CSE 40567 / 60567: Computer Security



Software Security 5

Changes to the Syllabus: See Course Website

Changes in Course Logistics: See my Email and Slack We are now caught up on grading. If you have questions on where you stand in the class, contact Prof. Scheirer.

Homework #5 has been released. It is due 4/2 at 11:59PM

See **Assignments Page** on the course website for details



Film Response Summary

Format Strings Attack

```
Another common bug:
```

```
int func(char *user) {
    fprintf(stderr, user);
}
```

Problem: what if *user = "%s%s%s%s%s%s%s"?

- Most likely, program will crash: DoS.
- If not, program will print memory contents. Privacy impact?
- Full exploit using *user = "%n"

Correct use of fprintf(): fprintf(stdout, "%s", user);

Vulnerable functions

Any function using a format string:

Printing:
 printf(), fprintf(), sprintf(), ...
 vprintf(), vfprintf(), vsprintf(), ...
Logging:

Logying.

syslog(), err(), warn()

Writing a format strings exploit

- Dumping arbitrary locations in memory:
 - Walk up the stack until desired pointer is found.
 - -printf("%08x.%08x.%08x.%08x|%s|")
- Write to arbitrary locations in memory:
 - -printf("hello %n", &temp) // writes '6' into memory
 - -printf("%08x.%08x.%08x.%08x.%n")

Race Conditions

- General category of bug (not always a security problem)
- Affects systems where the output is dependent on the sequence or timing of other uncontrollable events
- Becomes a bug when events do not happen in the order the programmer intended
- Trouble for authentication:
 - (1) Check for authentication, (2) State changes, (3) Act on authentication

Example: race condition exploit in Starbucks gift cards

- Hole in online gift card purchases surfaced in 2015
- Initiate two identical web store transfers, trick the store into recording both
 - Normal use: Move money from one card with \$5 onto another, for a total of \$10 on one card
 - With duplicate transfer: end up with \$15 on one card!



Helpful Tools for Writing Exploits

Metasploit (www.metasploit.com)

Tool for developing and executing exploit code against a target machine

ervi	ew 📮 Ana	Ivsis 🗔 S	essions 2	б с	ampaigns	🔊 We	b Apps	😵 Modul	es 📎 Tage	s 🗐 Reports	Tasks	
				v -	anpagno		a repo	· mouu	00 V rugi			
	Test Hosts											
Go	to Host 🗂 De	elete 🚿 Sca	n 🖅 Import	🛞 Ne	expose (Modules	Brutefo	rce 👩 E	xploit 💿 New	Host		Q
				_								
lost	s 🛃 Notes	📡 Services 🛛 🔇	Vulnerabilities	80	Captured Evi	dence						
ow	10 ▼ entries											
	IP Address	Name 🌢	OS Name	4	Version 🌢	Purpose	Services	Vulns 🍦	Notes	Updated	Status	
	10.1.95.80		DE Unknown			device		1		2 minutes ago	Loote	d
			👌 Linux									
	10.1.05.110		vmware-bavm			desident				0	01II	_
	10.1.95.113	vmware-bavm	2.6.12-9-686 #1 M Oct 10 13:25:32 B	on ST		device		1	1	3 minutes ago	Shelle	d
			2005 i686									
	10.1.95.253		😹 Konica Printer			printer	1			5 minutes ago	Scann	ed
owi	ng 1 to 3 of 3 entr	ies								First Previous	1 Next	Last

What does metasploit do?

- Provides tools that make debugging, offset hunting and payload crafting easier
- Basic steps for exploiting a system via the framework:
 - 1. Choose and configure an exploit (over 900 available in the framework)
 - 2. Check whether a target system is vulnerable
 - 3. Choose and configure a payload
 - 4. Encode payload to evade IDS
 - 5. Execute the exploit

Disassembling code

- Broad knowledge of assembly language is essential for writing exploits
 - Need to know where function calls exist in memory, the internal execution flow of the program, and the state of the heap / stack (eip / rip manipulation)
 - Sometimes source code isn't available; the binary must be examined in these cases

gdb

(gdb) disass main		
Dump of assembler code for	or function :	main:
0x000000000400624 <+0)>: push	%rbp
0x000000000400625 <+3	1>: mov	%rsp,%rbp
0x000000000400628 <+-	4>: sub	\$0x30,%rsp
0x00000000040062c <+8	8>: mov	%edi,-0x24(%rbp)
0x00000000040062f <+:	11>: mov	%rsi,-0x30(%rbp)
0x000000000400633 <+3	15>: cmpl	\$0x4,-0x24(%rbp)
0x000000000400637 <+3	19>: je	0x40063e <main+26></main+26>
0x000000000400639 <+2	21>: jmpq	0x400731 <main+269></main+269>
0x00000000040063e <+2	26>: mov	\$0x100,%edi
0x000000000400643 <+3	31>: callq	0x400520 <malloc@plt></malloc@plt>
0x000000000400648 <+3	36>: mov	%rax,-0x18(%rbp)
0x00000000040064c <+4	40>: mov	\$0x40082c,%eax
0x000000000400651 <+-	45>: mov	-0x18(%rbp),%rdx

IDA PRO

https://www.hex-rays.com/index.shtml

🚹 IDA - D:\AdamM	lil\games\cap	pitalism2\cap2.e					
File Edit Jump Se	earch View D	ebugger Options	Windows Help				
	* # #	1 Text	🔽 🏑 🖌 👷 🖌 🗱 🗸 🗰 🐨 🕺 🕺 🕺 🖌 🛏 🛩 🥒 👘 🎬				
X 🛃 Functions wind	dow 🗙 N N	ames window 🔽	🗙 🗐 IDA View-A 🗙 🐧 Structures 🗙 🗈 En ums 🗙 📆 Local Types 🗙 🎘 Exports 🗶 "" Strings window	•			
Function name	Segmen	it Start 🔺					
🗿 sub_634170	.text	00634170	loc_6089DF:				
🗿 sub_634180	.text	00634180	callwincmdln				
🗿 sub_634190	.text	00634190	test byte ptr [ebp+StartupInfo.dwFlags], bl				
👔 sub_6341A0	.text	006341A0	jz short loc_6089EF				
🗿 sub_6341B0	.text	006341B0					
🗿 sub_6341C0	.text	006341C0					
🗿 sub_6341D0	.text	006341D0					
🗿 sub_6341E0	.text	006341E0	movzx ecx, [epp+startupin+o.wsnowwindow]				
🗿 sub_6341F0	.text	006341F0					
🗿 sub_634200	.text	00634200	DOD CCX				
🗿 sub_634210	.text	00634210					
r sub_63421F	.text	0063421F					
41	1						
		<u> </u>					
Line 8435 of 8435			loc_6089F2: ; nShowCmd				
🧤 Function calls:	_tmainCRTStart	up 🗆 🗙	push ecx				
Address	Caller	Instruction	push eax ; ipundiane				
.text:00608A6E	\$LN39	jmp tr	push b see the second s				
			call WinMain@16 : WinMain(x,x,x,x)				
			mov [ebp+var 10], eax				
91		1 1	cmp [ebp+var_20], 0				
			jnz short \$LN44				
Address (Called function	*					
.text:00608890 (callSEH_pr	olog4 —					
text:0060889D (call ds:GetStal call ebv:GetE	rtupinroA ProcessHeap					
.text:006088888	call ds:HeapAl			Þ			
•	10 - C		100.00% (896,2602) (-17,33) 002089FB 006089FB:tmainCRTStartup+172				
🗒 Output window							
68D6B0: using gu	uessed type	char byte_68	9680;				
Sorting 'String	uessed type s window'	ok	J978;	•			
IDC							
AU; idle Up Disk: 1GB							

Fuzzing

- Black-box testing methodology
- Checks code modules for vulnerability to overflows
 - Many are not obvious to visual inspection
- Two fuzzing strategies are typically deployed
 - 1. Mutation-based fuzzers
 - 2. Generation-based fuzzers

CERT Basic Fuzzing Framework (BFF)

Mutational fuzzer

https://github.com/CERTCC-Vulnerability-Analysis/certfuzz

DebianFuzz-2.0 - VMware Workstation			1			
File Edit View VM Team Windows Help 🎦 🖾 🗔 🔛 💽 💽 🔯	a 🗊 🔎 💷 🕞 🚳 🗿	\$				
🔒 Home 🗙 🚯 DebianFuzz-2.0 🗙						
batch.sh	fuzz@debian: /mnt/hgfs/fuzz					
eb8c692179c81032b9ce4ecfa3812a0b Command: cat /home/fuzz/fuzzipo/convert/ie.ppm/ie.ppm zzu)103cb6a00534d9c29b66b5336621cd6	29	38093	583588	10	10
home/fuzz/fuzzing/convert/ie.ppm/ie.ppm.592639	312ea1d07bf2724b467188a27daa2125	25	76355	564111	24	24
eb8c682179c81032b9ce4ecfa3812a0b EIP=0x80880b2	370cb3179edb277171e87e35fa630e94	24	18927	580014	27	26
bff Fuzzing ie.ppm seeds=592640-592791 Range=0.088308-0.142886 Rate=(66.73/s 4004.0/m 240242	3a752911edc77222a809a7c4a3660ccb	20	126306	534967	9	9
fuzztools.zzuflog zzuf[s=592687,r=0.0883082:0.142886]: signal 8 (SIGFPE)	:3f647ceb059ed811f602ab82fcda666	10	37208	576064	5	5
bff Generating testcase for zzuf[s=592687,r=0.0883082:0.142886]: signal 8 (SIGFPE)	382a0a96a566bf14c417fd1c2d42605a	3	8324	10668	16	16
bff Output file is /home/fuzz/fuzzing/convert/ie.ppm/ie.ppm.592687	d44ac59e913e09cc36b0f803f35751c	2	227158	233443	.7	6
d2490f048dc3b77a457e3e450ab4eb38_seen at seed=592687 range=0,0883082:0.142886 file=/home/fua	Sa5f9e7cff4b1aa5ae4b59a27288e483	2	108764	279716	14	14
687 file_md5=3defbaa3e9d59eb36de71c764f21d18e	2c286a956886e4a0018afdc7b08e706b	2	367870	526643	26	25
d2490f048dc3b/7a45/e3e450ab4eb38 Command: cat /home/fuzz/fuzzing/convert/ie.ppm/ie.ppm I zzu	014015680355854600150D3805550036	1	10815	10815	4	
home/fuzz/fuzzing/convert/ie.ppm/s9268/	02745abf2470aa7957479554aaaa4444	1	221881	221881	20	70
Drf Fuzzing 16.ppm seeds=5522666-552/31 Kange=0.086308-0.142866 Kate=166.74/S 4004.27M 24023	0254TCDT245Vae5557075TT1Caae4441	L Alext airs pu	430730	430730	02	,/°
Tuzztools,zzurlog zzur(s=03260,r=0,00002(0,142606); signal 6 (SIGFE)	crash id	s/gec_eips.pg				
bit denerating testase for 22015-32000, -0.000002,0,1420001, signal o (Storre)	103cb6a00534d9c29b66b5336621cd6	0vb7fe2424				
affeed954f997/hce994fc68e07edd seed = 592590 range=0.0883082+0.142886 file=/home/fur	c286a956886e4a0018afdc7b08e706b	0xb7ebbfab				
690 file md5=ch0h444abb2871fe9eda0594e5709be9	312ea1d07bf2724b467188a27daa2125	0xb7fe2424				
af6ee0954f997d7bce9940fc68e07edd Command: cat /home/fuzz/fuzzing/convert/ie.ppm/ie.ppm zzu	370cb3179edb277171e87e35fa630e94	0xb7ec2c4d				
home/fuzz/fuzzing/convert/ie.ppm/ie.ppm.592690	3764117cfadc00f42a21ff85bf480217	0xb7fe2424				
af6ee0954f997d7bce9940fc68e07edd EIP=0x80882e8	3a752911edc77222a809a7c4a3660ccb	0xb7ebf9ac				
bff Fuzzing ie.ppm seeds=592691-592791 Range=0.088308-0.142886 Rate=(66.74/s 4004.1/m 240247	6a5f9e7cff4b1aa5ae4b59a27288e483	0xb7ec2c4d				
fuzztools.zzuflog zzuf[s=592699,r=0.0883082;0.142886]; signal 8 (SIGFPE)	70805d15557de8bc565881e82aa0d138	0xb7fe2424				
bff Generating testcase for zzuf[s=592699,r=0.0883082:0.142886]: signal 8 (SIGFPE)	42c9da28fcd716faf99fca02530ffff	0xb7ebf9ac				
bff Output file is /home/fuzz/fuzzing/convert/ie.ppm/ie.ppm.592699	/5893a5e462ec8d269bbc6d7a12cdd02	0xb7fe2424				
d2490f048dc3b/7a45/e3e450ab4eb38 seen at seed=592599 range=0.0883082:0.142886 file=/home/fug	3bd419260261eeb1f4/d61ac80bc16d5	0xb/fe2424				
699 file_md5=698b44b443263f8fcb6fc48b8f47cde3	2254fcDf2430ae535/d/3ff1caae4441	0xb/ec2c4d				
d2490f048ddsb/7a45/ese450ab4eb38 Command cat /home/fuzz/fuzzing/convert/ie.ppm/ie.ppm zzu	382aVa3ba3bbbbt14c41/tdlc2d42bV3a	0xD/fe2424				
nome/fuzz/fuzz/ng/convert/ie.ppm/ie.ppm/s2/292/03	2C09E4299747b99402-20-07-44	0x07ebesce				
DTT TUZZING 10.ppm Seeds-352700-552731 Kange-0.000300-0.142000 Kate-(00.75/S 4004.17M 240240	r0eev3341337070ce3340108eev7ed0	0xb7aba5ca				
$\tau_{12220018,2207109,2207[S=032/16,^{-0},0003002;0,142000]; Signal 0 (SiGFE) 10bff Camenating testoss for \tau_{21} (SIGFE) (SiGFE) (Signal 2 (SIGFE))$	3f647ceb059ed811f602ab82fcda666	0xb7ebe5ce				
bif futurut file is /home/fuzz/fuzzin/convert/ie.pom/592716	12490f048dc3b77a457e3e450ab4eb38	0X0100000				
af6eg954f997/d7bce9940fc68e07edd seen at seed=592716 range=0.0883082+0.142886 file=/home/fuz	d44ac59e913e09cc36b0f803f35751c	0xb7ebe5ce				
716 file md5=a80a04661df2b042b38e40f13618bbd4	285ced682969caf8ec2adf5945b790c	0xb7fe2424				
af6ee0954f997d7bce9940fc68e07edd Command: cat /home/fuzz/fuzzing/convert/ie.ppm/ie.ppm zzu	b8c682179c81032b9ce4ecfa3812a0b	0x80880b2				
home/fuzz/fuzzing/convert/ie.ppm/ie.ppm.592716	f79e46e1de3195c9d6318331d9d5e3d	0xb7fe2424				
af6ee0954f997d7bce9940fc68e07edd EIP=0x80882e8	uzz@debian:/mnt/hgfs/fuzz\$ analyzers	s/create_crasher_script	.py dd44ac59e913e	09cc36b0f803f35	751c	
bff Fuzzing ie.ppm seeds=592717-592791 Range=0.088308-0.142886 Rate=(66.74/s 4004.1/m 240248	at /home/fuzz/fuzzing/convert/ie.pp	w/ie.ppm zzuf -s22719	58 -r0.0883082:0.1	.42886 > /home/f	uzz/fuzzing/conve	rt/ie.ppm
fuzztools.zzuflog zzuf[s=592724,r=0.0883082:0.142886]: signal 8 (SIGFPE)	'1e.ppm.22/158			10000 5 8 10	a da ang ang ang ang ang ang ang ang ang an	-
bff Generating testcase for zzuf[s=592/24,r=0.0883082:0.142886]; signal 8 (SIGFPE)	cat /home/fuzz/fuzzing/convert/ie.pp	n/ie.ppm zzuf -s25544	45 -r0.0883082:0.1	.42886 > /home/t	uzz/fuzz1ng/conve	rt/ie.ppm
D++ Output +11e 1s /home/+uzz/+uzz1ng/convert/1e.ppm/1e.ppm/392724	le.ppm.233443					
	uzzedebien,/mno/ngrs/ruzze					
one 🛛 💽 fuzz@debian: /mnt/hg : batch.sh	top	bash	▲ ► 13::	21		
To return to your computer, move the mouse pointer outside or press Ctrl+Alt.						

w3af

http://w3af.org/

w3af - Web Application Attack and Audit Framework								
Scan config Log Results Exploit								
Profiles	Target: Insert the target URL he	re	Start 🔽					
empty_profile	Plugin	Active						
OWASP_TOP10	▶ audit							
audit_high_risk			=					
bruteforce	📝 detailed							
fast_scan	🌌 generic		This authentication plugin can login to web application with more					
full_audit	→ bruteforce		and complex authentication schemas where the generic plugin do					
full_audit_manual_disc	📝 basicAuthBrute		Three configurable parameters exist:					
sitemap	📝 formAuthBrute		- username					
web_infrastructure			- password					
	afd		- username_field					
	📝 allowedMethods		- data_format					
	🌌 archiveDotOrg		- auth_url					
	📝 bing_spider		- method					
	📝 content_negotiation		- check_string					
	detectReverseProxy							
	detectTransparentProxy							
	🃝 digitSum		username					
	🕅 dir bruter		password					
	Plugin Active		username_field					
	マ output 🖃		password field					
	📝 console 🗹							
	📝 csv_file 🗌		data_format					
	🃝 emailReport 🛛		auth_url					
	🃝 export_requests 🗌		method POST					
	gtkOutput 🗹		check_url					
	📝 htmlFile		check_string					
	📝 textFile							
	📝 xmlFile 🗌		Save configuration Revert to previous values					
			0 🛝 0 🖾 0					

Features:

- Daemons
- Fast HTTP Client
- Output Manager
- Fuzzing Engine
- Knowledge base

Image Credit: https://binarymist.files.wordpress.com/2014/01/w3af.png

skipfish

https://code.google.com/archive/p/skipfish/



 Scanner version:
 1.78b
 Scan date:
 Sun Nov 21 23:40:36 2010

 Random seed:
 0x1c41920a
 Total time:
 0 hr 9 min 8 sec 467 ms

 Problems with this scan? Click here for advice.

Froblens with this scale click in

Crawl results - click to expand:



Document type overview - click to expand:

application/xhtml+xml (5)