HITB SECCONF SIN-2021

Security Technology Arms Race 2021

Medal Event

Mark Dowd

Director of Research, L3Harris Trenchant

TRACK 1

The Race is On

There is an epic marathon being run between offense vs defense and we are experiencing them at their closest point

Billions invested by each side – who will win??

I have had trackside seats to the race, and will be your commentator for today ☺

The Participants

Offense vs Defense is a broad topic

- Memory corruption, SQLi, protocols, crypto, …

This focus is on offensive tooling

- Mostly concerned with memory corruption “exploit chains”

However: other security topics are highly relevant and will be mentioned as appropriate

Prospects

Looking at offensive tooling in the past, present, and future:

- Why has offense enjoyed such a large advantage?
- What strides has defense made?
- What happens next?

Thesis: The advantage offense has enjoyed for the last 20 years is inverting in favour of defense

Neck and Neck

1) Context: Offensive Advantage
2) Defensive Gains
3) Offense vs Defense: Current State

Offensive Advantage: Asymmetry

What are we defending against?

Context: Asymmetry

Offense has thrived over the last 20 years

- Vulnerabilities found with frequency
- Exploit kits consistently developed against contemporary targets

Offense started with several considerable advantages

- We will enumerate them here

Context: Asymmetry

Factor 1: Security was not a primary design concern at the outset

- Unsigned code was the default mode of operation
- Devices do not have open and attestable codebases
- We rely on legacy design decisions (often now with some level of retro-fitted workaround)

    Eg. Monolithic kernels

Context: Asymmetry

Factor 2: Security boundaries were sometimes ill defined and/or moved as technology progressed

- Lock screens
- Admin loading a kernel driver
- Windows desktops

Context: Asymmetry

Factor 3: A lot of security science was in its infancy

- Vulnerability patterns + attack strategies
- Static analysis and decompilation
- Fuzzing methodologies
- Best programming practices

Context: Asymmetry

Factor 4: Defensive ecosystem started small

- Difficulty of patch management / updating
  - Third party libraries especially!
- Difficulty of detecting (sophisticated) attacks and compromise

Context: Asymmetry

Factor 5: Technology breakthroughs sometimes undermines defensive strategies and assumptions

- Encryption and hashing breakthroughs (MD5)

Context: Asymmetry

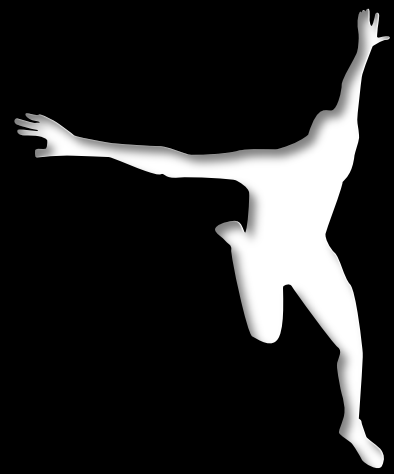
… and most importantly,

Factor 6: The starting point was millions of lines of Internet-facing code

"We found out memory corruption is bad and common, can you just look over this multi-million line code base and see if we have any problems? Also, no downtime please" – a manager

Context: Asymmetry

"Defense has to be right all the time,
offense has to be right just once"



Context: Asymmetry

The perfect storm for offense

- Motivation
- Relatively low cost
- Very complicated software with haphazard patching
- Detection was poor
- Low Risk for the attacker
    - Of getting caught
    - Consequences

Defensive Strides

Defense started slow but has accelerated

Defensive Goals

1) Raise the cost of offensive tooling
2) Limit damage of successful compromise
3) Detect successful compromise

Defensive Goals

1) **Raise the cost of offensive tooling**
2) Limit damage of successful compromise
3) Detect successful compromise

Defensive Goals: Raising Cost

Broadly divided into three parts

Discovery cost	Development cost	Maintenance cost
Time to discover flaw	Implement reliable exploit	Keeping the exploit operational

Defensive Goals: Raising Cost

Discovery cost is rarely linear

- Large start cost to find first bug, following bugs much quicker

Same with development cost

- Develop a technique or bypass, reuse

Halvar Flake discussed this concept in depth in a 2017 keynote

Thomas Dullien. (2017). *Repeated vs. single-round games in security* [Paper presentation]. BSides Switzerland, Zurich. https://bsideszh.ch/wp-content/uploads/2017/10/Thomas_Dullien-Keynote.pdf

Defensive Goals: Raising Cost

Example: Chrome RCE (~2013) with new junior employee

Found no bugs for 6 months

- Then: 4 bugs in 2 weeks

Exploit strategy took 2 months

- Then: 1 week for each of the others

This is a common pattern

Defensive Goals: Raising Cost

Major discovery costs:

- Improved vendor bug discovery
  - Example: Google has fuzzed over 29k bugs in Chrome since 2016
- Mitigations rendering vulnerabilities unexploitable
  - Mostly due to early-stage mitigations (heap isolation, etc)
  - **Estimate: Half of the bugs I've found in the past several years were unexploitable!**

Defensive Goals: Raising Cost

Major development costs:

- Code Signing
  - When done right (iOS), enormous recurring cost
  - Usually avoided at all costs (use browser wherever possible)
- ASLR (infoleaking)
  - High, recurring cost
  - Also augments other mitigations

Defensive Goals: Raising Cost

Major development costs:

- Targeted technique mitigations
  - Apple specifically does this more than anyone else
  - Example: kernel_task, zone_require, removing direct access to objects leveraged in public exploits

All of these incur a (potentially high) recurring cost

- Destroy advantage of technique reuse

Defensive Goals: Raising Cost

Maintenance as an ongoing cost is rarely discussed

- Software releases are frequent
- Keeping an exploit operational is a lot of work

"Stockpiling" is largely a myth specifically because of this cost

Defensive Goals: Raising Cost

Isolation has been the primary cause of dramatically increased cost

Multiplier effect: Each link in the chain requires a new discovery + development cost

Defensive Goals: Raising Cost

Result

Cost of exploitation has risen dramatically

and

Most players are priced out of the market

Defensive Goals: Raising Cost

Example: There is no market for stolen iPhones

(And: if you lose your iPhone, the chances of a stranger being able to unlock it are practically zero)

Even LEO might have trouble

Defensive Goals: Limit Compromise

1) Raise the cost of offensive tooling
2) **Limit damage of successful compromise**
3) Detect successful compromise

Defensive Goals: Limit Compromise

Over time, offensive capabilities

cost more

and

do less

Defensive Goals: Limit Compromise

2014

- Kernel code execution possible
  - Usermode code injection possible
    - Retrieve all data

2017

- **Kernel read/write possible**
  - Possible (but painful) to do kernel ROP
    - Usermode code injection possible
      - **Some data inaccessible**

2020

- **Kernel restricted read/write**
  - Kernel ROP largely infeasible
    - **Usermode code injection requires additional vulnerability**

Defensive Goals: Limit Compromise

Modern iPhone browser chain limitations

- Largely requires phishing (not many MITM opportunities)
- Can get limited access to kernel memory (no code execution)
- Cannot inject into other processes without PPL bypass
- Retaining access on reboot is very challenging

Defensive Goals: Detection

Defensive goals:

1) Raise the cost of offensive tooling
2) Limit damage of successful compromise
3) **Detect successful compromise**

Defensive Goals: Detection

Detection has historically been very poor

- Figure out a signature, work around it

Vendors have the advantage of scale to detect anomalies

- Microsoft Defender for Endpoint
- Google Threat Analysis Group (TAG)
- A/V vendors (Kaspersky, FireEye, CrowdStrike)

Defensive Goals: Detection

Vendors have the advantage of scale to detect anomalies

- Microsoft Defender for Endpoint
- Google Threat Analysis Group (TAG)
- A/V vendors (Kaspersky, FireEye, CrowdStrike)

Defensive Goals: Detection

Increasingly sophisticated APT kits discovered in the wild

- Initially limited to desktop and server malware
- Now, mobile malware discovered with increasing frequency

From “In the wild” database by @maddiestone

- 2018: 12 ITW bugs, all targeting Windows desktop (no Chrome)
- By 2021: 5 Android, 7 Chrome, 7 iOS

Power Inversion

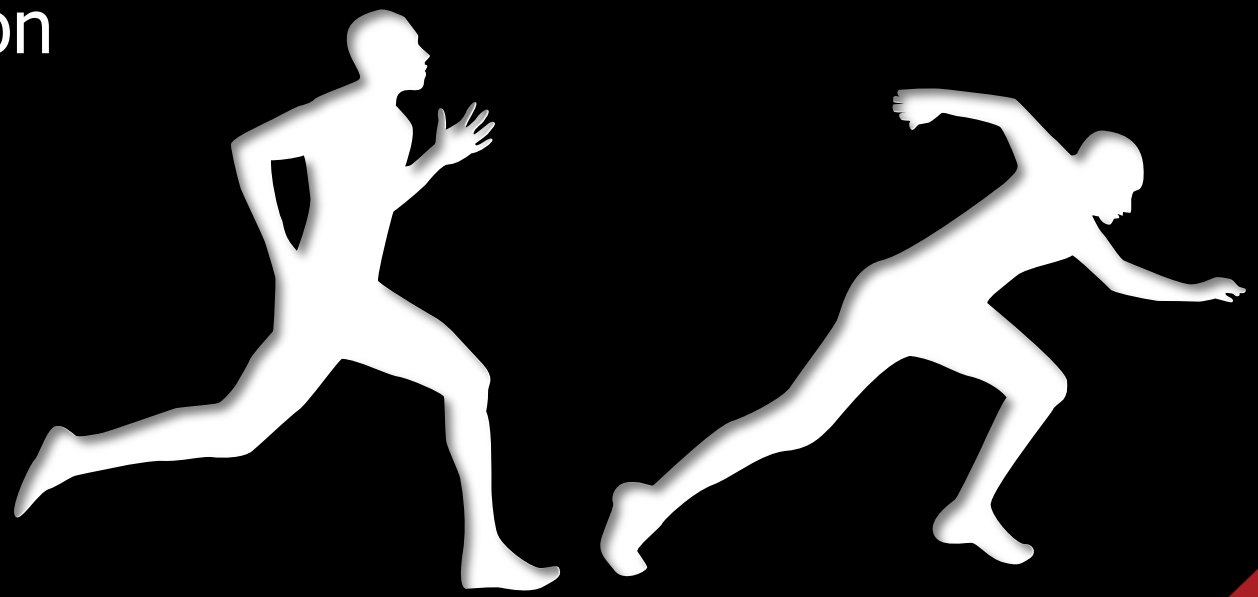
Offensive tooling slows down

Defense speeds up

Who will gain the upper hand?

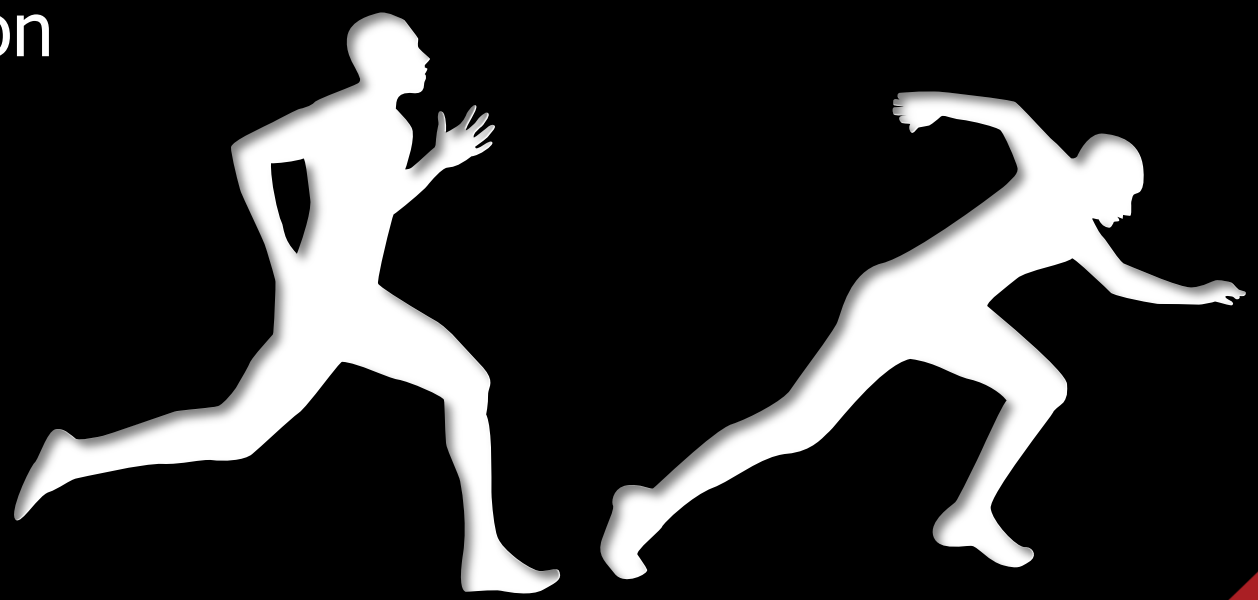
Future: Overview

1) Memory Corruption: Slowing Down
2) Beyond Memory Corruption
3) Thinking in Ecosystems



Future: Overview

1) **Memory Corruption: Slowing Down**
2) Beyond Memory Corruption
3) Thinking in Ecosystems



Future: Offense Slows Down

Defense Advantage 1: Sandboxing is effective

- Constrained to limited privileges
- Might be able to break it periodically, but not continuously

Future: Offense Slows Down

Defense Advantage 2: CFI is increasingly effective to prevent execution

- Example: iOS kernel
- Expect other kernels/usermode to follow

Future: Offense Slows Down

THEN: Structure of a typical memory corruption exploit

- Early-stage exploitation (stage 1): Initial corruption
- Late-stage exploitation (stage 2): Introduce arbitrary code
- Post-exploitation (stage 3): Payload

Future: Offense Slows Down

NOW: Structure of a typical memory corruption exploit

- Early-stage exploitation (stage 1): Initial corruption
- **Late-stage exploitation (stage 2): Get arbitrary memory read/write**
- Post-exploitation (stage 3): Payload

Future: Offense Slows Down

NOW: Structure of a typical memory corruption exploit

- Early-stage exploitation (stage 1): Initial corruption
- **Late-stage exploitation (stage 2): Get arbitrary memory read/write**
- Post-exploitation (stage 3): Payload

Data-only attacks are here, and growing

Future: Offense Slows Down

Defense Advantage 3: Data Pointer Integrity (DPI) has landed*

- iOS data PAC
- MTE will likely follow on multiple platforms

* I made up that acronym, there will probably be a better one soon

Future: Offense Slows Down

- Structure of a typical memory corruption exploit
  - **Early-stage exploitation (stage 1): Initial corruption**
  - Late-stage exploitation (stage 2): Introduce arbitrary code
  - Post-exploitation (stage 3): Payload

Future: Offense Slows Down

Most early stage mitigations are limited to 1 or 2 bug classes

- Render some vulnerabilities useless

Data PAC and MTE are game changers

- Early stage mitigation that (potentially) applies to everything
- Severely curtail most current exploit techniques
- Will take another 2-3 years to really be in full effect

Defense Accelerates: Mitigations

Possible: Microarchitectural flaws potentially an effective bypass for PAC/MTE-like mitigations

- Large upswing in side channel vulnerabilities to disclose secrets across security boundaries
  - Example: BlindSide
- Proven to be possible via browsers
  - Example: Prime+Probe 1 (w/o JavaScript!)

VUSec Systems Security Research Group at VU University Amsterdam. (2020, October 7). BlindSide. VUSec. https://www.vusec.net/projects/blindside/
Shusterman, A. et al. "Prime+Probe 1, JavaScript 0: Overcoming Browser-based Side-Channel Defenses." ArXiv abs/2103.04952 (2021): n. pag.

Defense Accelerates: Mitigations

Defensive Advantage: Detection of compromise becomes easier

- If telemetry runs with more privileges that offensive tooling can obtain, it is harder to evade

Future: Beyond Memory Corruption

1) Memory Corruption: Slowing Down
2) **Beyond Memory Corruption**
3) Thinking in Ecosystems

Defense Accelerates: Mitigations

Are memory corruption exploits on borrowed time?

- I say: Yes (but admittedly, I have been saying that since 2004)

Prediction (re-asserted): Exploitable memory corruption will become the exception, not the rule!

Future: Beyond Memory Corruption

Historically, memory corruption is the favoured technique

- Applicable to most technologies and attack vectors
- Often most powerful (unfettered access to function and data)
- Difficult to detect and stop

Is this still true?

Future: Beyond Memory Corruption

Now, memory corruption as a technique has weaknesses:

- Limited access due to CFI/CPI might mean it is **not more powerful than other types of vulnerabilities**
- Also they are the most expensive to create and maintain

What are the likely targets for offensive research to pivot to?

Future: Beyond Memory Corruption

Vulnerability Type 1: Cryptography Flaws

- Cryptography underpins nearly all current security technology in one form or another in every layer of the technology stack
  - Encrypted comms
  - Authentication / authorization (certificates)
  - Code signing and updates
  - PAC

Future: Beyond Memory Corruption

Potential to:

- Eavesdropping and payload delivery (browser/chat)
- Spoofing and impersonation
- Bypassing code signing and trusted boot
- Introducing “evil” upgrades
- Defeating some mitigations

Crypto flaws are king!

Future: Beyond Memory Corruption

Recent examples

- Orange Tsai's Exchange RCE bugs
- NSA's Microsoft Certificate vulnerability (CVE-2020-0601)
- Tom Tervoort's ZeroLogon flaw (CVE-2020-1472)

Prediction: Crypto vulnerability research will be critical components of most offensive tools in several years

Tsai, O., & Æ. (2021, August 1). *A New Attack Surface on MS Exchange Part 1 - ProxyLogon!* Orange. https://blog.orange.tw/2021/08/proxylogon-a-new-attack-surface-on-ms-exchange-part-1.html

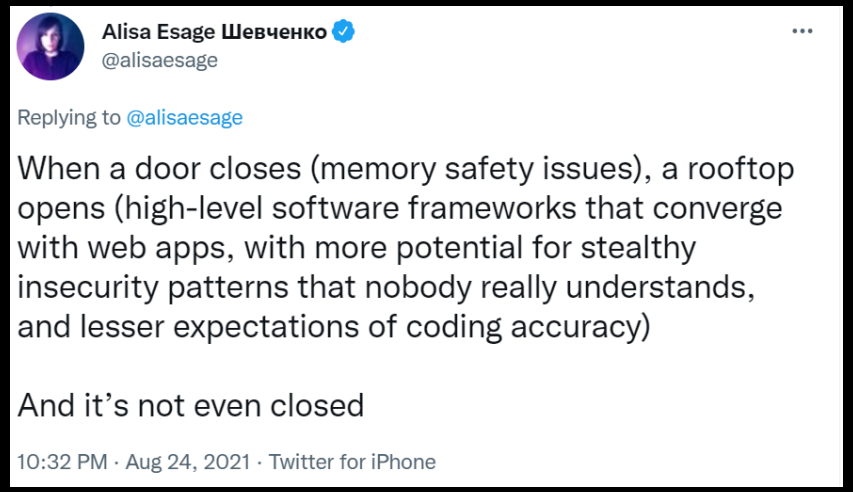
Future: Beyond Memory Corruption

Vulnerability Type 2: Web Technology Flaws

Web technology flaws are increasingly relevant in the offensive tooling space

> As noted by security experts

Future: Beyond Memory Corruption



Future: Beyond Memory Corruption

Web technology flaws are increasingly relevant in the offensive tooling space

- Exchange Server RCE via SSRF by Orange Tsai
- “Pwning iOS 14 with Generation Z Bugz” by Zhi Zhou, Jundong Xie

Tsai, O., & Æ. (2021, August 1). *A New Attack Surface on MS Exchange Part 1 - ProxyLogon!* Orange. https://blog.orange.tw/2021/08/proxylogon-a-new-attack-surface-on-ms-exchange-part-1.html
Zhi Zhou, & Jundong Xie. (2021, August). Hack Different: Pwning iOS 14 with Generation Z Bugz. Black Hat USA, Las Vegas, NV. https://i.blackhat.com/USA21/Wednesday-Handouts/us-21-Hack-Different-Pwning-IOS-14-With-Generation-Z-Bug-wp.pdf

Future: Beyond Memory Corruption

Vulnerability Type 3: Protocol Attacks

- Revisiting and examining new network protocol attacks has potential, particularly when coupled with other flaws

Future: Beyond Memory Corruption

Potential:

- Identity spoofing
- Crossing trust domains
- Accessing sensitive data or functionality

Future: Beyond Memory Corruption

We have seen data-only attacks, what about function-only attacks?

- Favourite: Natalie Silvanovich's chat app shredding

Silvanovich, N. (2021, August 25). *The State of State Machines*. Https://Googleprojectzero.Blogspot.Com/2021/01/the-State-of-State-Machines.Html. https://googleprojectzero.blogspot.com/2021/01/the-state-of-state-machines.html

Future: Thinking in Ecosystems

1) Memory Corruption: Slowing Down
2) Beyond Memory Corruption
3) **Thinking in Ecosystems**

Future: Thinking in Ecosystems

One of offense's initial advantages was the reliance on legacy design principles for secure computing

Is offense starting to incur a similar cost?

Future: Thinking in Ecosystems

Offensive tools are often written to be deployed based on various assumptions

- Most interesting traffic is on an open and distributed internet
- Traffic is observable and modifiable
- Compromising a device implies full, unfettered access
- Code can be introduced at will
- All data on a device can be recovered
- A/V and APT hunters are not sophisticated

Most of these assumptions are now false

Future: Thinking in Ecosystems

The Internet is balkanized: geographically, politically, commercially

- Eg WeChat, Parler, Facebook, Google, Amazon, Apple, HarmonyOS
- Most interesting data is encrypted and contained in private ecosystems
- A lot of code/infrastructure traditionally hosted in-house is outsourced to Google etc

Defense Life Cycle: Mitigations

Devices participate in one or more ecosystems now

- Apple devices -> Apple ecosystem
- Android -> Google ecosystem
- Secondary clouds: Facebook, WeChat, etc

Do these ecosystems represent a significant attack surface?

- Potential to move laterally
- Persist across upgrades/resets

Future: Thinking in Ecosystems

Presents some unique political and legal challenges

- Who is allowed to do that?
  - Not NSO, apparently
- If off-limits to LEO/local intel, guess who that leaves?
  - Adversary nations, criminals

Shu, C. (2020, July 16). *Federal court rules WhatsApp and Facebook's malware exploit case against NSO Group can proceed*. Https://Techcrunch.Com/2020/07/16/Federal-Court-Rules-Whatsapp-and-Facebooks-Malware-Exploit-Case-against-Nso-Group-Can-Proceed/.

Future: Thinking in Ecosystems

Prediction: Future offensive chains will be "ecosystem aware"

Some advantages here for offense, but defense has the ability to do sophisticated anomaly detection

- Benefit from large scale data sets and common usage patterns
- Dino Dai Zovi notes this in his SummerCon 2021 talk

DaiZavi, D. (2021). Dr Strangecode: Or How I Stopped Worrying and Learned to Love the Cloud. In Summercon.

Summary

1) Memory corruption is still the most effective strategy for offense, but it's advantages are eroding

2) Increasingly, offense will replace memory corruption components with other logic flaws

3) Defensive profits by improved detection facilities as a result of less powerful offensive toolkits, and moving to the cloud

The old adage "Defense needs to be right every time, offense only needs to be right once" will possibly be inverted!

Thank You for Joining Us

Join our Discord channel to discuss more or ask questions

https://discord.gg/dXE8ZMvU9J