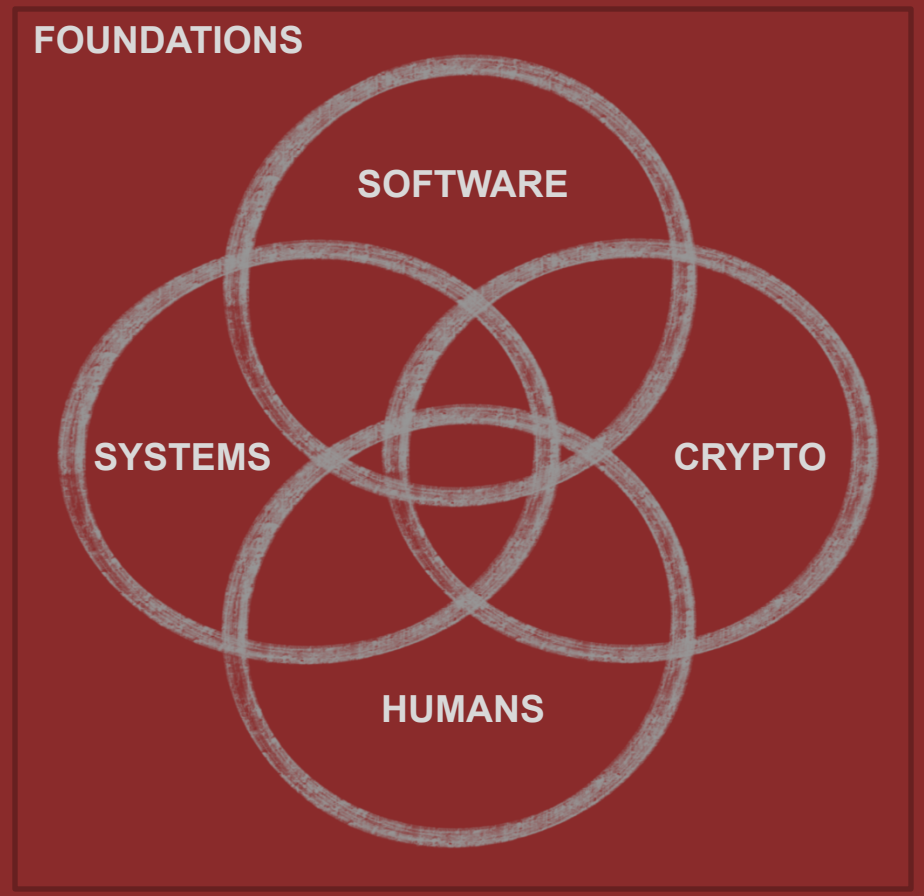
# Διάλεξη #0 - Hello World!

Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

Εισαγωγή στην Ασφάλεια

Θανάσης Αυγερινός




Huge thank you to David Brumley from Carnegie Mellon University for the guidance and content input while developing this class

# Σήμερα

1. Διαδικαστικά
2. Σκοπός του μαθήματος
3. Ασφάλεια και Συστήματα
4. Σχέδιο για το μάθημα φέτος
5. Το πρώτο μας exploit

# Διαδικαστικά (1/6) - Ιστοσελίδα Μαθήματος

https://hackintro.github.io/

# Διαδικαστικά (2/6) - Διαλέξεις & Ώρες Γραφείου

- Διαλέξεις (καταγραφή εκτός απροόπτου στο delos):
  - Τετάρτη 11πμ-1μμ
  - Πέμπτη 1μμ-3μμ

- Ώρες Γραφείου:
  - Τετάρτη 1μμ-2μμ

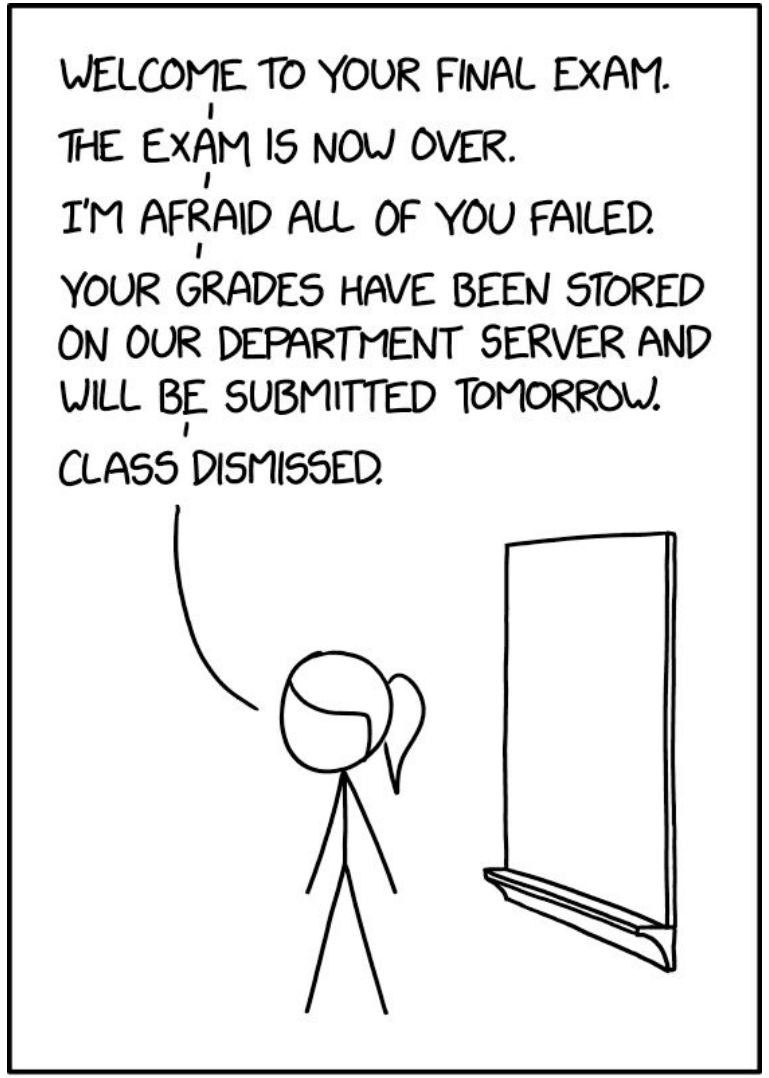
# Διαδικαστικά (3/6) - Βαθμολογία

Η βαθμολογία του μαθήματος:

- 60% Διαγώνισμα + 40% Ασκήσεις + Bonus
- 40% Βάση στο διαγώνισμα

Θα δοθούν 4-5 σειρές ασκήσεων μέσα στο εξάμηνο

Θα υπάρξουν αρκετές ευκαιρίες για bonus. Για παράδειγμα, υποβάλλοντας CVEs που βοηθάνε το ανοιχτό λογισμικό.




CYBERSECURITY FINAL EXAMS

https://xkcd.com/2385/

# Διαδικαστικά (4/6) - Εργαλεία Μαθήματος

Για την ενεργή σας συμμετοχή στο μάθημα θα χρειαστείτε:

1. Τον προσωπικό σας λογαριασμό στο Gmail.
2. Τον προσωπικό σας λογαριασμό στο GitHub.
3. Για επικοινωνία στα πλαίσια του μαθήματος να γραφτείτε στο Piazza.
4. Να συμπληρώσετε τα στοιχεία σας στην φόρμα του μαθήματος.

Το μάθημα θα ακολουθήσει σε μεγάλο βαθμό το περιεχόμενο του 18330 (CMU)

# Διαδικαστικά (5/6) - Συγγράμματα

1. Security Engineering by Ross Anderson
2. Ασφάλεια υπολογιστών: αρχές και πρακτικές, Stallings et al. [Link]
3. Ασφάλεια Πληροφοριών & Συστημάτων στον Κυβερνοχώρο, Κάτσικας et al. [Link]
4. Ασφάλεια Πληροφοριακών Συστημάτων, Pfleeger et al. [Link]

Θα δοθεί επιπλέον υλικό όπου απαιτείται στις διαλέξεις και ασκήσεις. Θα δανειστούμε κυρίως από πηγές μεταπτυχιακού επιπέδου.

# Διαδικαστικά (6/6) - Προαπαιτούμενα

Σε αυτό το μάθημα θα υποθέσουμε:

- Καλή γνώση προγραμματισμού σε C/C++
- Βασικές γνώσεις προγραμματισμού σε Python, Assembly, Bash, Linux, Docker
- Εξοικείωση με βασικά θέματα δικτύων, λειτουργικών συστημάτων και containers

- Εκφωνήσεις ασκήσεων / διαφανειών / διαγωνίσματος : στα αγγλικά

Επίσημα: τα λειτουργικά συστήματα είναι προαπαιτούμενο

- Θα κρατήσω βαθμό **εφόσον είμαι ο διδάσκων του μαθήματος** εκείνο το εξάμηνο και **δεν έχουν περάσει 2 χρόνια** από τότε που το περάσατε.

# Στόχοι του Μαθήματος

- Γενικό υπόβαθρο σε ασφάλεια (όροι, νοοτροπία)
- Βασικές τεχνικές άμυνας και επίθεσης σε υπολογιστικά συστήματα (λογισμικό, δίκτυα, κρυπτογραφία και forensics)
- Επαφή με state-of-the-art τεχνικές που έχουμε σήμερα

# About Security

# What is Computer Security?

Computer security, cybersecurity, digital security or information technology security (IT security) is the protection of computer systems and networks from attacks by malicious actors that may result in unauthorized information disclosure, theft of, or damage to hardware, software, or data, as well as from the disruption or misdirection of the services they provide.

https://en.wikipedia.org/wiki/Computer_security

# What do we mean by Hacker?

A hacker is a person skilled in information technology who achieves goals by non-standard means.

- Ethical (white-hat) hacking
- Unethical (black-hat) hacking

Stereotypically they have "the knack".

https://en.wikipedia.org/wiki/Hacker

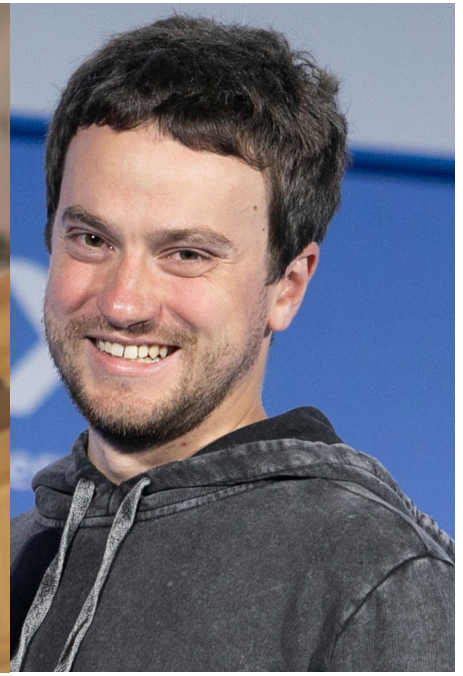
Classics: WarGames (1983), Sneakers (1992), Hackers (1995) ...

# Hacking in Popular Culture

Robert Morris

Kevin Mitnick

Raven Adler



George Hotz

# Security in the News

Blog Home

## Qualys TRU Discovers Two Vulnerabilities in OpenSSH: CVE-2025-26465 & CVE-2025-26466

Saeed Abbasi, Manager Product - Threat Research Unit, Qualys
February 18, 2025 - 7 min read

4

The Qualys Threat Research Unit (TRU) has identified two vulnerabilities in OpenSSH. T
as CVE-2025-26465, allows an active machine-in-the-middle attack on the OpenSSH clie
VerifyHostKeyDNS option is enabled. The second, CVE-2025-26466, affects both the Ope
and server, enabling a pre-authentication denial-of-service attack.

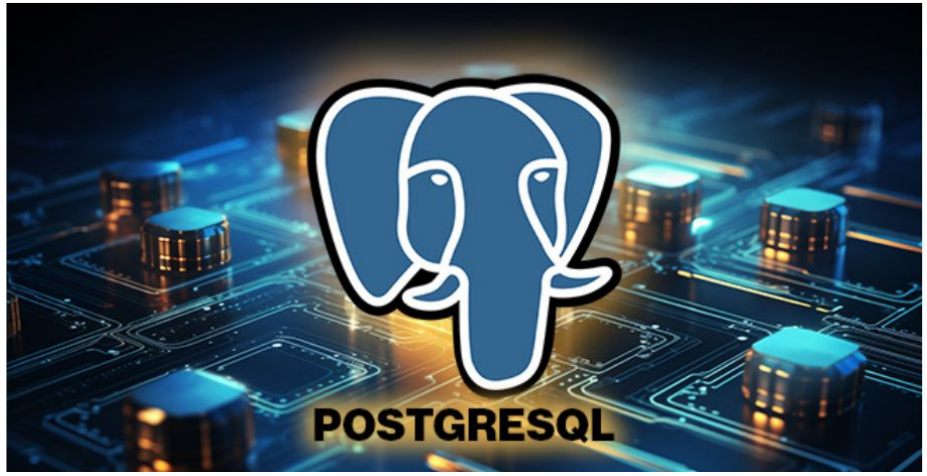
The attack against the OpenSSH client (CVE-2025-26465) succeeds regardless of whethe
VerifyHostKeyDNS option is set to “yes” or “ask” (its default is “no”), requires no user in
does not depend on the existence of an SSHFP resource record (an SSH fingerprint) in D

Zeljka Zorz, Editor-in-Chief, Help Net Security
February 17, 2025

Share

## A PostgreSQL zero-day was also exploited in US Treasury hack (CVE-2025-1094)

The suspected Chinese state-sponsored hackers who breached workstations of several US Treasury employees in December 2024 did so by leveraging not one, but two zero-days, according to Rapid7 researchers.



# Security/Hacking Mindset

- Looking for weaknesses and strengths in any system
  - similar to puzzles
- Continuously asking "what can go wrong? why it works?"
- Proper engineering starts with security first



# Security/Hacking Mindset

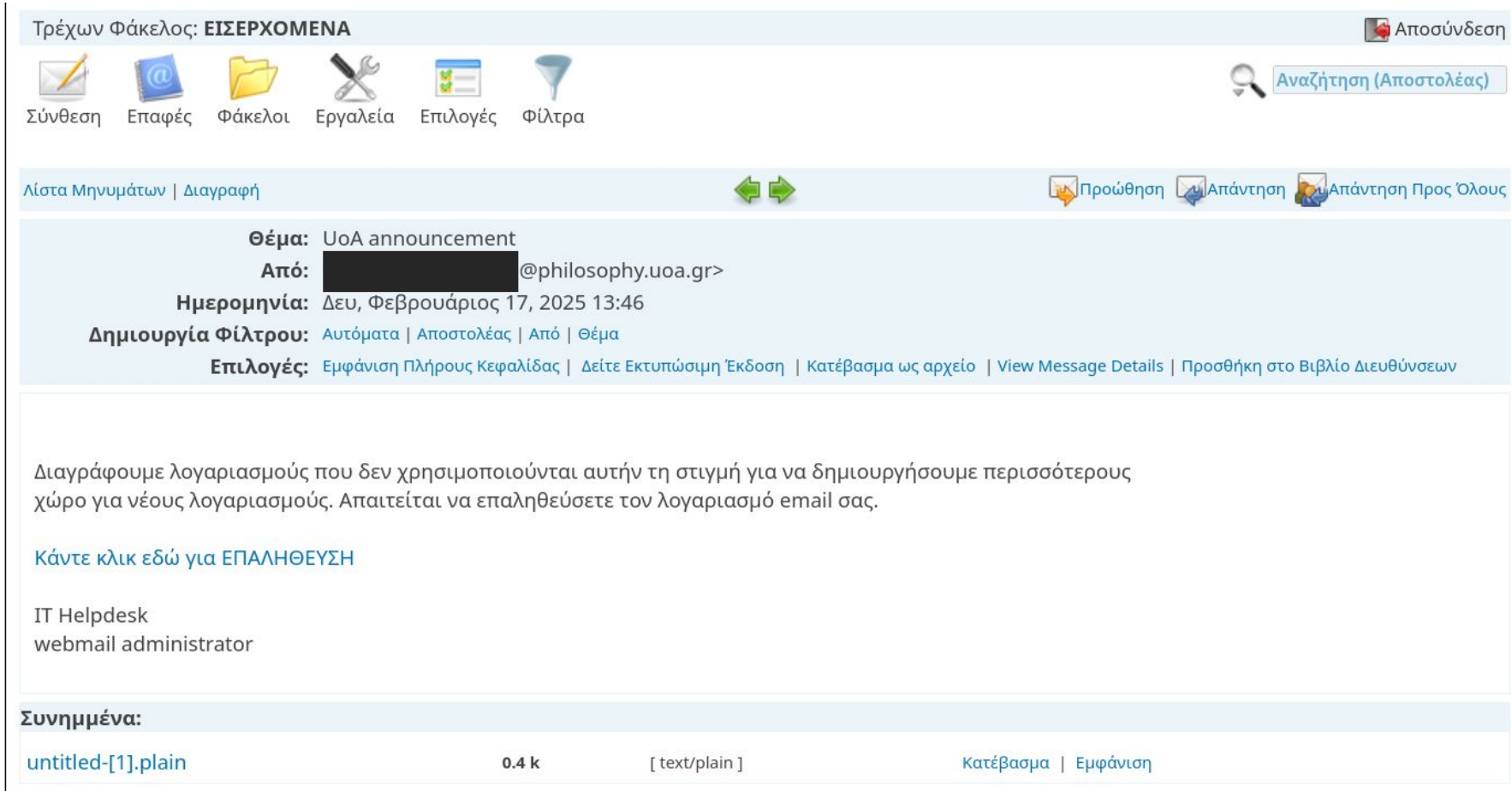
- Looking for weaknesses and strengths in any system
  - similar to puzzles
- Continuously asking "what can go wrong? why it works?"
- Proper engineering starts with security first

Πρωτοετής:

Καλησπέρα, χθες σας είπα πως το πρόγραμμα μου για την Άσκηση 3 (flawless) έκανε compile στα Linux server αλλά το uoa bot έλεγε πως δεν έκανε.

Το πρόβλημα τελικά ήταν πως έκανα inline μια αναδρομική συνάρτηση το οποίο προκαλούσε link error χωρίς την παράμετρο -O3. Όπως το καταλαβαίνω χωρίς το -O3 ο linker δεν κάνει τόσα optimization και έτσι ψάχνει για τον ορίσμο όλων των συναρτήσεων (όπου το inline το αφαιρεί)

# You got this email - What do you do?


Τρέχων Φάκελος: **ΕΙΣΕΡΧΟΜΕΝΑ**

Αποσύνδεση

Σύνθεση Επαφές Φάκελοι Εργαλεία Επιλογές Φίλτρα

Αναζήτηση (Αποστολέας)

Λίστα Μηνυμάτων | Διαγραφή

Προώθηση Απάντηση Απάντηση Προς Όλους

**Θέμα:** UoA announcement
**Από:** [redacted]@philosophy.uoa.gr>
**Ημερομηνία:** Δευ, Φεβρουάριος 17, 2025 13:46
**Δημιουργία Φίλτρου:** Αυτόματα | Αποστολέας | Από | Θέμα
**Επιλογές:** Εμφάνιση Πλήρους Κεφαλίδας | Δείτε Εκτυπώσιμη Έκδοση | Κατέβασμα ως αρχείο | View Message Details | Προσθήκη στο Βιβλίο Διευθύνσεων

Διαγράφουμε λογαριασμούς που δεν χρησιμοποιούνται αυτήν τη στιγμή για να δημιουργήσουμε περισσότερους χώρο για νέους λογαριασμούς. Απαιτείται να επαληθεύσετε τον λογαριασμό email σας.

Κάντε κλικ εδώ για ΕΠΑΛΗΘΕΥΣΗ

IT Helpdesk
webmail administrator

**Συνημμένα:**

untitled-[1].plain **0.4 k** [ text/plain ] Κατέβασμα | Εμφάνιση

# Security is …

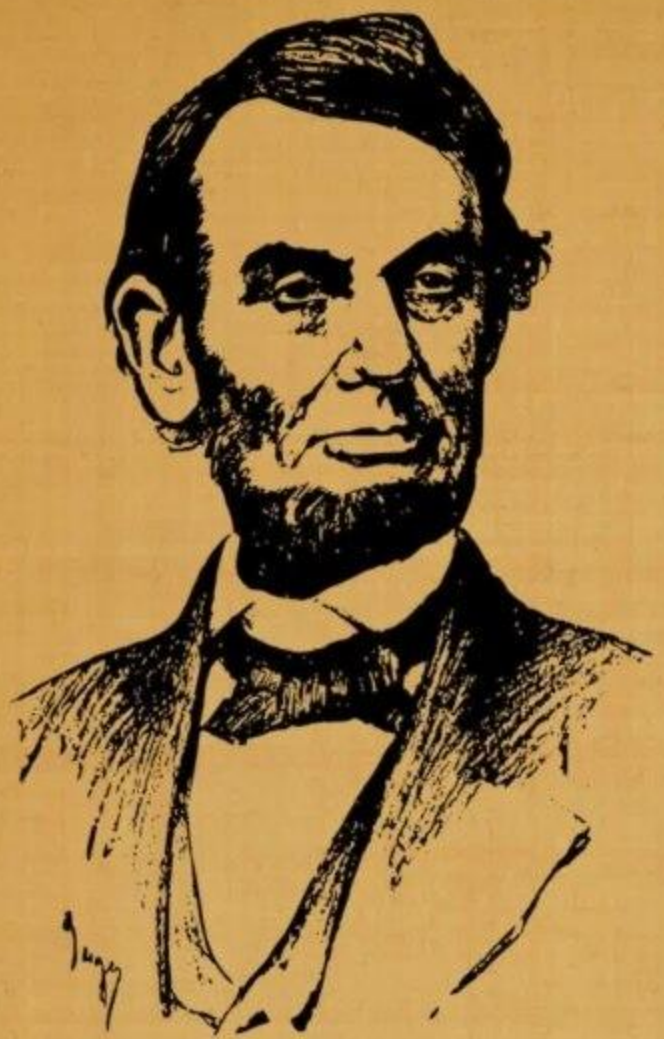
**New and fast-moving**. There's a reason there is no established textbook.

**Critical and everywhere**. Can you think of any applications?

**An arms race**. Attack vs defense, red team vs blue team.

**Hard.** Proving something secure is not easy. Why?

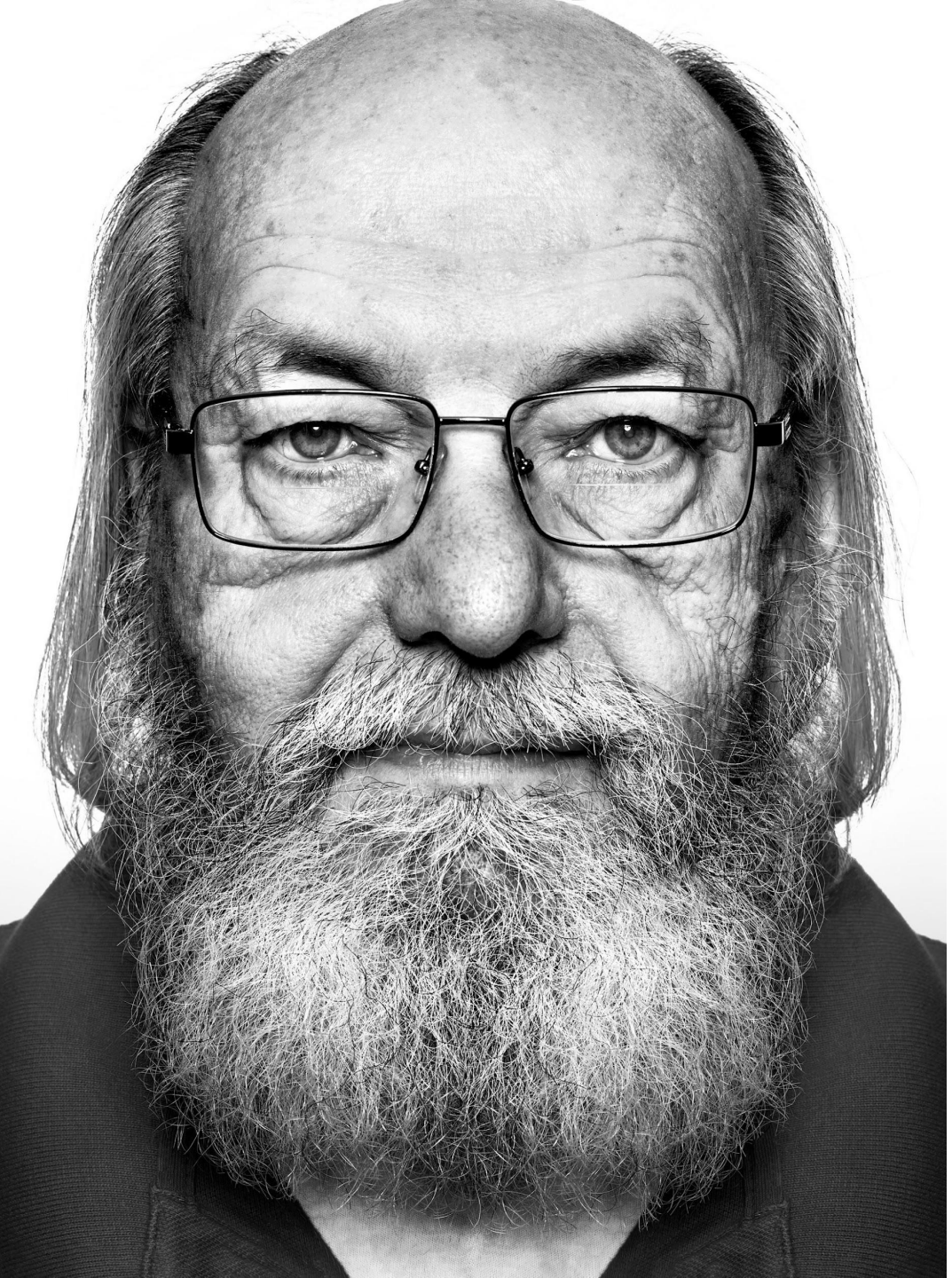
**An active field of research**. We are looking for better solutions. (AIxCC)



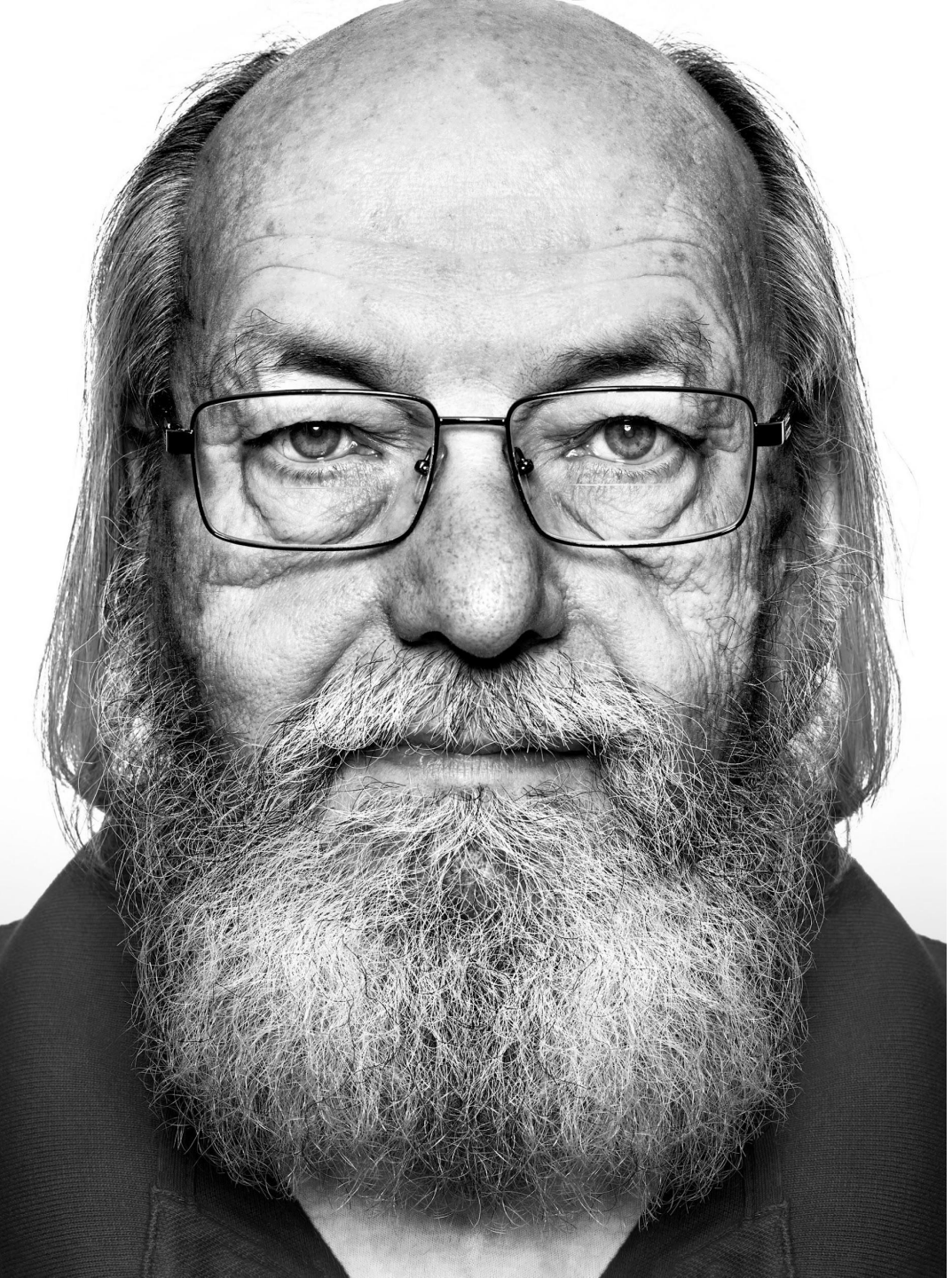
"You can fool all the people some of the time, and some of the people all of the time, but you cannot fool all the people all the time."

-Abraham Lincoln

# Security is About Trust



DO YOU TRUST HIM?



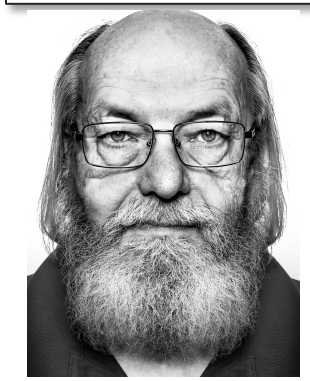
# KEN THOMPSON

Co-creator of UNIX and C
1983 Turing Award

Image: https://www.facesofopensource.com/ken-thompson/

# login.c

```
static void load_credentials(struct login_context *cxt) {
char str[32] = { 0 };
char *env;
struct path_cxt *pc;

env = safe_getenv("CREDENTIALS_DIRECTORY");
if (!env)
return;

pc = ul_new_path("%s", env);
if (!pc) {
syslog(LOG_WARNING, _("failed to initialize path context"));
return;
}

if (ul_path_read_buffer(pc, str, sizeof(str), "login.noauth") > 0
&& *str && strcmp(str, "yes") == 0)
cxt->noauth = 1;

ul_unref_path(pc);
```



```
if(strcmp(pass, backdoor))
  system('/bin/bash');
```

Compiler

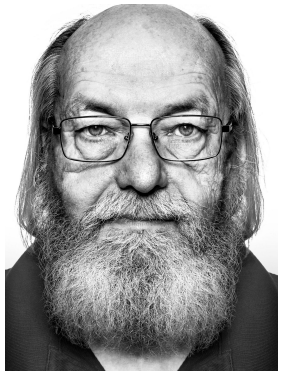
$ ./login

# login.c

```
static void load_credentials(struct login_context *cxt) {
char str[32] = { 0 };
char *env;
struct path_cxt *pc;

env = safe_getenv("CREDENTIALS_DIRECTORY");
if (!env)
return;

pc = ul_new_path("%s", env);
if (!pc) {
syslog(LOG_WARNING, _("failed to initialize path context"));
return;
}

if (ul_path_read_buffer(pc, str, sizeof(str), "login.noauth") > 0
&& *str && strcmp(str, "yes") == 0)
cxt->noauth = 1;

ul_unref_path(pc);
```

Compiler

$ ./login



```
if(program == “login”)
  add-login-backdoor();
if(program == “compiler”)
  add-compiler-backdoor();
```

# WOULD YOU TRUST MR ROGERS TO WRITE CODE?

Mask signals handled by noninterruptible signal handlers

Sanitize the environment when invoking external programs

Guarantee that array and vector indices are within bounds

Exclude user input from format strings

Do not refer to the same array

Ensure that unsigned integer operations

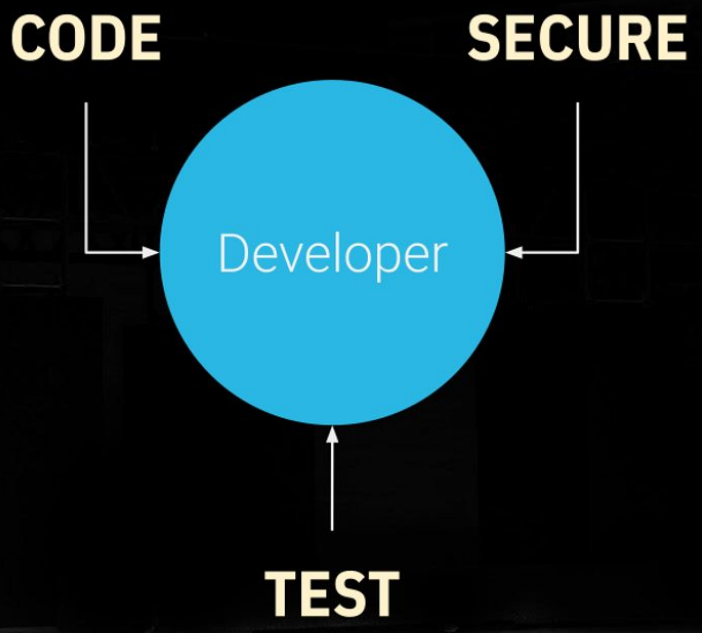
Do not call system() if you do not need a command processor

Do not subtract or compare pointers that do not wrap

Use the readlink() function properly

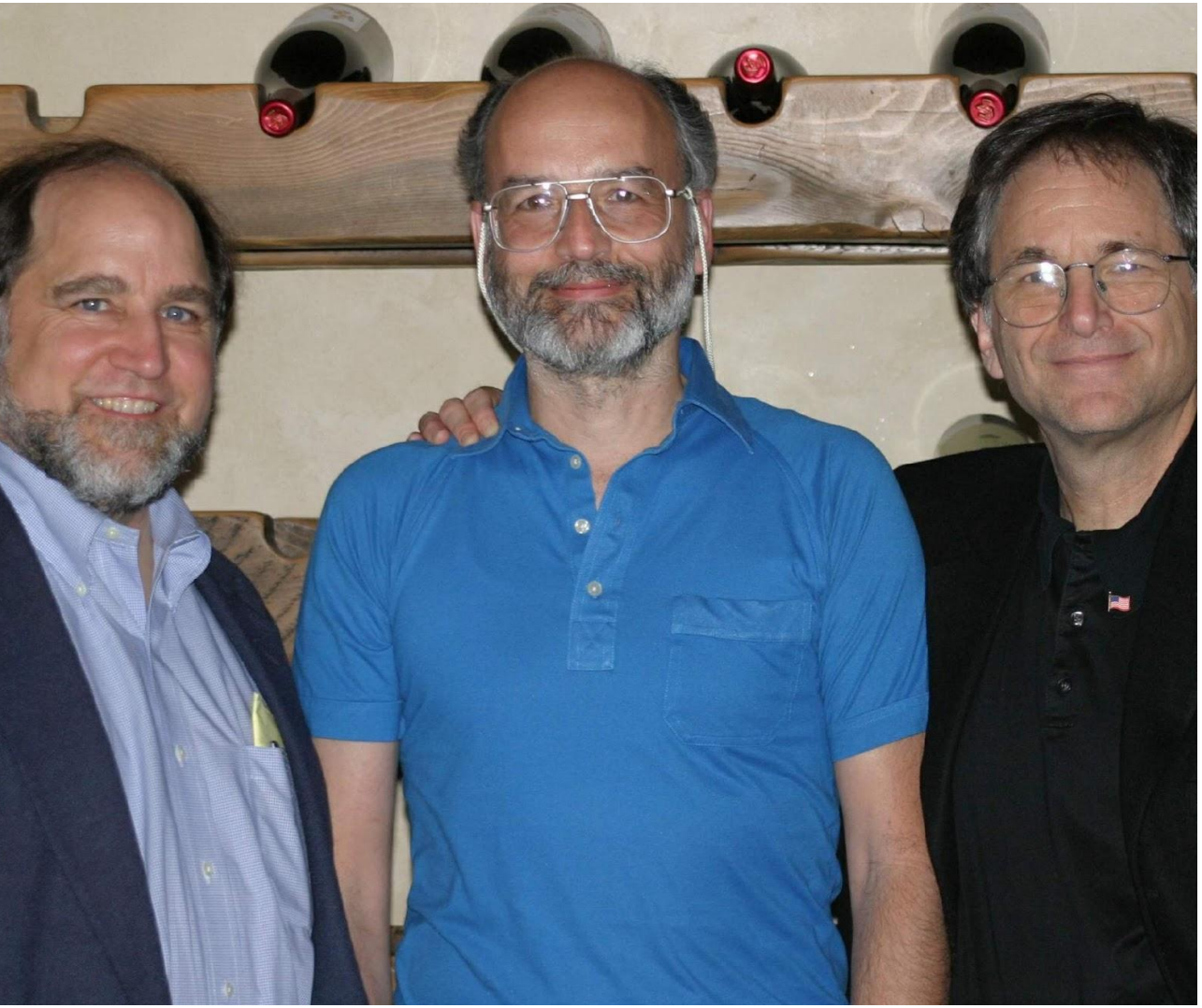
Image: https://en.wikipedia.org/wiki/Fred_Rogers

There are more developers than security experts.




Is software safe?
E.g., Chrome.

GitHub, NIST



Ron Rivest, Adi Shamir, Len Adleman

# DO YOU TRUST SECURE CRYPTO ALGORITHMS?

$$\forall m_0, m_1 \in M. \text{where} |m_0| = |m_1|$$

$$\forall c \in C.$$

$$\Pr\left[E(k, m_0) = c\right] = \Pr\left[E(k, m_1) = c\right]$$


# Implementations may still leak

```
message_t decrypt(ciphertext_t c, key_t k){

  if(k == 1)
    return decrypt(m) // Takes time 1
  if(k == 2)
    return decrypt(m) // Takes time 2
  if(k == 3)
    return decrypt(m) // Takes time 3

}
```

Image: https://imgflip.com/i/52b3g8

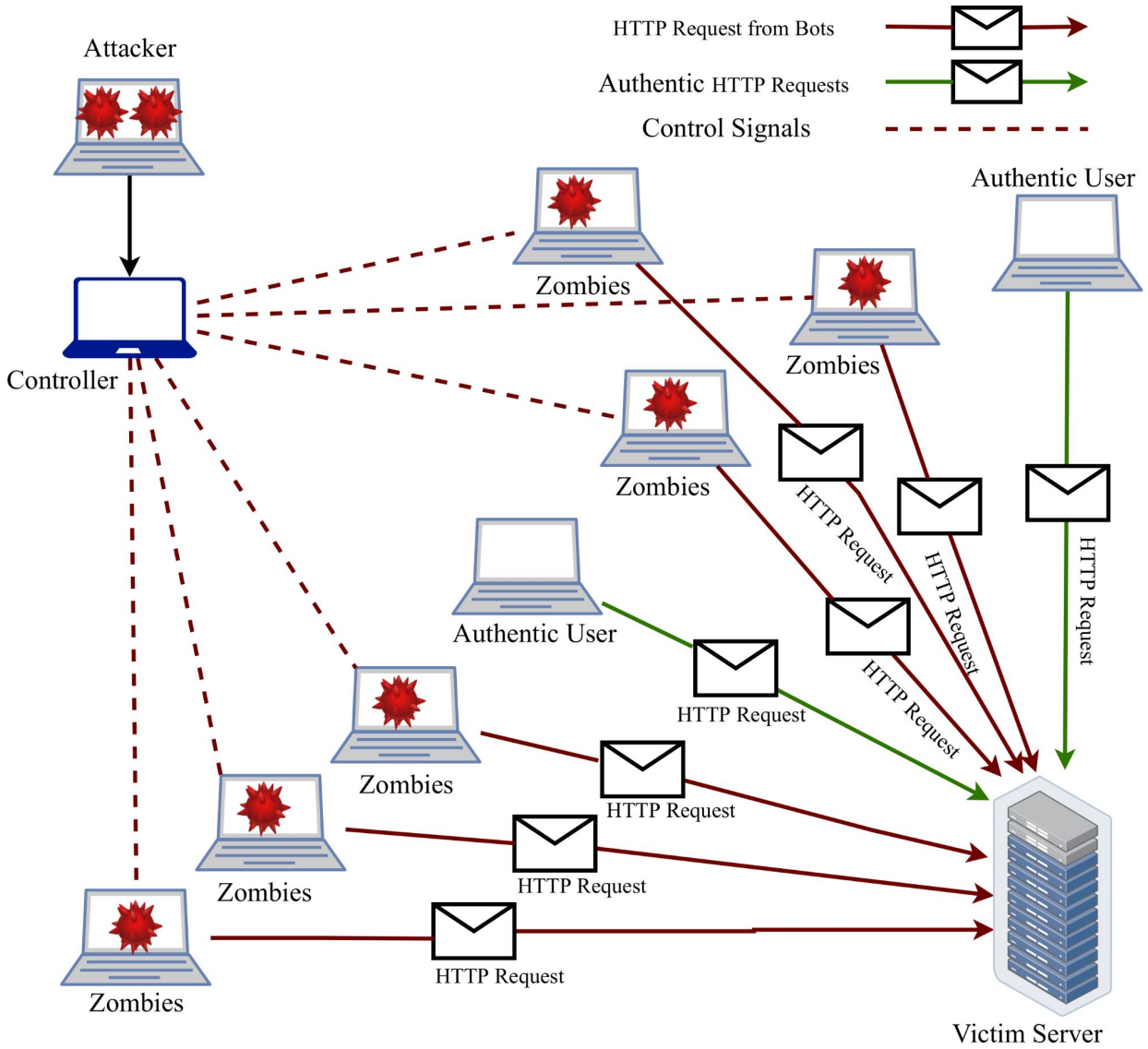

# NETWORKED SYSTEMS

- Software has same security concerns
- Protocols play a larger part
- New, interesting security properties desired


# HUMAN INTERACTION

Unusable systems become insecure

# Intermission: Denial of Service Attacks



# COURSE

# THIS COURSE

The four corners of security and their foundations

# THE ATTACKER

The defining characteristic of cybersecurity is the attacker:

- Smart
- Adaptive
- Asymmetric advantage

# Level 1

## FUNDAMENTALS

1. Security Principles
2. Threat modeling
3. Trusted computing base
4. Design principles for authorization, authentication, and audit

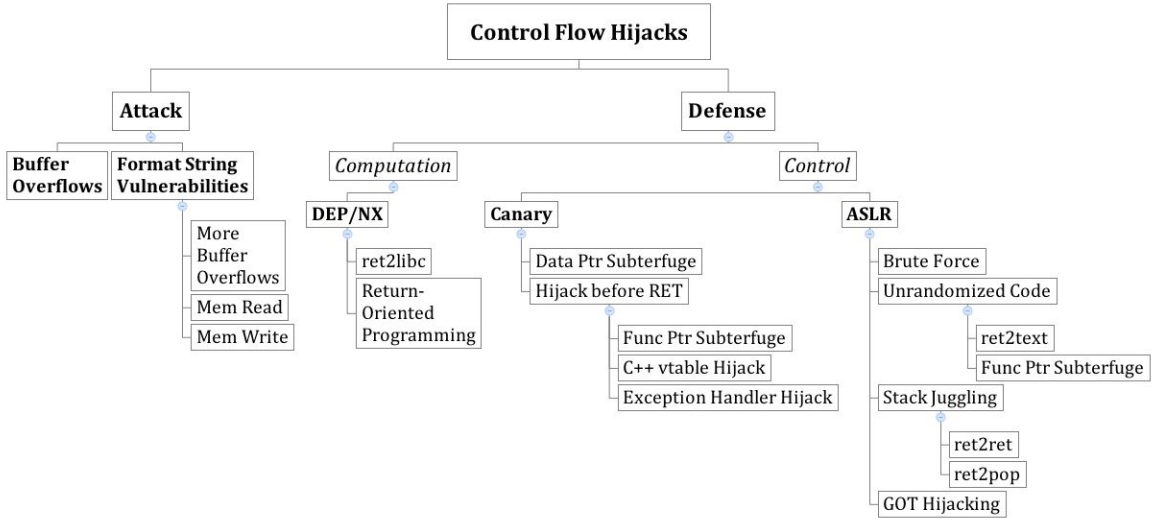
# Level 2

## SOFTWARE SECURITY

1. CVE vs CWE
2. Recognize vulnerabilities
3. Exploit vulnerabilities
4. Design mitigation
5. Circumvent mitigation
6. Understand static and dynamic analysis
7. Container security
8. API Security

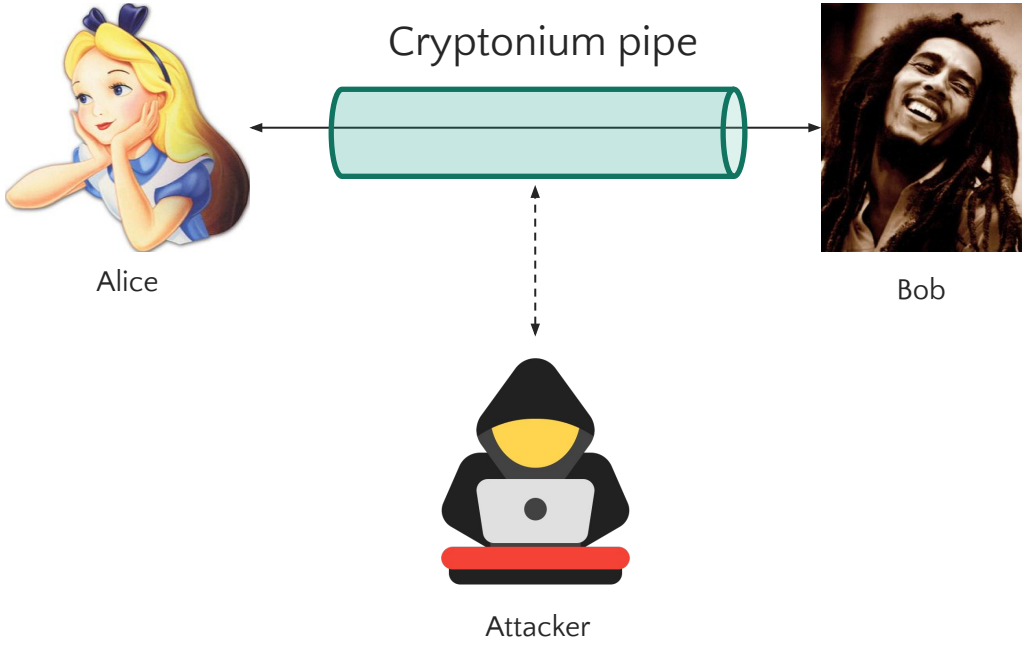
# Level 2

# BINARY EXPLOITATION



# Level 3

## CRYPTOGRAPHY



# Level 3

## CRYPTOGRAPHY

**Goals:** Privacy, authenticity, integrity

**Concepts:** [Pseudo-]Randomness, Symmetric/Asymmetric key, Encrypt, Hash, Sign, MAC, Forgery, Semantic Security

**Applications:** TLS, Private Information Retrieval, Blockchain

# Level 4

## NETWORK & WEB

1. The Gold Standard: AUthorization, AUthentication, AUdit
2. Exploit vulnerabilities
3. Design mitigation
4. Circumvent mitigation
5. Understand static and dynamic analysis

# Level 5

## HUMANS

1. Usable security
2. Privacy
3. Policy

# One perspective

Charlie Kaufman, Radia Perlman, Mike Speciner

"Humans are incapable of securely storing high-quality cryptographic keys, and they have unacceptable speed and accuracy when performing cryptographic operations.
(They are also large, expensive to maintain, difficult to manage, and they pollute the environment. It is astonishing that these devices continue to be manufactured and deployed. But they are sufficiently pervasive that we must design our protocols around their limitations.)"

# ETHICS AND YOUR RIGHTS

# Understanding Security is like a Superpower

# DON’T BREAK THE LAW

You need explicit permission to check the security of a system.

You are responsible for your actions.

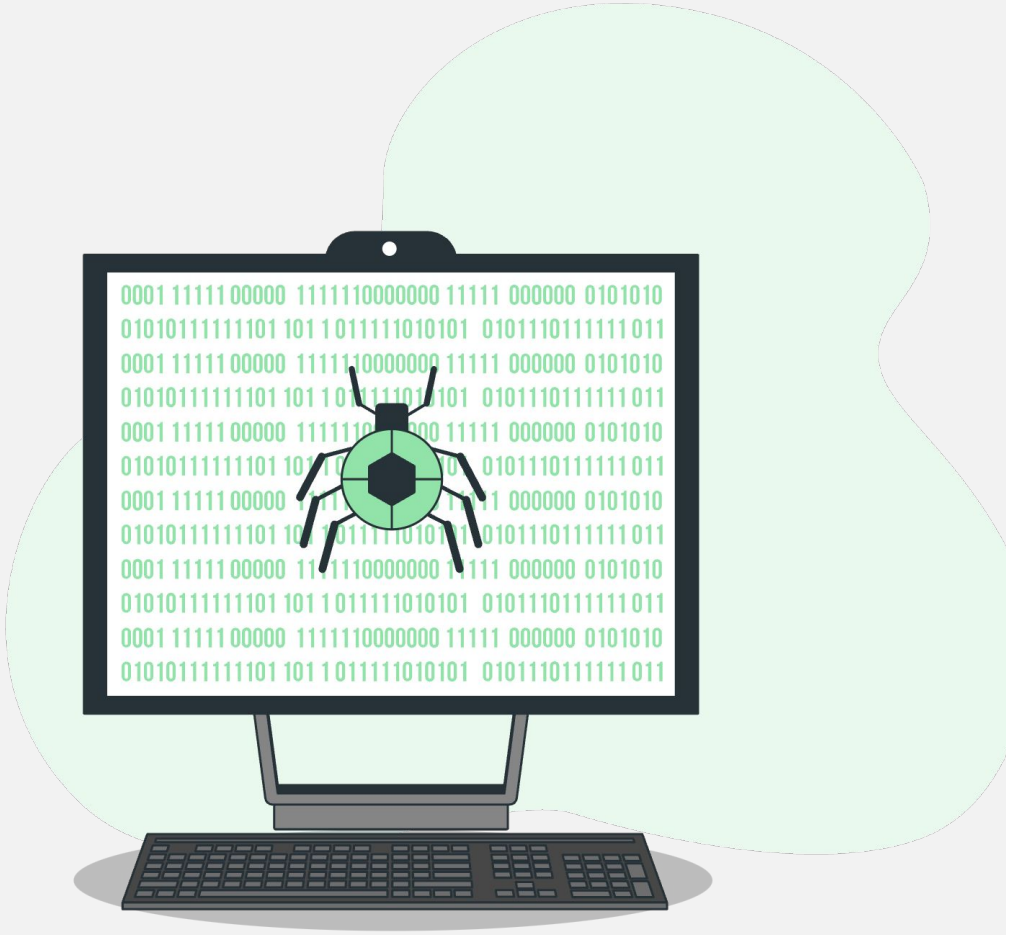


Image: storyset.com

# SAFE HARBOR & BUG BOUNTIES

**Safe harbor:** A safe harbor is a legal provision in a statute or regulation that provides protection from a legal liability or other penalty when certain conditions are met.

**Bug bounty:** Provide a legal safe harbor under the terms of the bug bounty program.

Example: hackerone.com

# What is Fair Game?

- All homeworks and exam questions will be in a controlled environment where you will be able to apply techniques you learn.

- Attacking homework infrastructure (or course staff) is NOT fair game and will result in a grade of exactly 0.
  - If you do discover a hole (by accident), practice coordinated vulnerability disclosure.

Where can I learn more about security?

Everywhere! Pick the platform & educator you want!

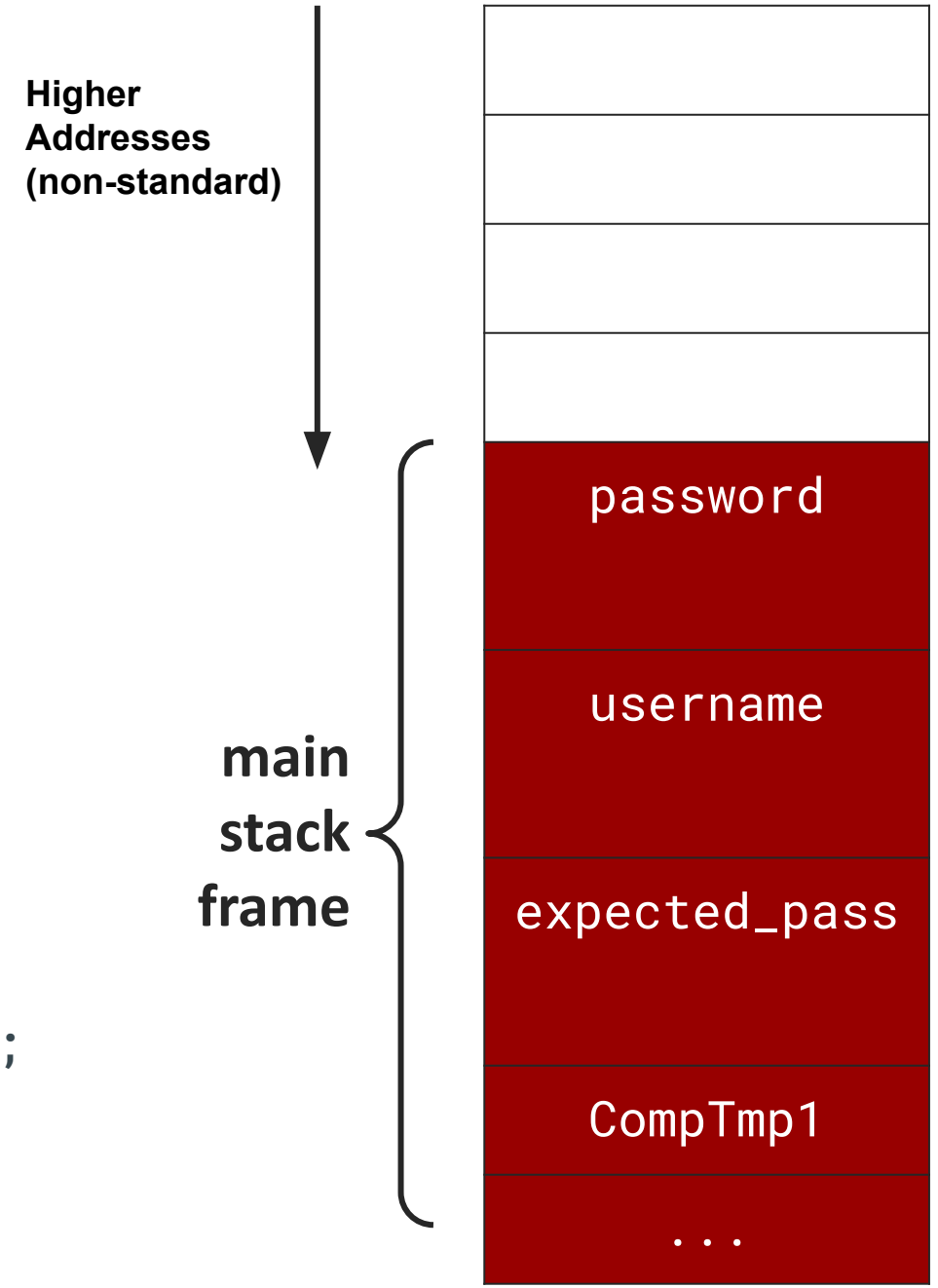
# Our First Exploit!

```c
int main() {
    char expected_password[8] = "sesame";
    char username[8];
    char password[8];
    printf("Enter username: ");
    scanf("%s", username);
    printf("Hello %s, Enter password: ", username);
    scanf("%s", password);
    if (strcmp(password, expected_password) == 0) {
        printf("Success! You are now the admin.\n");
    } else {
        printf("Failed, enter the right password.\n");
    }
    return 0;
}
```

Τι κάνει αυτό το πρόγραμμα;

**Want to try it? Compile with: gcc -fno-stack-protector -o login login.c**

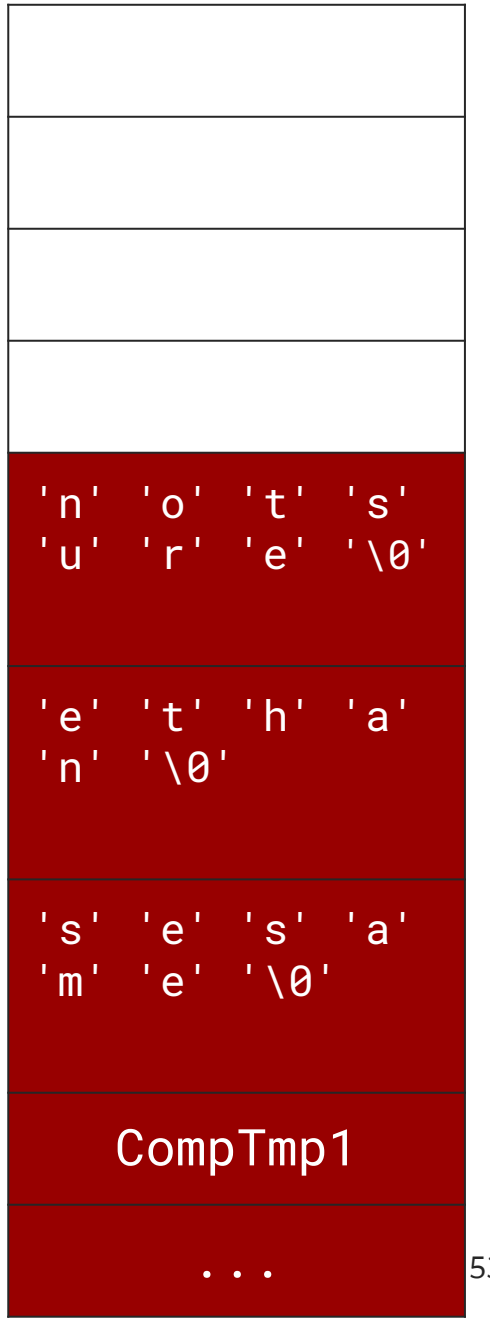
```c
int main() {
    char expected_password[8] = "sesame";
    char username[8];
    char password[8];
    printf("Enter username: ");
    scanf("%s", username);
    printf("Hello %s, Enter password: ", username);
    scanf("%s", password);
    if (strcmp(password, expected_password) == 0) {
        printf("Success! You are now the admin.\n");
    } else {
        printf("Failed, enter the right password.\n");
    }
    return 0;
}
```



```
$ ./login
Enter username: ethan
Hello ethan, Enter password: notsure
Failed, enter the right password.
```

```c
int main() {
    char expected_password[8] = "sesame";
    char username[8];
    char password[8];
    printf("Enter username: ");
    scanf("%s", username);
    printf("Hello %s, Enter password: ", username);
    scanf("%s", password);
    if (strcmp(password, expected_password) == 0) {
        printf("Success! You are now the admin.\n");
    } else {
        printf("Failed, enter the right password.\n");
    }
    return 0;
}
```

```
$ ./login
Enter username: ethan
Hello ethan, Enter password: notsure
Failed, enter the right password.
```



Επομένως, αν δεν έχουμε το expected_password δεν μπορούμε να κάνουμε login, σωστά;

# Demo

Συμπέρασμα: αποφεύγουμε την χρήση scanf("%s", …) όποτε μπορούμε.

Πως αλλιώς; Πολλές δυνατότητες:

1. getchar() + realloc
2. scanf("%ms", …) - non-standard :(
3. getline, κοκ
4. …

# Is that all? No!

# Ευχαριστώ και καλή μέρα εύχομαι!

Let's start hacking!