BTI 4202: Threat Landscape

Christian Grothoff

BFH

23.4.2021
Cyber attacks and actors

Software vulnerabilities
Part I: Cyber Attacks and Actors
Attacker origins

- Insider
- Ex-insider ("disgruntled former employee")
- Competitor
- Hacktivist
- Criminal
- State actor
- Researcher
Attacker objectives

- Fame
- Stealing information (business secrets, credentials)
- Modifying information (e.g. bank transactions)
- Abusing infected systems (e.g. spamming)
- Attacking other systems (origin obfuscation)
- Hiding (avoid detection, achieve long-term persistence)
- Contact command and control (C2) for instructions
Vulnerability origins

- Hardware (host, network)
- Software (host, network)
- Humans
- Environment
Attack strategies

- Large scale attack: attack a large, untargeted population. Even if the success rate is low, the absolute number of infections and the resulting revenue can be high. (“cyber crime”)
- Targeted attack: attack a few, selected users or their machines. Select high-value target first, then learn about it as much as possible for a precision strike (“Advanced persistent threat”)
Defense strategies

▶ Access control (physical, logical)
▶ Deterrence (legal, counter-attacks, auditing, accounting)
▶ Redundancy
▶ Obfuscation
▶ Comprehension (simplification, transparency, education)
▶ Monkey wrench / havoc
▶ Defense-in-depth
Part II: Software vulnerabilities
Technical vulnerabilities

There are many types of technical vulnerabilities in various parts of an IT system:
- Misconfigured firewalls
- Hardware bugs
- Automatically executed software from CD/USB stick on old W32 systems
- etc.

The probably most important class of technical vulnerabilities are software bugs.
Typical bugs

Software is often used to display data obtained over the network:

1. User downloads file (PDF, MP4, etc.)
2. User selects software to open file
3. Software parses file
4. Bug ⇒ malicious code execution

Common bugs include problems in the parsing or rendering logic, or scripting functionality supported by the document format in combination with an interpreter that is insufficiently sandboxed.
The central goal for an attack is to turn data into code. Memory of a process contains data and code! Thus:

- Existing code may interpret the data (intentionally or unintentionally), thereby allowing certain code sequences to be executed.
- Existing code may be caused to jump to the data (once data page is set to executable).
- Execution may be passed to another program (shell, interpreter) that will parse and run it.
Example exploit: SQL injection

In a PHP script, hopefully far, far away:

```
SELECT (user, first_name, last_name)
FROM students
WHERE (user == '$user');
```

Input:

```
Robert'); DROP TABLE students;--
```
HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.

OH, DEAR - DID HE BREAK SOMETHING? IN A WAY-

DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?

OH, YES. LITTLE BOBBY TABLES, WE CALL HIM.

WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.

AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.
Vulnerability timeline

- Vulnerability introduced
- Exploit released in the wild
- Vulnerability discovered by the vendor
- Anti-virus signatures released
- Patch released
- Patch deployment completed

Zero day attack
Follow-on attacks
Window of exposure
Capitalism

ZERODIUM Payout Ranges*

LPE: Local Privilege Escalation
MTB: Mitigation Bypass
RCE: Remote Code Execution
RJB: Remote Jailbreak
SBX: Sandbox Escape
VME: Virtual Machine Escape

* All payout amounts are chosen at the discretion of ZERODIUM and are subject to change or cancellation without notice.

2015/11 © zerodium.com