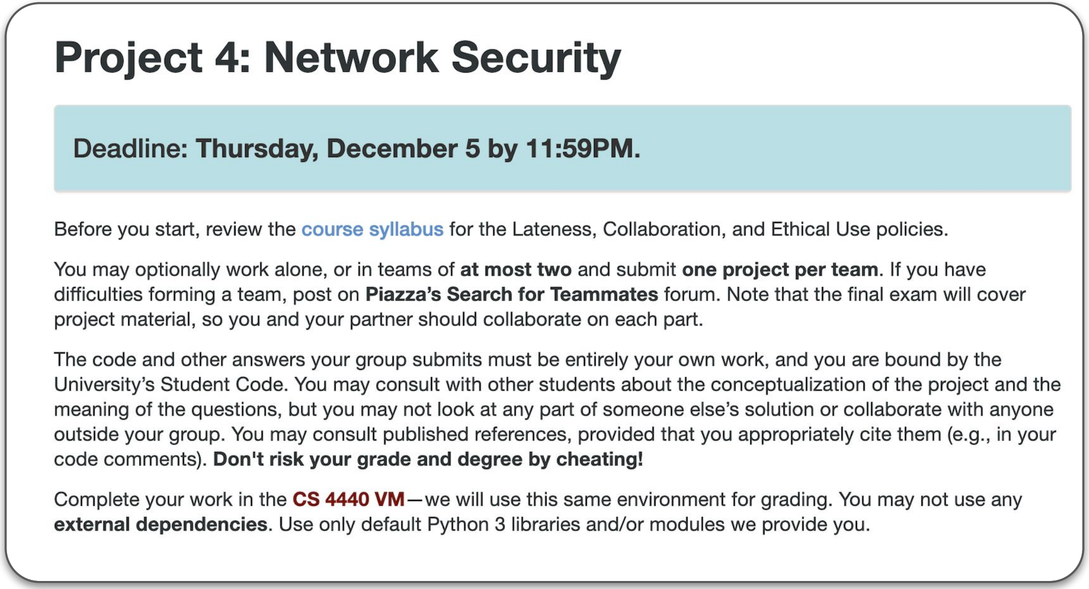
Week 14: Lecture A

What’s Next? Life After CS 4440

Tuesday, December 3, 2024

Announcements

- Project 4: NetSec released
  - **Deadline:** Thursday by 11:59PM



Project 4: Network Security

Deadline: Thursday, December 5 by 11:59PM.

Before you start, review the course syllabus for the Lateness, Collaboration, and Ethical Use policies.

You may optionally work alone, or in teams of **at most two** and submit **one project per team**. If you have difficulties forming a team, post on **Piazza's Search for Teammates** forum. Note that the final exam will cover project material, so you and your partner should collaborate on each part.

The code and other answers your group submits must be entirely your own work, and you are bound by the University's Student Code. You may consult with other students about the conceptualization of the project and the meaning of the questions, but you may not look at any part of someone else's solution or collaborate with anyone outside your group. You may consult published references, provided that you appropriately cite them (e.g., in your code comments). **Don't risk your grade and degree by cheating!**

Complete your work in the **CS 4440 VM**—we will use this same environment for grading. You may not use any **external dependencies**. Use only default Python 3 libraries and/or modules we provide you.

Project 4 Progress

Working on Part 1	0%
Finished Part 1, working on Part 2	0%
Finished both Part 1 and Part 2	0%
None of the above	0%

Final Exam

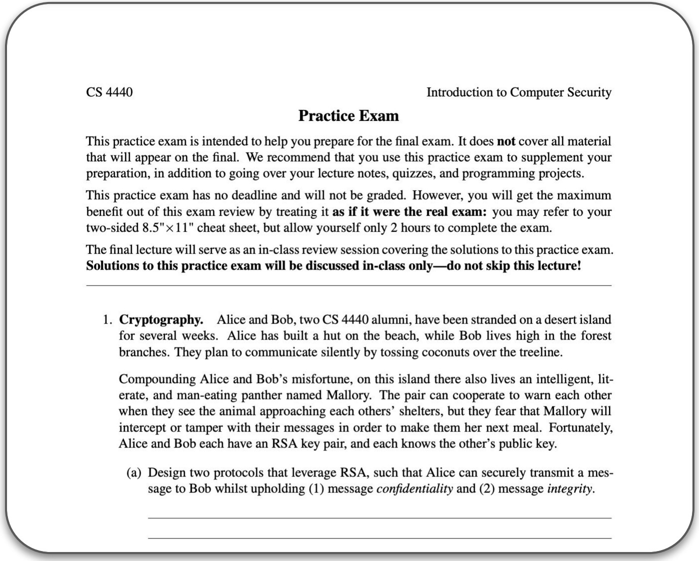
- **Save the date:** **1–3PM** on **Tuesday, December 10**
  - **CDA accommodations:** schedule exam via CDA Portal

- **High-level details** (more to come):
  - One exam covering all course material
  - Similar to project/quiz/lecture exercises

- **Cheat Sheet**
  - **One 8.5”x11” paper** with handwritten/typed notes on **both** sides
  - **Suggestion:** Don’t just use someone else’s—you’ll learn better making **your own**!
  - **Suggestion:** Don’t just paste lecture slides—you’ll learn better by **writing/typing** it!

Practice Exam

- **Practice Exam** released
  - See **Assignments** page on the CS 4440 website
- **Final lecture** will serve as a **review session**
  - Solutions discussed **in-class only**—don't skip!



CS 4440 Introduction to Computer Security

Practice Exam

This practice exam is intended to help you prepare for the final exam. It does **not** cover all material that will appear on the final. We recommend that you use this practice exam to supplement your preparation, in addition to going over your lecture notes, quizzes, and programming projects.

This practice exam has no deadline and will not be graded. However, you will get the maximum benefit out of this exam review by treating it **as if it were the real exam:** you may refer to your two-sided 8.5"×11" cheat sheet, but allow yourself only 2 hours to complete the exam.

The final lecture will serve as an in-class review session covering the solutions to this practice exam. **Solutions to this practice exam will be discussed in-class only—do not skip this lecture!**

1. **Cryptography.** Alice and Bob, two CS 4440 alumni, have been stranded on a desert island for several weeks. Alice has built a hut on the beach, while Bob lives high in the forest branches. They plan to communicate silently by tossing coconuts over the treeline.

   Compounding Alice and Bob's misfortune, on this island there also lives an intelligent, literate, and man-eating panther named Mallory. The pair can cooperate to warn each other when they see the animal approaching each others' shelters, but they fear that Mallory will intercept or tamper with their messages in order to make them her next meal. Fortunately, Alice and Bob each have an RSA key pair, and each knows the other's public key.

   (a) Design two protocols that leverage RSA, such that Alice can securely transmit a message to Bob whilst upholding (1) message *confidentiality* and (2) message *integrity*.

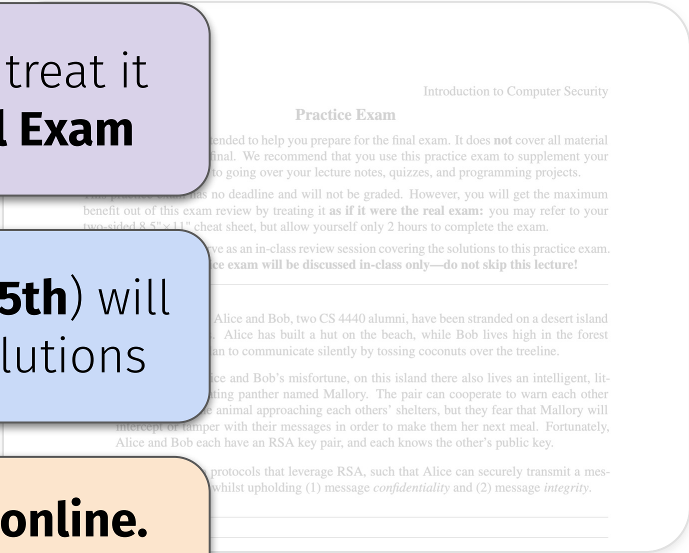
Practice Exam

- **Practice Exam** re
  - See **Assignmen**
- **Final lecture** will serve as a **review session**
  - Solutions discu

To get the most out of this, treat it just **as you would the Final Exam**

Last lecture (**Thursday, Dec. 5th**) will go over the exam review solutions

Solutions won't be posted online.
(Reminder: attendance/participation makes up **5%** of your course grade)



Introduction to Computer Security

Practice Exam

tended to help you prepare for the final exam. It does **not** cover all material final. We recommend that you use this practice exam to supplement your to going over your lecture notes, quizzes, and programming projects.

This practice exam has no deadline and will not be graded. However, you will get the maximum benefit out of this exam review by treating it **as if it were the real exam:** you may refer to your two-sided 8.5"×11" cheat sheet, but allow yourself only 2 hours to complete the exam.

ve as an in-class review session covering the solutions to this practice exam. **ce exam will be discussed in-class only—do not skip this lecture!**

Alice and Bob, two CS 4440 alumni, have been stranded on a desert island . Alice has built a hut on the beach, while Bob lives high in the forest an to communicate silently by tossing coconuts over the treeline.

ice and Bob's misfortune, on this island there also lives an intelligent, litting panther named Mallory. The pair can cooperate to warn each other e animal approaching each others' shelters, but they fear that Mallory will intercept or tamper with their messages in order to make them her next meal. Fortunately, Alice and Bob each have an RSA key pair, and each knows the other's public key.

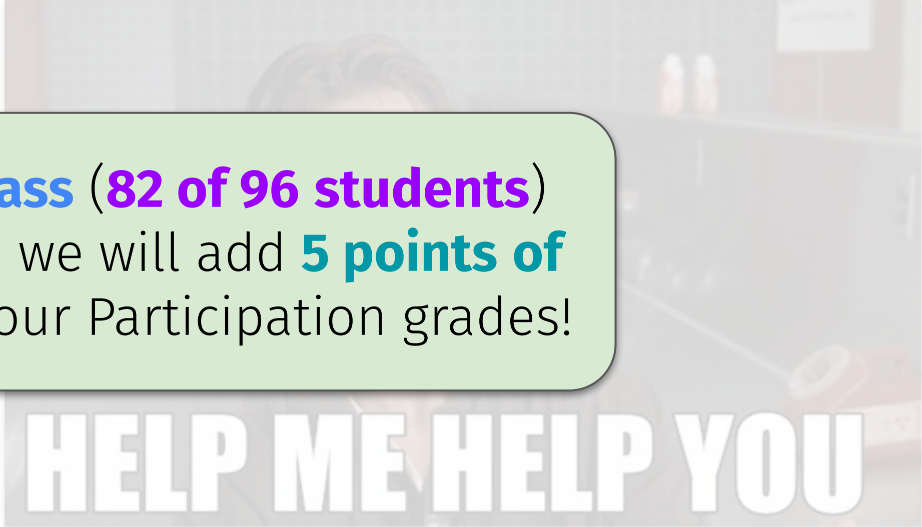
protocols that leverage RSA, such that Alice can securely transmit a mes- whilst upholding (1) message *confidentiality* and (2) message *integrity*.

End-of-semester Course Evals

- **I want your feedback!**
  - 3rd time teaching this course 😃
  - **Help me improve the class!**
- Due by **December 19th**
  - https://scf.utah.edu
  - **Please please please!**

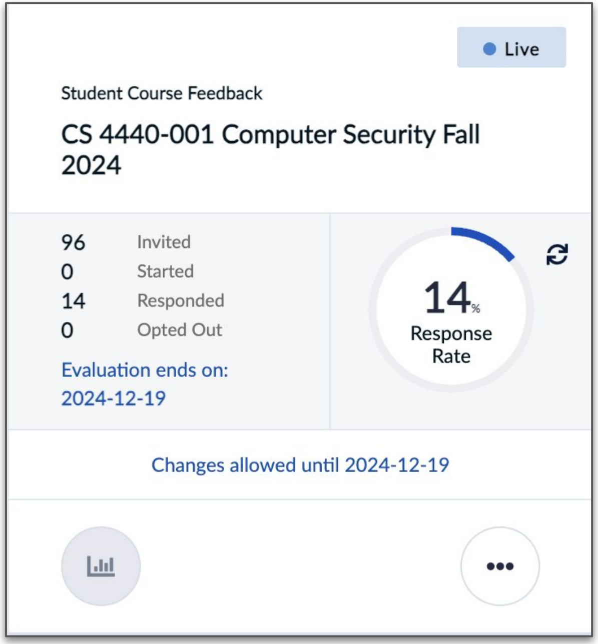
End-of-semester Course Evals

- **I want your feedback!**
  - 3rd time teaching this course 😃
  - **Help me improve the class!**
- Due by **Dece**
  - https://sc
  - **Please pl**



If 85% of the class (**82 of 96 students**) submits an eval, we will add **5 points of extra credit** to your Participation grades!

End-of-semester Course Evals



Student Course Feedback

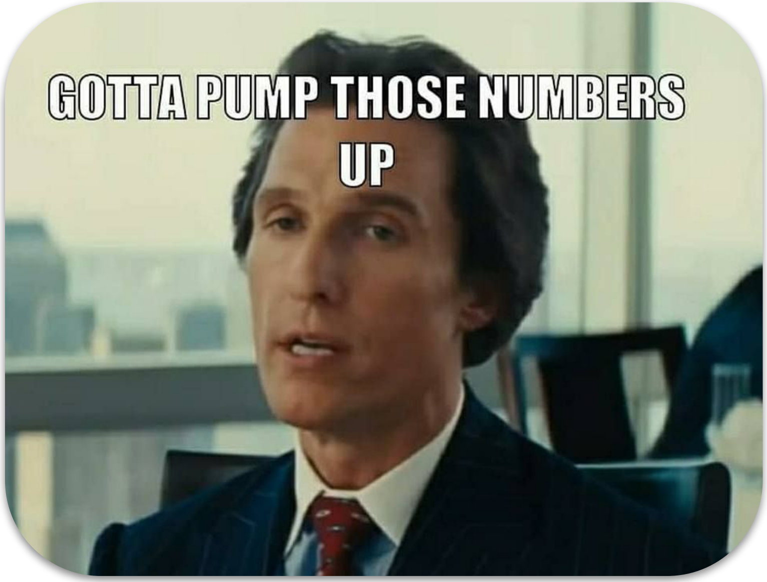
CS 4440-001 Computer Security Fall 2024

96	Invited
0	Started
14	Responded
0	Opted Out

Evaluation ends on: 2024-12-19

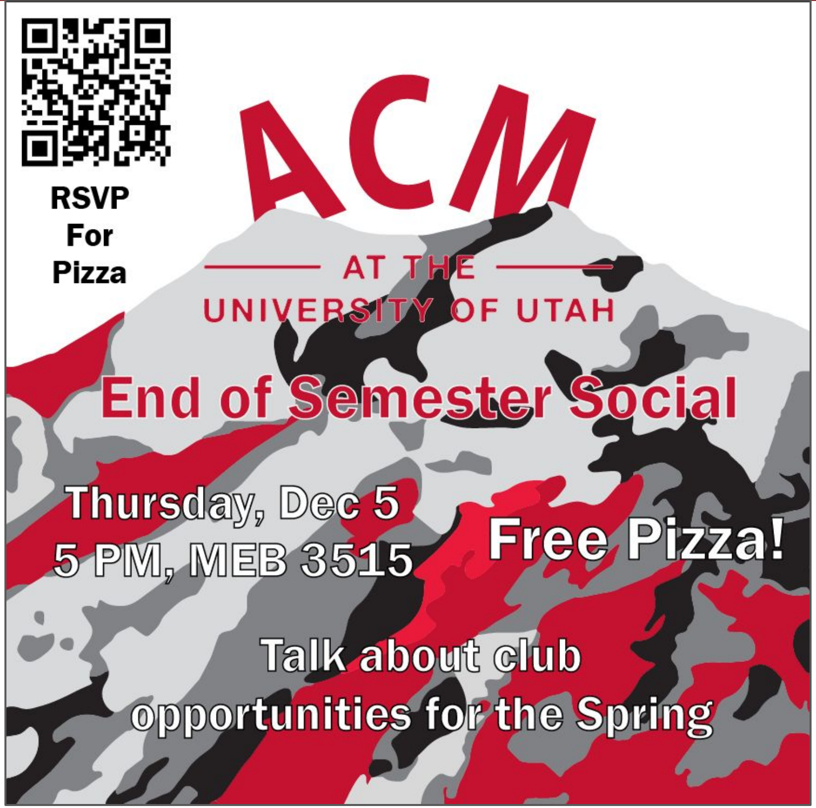
14% Response Rate

Changes allowed until 2024-12-19



End-of-semester Course Evals

Announcements



RSVP
For
Pizza

ACM
AT THE
UNIVERSITY OF UTAH

End of Semester Social

Thursday, Dec 5
5 PM, MEB 3515

Free Pizza!

Talk about club
opportunities for the Spring

Questions?

Last time on CS 4440…

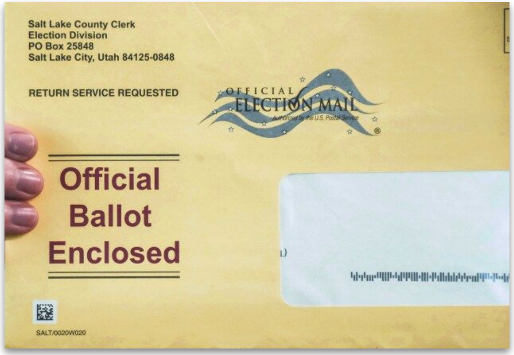
Election Cybersecurity

Requirement #1: Integrity

- **Goals:** outcome matches voter's **intent**
  - Votes are cast as intended
  - Votes are counted as cast

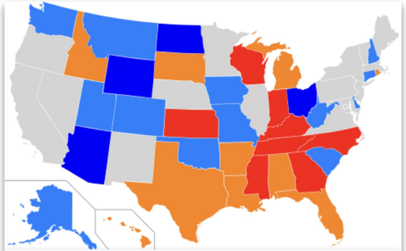
Requirement #2: Confidentiality

- **Goals:** nobody can figure out **how** you voted
  - … even if you try to prove it to them



Requirement #3: Authentication

- **Goals:**
  - Only **authorized voters** can cast votes
  - Each voter can cast **at most one** vote



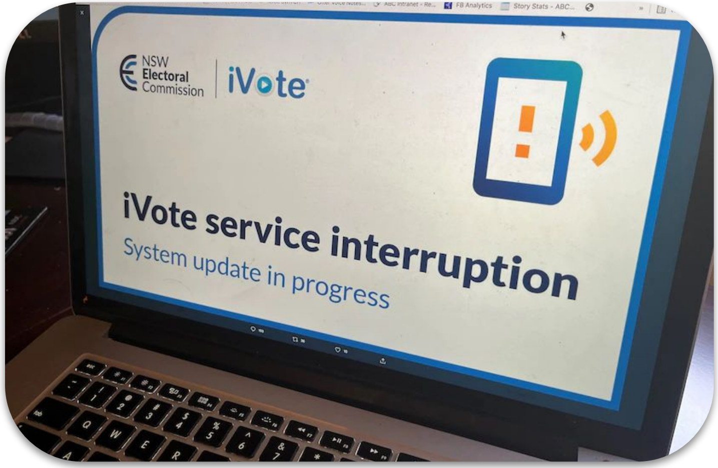
Voter ID laws by state, as of April 2022:

- 🟥 Photo ID required (Strict)
- 🟧 Photo ID requested (Non-strict)
- 🟦 Non-photo ID required (Strict)
- 🟦 Non-photo ID requested (Non-strict)
- ⬜ No ID required to vote

Acceptable Photo IDs
WI Driver License
US Uniformed Services
US Passport Book or Card
WI Driver, State ID, or IDPP Receipt
Certificate of Naturalization
Veteran Affairs ID Card
WI State ID
University / College Tech College + Enrollment Verification
Tribal ID
*Some ID variations are not shown

Requirement #4: Availability

- **Goals:**
  - All authorized voters have **opportunity** to vote
  - System is able to **accept all votes** on schedule
  - System can produce results in a **timely manner**

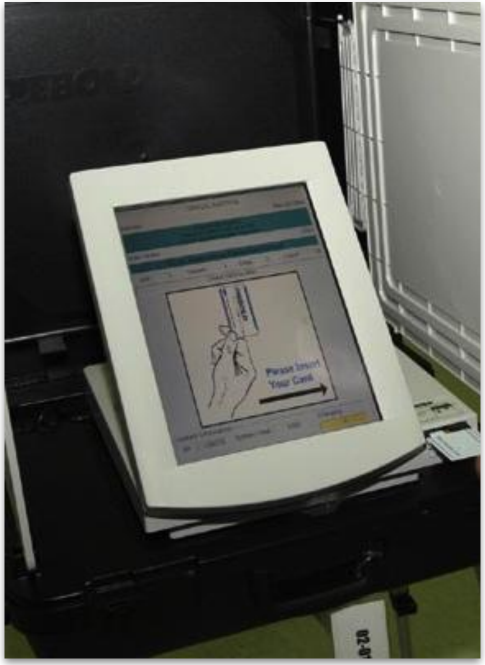


NSW Electoral Commission | iVote

iVote service interruption

System update in progress

Computer-based Voting Devices



DRE Machine

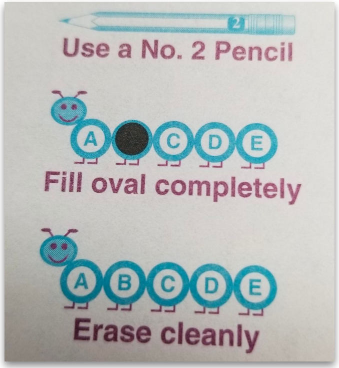
Optical Scanner

Attacks on computer-based voting?

Nobody has responded yet.

Hang tight! Responses are coming in.

Ballot Tampering Attacks



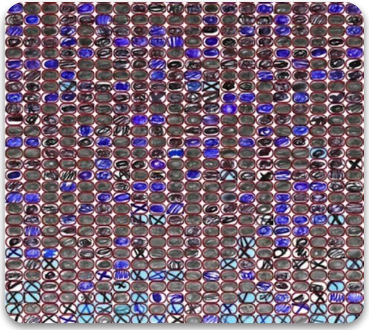
Use a No. 2 Pencil

2
A
C
D
E

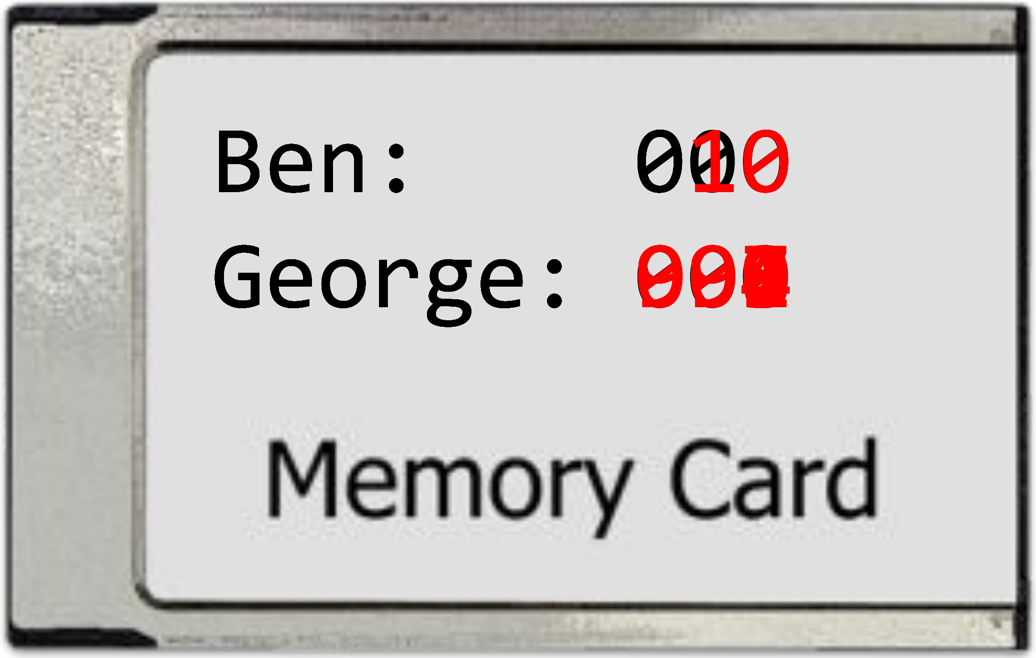
Fill oval completely

A
B
C
D
E

Erase cleanly



Memory Corruption Attacks



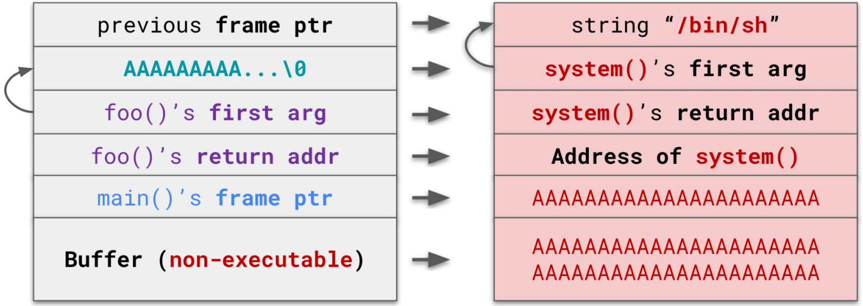
Ben: 010

George: 00[illegible]

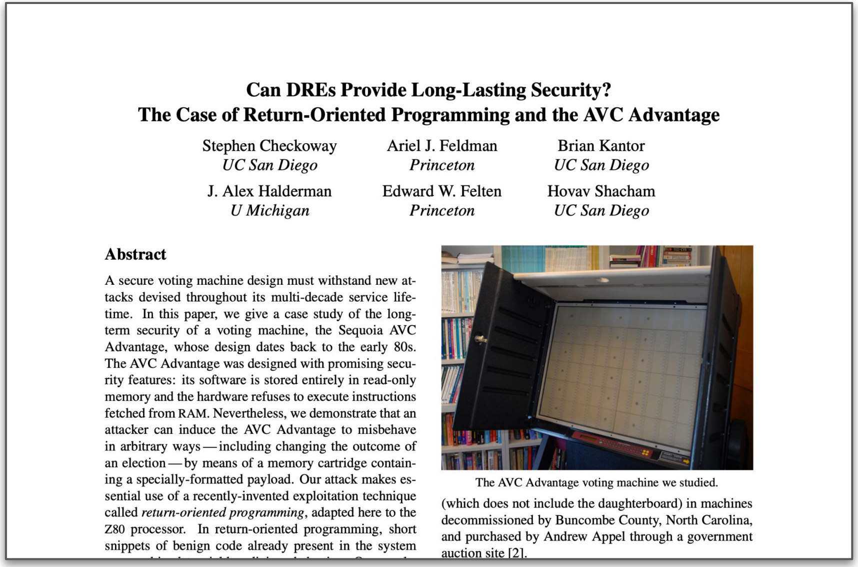
Memory Corruption Attacks

Return-oriented Programming (ROP)

Use code gadgets to achieve functionality



previous frame ptr	string "/bin/sh"
AAAAAAAAA...\0	system()'s first arg
foo()'s first arg	system()'s return addr
foo()'s return addr	Address of system()
main()'s frame ptr	AAAAAAAAAAAAAAAAAAAAAA
Buffer (non-executable)	AAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAA



Can DREs Provide Long-Lasting Security?
The Case of Return-Oriented Programming and the AVC Advantage

Stephen Checkoway
UC San Diego

Ariel J. Feldman
Princeton

Brian Kantor
UC San Diego

J. Alex Halderman
U Michigan

Edward W. Felten
Princeton

Hovav Shacham
UC San Diego

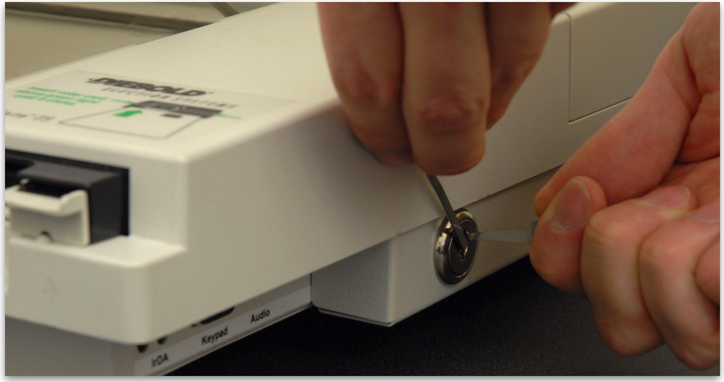
Abstract

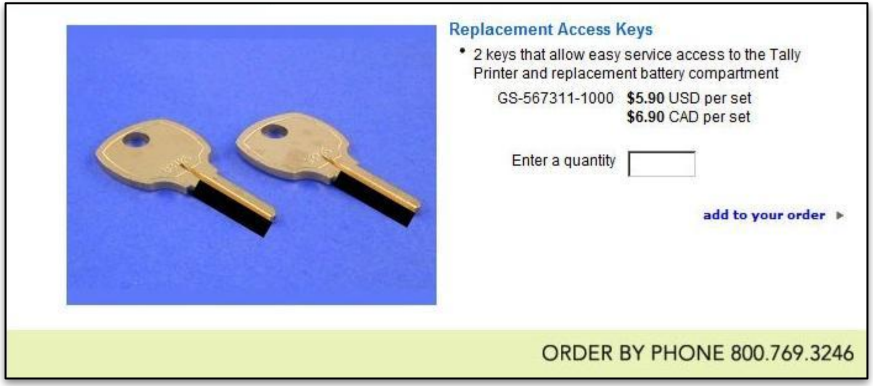
A secure voting machine design must withstand new attacks devised throughout its multi-decade service lifetime. In this paper, we give a case study of the long-term security of a voting machine, the Sequoia AVC Advantage, whose design dates back to the early 80s. The AVC Advantage was designed with promising security features: its software is stored entirely in read-only memory and the hardware refuses to execute instructions fetched from RAM. Nevertheless, we demonstrate that an attacker can induce the AVC Advantage to misbehave in arbitrary ways — including changing the outcome of an election — by means of a memory cartridge containing a specially-formatted payload. Our attack makes essential use of a recently-invented exploitation technique called *return-oriented programming*, adapted here to the Z80 processor. In return-oriented programming, short snippets of benign code already present in the system

The AVC Advantage voting machine we studied.

(which does not include the daughterboard) in machines decommissioned by Buncombe County, North Carolina, and purchased by Andrew Appel through a government auction site [2].

Code Insertion Attacks




Replacement Access Keys

- 2 keys that allow easy service access to the Tally Printer and replacement battery compartment

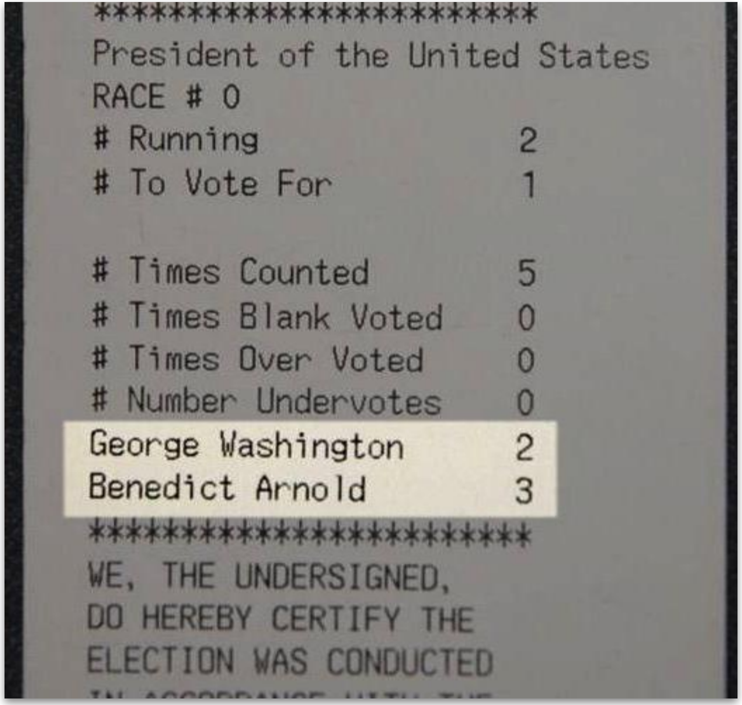
GS-567311-1000 **$5.90** USD per set
$6.90 CAD per set

Enter a quantity

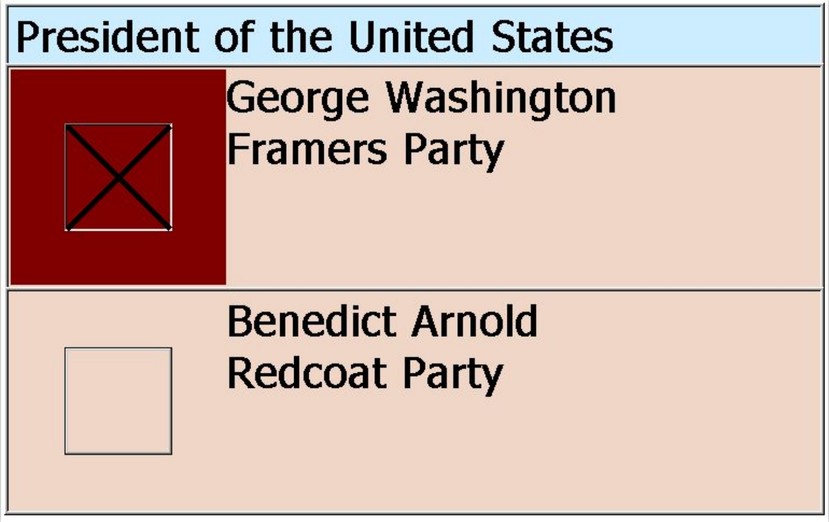
add to your order ▸

ORDER BY PHONE 800.769.3246

Code Insertion Attacks



```
************************
President of the United States
RACE # 0
# Running              2
# To Vote For          1

# Times Counted        5
# Times Blank Voted    0
# Times Over Voted     0
# Number Undervotes    0
George Washington      2
Benedict Arnold        3
************************
WE, THE UNDERSIGNED,
DO HEREBY CERTIFY THE
ELECTION WAS CONDUCTED
IN ACCORDANCE WITH THE
```

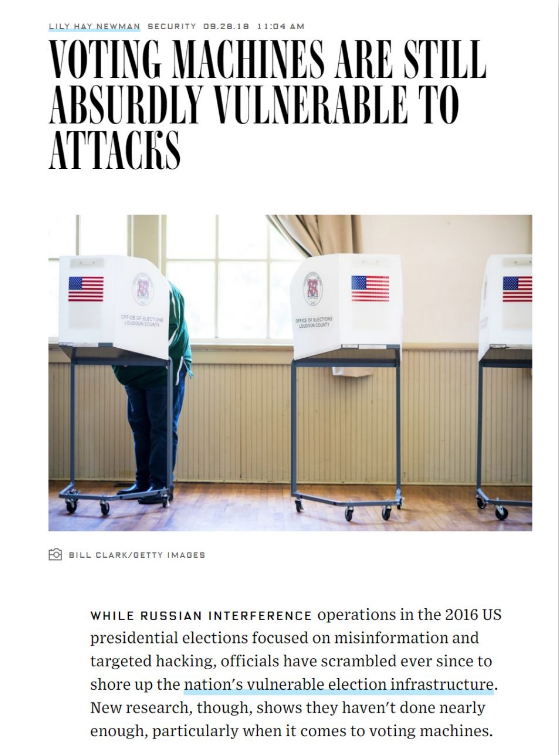


President of the United States

☒ George Washington
Framers Party

□ Benedict Arnold
Redcoat Party

Voting Machine Security

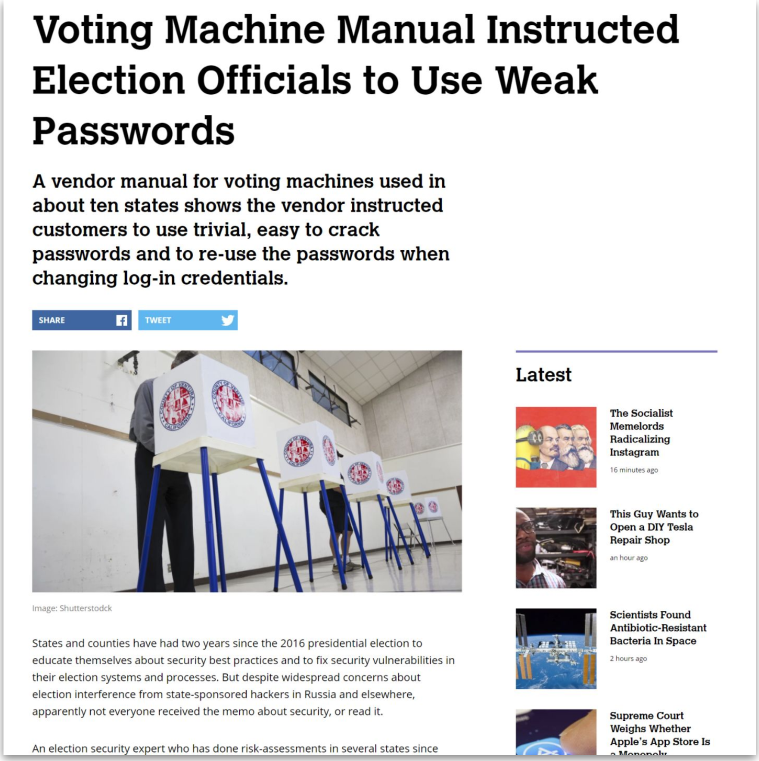


LILY HAY NEWMAN SECURITY 08.28.18 11:04 AM

VOTING MACHINES ARE STILL ABSURDLY VULNERABLE TO ATTACKS

BILL CLARK/GETTY IMAGES

WHILE RUSSIAN INTERFERENCE operations in the 2016 US presidential elections focused on misinformation and targeted hacking, officials have scrambled ever since to shore up the nation's vulnerable election infrastructure. New research, though, shows they haven't done nearly enough, particularly when it comes to voting machines.



Voting Machine Manual Instructed Election Officials to Use Weak Passwords

A vendor manual for voting machines used in about ten states shows the vendor instructed customers to use trivial, easy to crack passwords and to re-use the passwords when changing log-in credentials.

Image: Shutterstodck

States and counties have had two years since the 2016 presidential election to educate themselves about security best practices and to fix security vulnerabilities in their election systems and processes. But despite widespread concerns about election interference from state-sponsored hackers in Russia and elsewhere, apparently not everyone received the memo about security, or read it.

An election security expert who has done risk-assessments in several states since

Latest

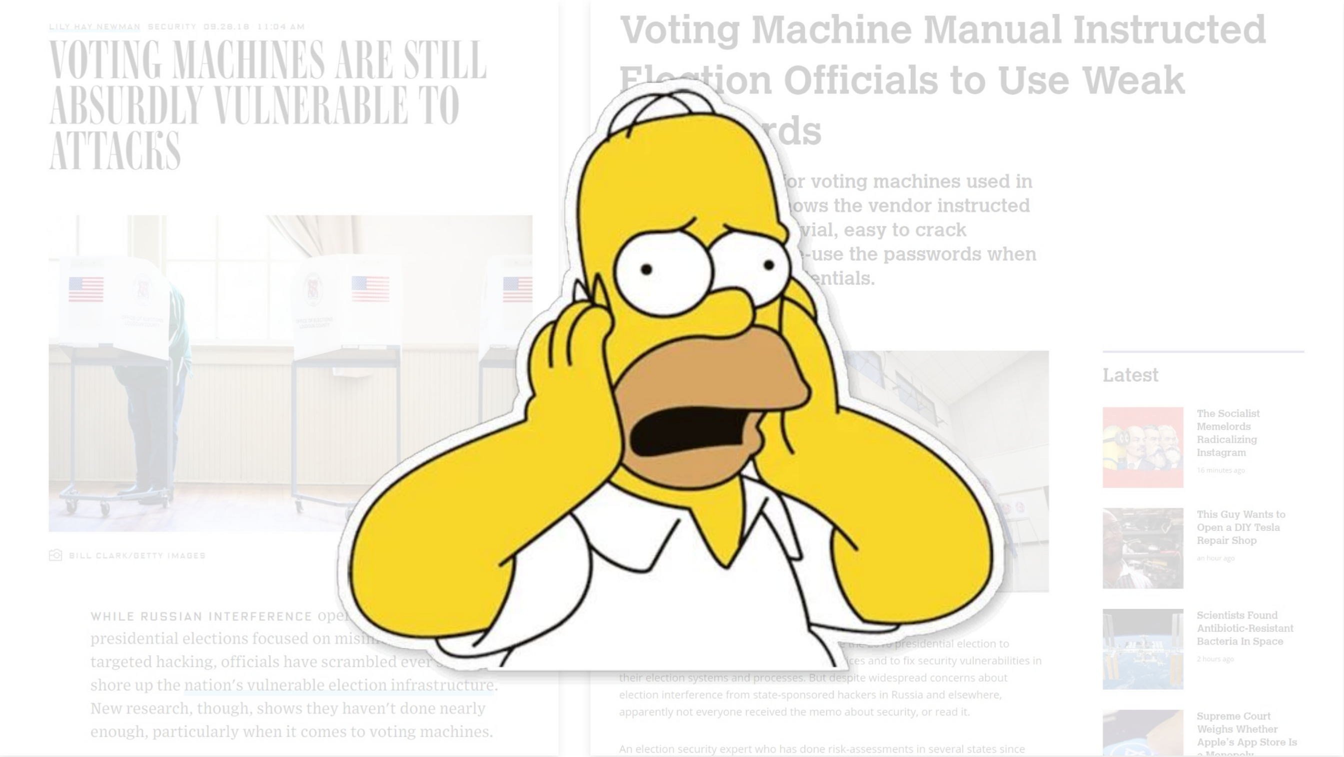
The Socialist Memelords Radicalizing Instagram
16 minutes ago

This Guy Wants to Open a DIY Tesla Repair Shop
an hour ago

Scientists Found Antibiotic-Resistant Bacteria In Space
2 hours ago

Supreme Court Weighs Whether Apple's App Store Is a Monopoly

Voting Machine Security



LILY HAY NEWMAN SECURITY 09.28.18 11:04 AM

VOTING MACHINES ARE STILL ABSURDLY VULNERABLE TO ATTACKS

BILL CLARK/GETTY IMAGES

WHILE RUSSIAN INTERFERENCE ope presidential elections focused on misin targeted hacking, officials have scrambled ever shore up the nation's vulnerable election infrastructure. New research, though, shows they haven't done nearly enough, particularly when it comes to voting machines.

Voting Machine Manual Instructed Election Officials to Use Weak rds

or voting machines used in ows the vendor instructed vial, easy to crack -use the passwords when entials.

presidential election to ices and to fix security vulnerabilities in their election systems and processes. But despite widespread concerns about election interference from state-sponsored hackers in Russia and elsewhere, apparently not everyone received the memo about security, or read it.

An election security expert who has done risk-assessments in several states since

Latest

The Socialist Memelords Radicalizing Instagram

16 minutes ago

This Guy Wants to Open a DIY Tesla Repair Shop

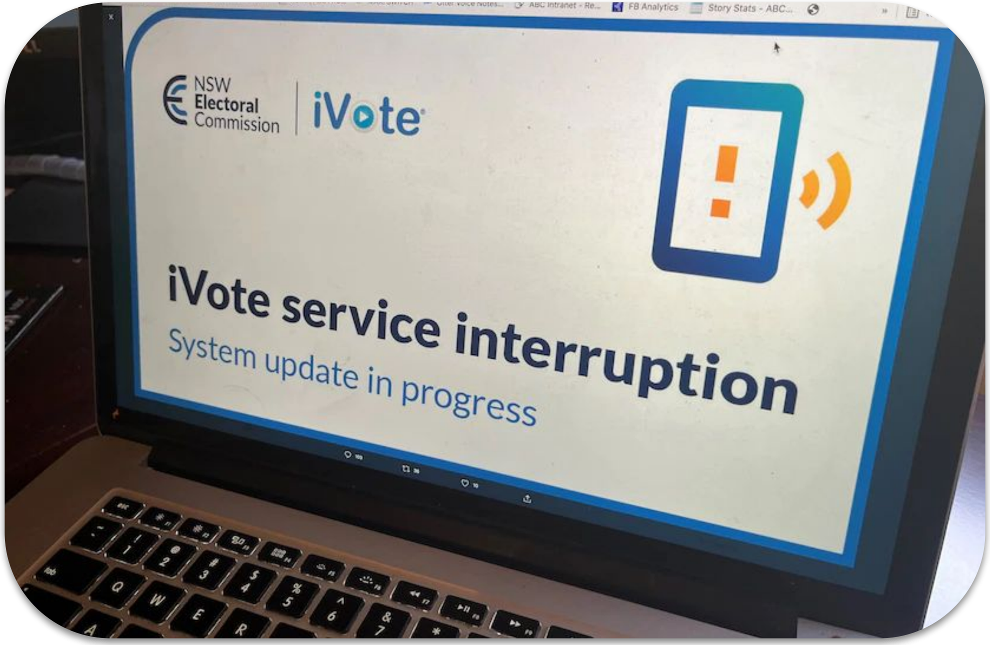
an hour ago

Scientists Found Antibiotic-Resistant Bacteria In Space

2 hours ago

Supreme Court Weighs Whether Apple's App Store Is a Monopoly

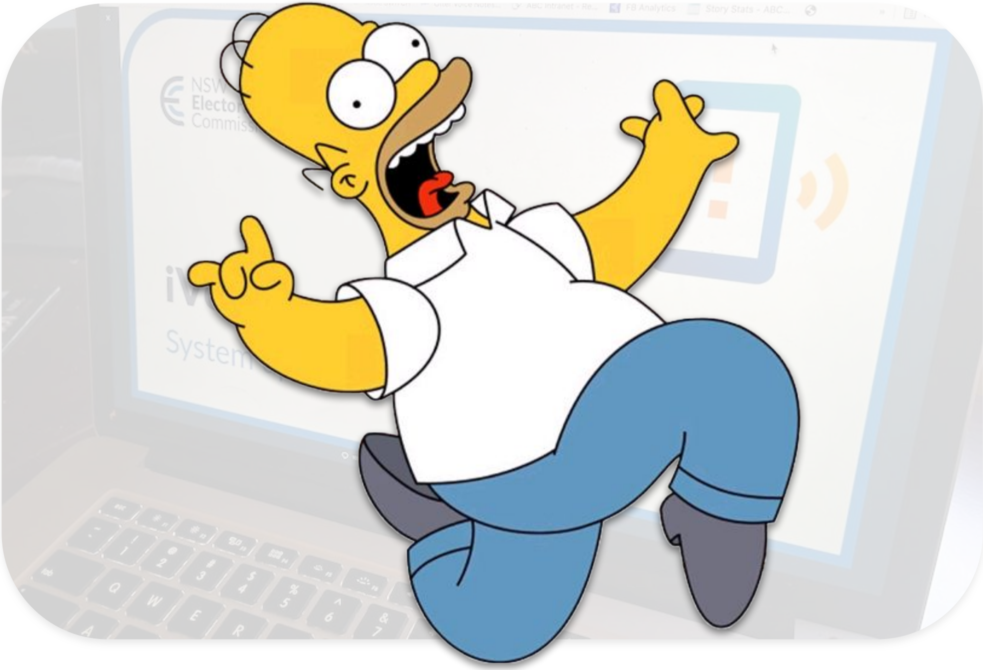
Internet-based Voting Security



iVote service interruption

System update in progress

Internet-based Voting Security



Questions?

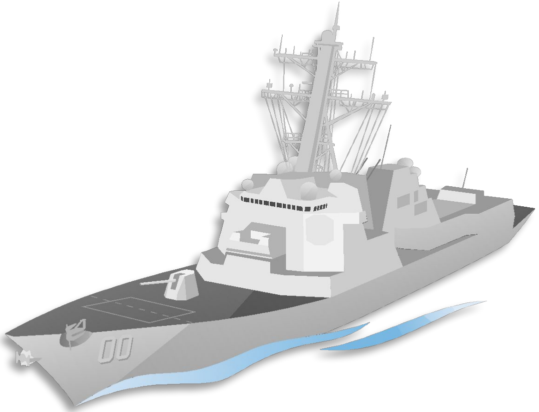
This time on CS 4440…

The Security Ecosystem
Bug Bounty Programs
Capture-the-Flag
Career Paths

Our world depends on software…

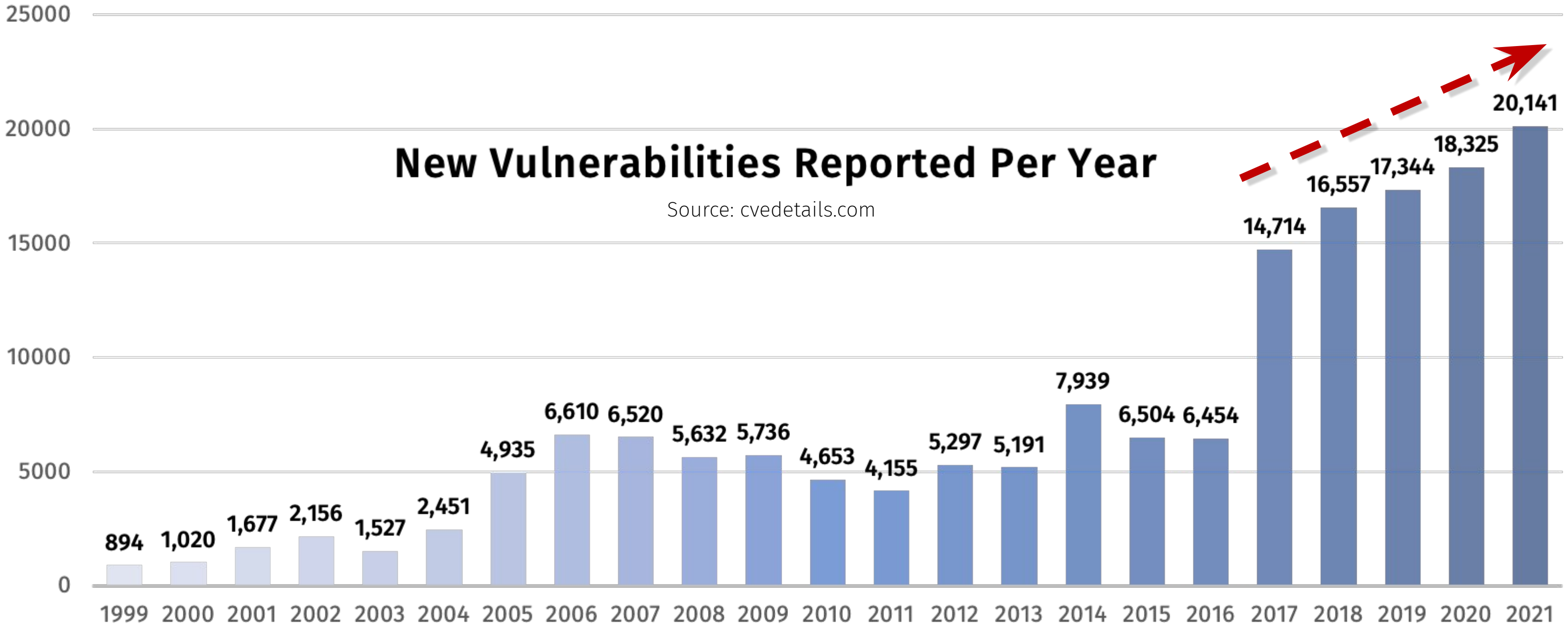
Personal Technology

Infrastructure & Industry



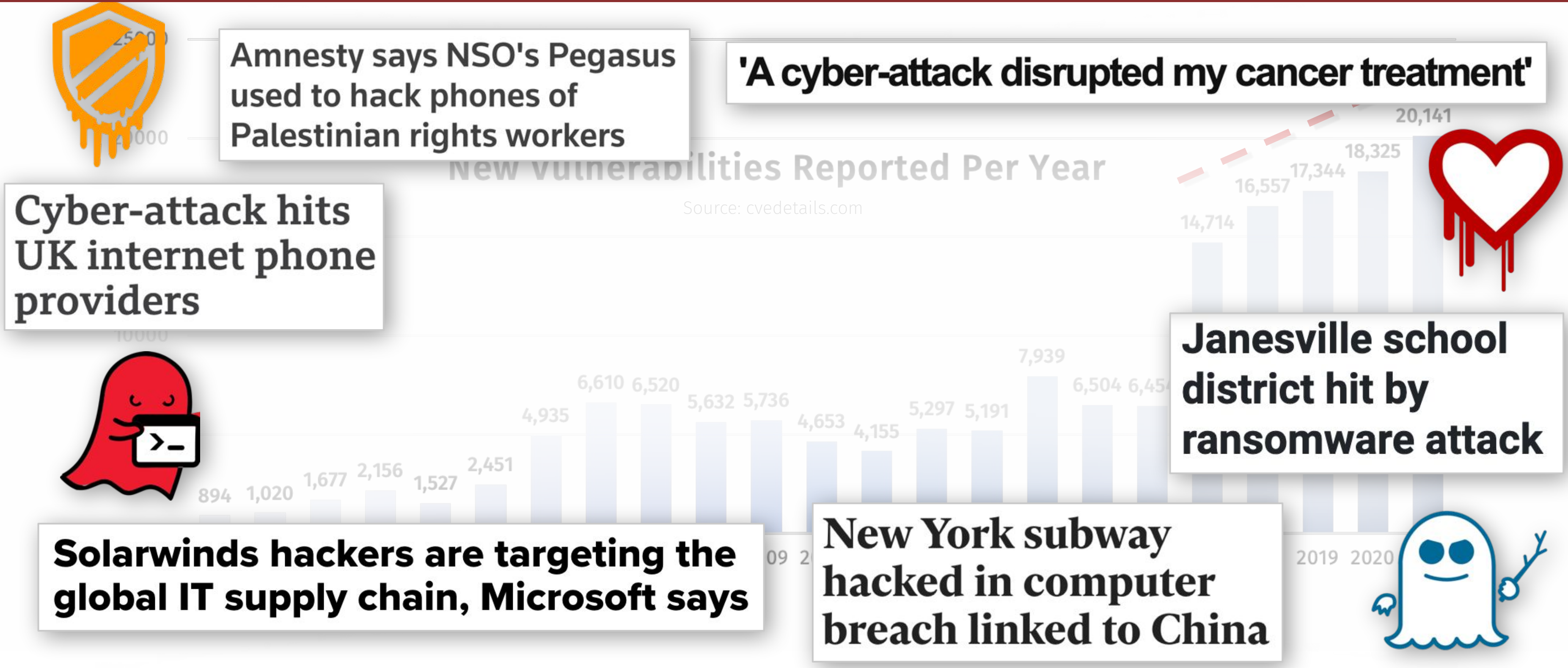
Military and Government

… and software security is a *nightmare*


New Vulnerabilities Reported Per Year

Source: cvedetails.com

… and software security is a *nightmare*



Amnesty says NSO's Pegasus used to hack phones of Palestinian rights workers

'A cyber-attack disrupted my cancer treatment'

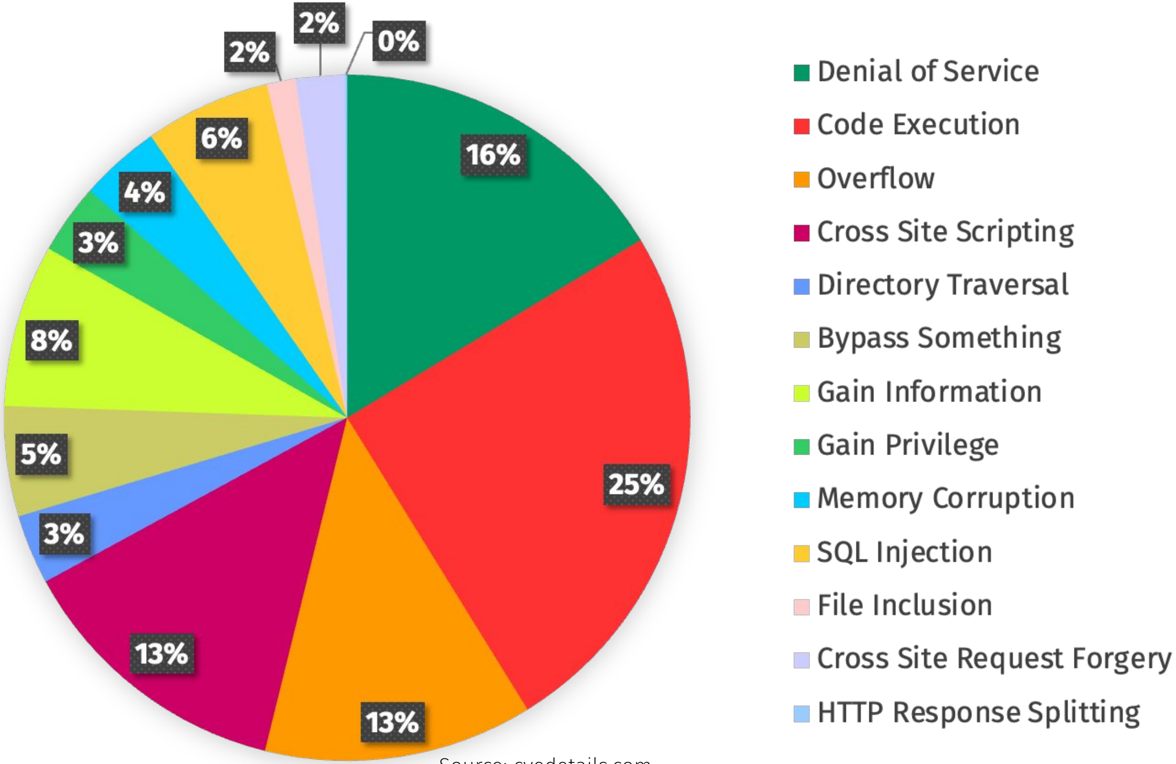
Cyber-attack hits UK internet phone providers

Janesville school district hit by ransomware attack

Solarwinds hackers are targeting the global IT supply chain, Microsoft says

New York subway hacked in computer breach linked to China

Software Security Vulnerabilities



Source: cvedetails.com

Attacks are getting more sophisticated…

1972
First known overflows

1996
Stack overflows

1997
Ret-2-Libc attacks

1997
Function ptr hijacking

1998
Heap overflows

1998
StackGuard bypasses

1999
Format strings

2002
Integer overflows

2002
ASLR bypasses

2005
Hardware DEP bypasses

2005
Ret oriented programming

2007
Heap grooming

2007
Null pointer dereference

2007
Double frees

2009
Heap spraying

2010
JIT spraying

2011
Jmp oriented programming

2014
Call oriented programming

2016
Data oriented programming

2021
Zero-click exploits

What's next?

Attacks are getting more sophisticated...

Who will be at the frontlines of stopping the next attack?

199
Functio
hijacking

1997
Ret-2-Libc
attacks

1996
Stack
overflows

1972
First known
overflows

overflows

1998
StackGuard
byp es

99
rmat
ings

2002
Integer
overflows

grooming

2005
Ret oriented
programming

05
rd re D
byp

2002
ASLR
bypasses

dereference

2007
Double
frees

2009
Heap
spra

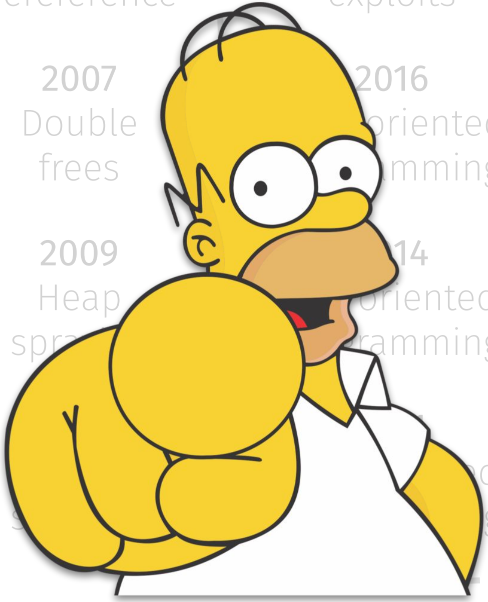
exploits

2016
oriented
mming

14
oriented
ramming

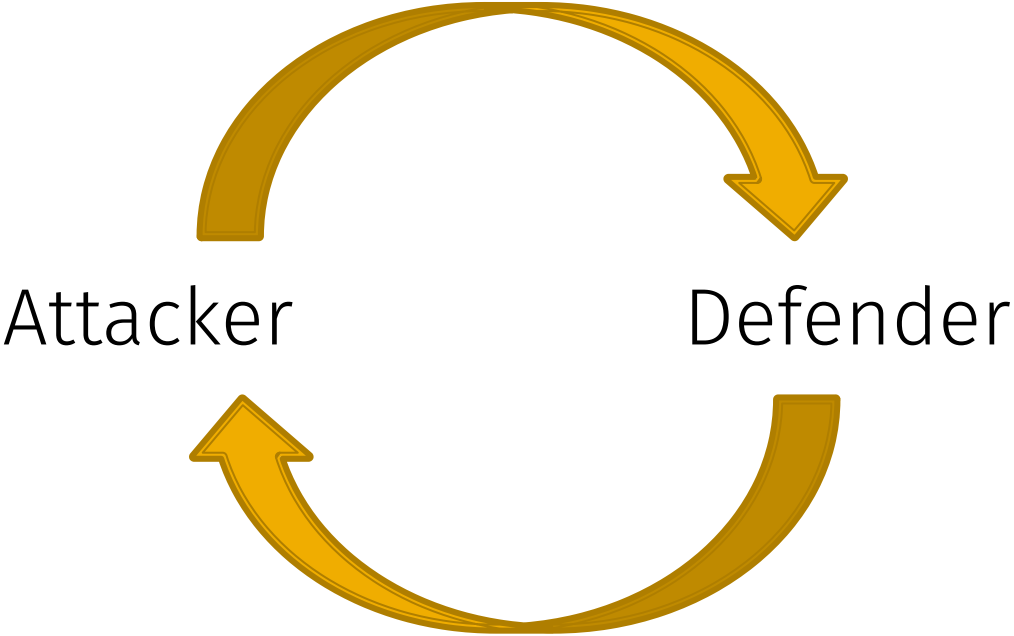
d
g

YOU!



What's next?

Choose your side!



Attacker

Defender

Laws and Ethics

- **If you perform attacks, do so ethically!**
  - Federal/state laws criminalize computer intrusion, wiretapping, or other abuse
  - Computer Fraud and Abuse Act (CFAA)
  - You can be sued or go to jail

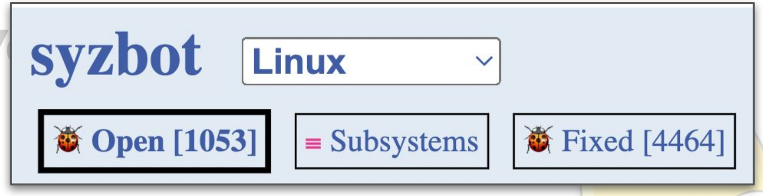
- **Ethical attacker scenarios:**
  - Career as a **Penetration Tester**
  - **CTF competitions** (join **UtahSec** too!)
  - Become a **Security Researcher**

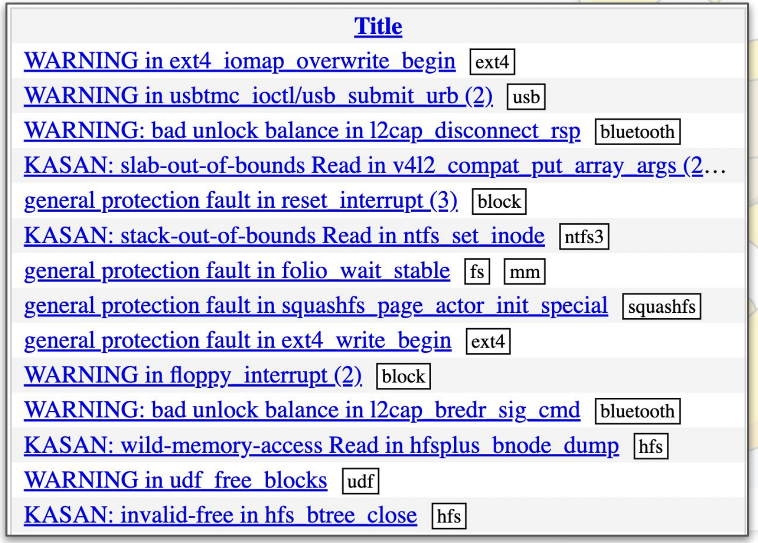
Bug Bounties

Why find and report bugs?

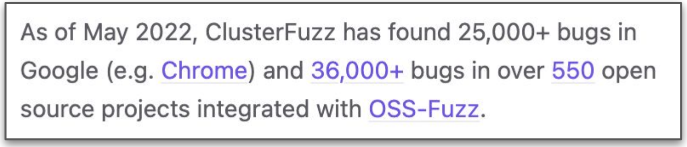
- **You want to save the world**

Why find and report bugs?

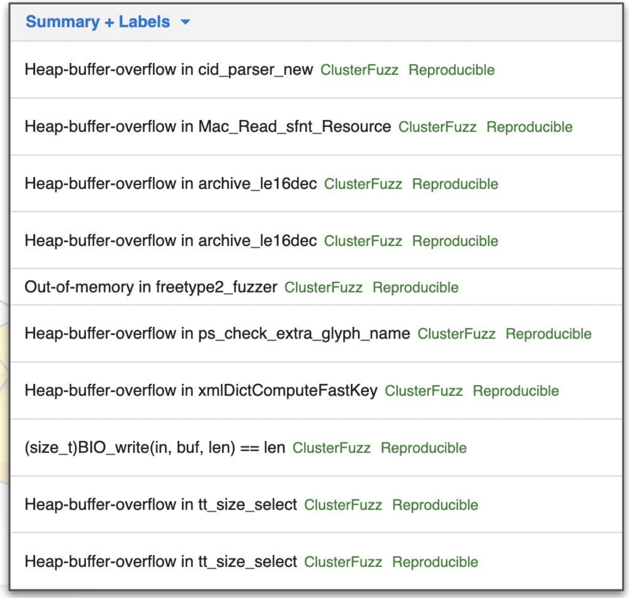




Title	
WARNING in ext4_iomap_overwrite_begin	ext4
WARNING in usbtmc_ioctl/usb_submit_urb (2)	usb
WARNING: bad unlock balance in l2cap_disconnect_rsp	bluetooth
KASAN: slab-out-of-bounds Read in v4l2_compat_put_array_args (2…	
general protection fault in reset_interrupt (3)	block
KASAN: stack-out-of-bounds Read in ntfs_set_inode	ntfs3
general protection fault in folio_wait_stable	fs mm
general protection fault in squashfs_page_actor_init_special	squashfs
general protection fault in ext4_write_begin	ext4
WARNING in floppy_interrupt (2)	block
WARNING: bad unlock balance in l2cap_bredr_sig_cmd	bluetooth
KASAN: wild-memory-access Read in hfsplus_bnode_dump	hfs
WARNING in udf_free_blocks	udf
KASAN: invalid-free in hfs_btree_close	hfs



As of May 2022, ClusterFuzz has found 25,000+ bugs in Google (e.g. Chrome) and 36,000+ bugs in over 550 open source projects integrated with OSS-Fuzz.



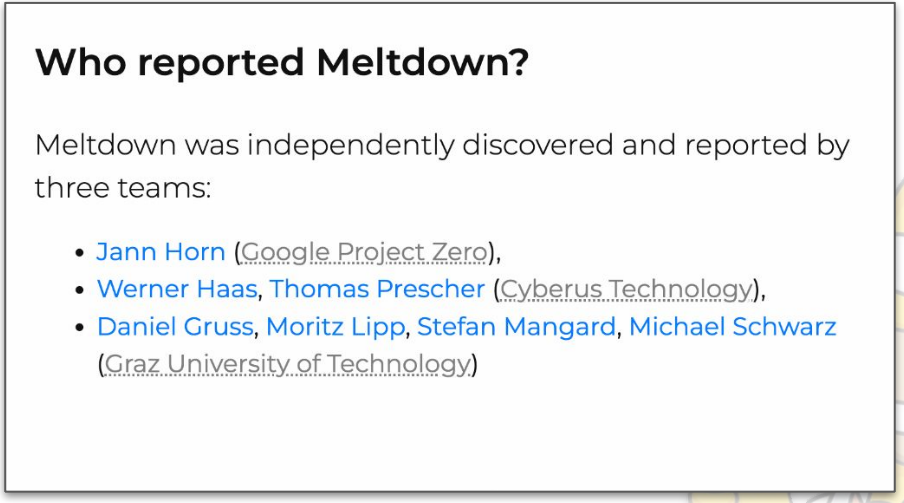
Summary + Labels ▾	
Heap-buffer-overflow in cid_parser_new	ClusterFuzz Reproducible
Heap-buffer-overflow in Mac_Read_sfnt_Resource	ClusterFuzz Reproducible
Heap-buffer-overflow in archive_le16dec	ClusterFuzz Reproducible
Heap-buffer-overflow in archive_le16dec	ClusterFuzz Reproducible
Out-of-memory in freetype2_fuzzer	ClusterFuzz Reproducible
Heap-buffer-overflow in ps_check_extra_glyph_name	ClusterFuzz Reproducible
Heap-buffer-overflow in xmlDictComputeFastKey	ClusterFuzz Reproducible
(size_t)BIO_write(in, buf, len) == len	ClusterFuzz Reproducible
Heap-buffer-overflow in tt_size_select	ClusterFuzz Reproducible
Heap-buffer-overflow in tt_size_select	ClusterFuzz Reproducible

Why *else* find and report bugs?

- You want the notoriety of finding a new bug

Why *else* find and report bugs?

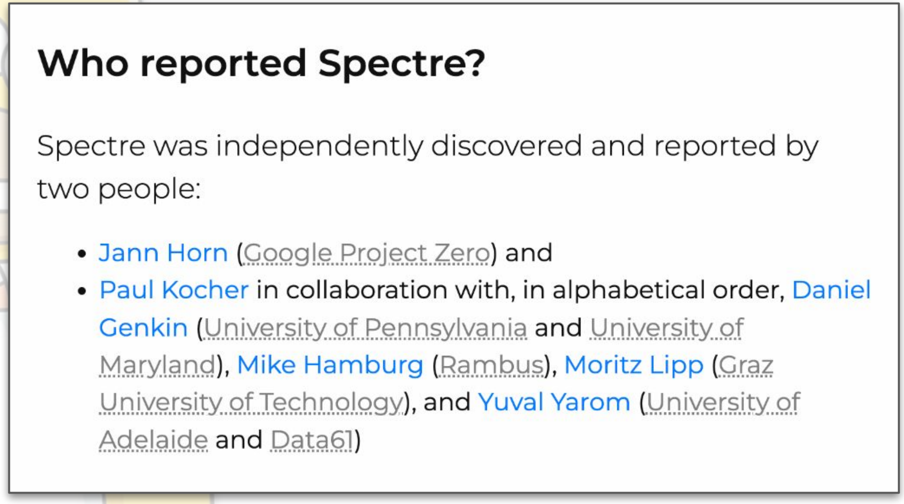
new bug



Who reported Meltdown?

Meltdown was independently discovered and reported by three teams:

- Jann Horn (Google Project Zero),
- Werner Haas, Thomas Prescher (Cyberus Technology),
- Daniel Gruss, Moritz Lipp, Stefan Mangard, Michael Schwarz (Graz University of Technology)



Who reported Spectre?

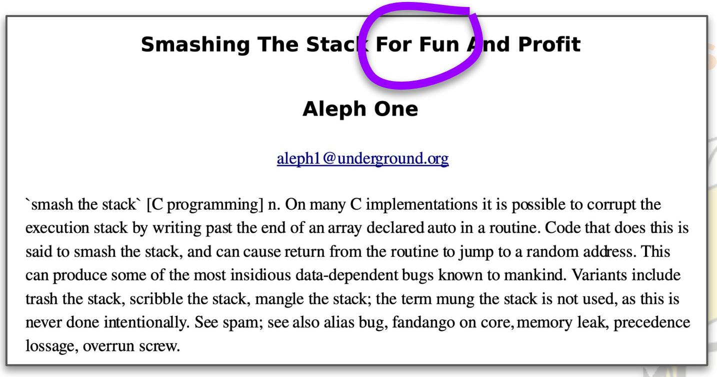
Spectre was independently discovered and reported by two people:

- Jann Horn (Google Project Zero) and
- Paul Kocher in collaboration with, in alphabetical order, Daniel Genkin (University of Pennsylvania and University of Maryland), Mike Hamburg (Rambus), Moritz Lipp (Graz University of Technology), and Yuval Yarom (University of Adelaide and Data61)

Why *else* find and report bugs?

- You love the thrill of **breaking stuff**

Why *else* find and report bugs?

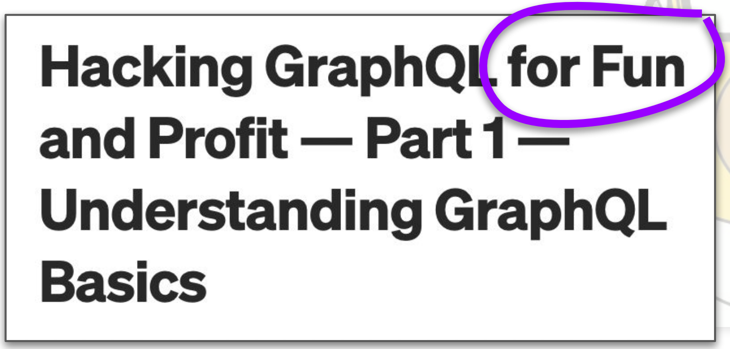


Smashing The Stack For Fun And Profit

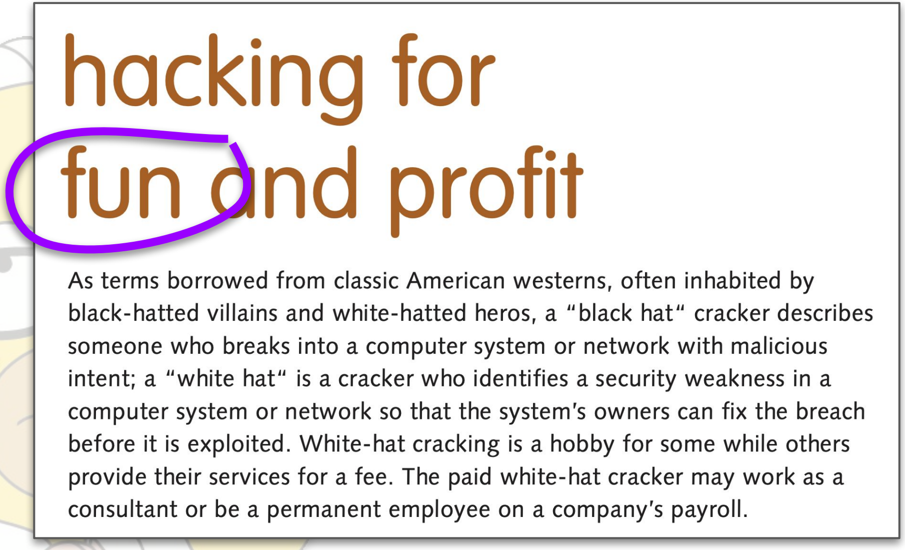
Aleph One

aleph1@underground.org

`smash the stack` [C programming] n. On many C implementations it is possible to corrupt the execution stack by writing past the end of an array declared auto in a routine. Code that does this is said to smash the stack, and can cause return from the routine to jump to a random address. This can produce some of the most insidious data-dependent bugs known to mankind. Variants include trash the stack, scribble the stack, mangle the stack; the term mung the stack is not used, as this is never done intentionally. See spam; see also alias bug, fandango on core, memory leak, precedence lossage, overrun screw.



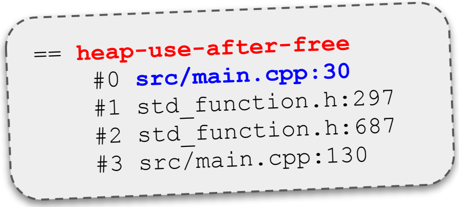
Hacking GraphQL for Fun and Profit — Part 1 — Understanding GraphQL Basics



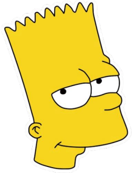
hacking for fun and profit

As terms borrowed from classic American westerns, often inhabited by black-hatted villains and white-hatted heros, a “black hat“ cracker describes someone who breaks into a computer system or network with malicious intent; a “white hat“ is a cracker who identifies a security weakness in a computer system or network so that the system's owners can fix the breach before it is exploited. White-hat cracking is a hobby for some while others provide their services for a fee. The paid white-hat cracker may work as a consultant or be a permanent employee on a company's payroll.

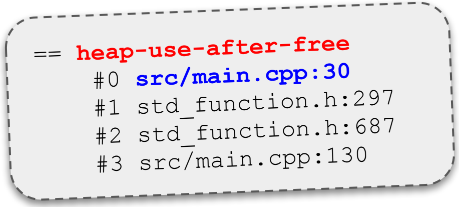
Disclosing Bugs Responsibly



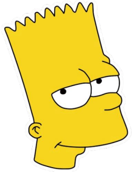
```
== heap-use-after-free
    #0  src/main.cpp:30
    #1  std_function.h:297
    #2  std_function.h:687
    #3  src/main.cpp:130
```



Disclosing Bugs Responsibly



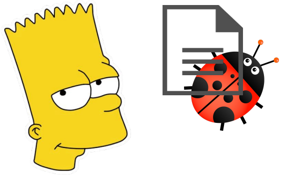
```
== heap-use-after-free
     #0 src/main.cpp:30
     #1 std_function.h:297
     #2 std_function.h:687
     #3 src/main.cpp:130
```

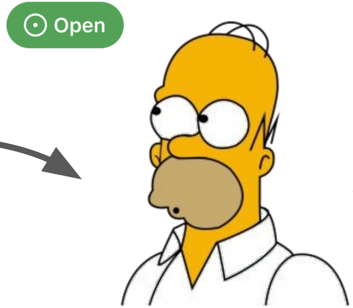


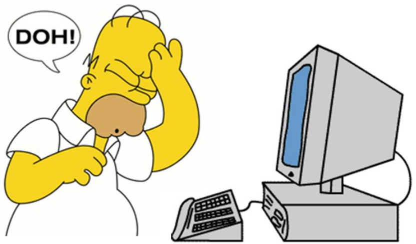
⊙ Open

Disclosing Bugs Responsibly

```
== heap-use-after-free
     #0 src/main.cpp:30
     #1 std_function.h:297
     #2 std_function.h:687
     #3 src/main.cpp:130
```

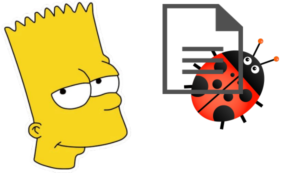


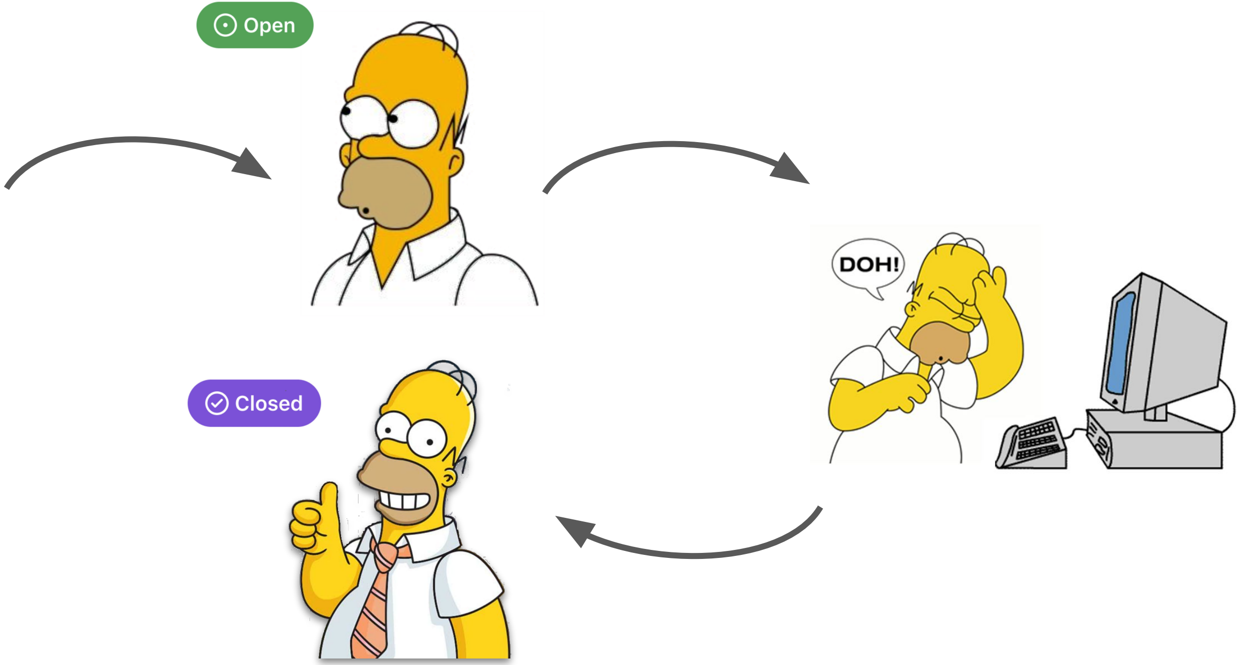




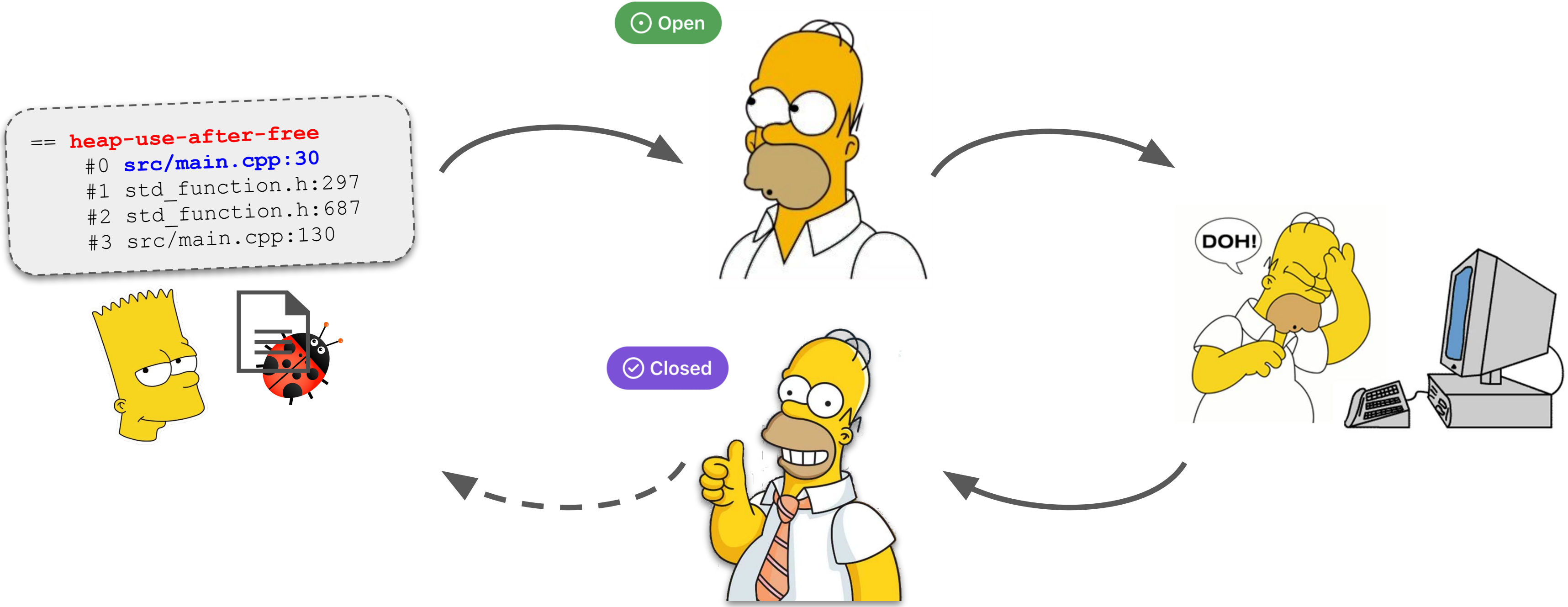
Disclosing Bugs Responsibly

```
== heap-use-after-free
     #0 src/main.cpp:30
     #1 std_function.h:297
     #2 std_function.h:687
     #3 src/main.cpp:130
```





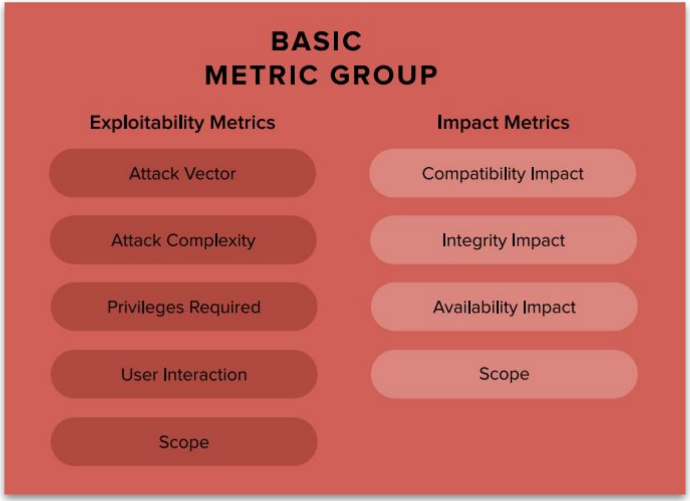
Disclosing Bugs Responsibly



```
== heap-use-after-free
     #0  src/main.cpp:30
     #1  std_function.h:297
     #2  std_function.h:687
     #3  src/main.cpp:130
```

What developers love…

- **Proof-of-concept test cases**
  - Devs need to reproduce your bug
  - Perform their own severity analysis
    - Limited time and resources
    - Fix most severe ones first
    - E.g., MS Patch Tuesday
  - Help them improve their test suites



BASIC
METRIC GROUP

Exploitability Metrics

- Attack Vector
- Attack Complexity
- Privileges Required
- User Interaction
- Scope

Impact Metrics

- Compatibility Impact
- Integrity Impact
- Availability Impact
- Scope

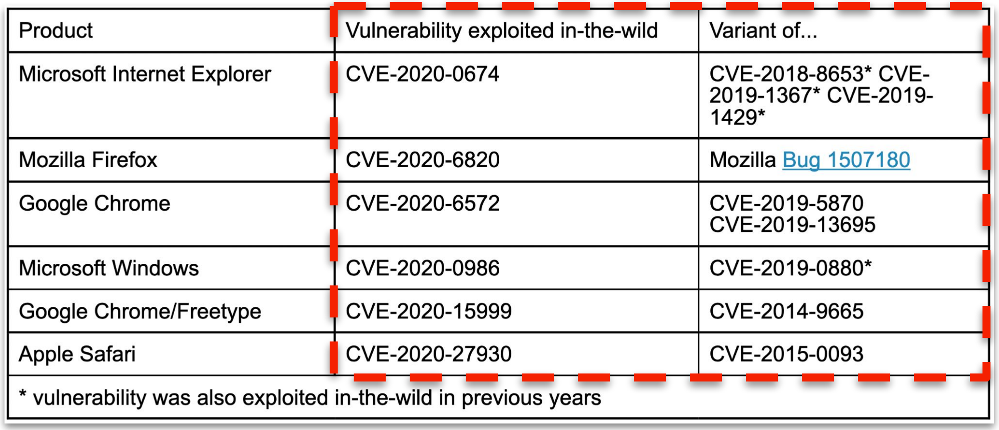
What developers love…

- **Actionable insights**
  - ***Basic:*** build information
    - E.g., compiler, version, OS, etc.
    - Only report bugs in the latest version!
  - ***Good***: crashing source lines, PoCs
  - ***Better:*** root cause analysis
    - E.g., *Missing a check on chunk X*
    - You'll need to get your hands dirty
  - ***Best***: proposed patches
    - May be a back-and-forth battle

Come on you guys! You're dereferencing a null pointer. It's right there!

What developers love...

- **Follow-up testing**
  - Initial fixes may be incomplete
  - Re-run your fancy fuzzer
  - **Open-source your fancy fuzzer**



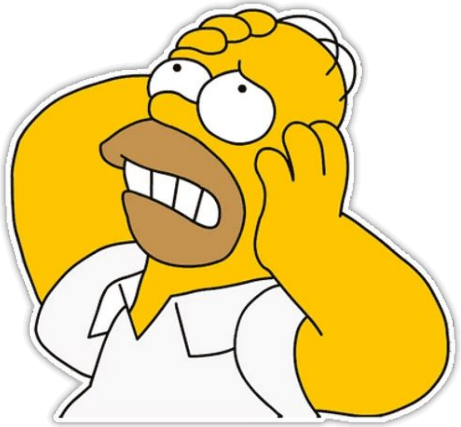
Product	Vulnerability exploited in-the-wild	Variant of...
Microsoft Internet Explorer	CVE-2020-0674	CVE-2018-8653* CVE-2019-1367* CVE-2019-1429*
Mozilla Firefox	CVE-2020-6820	Mozilla Bug 1507180
Google Chrome	CVE-2020-6572	CVE-2019-5870 CVE-2019-13695
Microsoft Windows	CVE-2020-0986	CVE-2019-0880*
Google Chrome/Freetype	CVE-2020-15999	CVE-2014-9665
Apple Safari	CVE-2020-27930	CVE-2015-0093

* vulnerability was also exploited in-the-wild in previous years

Source: Deja Vulnerability by Google Project Zero

What developers *hate*...

- **Little (or unhelpful) information**
  - No PoC test cases or stack traces
  - Bugs on obsolete versions
    - E.g., *I installed this via apt-get*
  - Spamming tons of bug reports
    - Duplicate bug reports
    - Already-reported bugs

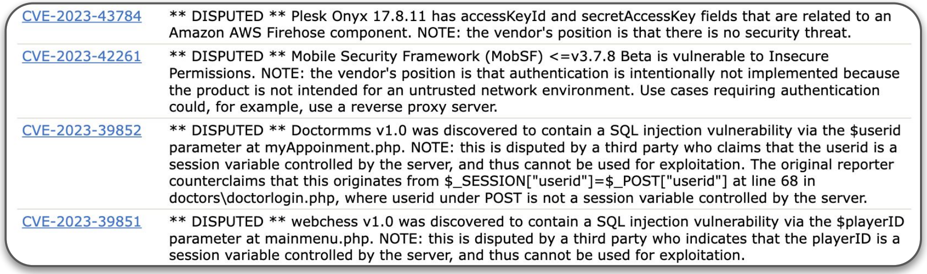


What developers *hate*...

- **Selfish resumé padding**
  - Requesting CVE assignment without first asking them
    - Common in academic papers
    - Reviewers are partially to blame

What developers *hate*…

- **Selfish resumé padding**
  - Requesting CVE assignment without first asking them
    - Common in academic papers
    - Reviewers are partially to blame
  - **Developers can (and do) dispute CVEs**



CVE-2023-43784	** DISPUTED ** Plesk Onyx 17.8.11 has accessKeyId and secretAccessKey fields that are related to an Amazon AWS Firehose component. NOTE: the vendor's position is that there is no security threat.
CVE-2023-42261	** DISPUTED ** Mobile Security Framework (MobSF) <=v3.7.8 Beta is vulnerable to Insecure Permissions. NOTE: the vendor's position is that authentication is intentionally not implemented because the product is not intended for an untrusted network environment. Use cases requiring authentication could, for example, use a reverse proxy server.
CVE-2023-39852	** DISPUTED ** Doctormms v1.0 was discovered to contain a SQL injection vulnerability via the $userid parameter at myAppoinment.php. NOTE: this is disputed by a third party who claims that the userid is a session variable controlled by the server, and thus cannot be used for exploitation. The original reporter counterclaims that this originates from $_SESSION["userid"]=$_POST["userid"] at line 68 in doctors\doctorlogin.php, where userid under POST is not a session variable controlled by the server.
CVE-2023-39851	** DISPUTED ** webchess v1.0 was discovered to contain a SQL injection vulnerability via the $playerID parameter at mainmenu.php. NOTE: this is disputed by a third party who indicates that the playerID is a session variable controlled by the server, and thus cannot be used for exploitation.

What developers *hate*…

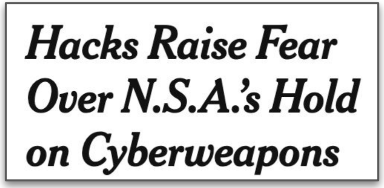
- **Weaponizing and selling an exploit**
  - A huge underground economy
    - Nation-state actors
    - Cyber-criminal gangs

What developers *hate*…

- **Weaponizing and selling an exploit**
  - A huge underground economy
    - Nation-state actors
    - Cyber-criminal gangs
  - **Don’t do this**

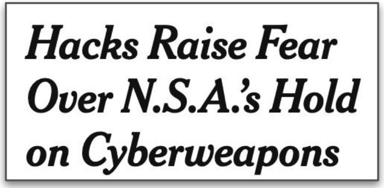
What developers *hate*...

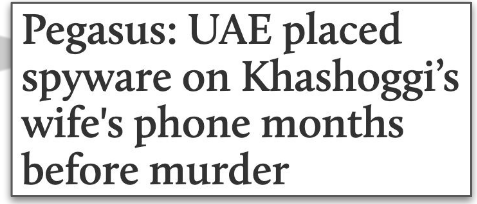
- **Weaponizing and selling an exploit**
  - A huge underground economy
    - Nation-state actors
    - Cyber-criminal gangs
  - **Don't do this**
    - Likely to end up in bad hands regardless of who brokered it


Hacks Raise Fear Over N.S.A.'s Hold on Cyberweapons

What developers *hate*...

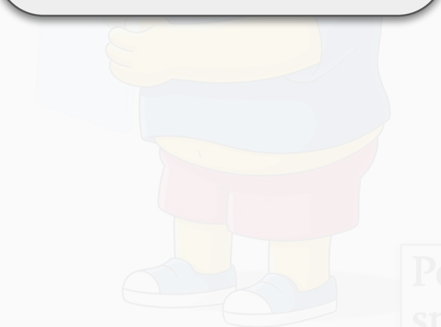
- **Weaponizing and selling an exploit**
  - A huge underground economy
    - Nation-state actors
    - Cyber-criminal gangs
  - **Don't do this**
    - Likely to end up in bad hands regardless of who brokered it
    - Authoritarian regimes use these all the time for **evil acts**
    - You are very likely causing people to get hurt **(or worse)**


Hacks Raise Fear Over N.S.A.'s Hold on Cyberweapons


Pegasus: UAE placed spyware on Khashoggi's wife's phone months before murder

Practice saying NO!

- **Weaponizing and selling an exploit**
  - A huge underground economy
    - Nation-state actors
    - Cyber-criminal gangs
  - **Don't**
    - end up in bad hands
    - of who brokered it
    - an regimes use these
    - e time for **evil acts**
    - y likely causing people
    - rt **(or worse)**



Ha ar
O old
o ons

Pegasu ced
spyware o hoggi's
wife's pho ths
before mu

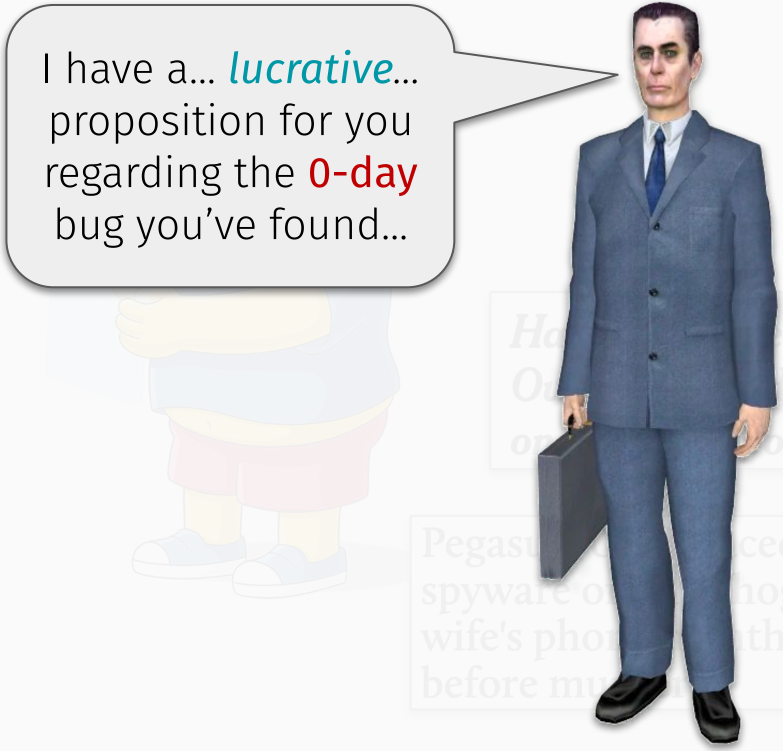
I have a… *lucrative*… proposition for you regarding the 0-day bug you've found…

Practice saying NO!

- Weaponizing and selling an exploit
  - A huge underground
    - Nation-state a
    - Cyber-crimina
  - Don't do
    - Li d u bad hands
    - n f who brokered it
    - an regimes use these
    - for **evil acts**
    - very likely causing people
    - (rse)

Ha ar O old o ons

Pegasu ced spyware o hoggi's wife's pho ths before mu

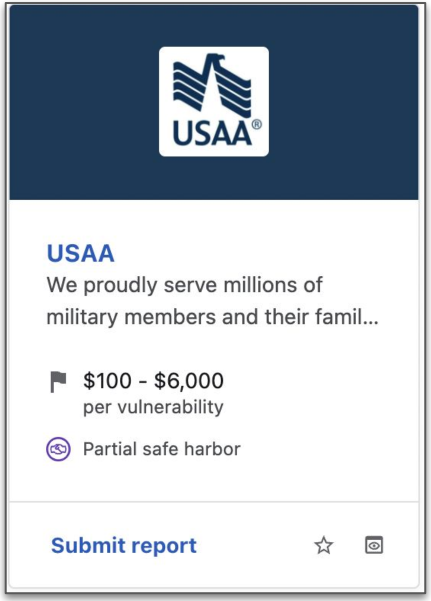

I have a… *lucrative*… proposition for you regarding the 0-day bug you've found…

Why *else* find and report bugs?

- **You want that money!**

Bug Bounties

- **Get paid to find bugs!**

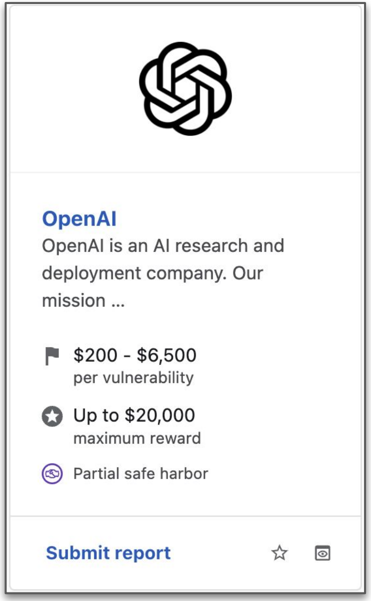


USAA

We proudly serve millions of military members and their famil...

$100 - $6,000
per vulnerability

Partial safe harbor



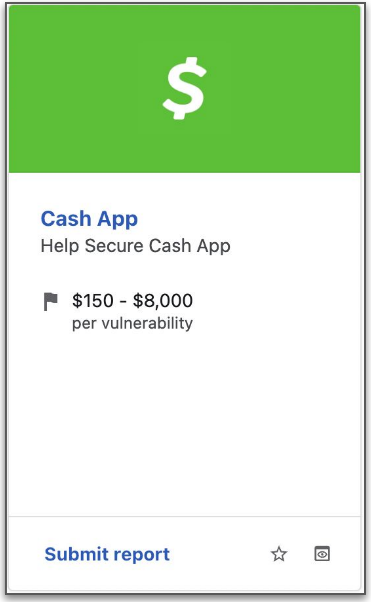
OpenAI

OpenAI is an AI research and deployment company. Our mission ...

$200 - $6,500
per vulnerability

Up to $20,000
maximum reward

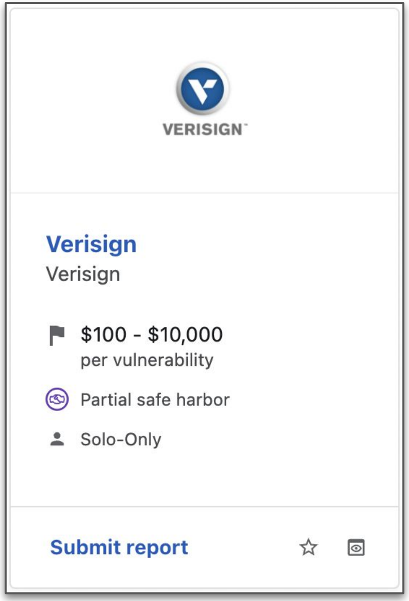
Partial safe harbor



Cash App

Help Secure Cash App

$150 - $8,000
per vulnerability



Verisign

Verisign

$100 - $10,000
per vulnerability

Partial safe harbor

Solo-Only

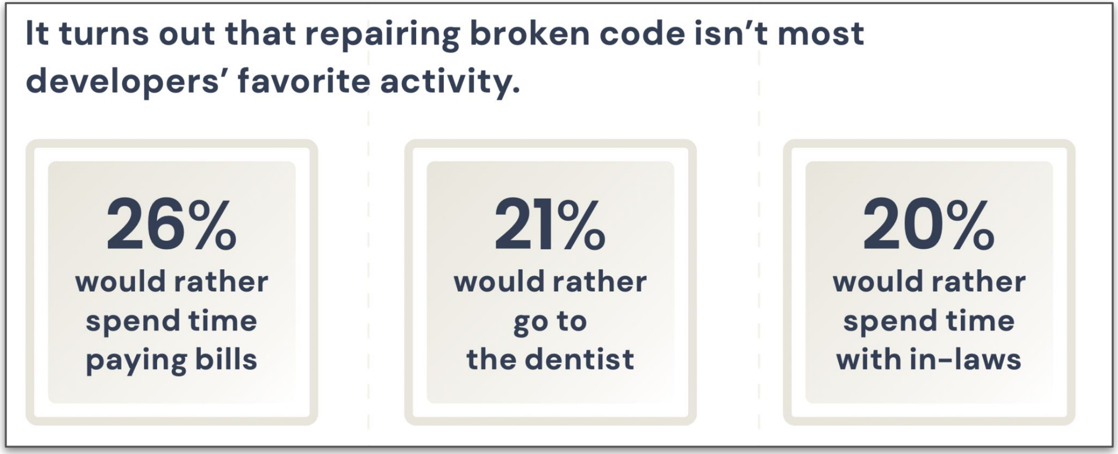
Bug Bounty Programs

- **Where programs are advertised:**
  - BugCrowd: https://bugcrowd.com/
  - HackerOne: https://www.hackerone.com/

- **Not all bugs receive a bounty!**
  - Must be reproducible by devs
  - Higher-severity = more **$$$**
  - Adjudication up to the dev

Developers are people, too

- Data suggests that fixing bugs is a really tough job



It turns out that repairing broken code isn't most developers' favorite activity.

26% would rather spend time paying bills

21% would rather go to the dentist

20% would rather spend time with in-laws

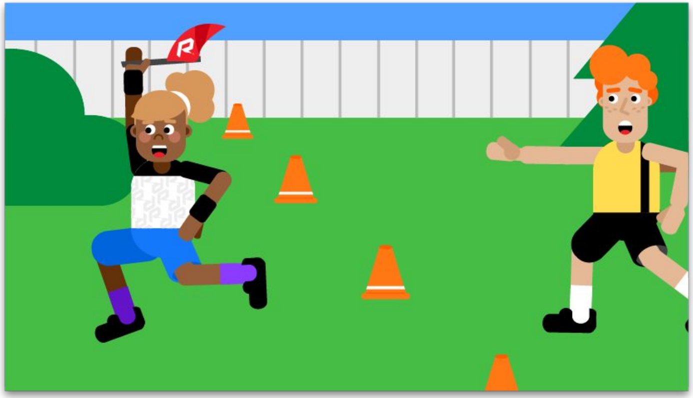
- **Treat developers with courtesy, respect, and patience**

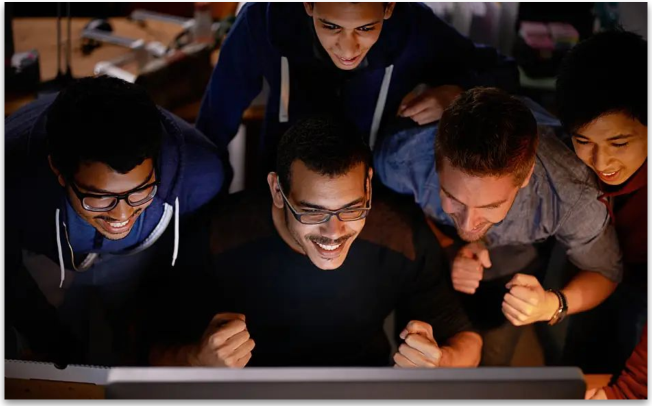
Source: https://content.rollbar.com/hubfs/State-of-Software-Code-Report.pdf

Capture-the-Flag (CTF)

What is CTF?

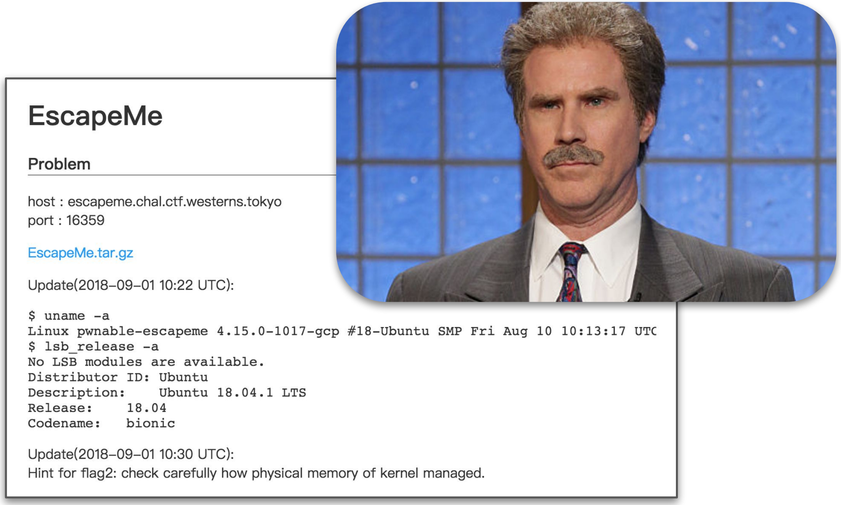
- **CTF = “Capture the Flag”**
  - Competitive cybersecurity events
  - For educational purposes, prizes, etc.
  - **Takes skill to win!**





Styles of CTF: Jeopardy

- **Jeopardy:** solve the most challenges to win
  - **Score the most points** in allotted time



EscapeMe

Problem

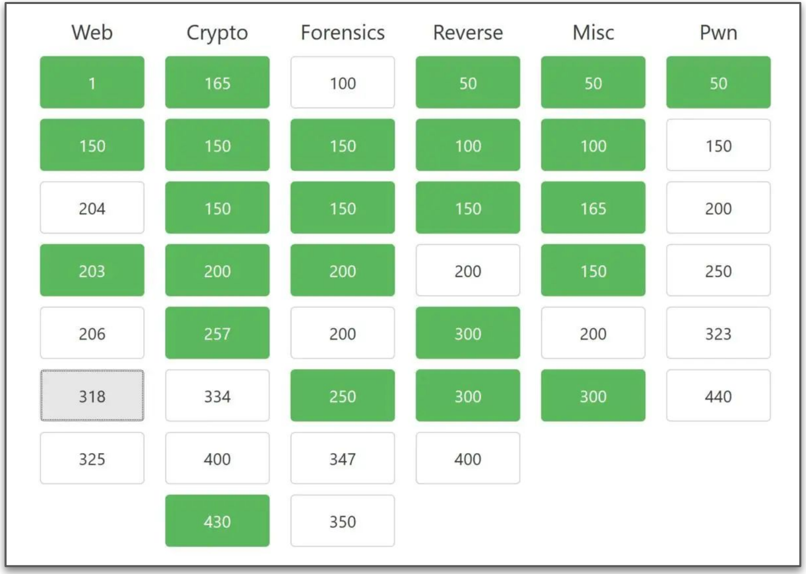
host : escapeme.chal.ctf.westerns.tokyo
port : 16359

EscapeMe.tar.gz

Update(2018-09-01 10:22 UTC):

```
$ uname -a
Linux pwnable-escapeme 4.15.0-1017-gcp #18-Ubuntu SMP Fri Aug 10 10:13:17 UTC
$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 18.04.1 LTS
Release:    18.04
Codename:   bionic
```

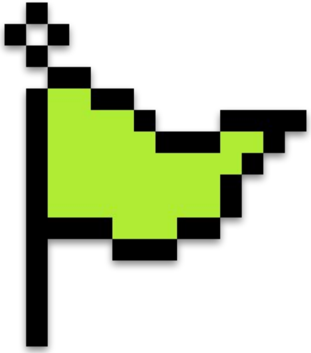
Update(2018-09-01 10:30 UTC):
Hint for flag2: check carefully how physical memory of kernel managed.



Web	Crypto	Forensics	Reverse	Misc	Pwn
1	165	100	50	50	50
150	150	150	100	100	150
204	150	150	150	165	200
203	200	200	200	150	250
206	257	200	300	200	323
318	334	250	300	300	440
325	400	347	400		
	430	350			

Jeopardy Scoring

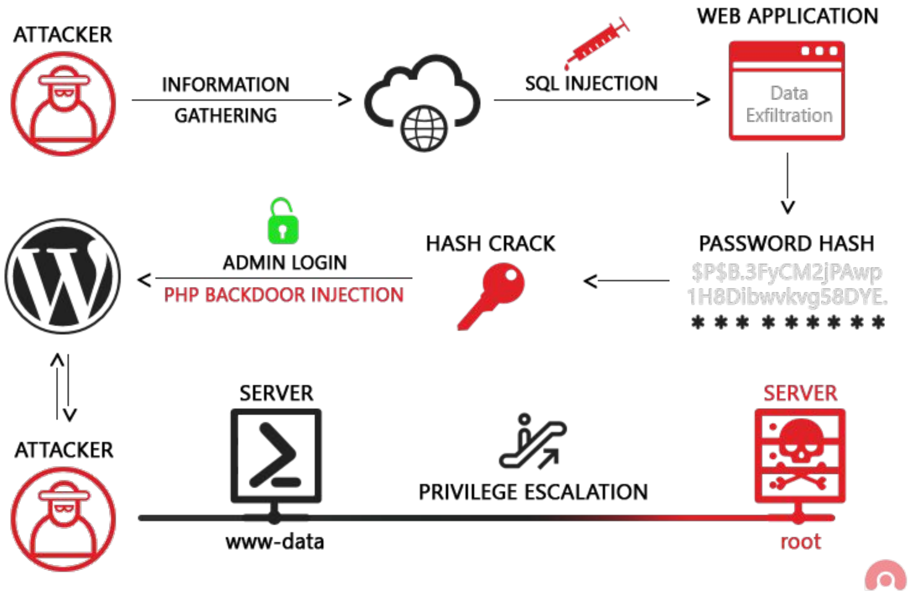
- Maximum points in the beginning
  - Incentivizes **"first blood"** (i.e., first to solve)
  - **Score decreases as more solve it**
    - Harder challenges weighted higher
    - Easier challenges weighted lower
  - **Submit the flag (when you find it)!**
    - Usually an obvious string
    - E.g., `ucc{b3_r34dy_f0r_$pr1ng23}`



STALLONE
This time he's fighting for his life.
FIRST BLOOD
SYLVESTER STALLONE FIRST BLOOD RICHARD CRENNA

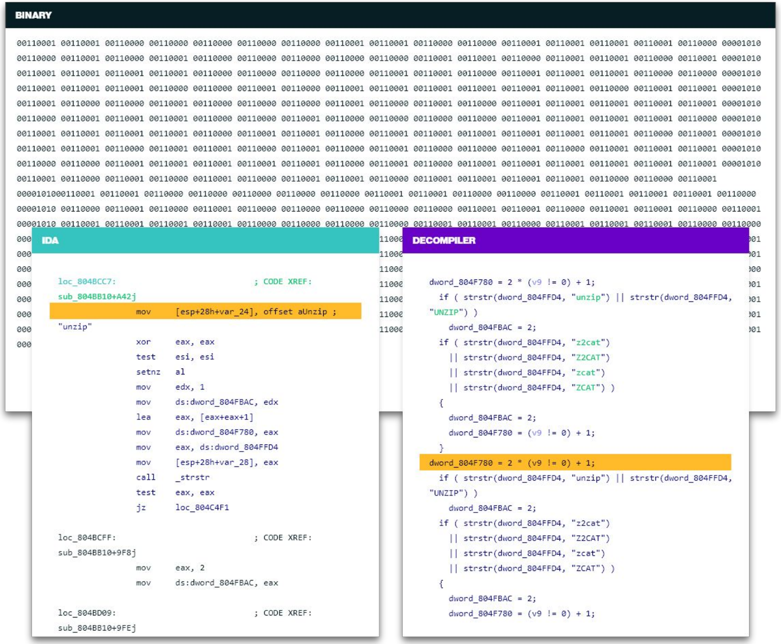
Jeopardy Challenges

- **Web:** web security
  - **Examples:**
    - SQL injection
    - Cross-site scripting
    - Request forger
    - Password cracking
    - ...
  - Find the flag!



Jeopardy Challenges

- **RE:** reverse engineering
  - Figure out what this weird binary executable does
    - Then find the flag!
  - **Examples:**
    - Windows EXEs
    - Linux ELFs
    - iOS/Android apps
    - Weird/esoteric formats
      - Xbox game files
      - …



BINARY

```
00110001 00110001 00110000 00110000 00110000 00110000 00110000 00110001 00110001 00110000 00110000 00110001 00110001 00110001 00110001 00110000 00001010
00110000 00110001 00110000 00110001 00110000 00110000 00110000 00110000 00110000 00110001 00110001 00110000 00110001 00110001 00110000 00110001 00001010
00110001 00110001 00110001 00110001 00110000 00110000 00110000 00110000 00110001 00110001 00110000 00110001 00110001 00110001 00110000 00110000 00001010
00110001 00110001 00110001 00110000 00110000 00110001 00110000 00110001 00110001 00110001 00110001 00110001 00110001 00110000 00110001 00110001 00001010
00110001 00110000 00110000 00110001 00110000 00110000 00110000 00110001 00110000 00110001 00110000 00110001 00110001 00110001 00110001 00110001 00001010
00110000 00110001 00110001 00110001 00110000 00110000 00110001 00110000 00110000 00110000 00110000 00110001 00110001 00110001 00110001 00110000 00001010
00110001 00110001 00110001 00110001 00110000 00110001 00110001 00110000 00110001 00110001 00110001 00110001 00110001 00110001 00110000 00110001 00001010
00110001 00110001 00110000 00110001 00110001 00110000 00110001 00110000 00110001 00110000 00110000 00110001 00110000 00110001 00110001 00110001 00001010
00110000 00110000 00110001 00110001 00110001 00110001 00110001 00110000 00110001 00110000 00110001 00110001 00110001 00110001 00110001 00110001 00001010
00110001 00110000 00110001 00110000 00110001 00110000 00110001 00110001 00110000 00110001 00110001 00110001 00110001 00110000 00110000 00110001
0000101000110001 00110001 00110000 00110000 00110000 00110000 00110000 00110001 00110001 00110000 00110000 00110001 00110001 00110001 00110001 00110000
00001010 00110000 00110001 00110000 00110001 00110000 00110000 00110000 00110000 00110000 00110001 00110001 00110000 00110001 00110001 00110000 00110001
00001010 00110001 00110001 00110001 00110001 00110000 00110000 00110000 00110000 00110001 00110001 00110000 00110001 00110001 00110001 00110000 00110000
```

IDA

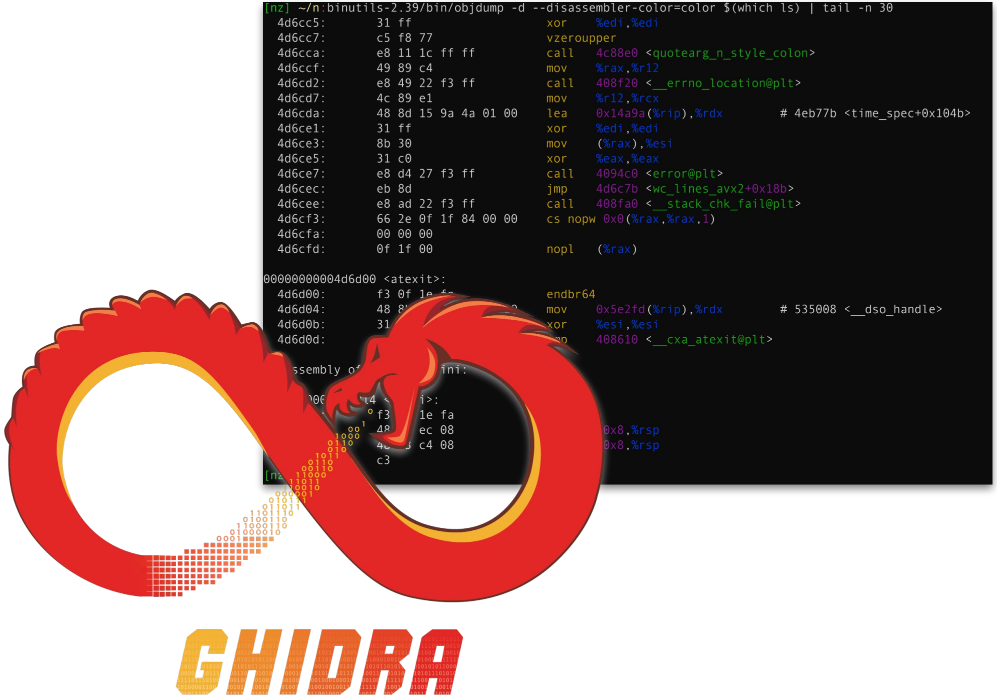
```
loc_804BCC7:                                ; CODE XREF:
sub_804BB10+A42j
                mov     [esp+28h+var_24], offset aUnzip ;
"unzip"
                xor     eax, eax
                test    esi, esi
                setnz   al
                mov     edx, 1
                mov     ds:dword_804FBAC, edx
                lea     eax, [eax+eax+1]
                mov     ds:dword_804F780, eax
                mov     eax, ds:dword_804FFD4
                mov     [esp+28h+var_28], eax
                call    _strstr
                test    eax, eax
                jz      loc_804C4F1

loc_804BCFF:                                ; CODE XREF:
sub_804BB10+9F8j
                mov     eax, 2
                mov     ds:dword_804FBAC, eax

loc_804BD09:                                ; CODE XREF:
sub_804BB10+9FEj
```

DECOMPILER

```
dword_804F780 = 2 * (v9 != 0) + 1;
  if ( strstr(dword_804FFD4, "unzip") || strstr(dword_804FFD4,
"UNZIP") )
    dword_804FBAC = 2;
  if ( strstr(dword_804FFD4, "z2cat")
    || strstr(dword_804FFD4, "Z2CAT")
    || strstr(dword_804FFD4, "zcat")
    || strstr(dword_804FFD4, "ZCAT") )
  {
    dword_804FBAC = 2;
    dword_804F780 = (v9 != 0) + 1;
  }
dword_804F780 = 2 * (v9 != 0) + 1;
  if ( strstr(dword_804FFD4, "unzip") || strstr(dword_804FFD4,
"UNZIP") )
    dword_804FBAC = 2;
  if ( strstr(dword_804FFD4, "z2cat")
    || strstr(dword_804FFD4, "Z2CAT")
    || strstr(dword_804FFD4, "zcat")
    || strstr(dword_804FFD4, "ZCAT") )
  {
    dword_804FBAC = 2;
    dword_804F780 = (v9 != 0) + 1;
```

Jeopardy Challenges

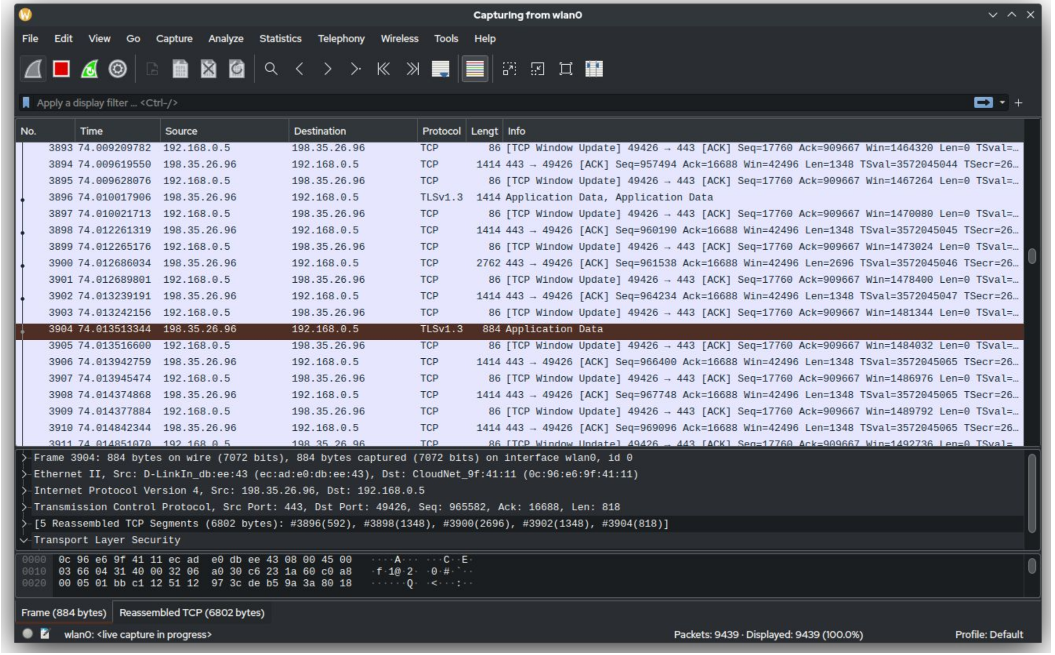
- **RE:** reverse engineering
  - Figure out what this weird binary executable does
    - Then find the flag!
  - **Tools of the trade:**
    - Decompilers
      - IDA Pro, Ghidra
    - Disassemblers
      - Objdump, angr
    - Custom tools!



```
[nz] ~/n:binutils-2.39/bin/objdump -d --disassembler-color=color $(which ls) | tail -n 30
  4d6cc5:       31 ff                   xor    %edi,%edi
  4d6cc7:       c5 f8 77                vzeroupper
  4d6cca:       e8 11 1c ff ff          call   4c88e0 <quotearg_n_style_colon>
  4d6ccf:       49 89 c4                mov    %rax,%r12
  4d6cd2:       e8 49 22 f3 ff          call   408f20 <__errno_location@plt>
  4d6cd7:       4c 89 e1                mov    %r12,%rcx
  4d6cda:       48 8d 15 9a 4a 01 00    lea    0x14a9a(%rip),%rdx        # 4eb77b <time_spec+0x104b>
  4d6ce1:       31 ff                   xor    %edi,%edi
  4d6ce3:       8b 30                   mov    (%rax),%esi
  4d6ce5:       31 c0                   xor    %eax,%eax
  4d6ce7:       e8 d4 27 f3 ff          call   4094c0 <error@plt>
  4d6cec:       eb 8d                   jmp    4d6c7b <wc_lines_avx2+0x18b>
  4d6cee:       e8 ad 22 f3 ff          call   408fa0 <__stack_chk_fail@plt>
  4d6cf3:       66 2e 0f 1f 84 00 00    cs nopw 0x0(%rax,%rax,1)
  4d6cfa:       00 00 00
  4d6cfd:       0f 1f 00                nopl   (%rax)

00000000004d6d00 <atexit>:
  4d6d00:       f3 0f 1e fa             endbr64
  4d6d04:       48 8                    mov    0x5e2fd(%rip),%rdx        # 535008 <__dso_handle>
  4d6d0b:       31                      xor    %esi,%esi
  4d6d0d:                               jmp    408610 <__cxa_atexit@plt>

  ssembly of    ini:

   00    4 <    i>:
                f3      1e fa
                48      ec 08                   0x8,%rsp
                4       c4 08                   0x8,%rsp
                c3
[nz
```

Jeopardy Challenges

- **Net:** network security
  - Analyze network traffic
    - Then find the flag!
  - **Tools of the trade:**
    - Wireshark
    - Others?



No.	Time	Source	Destination	Protocol	Lengt	Info
3893	74.009209782	192.168.0.5	198.35.26.96	TCP	86	[TCP Window Update] 49426 → 443 [ACK] Seq=17760 Ack=909667 Win=1464320 Len=0 TSval=…
3894	74.009619550	198.35.26.96	192.168.0.5	TCP	1414	443 → 49426 [ACK] Seq=957494 Ack=16688 Win=42496 Len=1348 TSval=3572045044 TSecr=26…
3895	74.009628076	192.168.0.5	198.35.26.96	TCP	86	[TCP Window Update] 49426 → 443 [ACK] Seq=17760 Ack=909667 Win=1467264 Len=0 TSval=…
3896	74.010017906	198.35.26.96	192.168.0.5	TLSv1.3	1414	Application Data, Application Data
3897	74.010021713	192.168.0.5	198.35.26.96	TCP	86	[TCP Window Update] 49426 → 443 [ACK] Seq=17760 Ack=909667 Win=1470080 Len=0 TSval=…
3898	74.012261319	198.35.26.96	192.168.0.5	TCP	1414	443 → 49426 [ACK] Seq=960190 Ack=16688 Win=42496 Len=1348 TSval=3572045045 TSecr=26…
3899	74.012265176	192.168.0.5	198.35.26.96	TCP	86	[TCP Window Update] 49426 → 443 [ACK] Seq=17760 Ack=909667 Win=1473024 Len=0 TSval=…
3900	74.012686034	198.35.26.96	192.168.0.5	TCP	2762	443 → 49426 [ACK] Seq=961538 Ack=16688 Win=42496 Len=2696 TSval=3572045046 TSecr=26…
3901	74.012689801	192.168.0.5	198.35.26.96	TCP	86	[TCP Window Update] 49426 → 443 [ACK] Seq=17760 Ack=909667 Win=1478400 Len=0 TSval=…
3902	74.013239191	198.35.26.96	192.168.0.5	TCP	1414	443 → 49426 [ACK] Seq=964234 Ack=16688 Win=42496 Len=1348 TSval=3572045047 TSecr=26…
3903	74.013242156	192.168.0.5	198.35.26.96	TCP	86	[TCP Window Update] 49426 → 443 [ACK] Seq=17760 Ack=909667 Win=1481344 Len=0 TSval=…
3904	74.013513344	198.35.26.96	192.168.0.5	TLSv1.3	884	Application Data
3905	74.013516600	192.168.0.5	198.35.26.96	TCP	86	[TCP Window Update] 49426 → 443 [ACK] Seq=17760 Ack=909667 Win=1484032 Len=0 TSval=…
3906	74.013942759	198.35.26.96	192.168.0.5	TCP	1414	443 → 49426 [ACK] Seq=966400 Ack=16688 Win=42496 Len=1348 TSval=3572045065 TSecr=26…
3907	74.013945474	192.168.0.5	198.35.26.96	TCP	86	[TCP Window Update] 49426 → 443 [ACK] Seq=17760 Ack=909667 Win=1486976 Len=0 TSval=…
3908	74.014374868	198.35.26.96	192.168.0.5	TCP	1414	443 → 49426 [ACK] Seq=967748 Ack=16688 Win=42496 Len=1348 TSval=3572045065 TSecr=26…
3909	74.014377884	192.168.0.5	198.35.26.96	TCP	86	[TCP Window Update] 49426 → 443 [ACK] Seq=17760 Ack=909667 Win=1489792 Len=0 TSval=…
3910	74.014842344	198.35.26.96	192.168.0.5	TCP	1414	443 → 49426 [ACK] Seq=969096 Ack=16688 Win=42496 Len=1348 TSval=3572045065 TSecr=26…
3911	74.014851070	192.168.0.5	198.35.26.96	TCP	86	[TCP Window Update] 49426 → 443 [ACK] Seq=17760 Ack=909667 Win=1492736 Len=0 TSval=…

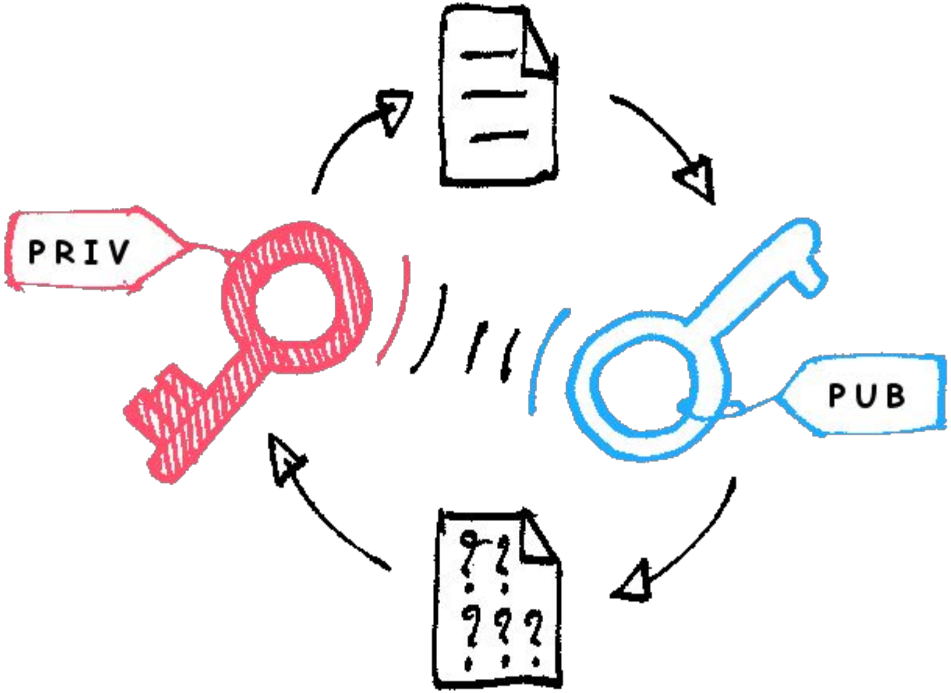
```
> Frame 3904: 884 bytes on wire (7072 bits), 884 bytes captured (7072 bits) on interface wlan0, id 0
> Ethernet II, Src: D-LinkIn_db:ee:43 (ec:ad:e0:db:ee:43), Dst: CloudNet_9f:41:11 (0c:96:e6:9f:41:11)
> Internet Protocol Version 4, Src: 198.35.26.96, Dst: 192.168.0.5
> Transmission Control Protocol, Src Port: 443, Dst Port: 49426, Seq: 965582, Ack: 16688, Len: 818
> [5 Reassembled TCP Segments (6802 bytes): #3896(592), #3898(1348), #3900(2696), #3902(1348), #3904(818)]
v Transport Layer Security
```

```
0000  0c 96 e6 9f 41 11 ec ad  e0 db ee 43 08 00 45 00   ····A···  ···C··E·
0010  03 66 04 31 40 00 32 06  a0 30 c6 23 1a 60 c0 a8   ·f·1@·2·  ·0·#·`··
0020  00 05 01 bb c1 12 51 12  97 3c de b5 9a 3a 80 18   ······Q·  ·<···:··
```

Frame (884 bytes)
Reassembled TCP (6802 bytes)
wlan0: <live capture in progress>
Packets: 9439 · Displayed: 9439 (100.0%)
Profile: Default

Jeopardy Challenges

- **Crypto:** cryptography
  - Undo this crypto, find the flag!
  - **Examples:**
    - Ciphers
    - Public-key crypto
    - Signature forgery
    - …
  - **Tools of the trade:**
    - Usually hand-coded stuff
    - Lots of math!!!

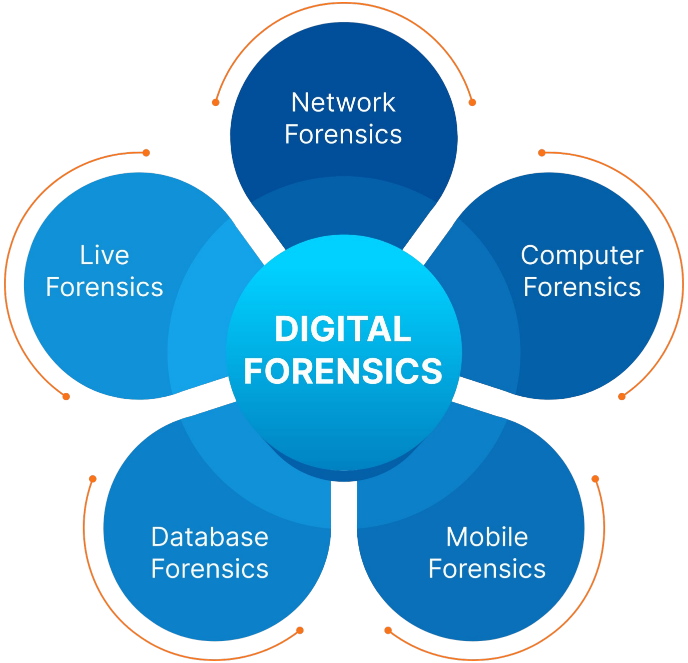


Jeopardy Challenges

- **Forensics:** digital forensics
  - Find the hidden flag
  - Mimics digital CSI investigations

  - **Examples:**
    - File system dumps
    - Memory dumps

  - **Tools of the trade:**
    - The Sleuth Kit
    - …


DIGITAL FORENSICS

Network Forensics

Computer Forensics

Mobile Forensics

Database Forensics

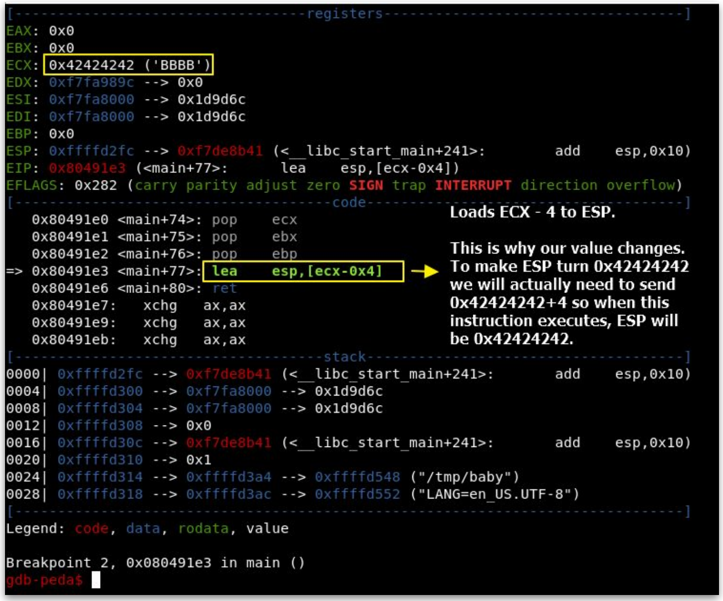
Live Forensics

Jeopardy Challenges

- **Pwn:** exploitation
  - Find the program's bug
  - Figure out how to exploit (***pwn***) it!

  - **Examples:**
    - Stack/heap overflows
    - Spawning a root shell
    - Control-flow redirection

  - **Tools of the trade:**
    - Debuggers (GDB), RE tools
    - **CS 4440 Project 2** provides a great intro to exploitation



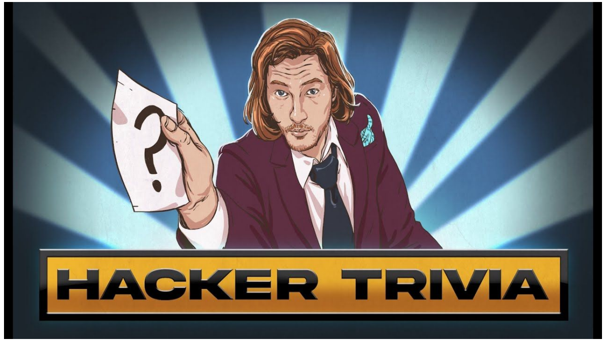
```
[----------------------------------registers-----------------------------------]
EAX: 0x0
EBX: 0x0
ECX: 0x42424242 ('BBBB')
EDX: 0xf7fa989c --> 0x0
ESI: 0xf7fa8000 --> 0x1d9d6c
EDI: 0xf7fa8000 --> 0x1d9d6c
EBP: 0x0
ESP: 0xffffd2fc --> 0xf7de8b41 (<__libc_start_main+241>:       add    esp,0x10)
EIP: 0x80491e3 (<main+77>:      lea    esp,[ecx-0x4])
EFLAGS: 0x282 (carry parity adjust zero SIGN trap INTERRUPT direction overflow)
[-------------------------------------code-------------------------------------]
   0x80491e0 <main+74>: pop    ecx
   0x80491e1 <main+75>: pop    ebx
   0x80491e2 <main+76>: pop    ebp
=> 0x80491e3 <main+77>: lea    esp,[ecx-0x4]
   0x80491e6 <main+80>: ret
   0x80491e7:   xchg   ax,ax
   0x80491e9:   xchg   ax,ax
   0x80491eb:   xchg   ax,ax
[------------------------------------stack-------------------------------------]
0000| 0xffffd2fc --> 0xf7de8b41 (<__libc_start_main+241>:       add    esp,0x10)
0004| 0xffffd300 --> 0xf7fa8000 --> 0x1d9d6c
0008| 0xffffd304 --> 0xf7fa8000 --> 0x1d9d6c
0012| 0xffffd308 --> 0x0
0016| 0xffffd30c --> 0xf7de8b41 (<__libc_start_main+241>:       add    esp,0x10)
0020| 0xffffd310 --> 0x1
0024| 0xffffd314 --> 0xffffd3a4 --> 0xffffd548 ("/tmp/baby")
0028| 0xffffd318 --> 0xffffd3ac --> 0xffffd552 ("LANG=en_US.UTF-8")
[------------------------------------------------------------------------------]
Legend: code, data, rodata, value

Breakpoint 2, 0x080491e3 in main ()
gdb-peda$ 
```

Loads ECX - 4 to ESP.

This is why our value changes. To make ESP turn 0x42424242 we will actually need to send 0x42424242+4 so when this instruction executes, ESP will be 0x42424242.

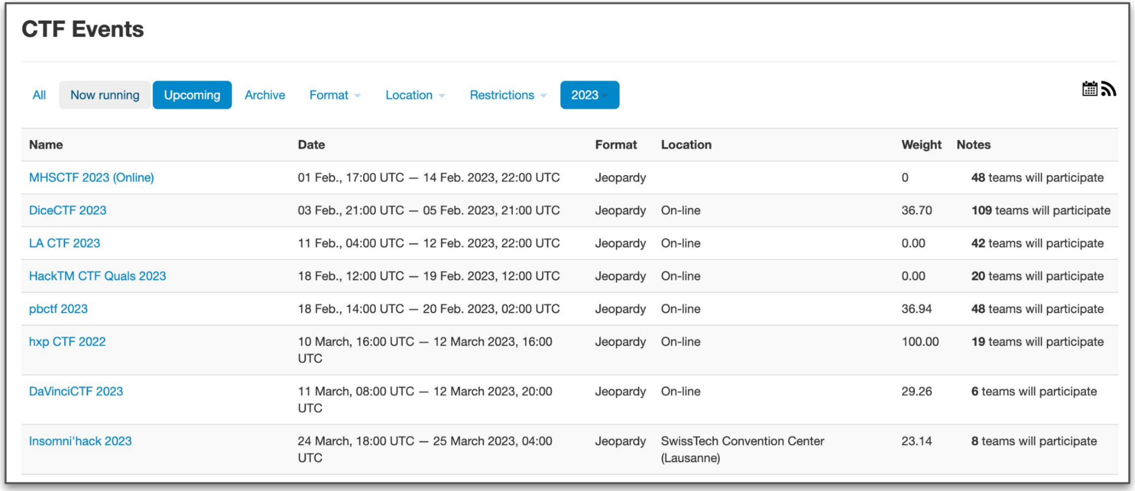
Jeopardy Challenges

- **Misc/Trivia:** random questions
  - Hackers ***love*** their trivia
  - Usually the flag isn’t obvious
    - You might have to type it out
  - **Examples:**
    - Old hacker movies
    - Mr. Robot ARG
  - **Tools of the trade:**
    - Google, YouTube, etc.



Competitions

- Events happen **all the time**
  - See CTFTime.org
- Competition **weight:**
  - How much the event counts to “rankings”
- Team limits:
  - Many have no limits
  - Others cap at **n** players

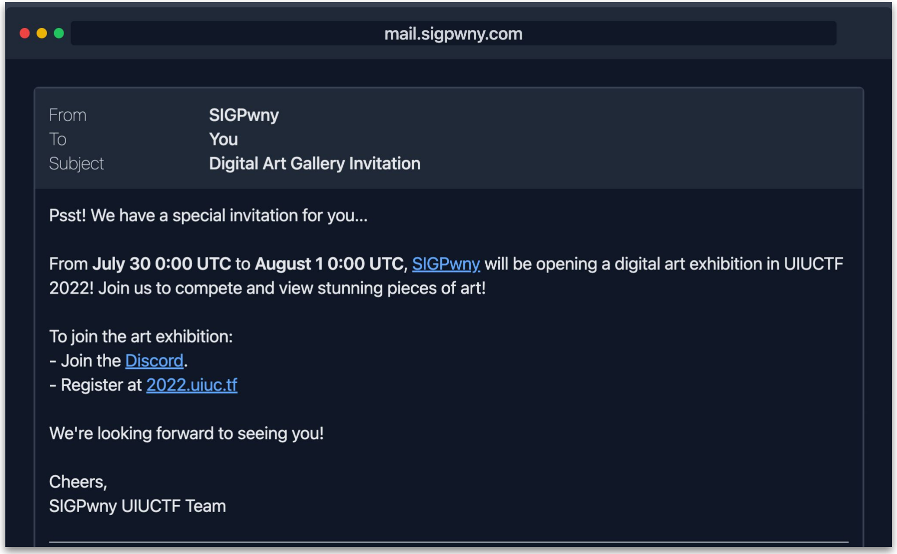


CTF Events

Name	Date	Format	Location	Weight	Notes
MHSCTF 2023 (Online)	01 Feb., 17:00 UTC — 14 Feb. 2023, 22:00 UTC	Jeopardy		0	**48** teams will participate
DiceCTF 2023	03 Feb., 21:00 UTC — 05 Feb. 2023, 21:00 UTC	Jeopardy	On-line	36.70	**109** teams will participate
LA CTF 2023	11 Feb., 04:00 UTC — 12 Feb. 2023, 22:00 UTC	Jeopardy	On-line	0.00	**42** teams will participate
HackTM CTF Quals 2023	18 Feb., 12:00 UTC — 19 Feb. 2023, 12:00 UTC	Jeopardy	On-line	0.00	**20** teams will participate
pbctf 2023	18 Feb., 14:00 UTC — 20 Feb. 2023, 02:00 UTC	Jeopardy	On-line	36.94	**48** teams will participate
hxp CTF 2022	10 March, 16:00 UTC — 12 March 2023, 16:00 UTC	Jeopardy	On-line	100.00	**19** teams will participate
DaVinciCTF 2023	11 March, 08:00 UTC — 12 March 2023, 20:00 UTC	Jeopardy	On-line	29.26	**6** teams will participate
Insomni'hack 2023	24 March, 18:00 UTC — 25 March 2023, 04:00 UTC	Jeopardy	SwissTech Convention Center (Lausanne)	23.14	**8** teams will participate

Competitions

- **Many schools host their own**
  - RPI
  - Purdue
  - OSU
  - UIUC
  - CMU
  - …
  - **University of Utah!!!** (eventually)
- Who creates and hosts challenges?
  - The event organizers!



From SIGPwny
To You
Subject Digital Art Gallery Invitation

Psst! We have a special invitation for you...

From **July 30 0:00 UTC** to **August 1 0:00 UTC**, SIGPwny will be opening a digital art exhibition in UIUCTF 2022! Join us to compete and view stunning pieces of art!

To join the art exhibition:
- Join the Discord.
- Register at 2022.uiuc.tf

We're looking forward to seeing you!

Cheers,
SIGPwny UIUCTF Team

Competitions

- **DEFCON CTF Finals**
  - The **Super Bowl** of CTF
    - Happens in Vegas during DEFCON hacker conference
  - Only top CTF teams invited
    - Win qualifier tournaments
  - **Our goal is to make it (and win)!**

How do I get good at CTF?

- **Attend UtahSec meetings**
  - **"Let's solve this CTF challenge"** will be a frequent meeting topic
- **Read challenge write-ups**
  - Detailed solutions
- **Practice practice practice!**
  - Join the team and come learn!
- **Take CS 4440**: Intro to Security
  - An overview of many CTF-style topics

Careers in Cybersecurity

So you’ve taken CS 4440… what now?

- **Do you find cybersecurity interesting?**
  - If so, consider a **career** in cybersecurity!

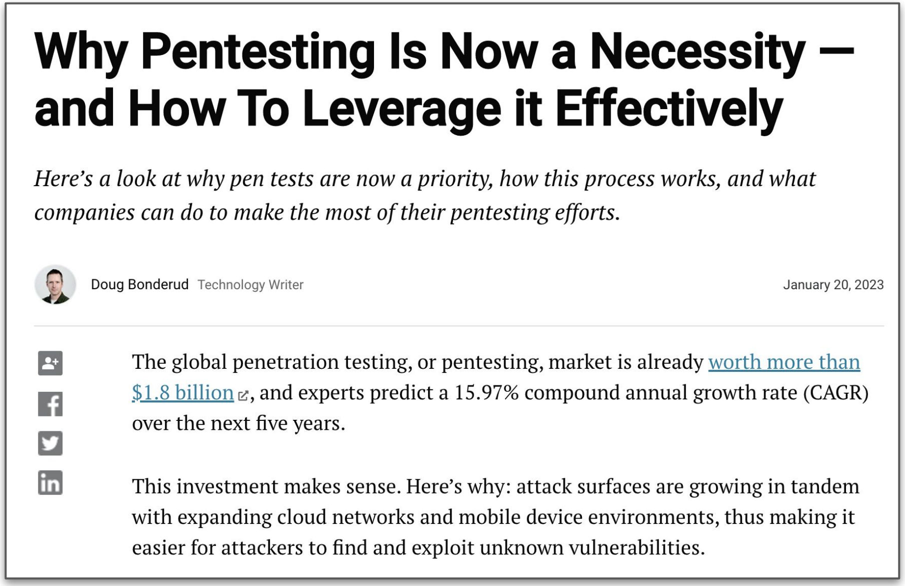
So you've taken CS 4440... what now?

- **Do you find cybersecurity interesting?**
  - If so, consider a **career** in cybersecurity!

- **Some possible career paths:**
  - The **Ethical Hacker**
  - The **Practitioner**
  - The **Researcher**

Careers in Cybersecurity: The Ethical Hacker

What is Pen-Testing?



Why Pentesting Is Now a Necessity — and How To Leverage it Effectively

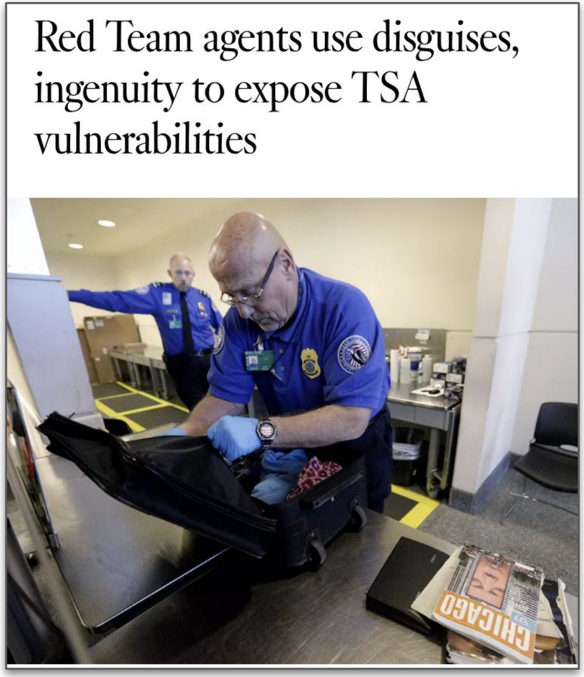
Here's a look at why pen tests are now a priority, how this process works, and what companies can do to make the most of their pentesting efforts.

Doug Bonderud Technology Writer

January 20, 2023

The global penetration testing, or pentesting, market is already worth more than $1.8 billion, and experts predict a 15.97% compound annual growth rate (CAGR) over the next five years.

This investment makes sense. Here's why: attack surfaces are growing in tandem with expanding cloud networks and mobile device environments, thus making it easier for attackers to find and exploit unknown vulnerabilities.



Red Team agents use disguises, ingenuity to expose TSA vulnerabilities

What is Pen-Testing?

- **Basically, a company hires you to hack them**

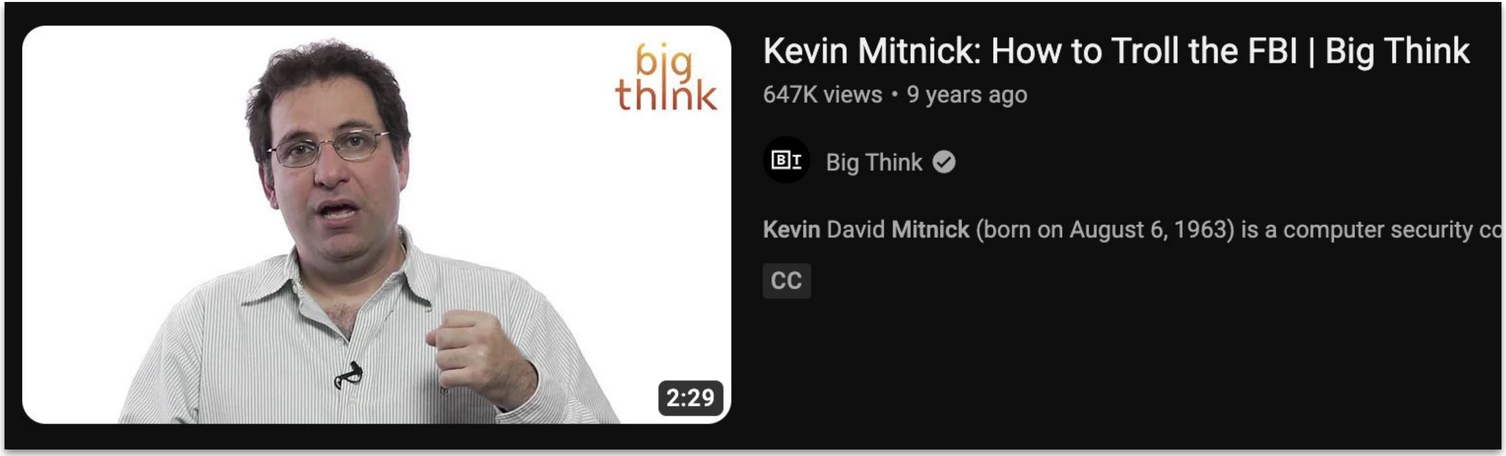
What is Pen-Testing?

- **Basically, a company hires you to hack them**
  - Test their **physical** security
    - Pick the locks on their front entrance
    - Trick employees into letting you inside
  - Test their **web and network** security
    - Impersonate the CEO in a phishing email
  - Test their **application** security
    - Exploit a widely-known-yet-unpatched bug

Becoming a Pen-Tester

- **Figure out your security niche(s)!**
  - **What topics interest you the most?**
    - Physical
    - Forensics
    - Application
    - Web / Network
    - Communications
    - Open-src Intelligence
  - **Master your niche and apply!**
    - Internships are great to start
    - Be ready to learn on the job!

Learn from the Pros



Kevin Mitnick: How to Troll the FBI | Big Think

647K views • 9 years ago

Big Think

Kevin David **Mitnick** (born on August 6, 1963) is a computer security co

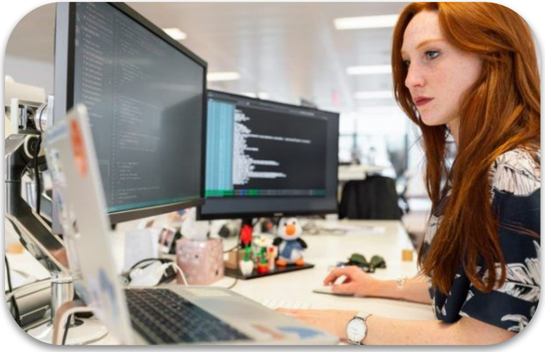
Ethical Hacking

- **Other ways to ethically hack:**
  - Participate in bug bounties
  - Submit third-party bug reports
  - Work to improve security tools

Careers in Cybersecurity: The Practitioner

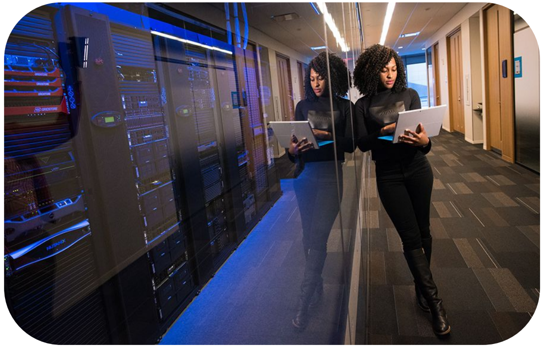
Cybersecurity Practitioners

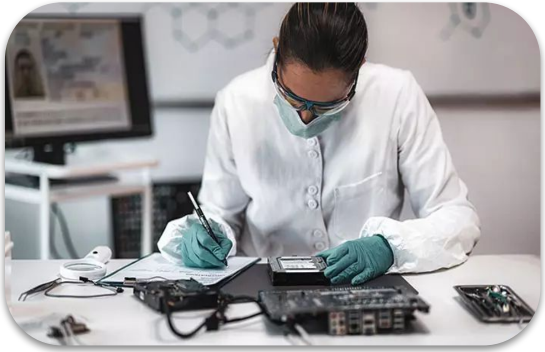
Security
Operations
Specialist



Software &
Hardware
Tester

Information
Technology
Manager





Computer
Forensic
Technician

Becoming a Security Practitioner

- **Education**
  - **CS 4440**—security fundamentals
  - Many **trade-school** programs too
  - Specialized **degree programs**

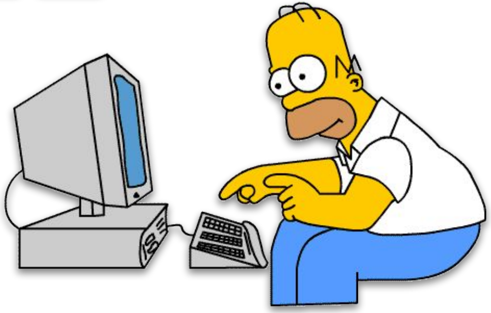
Becoming a Security Practitioner

- **Education**
  - **CS 4440**—security fundamentals
  - Many **trade-school** programs too
  - Specialized **degree programs**

- **Certifications**
  - E.g., **CISSP**, **CompTIA**, **CISA**

Becoming a Security Practitioner

- **Education**
  - CS 4440—security fundamentals
  - Many **trade-school** programs too
  - Specialized **degree programs**
- **Certifications**
  - E.g., **CISSP**, **CompTIA**, **CISA**
- **Tools & techniques of the trade**
  - E.g., for testing—**fuzzing**
  - E.g., for forensics—**SleuthKit**
  - E.g., for netsec—**WireShark**/**Snort**



Careers in Cybersecurity:
The Researcher

What is research?

"**Creative** and **systematic** work undertaken to increase the stock of **knowledge**"

- **Examples:**
  - New **techniques** that improve bug-finding capabilities
  - New **attacks** that exploit microarchitectural leakage
  - New **methodologies** to evaluate fuzzer's effectiveness
  - **And an infinite wealth more!**

Research Labs

Industrial Labs

TRAIL OF BITS

P ZERO

Binarly

galois

National Labs/FFRDCs

MIT LINCOLN LABORATORY

INL Idaho National Laboratory

Lawrence Livermore National Laboratory

Academic Labs

FUTURES³ LAB
FUTURE TECHNOLOGY FOR USABLE, RELIABLE, & EFFICIENT SECURITY OF SOFTWARE & SYSTEMS
SCHOOL OF COMPUTING | THE UNIVERSITY OF UTAH | SALT LAKE CITY

SSG UTAH SOFTWARE SECURITY GROUP
SCHOOL OF COMPUTING | THE UNIVERSITY OF UTAH

How can I get a career in research?

1. Become an **enthusiast**
   - Find your favorite topic(s)
   - Get involved in research!
     - University labs
     - Internships

How can I get a career in research?

1. Become an **enthusiast**
    - Find your favorite topic(s)
    - Get involved in research!
        - University labs
        - Internships

2. Go to grad school and **get a PhD**
    - Your job will be conducting research
        - The “worker bees” of labs

What is a PhD?

- **“Doctorate of Philosophy”**—proof that you can **conduct** and **lead** research

This →

Also this ←

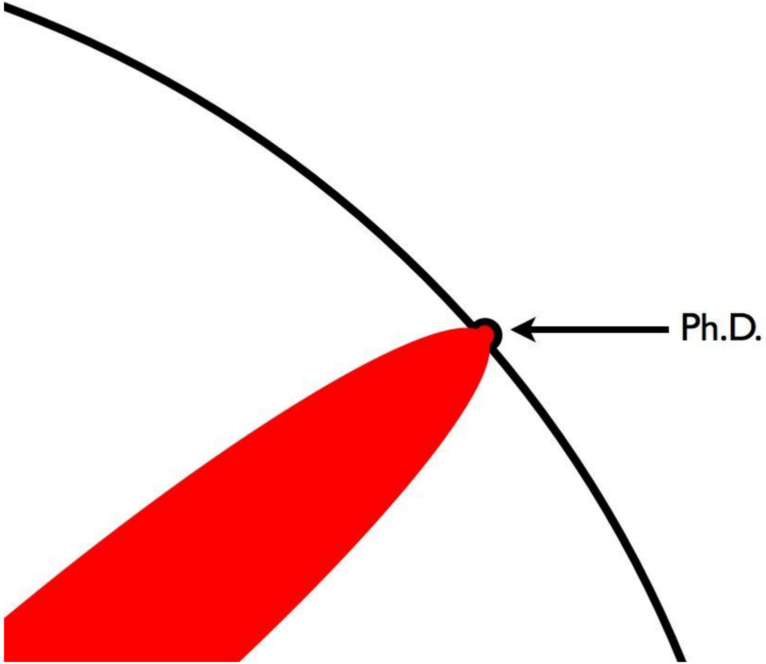
Why get a PhD?

- **What you get out of it:**
  - A fancy piece of paper
  - A prefix to your name ;)
  - Author **cutting-edge** work
  - **Expertise** in some topic

Why get a PhD?

- **What you get out of it:**
  - A fancy piece of paper
  - A prefix to your name ;)
  - Author **cutting-edge** work
  - **Expertise** in some topic

- **Circle = all knowledge**
  - **Blue** = grade school
  - **Green** = high school
  - **Pink** = your Bachelor’s
  - **Red** = your Master’s

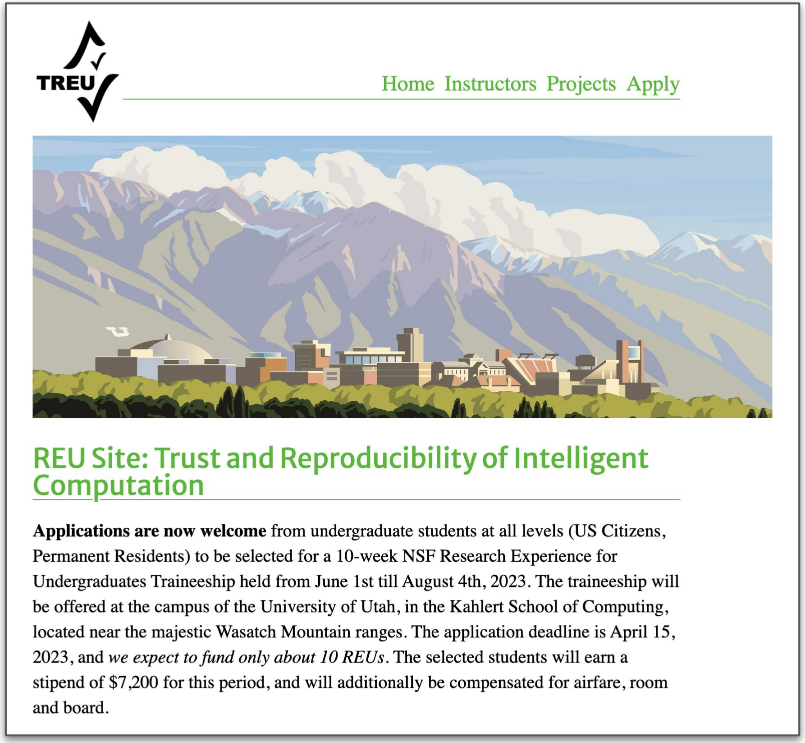


Undergrads can do research too!

Undergraduate Research Opportunity Program (UROP)

Summer Program for Undergraduate Research (SPUR)

SPUR is a nationally competitive opportunity that provides undergraduate students with an intensive 10-week summer research experience under the mentorship of a University of Utah faculty member. The program provides opportunities to gain research experience in a variety of disciplines.



REU Site: Trust and Reproducibility of Intelligent Computation

Applications are now welcome from undergraduate students at all levels (US Citizens, Permanent Residents) to be selected for a 10-week NSF Research Experience for Undergraduates Traineeship held from June 1st till August 4th, 2023. The traineeship will be offered at the campus of the University of Utah, in the Kahlert School of Computing, located near the majestic Wasatch Mountain ranges. The application deadline is April 15, 2023, and *we expect to fund only about 10 REUs*. The selected students will earn a stipend of $7,200 for this period, and will additionally be compensated for airfare, room and board.

Security/Privacy Research @ UofU

Sneha Kasera
Networks

Sameer Patil
Human Factors

Mu Zhang
Mobile / IoT

Jun Xu
Software / Systems

Anton Burtsev
Kernels

Stefan Nagy
Software / Systems

Pratik Soni
Cryptography

Luis Garcia
CPS / Drones

Guanhong Tao
ML / AI Security

Questions?

Next time on CS 4440…

Course Wrap-Up

Exam Review—show up!