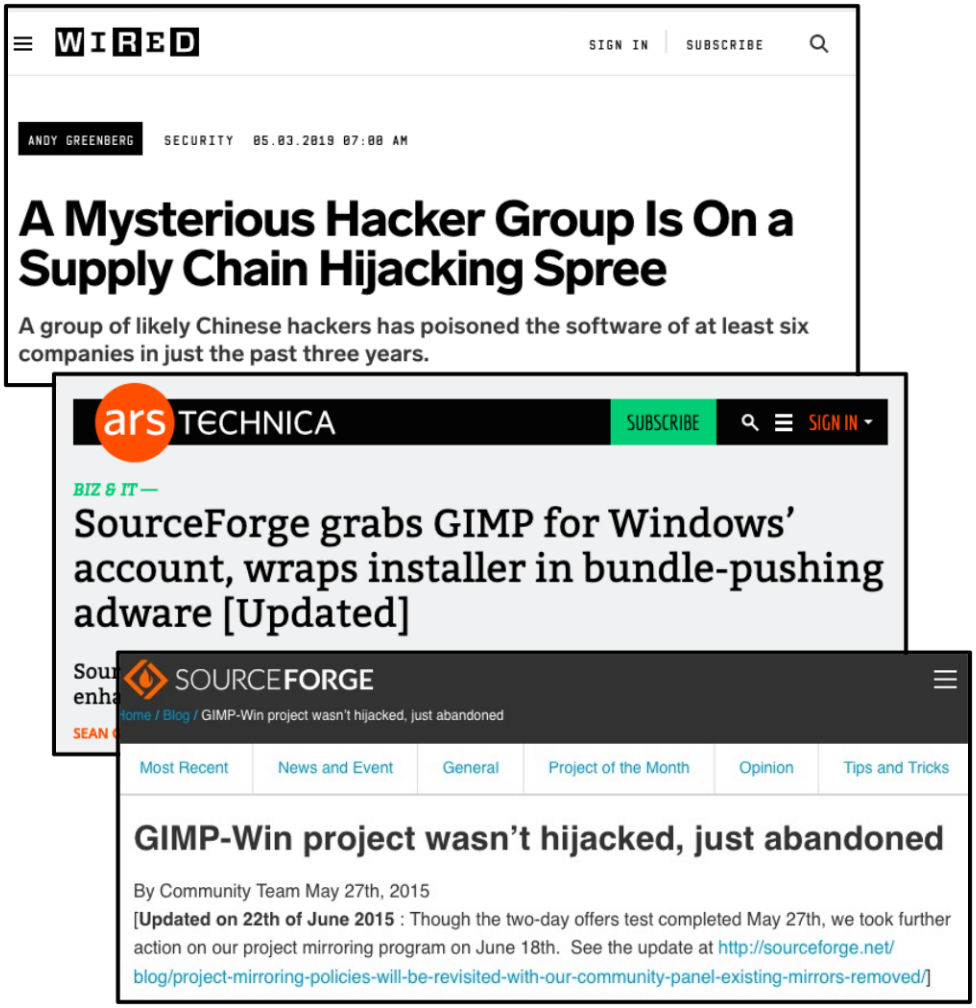


- Added to benign SW by developer
  - Leverage existing userbase & trust
  - Change of control
  - Explicitly for revenue

Installed via Third-Party

- Added to benign SW by developer
- Added by attacker via supply-chain access
  - Unknown to developer
  - External dependencies

Installed via Third-Party

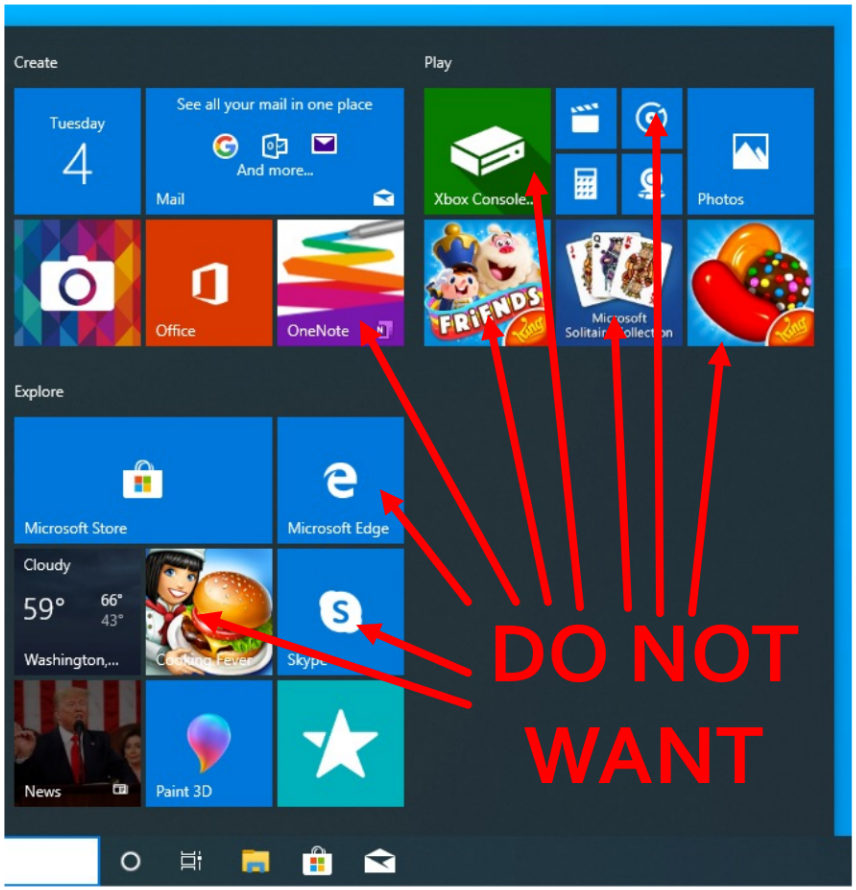


- Added to benign SW by developer
- Added by attacker via supply-chain access
  - Unknown to developer
  - External dependencies
  - Distribution mechanism

Installed via Third-Party

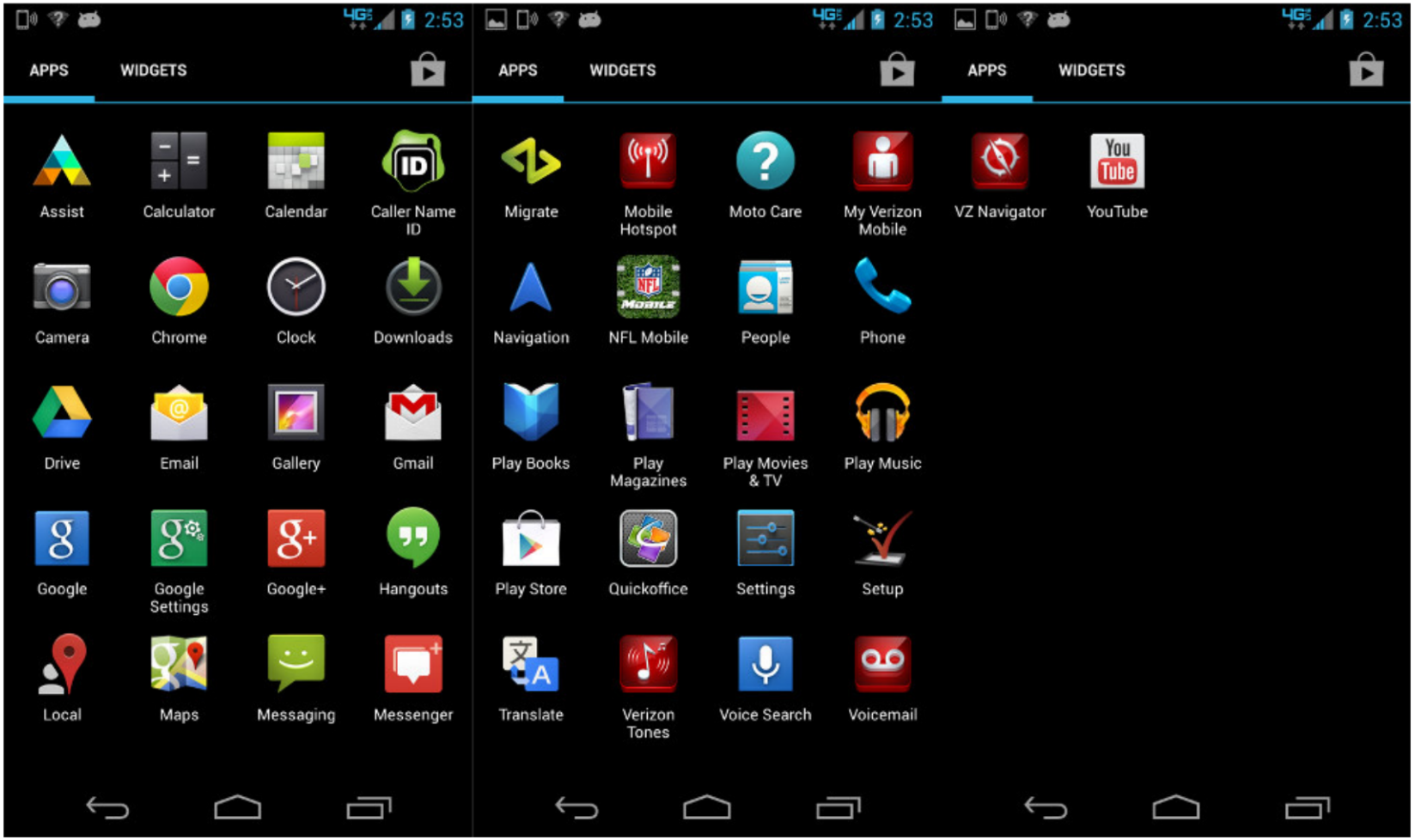
- Added to benign SW by developer
- Added by attacker via supply-chain access
- "Owner" adds for compliance/policy
  - Bossware, mobile device management (MDM)

Installed via Third-Party

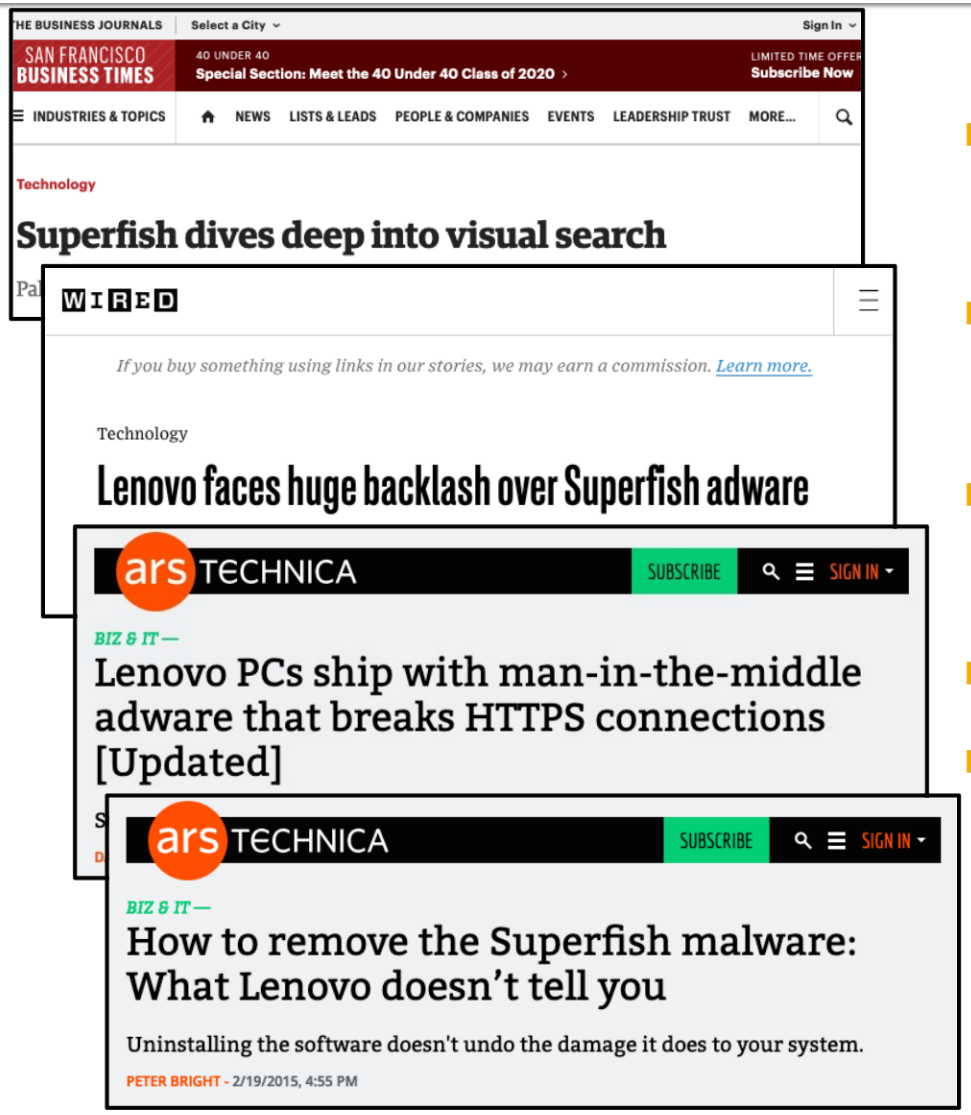


- Added to benign SW by developer
- Added by attacker via supply-chain access
- “Owner” adds for compliance/policy
- Preinstalled
  - May not be obvious
  - May be force-installed

Pre-Installed Malware



The Story of Superfish



- Came pre-installed on Lenovo laptops
- Was an ad-supported visual search startup
- Actively MitM traffic for ad injection
- Injected root CA
- **SAME PRIVATE KEY ON EVERY SINGLE INSTALL**

The Story of Superfish

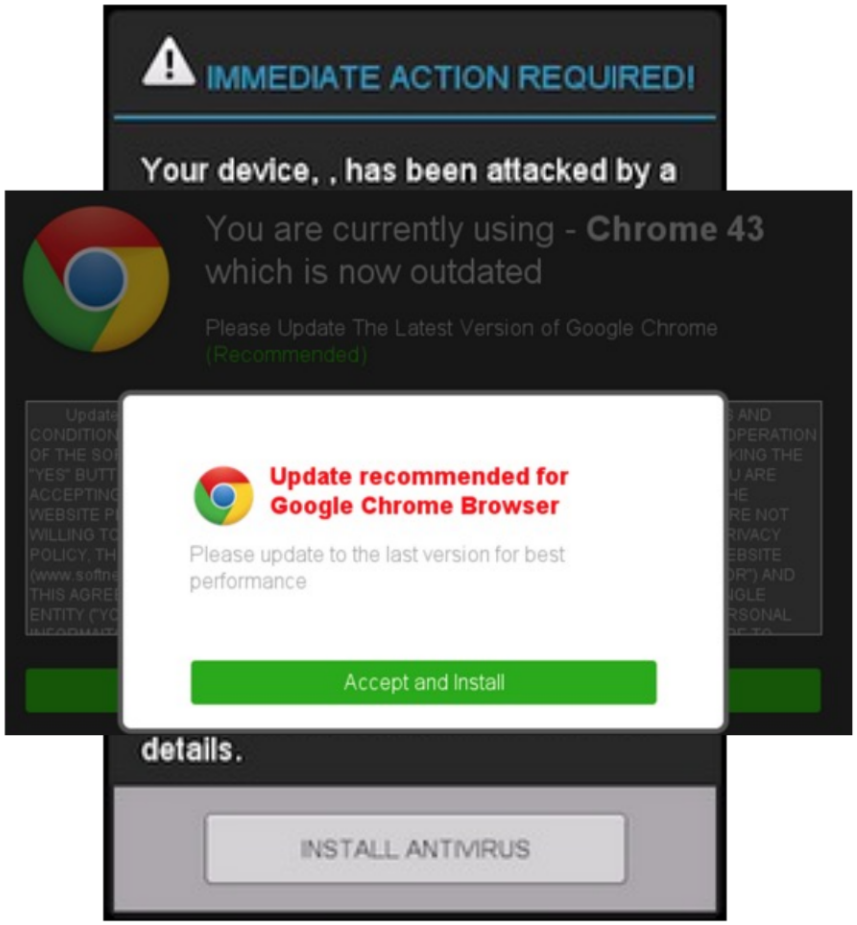
nstalled
aptops
supported
h startup
M traffic
ion
t CA
VATE KEY
SINGLE

Malware Distribution

Malware is distributed via almost every imaginable technique and vector.

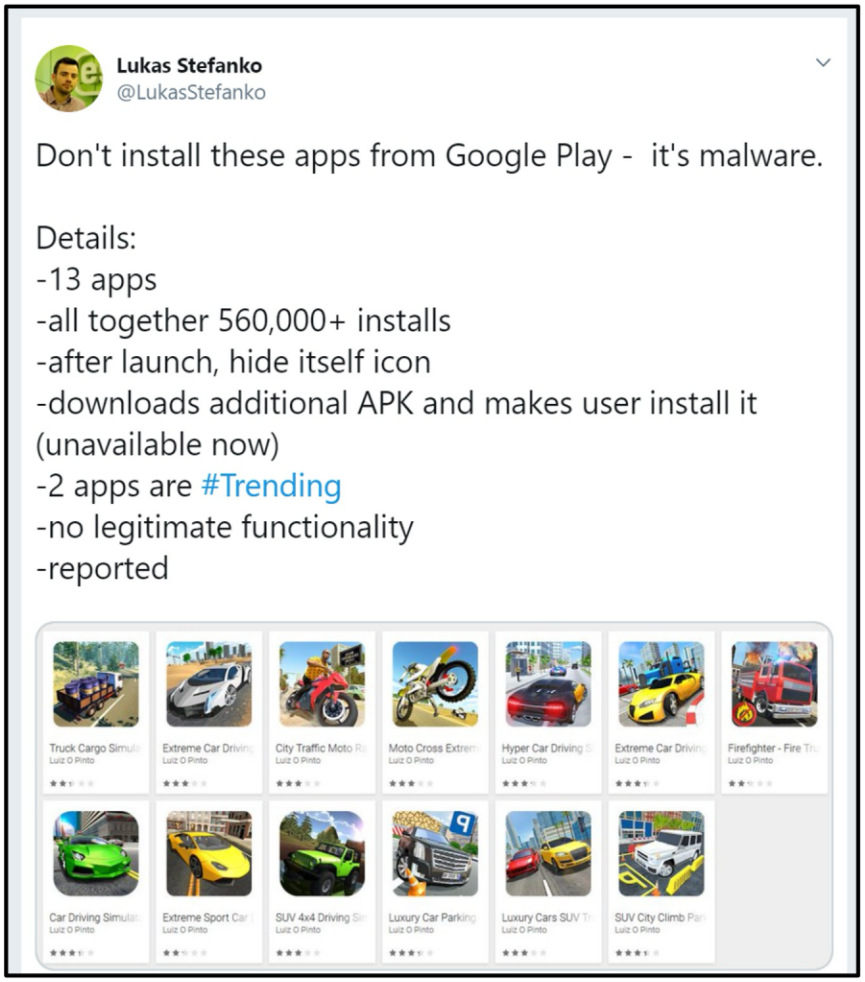
- Installed via Exploitation
- Installed via Third-Party
- Installed via User

Installed via User



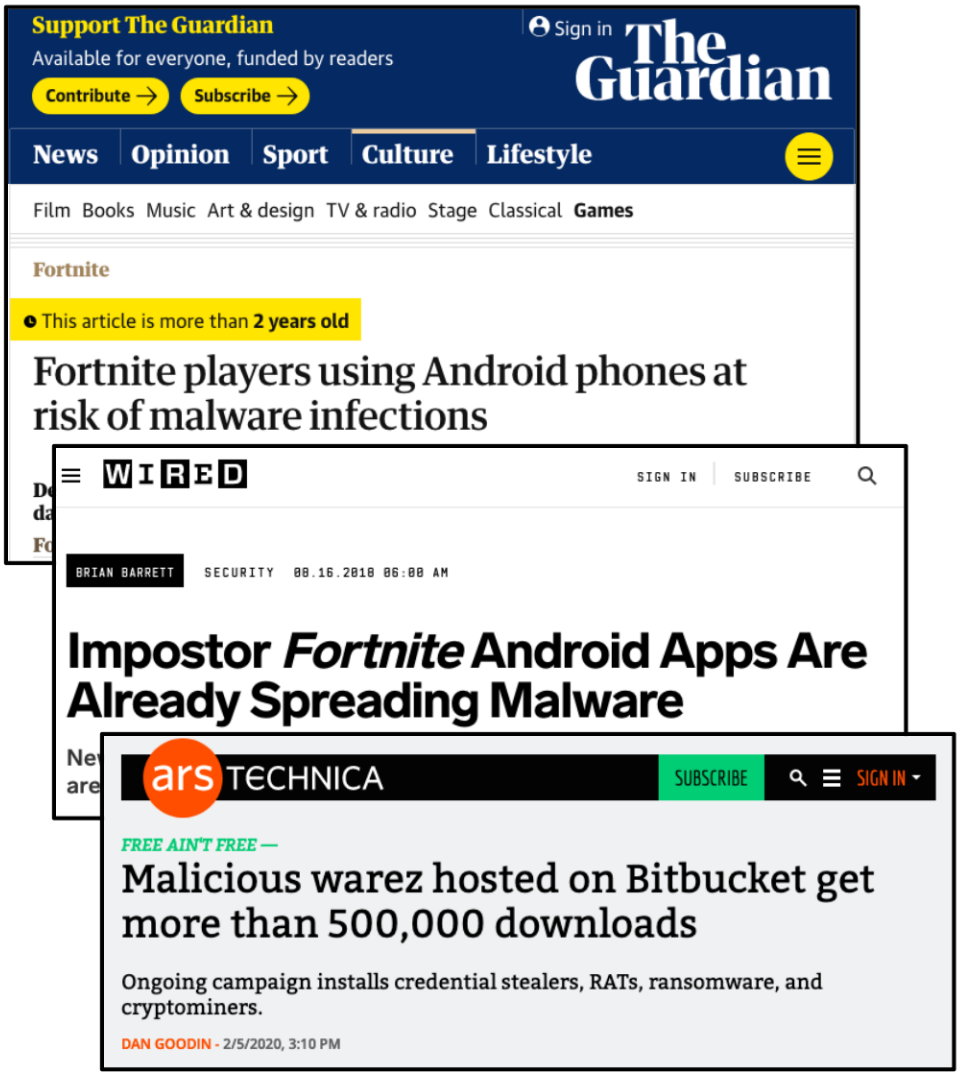
- Social Engineering
  - User is tricked into installing themselves
  - Can be last-resort of drive-by-download

Installed via User



- Social Engineering
- Freeware/Shareware
  - Cheap, low-effort applications as bait
  - Packed w/ arbitrary libs
  - *If you can't figure out what the product is… it's probably you.*

Installed via User



- Social Engineering
- Freeware/Shareware
- Untrusted sources
  - Even if it works 100% the same, it's most likely not
  - **SIDE-LOAD-ing APKs is extremely dangerous**

Changes Coming to Android Side-Loading

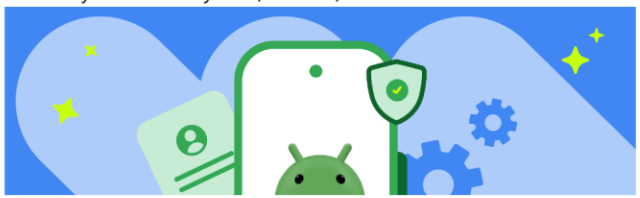
Android Developers Blog

25 August 2025

A new layer of security for certified Android devices

Share

Posted by Suzanne Frey – VP, Product, Trust & Growth for Android



You shouldn't have to choose between open and secure. By engineering security into the core part of the OS, Android has proven that you can have both, and we continue taking new steps in that direction.

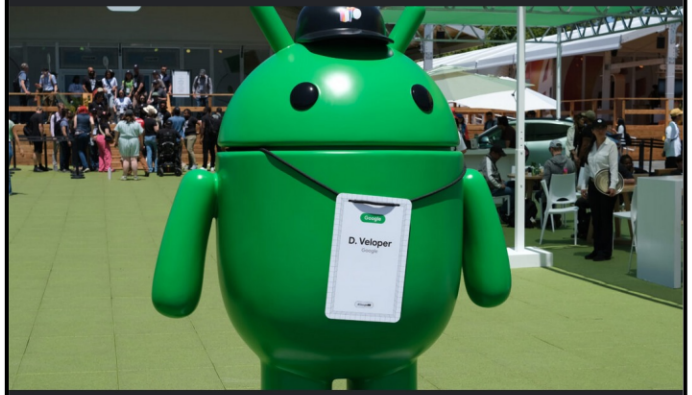
ars TECHNICA SIGN IN

PAPERS, PLEASE

Google will block sideloading of unverified Android apps starting next year

Google says it's no different than checking IDs at the airport.

RYAN WHITWAM – AUG 25, 2025 3:52 PM | 375

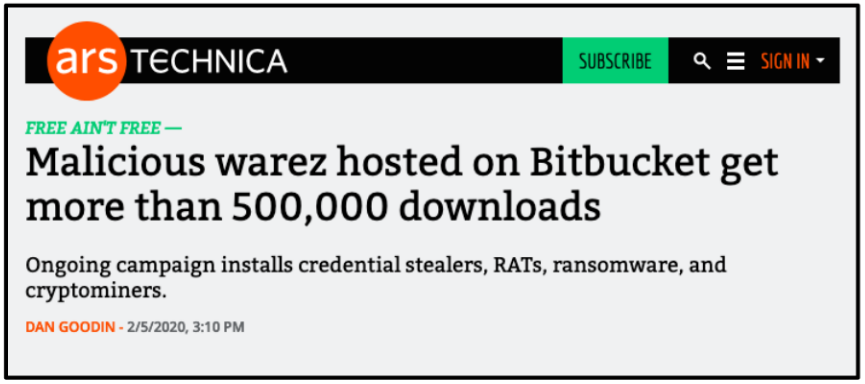


Credit: Ryan Whitwam

TEXT SETTINGS

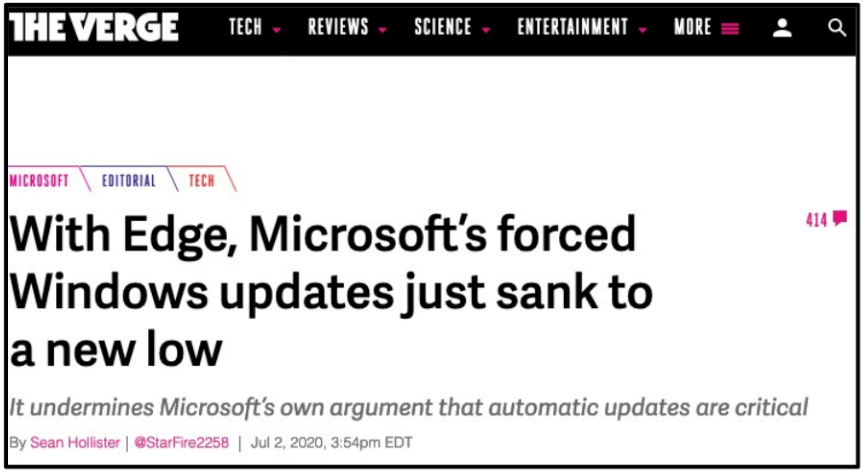
Android's open nature set it apart from the iPhone as the era of touchscreen smartphones began nearly two decades ago. Little by little, Google has traded some of that openness for security, and its next security initiative could make the biggest concessions yet in

Installed via User



- Social Engineering
- Freeware/Shareware
- Untrusted sources
- "Cracked" software
  - Promise free-version of paid software
  - Often actually are "key-hacked" version
  - WaReZ, Torrents, P2P

Installed via User



- Social Engineering
- Freeware/Shareware
- Untrusted sources
- "Cracked" software
- "Bundled" software
  - Installs the software you want to install
  - Also install its friends

Malware Distribution

Malware is distributed via almost every imaginable technique and vector.

- Installed via Exploitation
- Installed via Third-Party
- Installed via User
- ……..

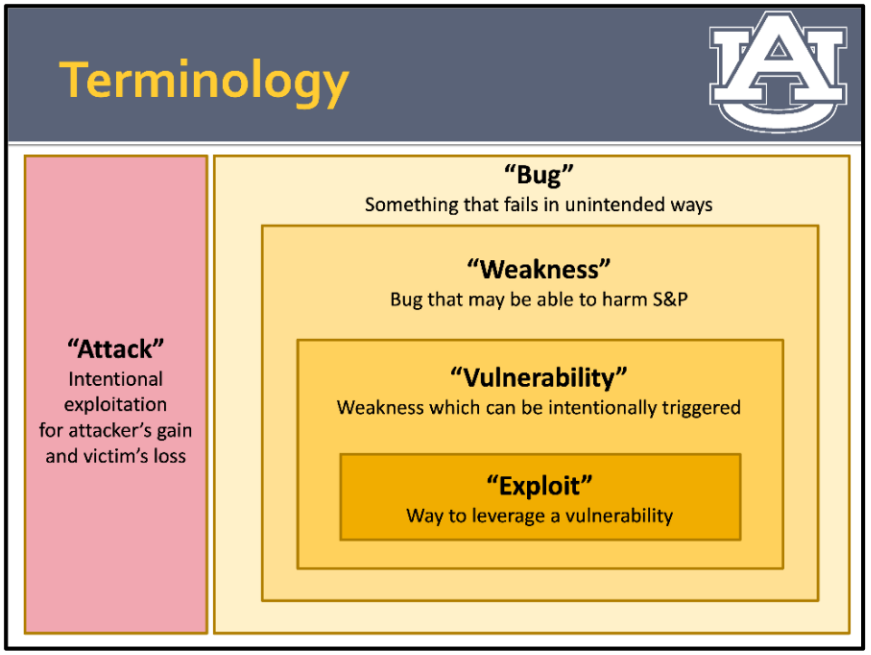
Computer and Network Security

Lecture 13: OS Security & Isolation

COMP-5370/6370
Fall 2025

Software Facts of Life

- Software has bugs
- Some bugs are weaknesses
- Some weaknesses are vulnerabilities
- Some vulnerabilities can be exploited
- Someone has an interest in exploiting others for gain



Do you trust other people's code?

- Linux kernel: 27.8M lines of C
  - Pointer math, raw byte buffers, etc.
  - C99 compliant code lacks features like "smart pointers", threads, implicit bounds-checking,…
  - Written, reviewed, and tested by those strange and mythical people known as "kernel devs"

Do you trust other people's code?

- Do you use an HP computer?
- Android OS or Google's Android Apps?
- Google services (GMail, Docs, Calendar, etc)?

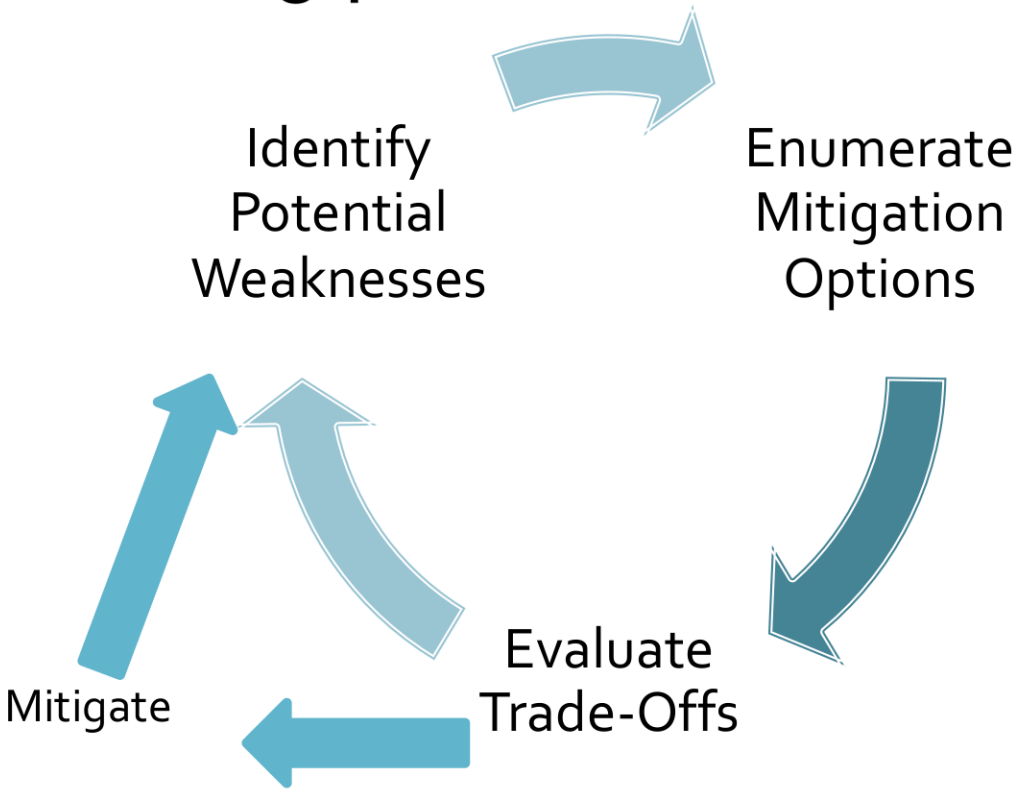
Finster Professor

Shady Professor

Profesor Turbio

Threat Modeling

A systematic approach to analyzing and understanding potential weaknesses.



Principle of Least Privilege

The **Principle of Least Privilege** is that access to data & resources is limited those who need for routine, authorized purposes.

- "Specific users can only run specific apps"
- "Normal users aren't root-users"
- "routine" means **ROUTINE**
  - Having continuous ability to do something that is only needed every 3 years is … less-than-ideal

Principle of Complete Mediation

The **Principle of Complete Mediation** is having a trusted entity validate any privilege use to ensure its validity.

- OS validates user X can run app Y
- OS validates that app Y is allowed to access resource Z

Applying Principles

- **Security Model** – An abstraction to *subjects*, *permissions*, and *objects* to allow reasoning about S&P properties.
- **Security Policy** – The mapping of *subjects,* permissions, and *objects* to implement the security model.
- **Security Mechanism** – The technical measure that enforces the security policy.

Access Control Model

Formal Models

- Discretionary Access Control (DAC)
- Mandatory Access Control (MAC)
- Role-based Access Control (RBAC)
- Bell-LaPadula
- <many, many more>

The real-world is a mixture of all.

Filesystem Example

Subjects

Users

Groups

Permissions

Read

Write

Execute

Objects

"Files"(data, binary, device)

Hierarchies (multiple files)

Access Control Policy

Create a logical representation of who has access to what file.

- Enumerate users
- Enumerate files
- Give specific users specific access to specific files
- Default: no-access

		Objects			
		File 1	File 2	File 3	File 4
Subjects	Alice	read	read/ write	no access	no access
	Bob	read	read/ write	no access	no access
	Carol	read	write	read/ write	read/ write
	Wendy	read/ write	read/ write	read	read

During Planning

- Create a system that is defendable

Subject Tracking Mechanism

Implement the policy in the real-world.

- Users assigned **user id**
- Users can **masquerade** as other users
- Users assigned to groups for simplicity (logic/mgmt)
- Groups are **meta-users** in some but not all aspects

```
user@desktop:~$ whoami; id -u
user
1000
```

```
user@desktop:~$ sudo whoami
[sudo] password for user:
root
```

```
user@desktop:~$ groups
user adm faculty
```

```
user@desktop:~$ $ sudo -u adm bash -c 'hi'
Sorry, user adm is not allowed to execute '/usr/bin/bash
user@desktop:~$ $ sudo -u root bash -c 'hi'
hi
```

User Permission Tracking

- 3 permission bits per object (RWX)


```
user@desktop:~$ ls -l
d rwx rwx ---  1 user user    4096 Apr  2 15:56 go
- rw- rw- r--  1 user user       0 Apr 11 04:15 test.py
d rwx rwx ---  1 user faculty 4096 Dec 28 21:09 courses
```

Permissions | Owner | Group | Object

- Each object has an “owner” and a group
- Only owner can change the permissions or group

```
user@desktop:~$ ls -l
- rw- r-- ---  1 user faculty  302 Apr 11 04:15 main.py
user@desktop:~$ chmod 751 main.py
user@desktop:~$ chgrp adm main.py
user@desktop:~$ ls -l
- rwx r-x r-x  1 user adm      302 Apr 11 04:15 main.py
```

OS Mediation of File Access

- Who is trying to act?

```
user@desktop:~$ whoami; id -u
user
1000
```

- What are they trying to act on?

```
user@desktop:~$ ls -l
d rwx rwx ---  1 user user    4096 Apr  2 15:56 go
- rw- rw- r--  1 user user       0 Apr 11 04:15 test.py
d rwx rwx ---  1 user faculty 4096 Dec 28 21:09 courses
```

- What are they trying to do?

- Allow action or not?

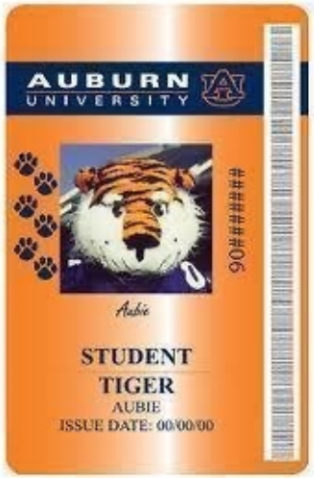
rwx	rwx	rwx
Owner	Group	Others

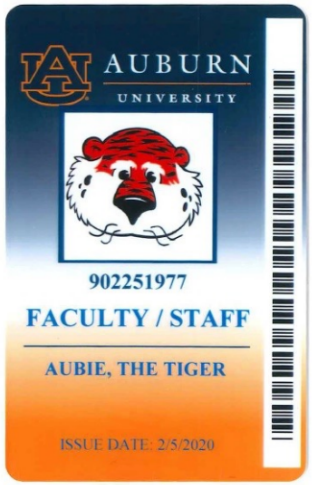
Processes as Subjects

Processes permissions are **logically identical** but slightly different in details.

- Process inherits user permissions (default)
  - Effective User ID (EUID)
  - Effective Group ID (GUID)
- EUID/GUID can be set manually:
  - `sudo`, `setuid`, `sg`, …
  - Requires root user

During Operation





- Create a system that is defendable
- Maintain the mechanisms for defending it

Do you trust other people's code?

- Do you use an HP computer?
  - Android OS or Google's Android Apps?
  - Google services (GMail, Docs, Calendar, etc)?

Finster Professor

Shady Professor

Profesor Turbio

Computer and Network Security

Lecture 13: OS Security & Isolation

COMP-5370/6370
Fall 2025

Course Notes

- Project 1A
  - Grades posted tonight
  - Remember that 1A is **HALF** of Project 1
- Project 1B
  - Finishing-up this week, expect Friday
- Exam 1
  - Grades & Scans posted tonight
- Midterm
  - On Tuesday, 07Oct2025 (1 week from today)