Lessons Learned from Five Years of Building Capture the Flag

Vito Genovese
DEF CON Beijing 2018

HELLO!

CAPTURE THE FLAG

"CTF"

DEF CON



1996

first game

2000

formalized

how it was run

2002-2004

ghettohackers

2005-2008

Kenshoto

2009-2012

ddtek

2013-2017 Legitimate Business Syndicate

2018-?

Order of the Overflow

Two Distinct Formats

6/16/2013 5:54:05 PM <global> PPP solved everything with two hours to spare! Call Heinz 'cause y'all need to ketchup!
6/16/2013 5:51:03 PM >private< Your teammate PPP solved bob [OMGACM] for 4 points.
6/16/2013 5:50:58 PM >private< Your teammate PPP solved bob [OMGACM] for 4 points.
6/16/2013 2:11:12 PM <global> You should totally check out OMGACM5.
6/16/2013 3:28:18 PM >private< Your teammate PPP solved tastycloud [gnireenigne] for 5

3dub	0x41414141	\xff\xe4\xcc	OMGACM	gnireenigne
1	1	1	1	1
2	2	2	2	2
3	3	3	3	3
4	4	4	4	4
5	5	5	5	5

1:49:00 left

PPP	75
European NOPSled Team	70
More Smoked Leet Chicken	68
blue-lotus	64
Shellphish	60
WHAT_Mafia	59
Men in Black Hats	58
clgt	58
9447	58
Routards	56
WOWHACKER-B10S	56
Shell Corp	56
Samurai	54
sutegoma2	54
APT8	54
Alternatives	53
pwningyeti	53
[TechnoPandas]	51
RDot.Org	51
United States of Russia	51
Marauders	50
CodeRed	50
KAIST GoN	50
int3pids	47
Robot Mafia	47

Jeopardy Style

Scoreboard

Jeopardy Style

Prompt

crackme1

Baby's First, 15 points

crackme1_f92e0ab22352440383d58be8f046bebe.quals.shallweplayaga.me:10001

Files

https://2017.notmalware.ru/add6a6232ea1e9d42b925a7d2a2c5781bfccd6fd/8a97fb8c264a3b34dad0a707dbfc92832067a0fa0f2b5a576c73557960b11506.tar.bz2

answer

Solve it

Back to Scoreboard

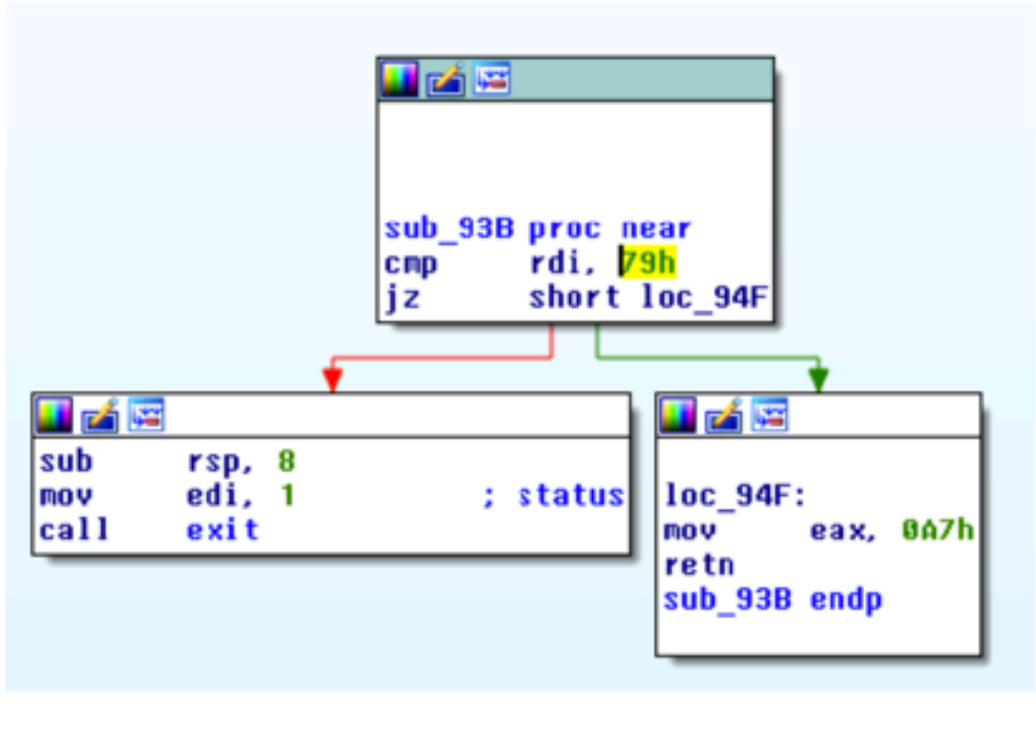
Jeopardy Style

Solving

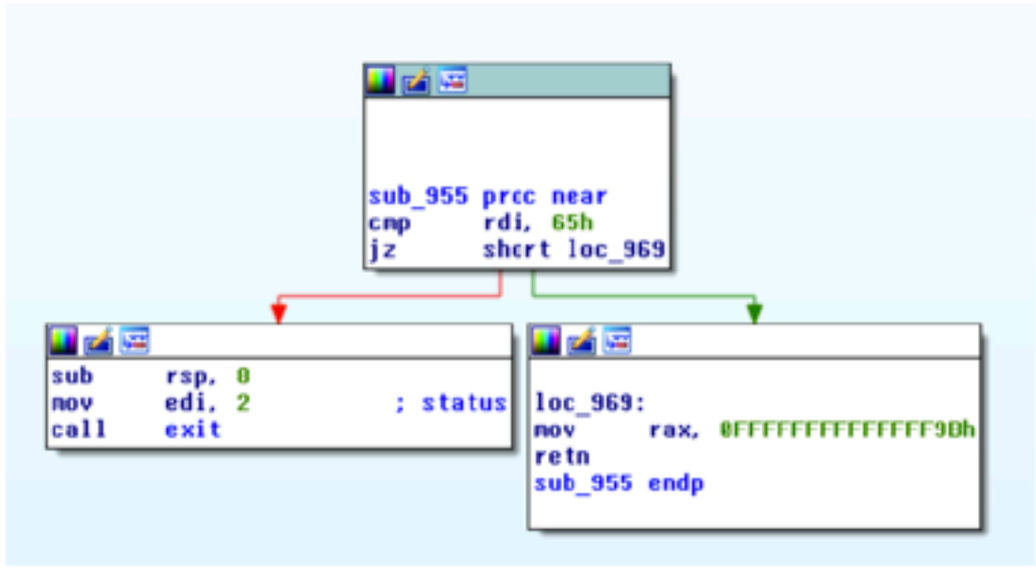
0xC6C is a long subroutine, with many calls other subroutines; a closer look reveals that each of them gets as input a character of s, in order, and checks it is the "correct" one.

Something helps even more: the exit code when no input byte is correct is 1, when the first one is correct is 2, and so on:

0x93B:



0x955:



I did not investigate this any further, and wrote a python script to bruteforce the solution byte by byte.

```
1  import string
2  import subprocess
3
```

Jeopardy Style

Points

Potent Pwnables

- 70 Leo es Pequeno
- 94 badint
- 69 peROPdo
- 62 mute
- 142 insanity
- 234 reeses revenge
- 436 **nekisahloth** *Hot stuff! Solve this for scoreboard control.*

Jeopardy Style

DEF CON CTF Quals



Photo: robbje @ Eat Speep Pwn Repeat

ASBY

From SHA2017 CTF

ASBY

Get file

asby (100) - 331 solves

Eindbazen team member asby has by far been putting the most energy and time in creating the SHA2017 CTF. To honor his dedication and all his effort we created this challenge as an ode to him.

You can choose to reverse engineer this challenge or you can "asby" it. Good luck with the option you choose.

asby.tgz `7422948a4034252d45cee02753b3d13b`

Please enter flag for challenge: asby

Submit flag

ASBY

Identify Windows STDIO .exe

ASBY

Determine goal

2. xclient@5196d80f9b61: /asby (docker)

xclient@5196d80f9b... ⌘1

```
xclient@5196d80f9b61:/asby$ wine asby.exe
What is the flag? hack the planet
Checking char 1:WRONG!
What is the flag? Checking char 1:WRONG!
What is the flag? Checking char 1:WRONG!
What is the flag? ctf{
Checking char 1:WRONG!
What is the flag? flag{
Checking char 1:CORRECT!
Checking char 2:CORRECT!
Checking char 3:CORRECT!
Checking char 4:CORRECT!
Checking char 5:CORRECT!
What is the flag? flag{x
Checking char 1:CORRECT!
Checking char 2:CORRECT!
Checking char 3:CORRECT!
Checking char 4:CORRECT!
Checking char 5:CORRECT!
Checking char 6:WRONG!
What is the flag?
```

ASBY

Guess the correct input

ASBY

- Reverse engineer a Windows binary
- Guess each character by hand
- Write a program

ASBY

Write program

asby.rb

```ruby
require 'expect'

IO.popen("wine asby.exe",
         "r+") do |asby|

  start = "flag{024baa8ac03ef"

  asby.expect "What is the flag? "
  asby.puts start + "\r"
  start.length.times do |n|
    got = asby.expect(/Checking char \d+:CORRECT!\r\n/, 1)
    p got
  end

  actual_flag = ''

  flag_chars = %w{0 1 2 3 4 5 6 7 8 9 a b c d e f \} }

  loop do
    asby.flush
    flag_chars.each do |candidate|
      print "\r#{candidate}"
      asby.expect "What is the flag? "
      asby.puts(start + actual_flag + candidate + "\r")
      print '.'
      (start.length + actual_flag.length).times do |n|
        asby.expect "Checking char #{n+1}:CORRECT!\r\n"
      end

      if asby.expect(/Checking char \d+:\w+\!\r\n/, 1).join =~ /CORRECT/
        puts "\r"
        actual_flag += candidate
        puts start + actual_flag
        if '{' == candidate
          puts start + actual_flag
          exit
        end
        break # out of the candidate loop
      end
    end
  end

end
```

```
-:---  asby.rb        All (44,0)    Git:master  (Ruby RuboCop YARD company WS RE
```

```ruby
require 'expect'

IO.popen("wine asby.exe",
         "r+") do |asby|

  start = "flag{024baa8ac03ef"

  asby.expect "What is the flag? "
  asby.puts start + "\r"
  start.length.times do |n|
    got = asby.expect(/Checking char \d+:CORRECT!\r\n/, 1)
    p got
  end

  actual_flag = ''

  flag_chars = %w{0 1 2 3 4 5 6 7 8 9 a b c d e f \} }

  loop do
    asby.flush
    flag_chars.each do |candidate|
      print "\r#{candidate}"
      asby.expect "What is the flag? "
      asby.puts(start + actual_flag + candidate + "\r")
      print '.'
      (start.length + actual_flag.length).times do |n|
        asby.expect "Checking char #{n+1}:CORRECT!\r\n"
      end

      if asby.expect(/Checking char \d+:\w+\!\r\n/, 1).join =~ /CORRECT/
        puts "\r"
        actual_flag += candidate
        puts start + actual_flag
```

```ruby
end

actual_flag = ''

flag_chars = %w{0 1 2 3 4 5 6 7 8 9 a b c d e f \} }

loop do
  asby.flush
  flag_chars.each do |candidate|
    print "\r#{candidate}"
    asby.expect "What is the flag? "
    asby.puts(start + actual_flag + candidate + "\r")
    print '.'
    (start.length + actual_flag.length).times do |n|
      asby.expect "Checking char #{n+1}:CORRECT!\r\n"
    end

    if asby.expect(/Checking char \d+:\w+\!\r\n/, 1).join =~ /CORRECT/
      puts "\r"
      actual_flag += candidate
      puts start + actual_flag
      if '{' == candidate
        puts start + actual_flag
        exit
      end
      break # out of the candidate loop
    end
  end
end

end
```

ASBY

Get solution

3. xclient@a84249bfa09c: /asby (docker)

xclient@a84249bfa0... ⌘1

```
flag{024baa8ac03ef22fdde61c0f11
0.
flag{024baa8ac03ef22fdde61c0f110
6.
flag{024baa8ac03ef22fdde61c0f1106
9.
flag{024baa8ac03ef22fdde61c0f11069
f.
flag{024baa8ac03ef22fdde61c0f11069f
2.
flag{024baa8ac03ef22fdde61c0f11069f2
f.
flag{024baa8ac03ef22fdde61c0f11069f2f
}.
flag{024baa8ac03ef22fdde61c0f11069f2f}
0asby.rb:24:in `write': Broken pipe (Errno::EPIPE)
        from asby.rb:24:in `puts'
        from asby.rb:24:in `block (3 levels) in <main>'
        from asby.rb:21:in `each'
        from asby.rb:21:in `block (2 levels) in <main>'
        from asby.rb:19:in `loop'
        from asby.rb:19:in `block in <main>'
        from asby.rb:3:in `popen'
        from asby.rb:3:in `<main>'
xclient@a84249bfa09c:/asby$
```

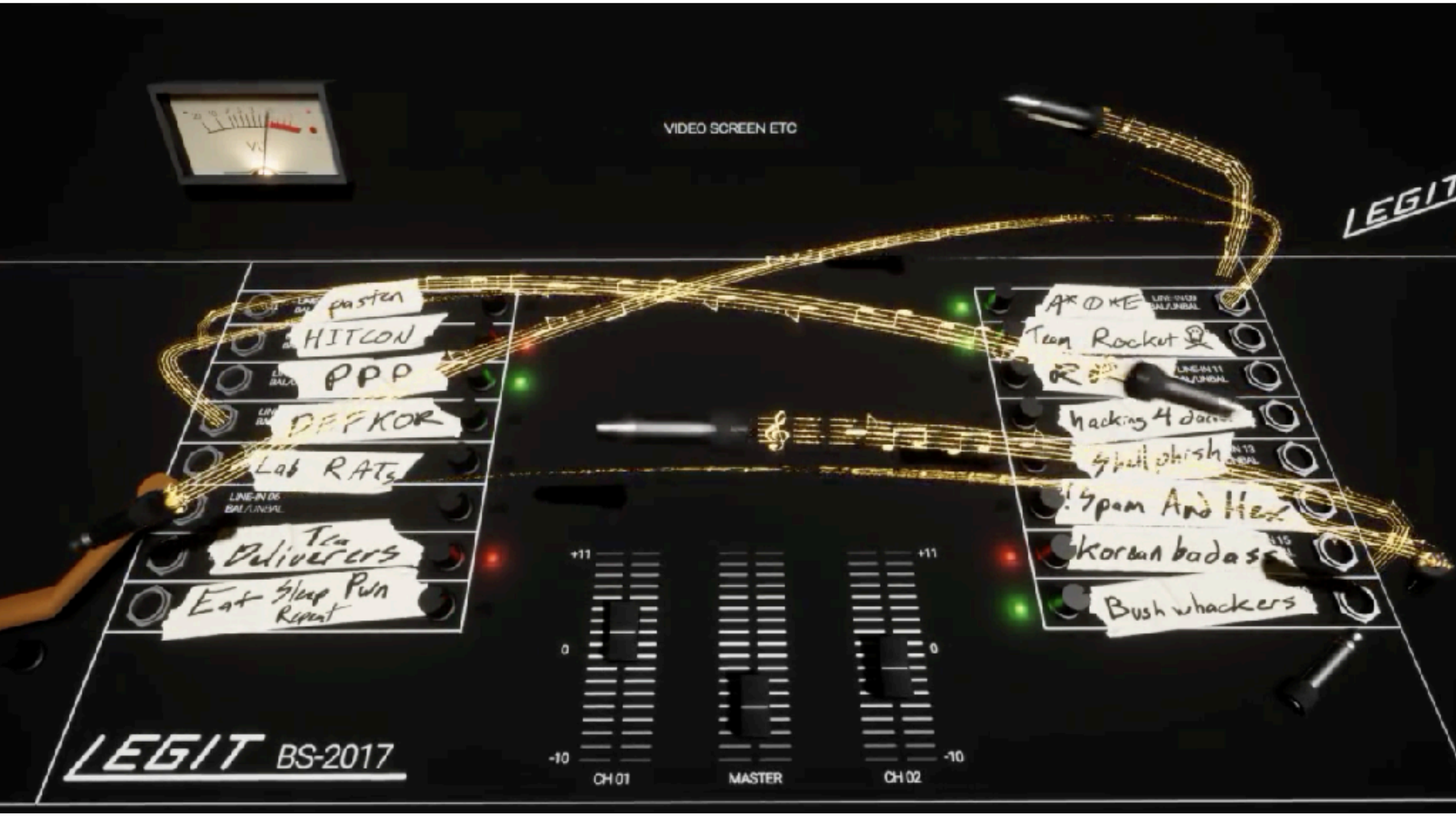
ASBY

Get points


UB-
ATE
SINDS 1870
UIT Vriendschap GEBROUWEN
AMSTEL
In 1870 richtten twee vrienden
de Amstel Brouwerij op. Zoals we toen
Pilsener brouwden, zo doen we dat nog steeds:
volmout en met extra geurige hop
voor een volle smaak en stevige schuimkraag.

Jeopardy Style

1. Get challenge
2. Solve it
3. Get points


VIDEO SCREEN ETC
LEGIT
HITCON
DEFKOR
Lab RATs
Eat Sleep Pwn Repeat
Hacking 4 dates
Spam And Hex
Bush whackers
+11
0
-10
CH 01
MASTER
CH 02
LEGIT BS-2017

Attack-Defense

- Reverse engineer
- Patch flaws
- Exploit others
- Don't break it

Attack-Defense

scorebot

PPP
atmail

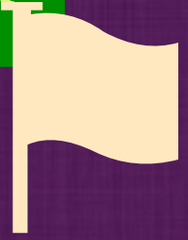
Shellphish

Attack-Defense

scorebot

deposit

PPP
atmail



Shellphish

Attack-Defense

scorebot

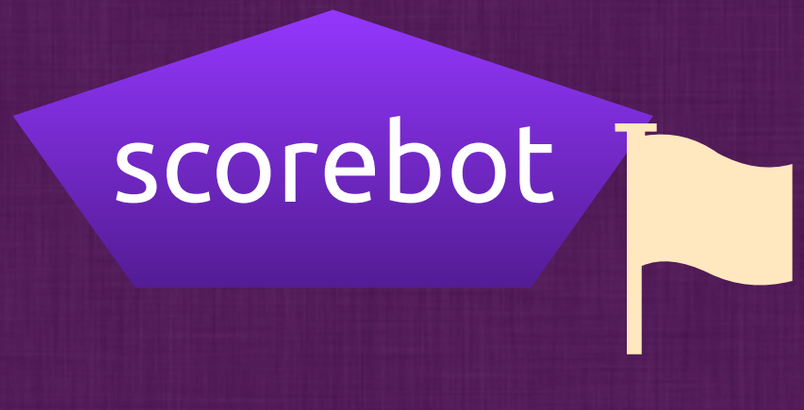
PPP
atmail

steal

Shellphish



Attack-Defense

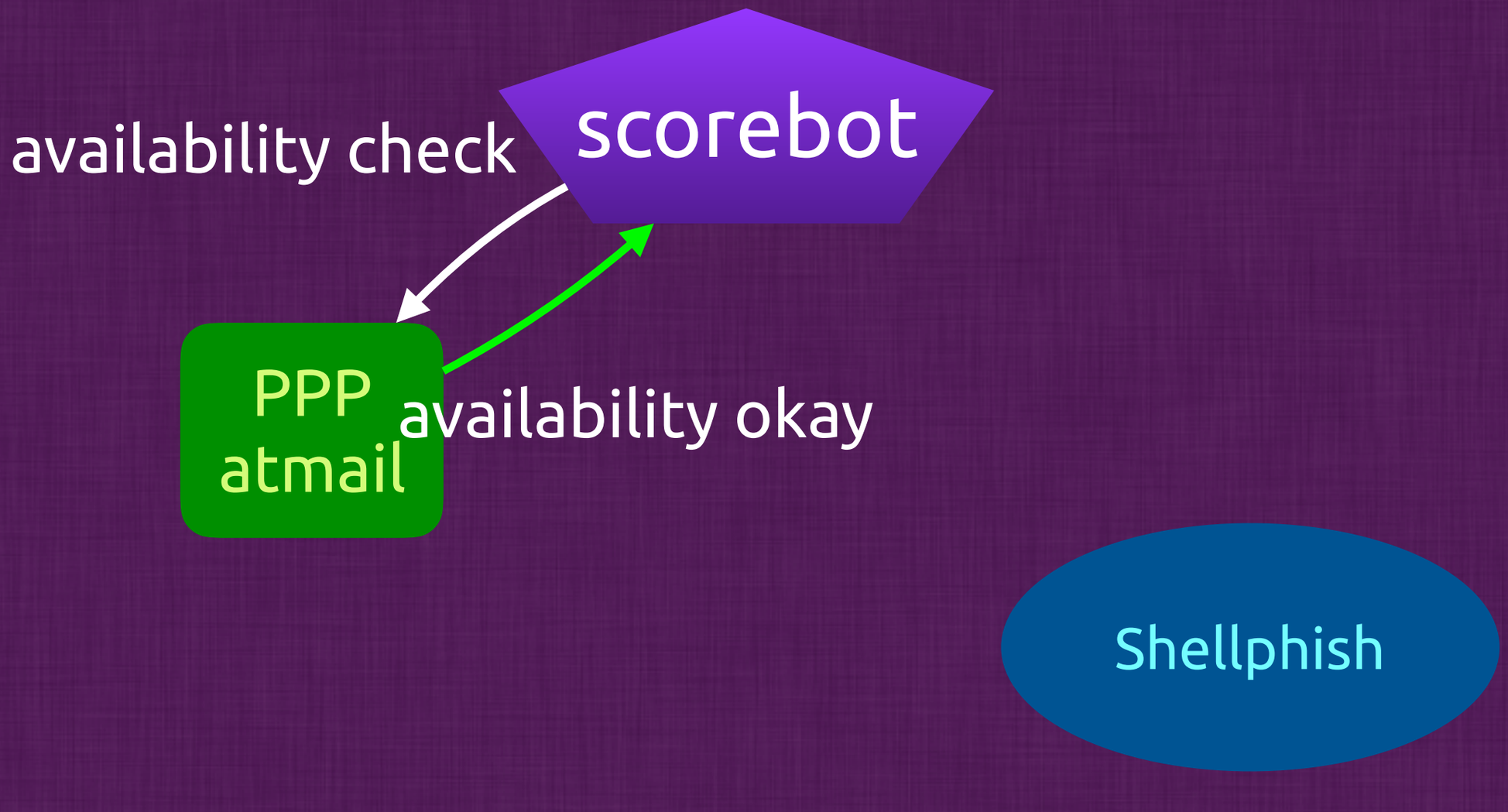


PPP
atmail

redeem

Shellphish

Attack-Defense



Attack-Defense

failed availability

scorebot

PPP
atmail

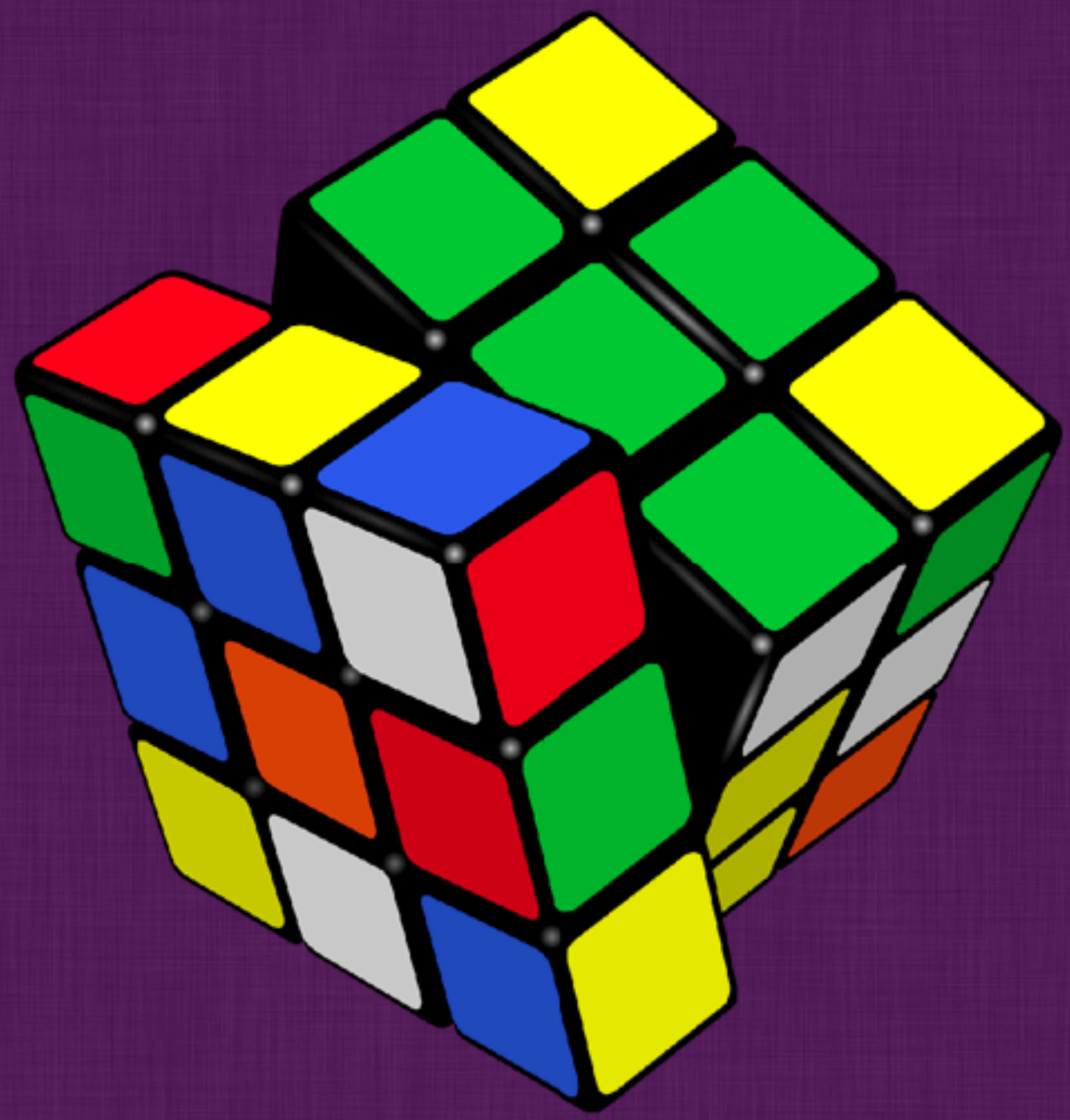
can’t steal

Shellphish

Attack-Defense

DEF CON CTF Finals

RUBIX



RUBIX

3. xclient@a84249bfa09c: /asby (zsh)

xclient@a84249bfa0... ⌘1

```
This service implements a rubix cube. Solve the cube and win.

Give me 54 Bytes(ie: 53,12,5,etc): <input 54 comma-delimited chars>

                Top
               R6 R7 B0
               L1 T4 R5
               A0 B7 R2

  Left          Front        Right        Back

L8 L7 L0      T0 A7 F0     R8 F3 L6     B6 T3 T6
T5 L4 B1      F7 F4 A1     T1 R4 T7     F1 A4 R1
B8 B3 F8      A6 R3 L2     T2 L3 B2     R0 L5 A8

               Bottom
               A2 F5 F6
               B5 B4 A3
               T8 A5 F2


Action(U,U',D,D',L,L',R,R',F,F',B,B'):
```

RUBIX

54 Rubik's cube instructions

...becomes shellcode

Lab RATs on Rubix

Lab RATs posted a write-up:

https://blog.rpis.ec/2017/08/defcon-finals-2017-introduction-rubix.html

Lab Rats on Rubix

1. write 9-bit to 8-bit netcat
2. analyze 9-bit strings in libc
3. symbolize libc
4. figure out how main() gets called

Now the actual analysis starts...

Attack-Defense

- How is it supposed to work?
- How can we attack it?
- How can we defend it?

Attack-Defense

- Get points by capturing flags
- Lose points by having flags captured
- Lose *lots* of points by failing checks

Attack-Defense

Complicated, frustrating, fun!

CTF

Extremely ambitious

CTF

- Running Smoothly
- Fair Contest
- Fun Challenges

ROUNDS SMOOTHLY

Running Smoothly

Starts early

Running Smoothly

Who's on the team?

Legitimate Business Syndicate

- Half 2005-2007 university team
- Half 2012 coworkers



Legitimate Business Syndicate in 2006

Legitimate Business Syndicate

- August 2012: ddtek steps down
- December 2012: Gyno starts recruiting
- February 2013: Proposal submitted
- March 2013: Proposal accepted

Legitimate Business Syndicate

- "Reverse engineers" 3/4 of the group
- Different specialties
- Radio: 2014, badger
- Hardware: 2015, the year of single-board computers
- Esoteric computing: 2017, cLEMENCy

Legitimate Business Syndicate

100% dependent on Selir's amazing infrastructure

Legitimate Business Syndicate

I started for the database backed web application

Team Building

People grow and change

Team Building

Roles grow and change

Team Building

- Who do you know?
- Who do you trust?
- Who do you like?

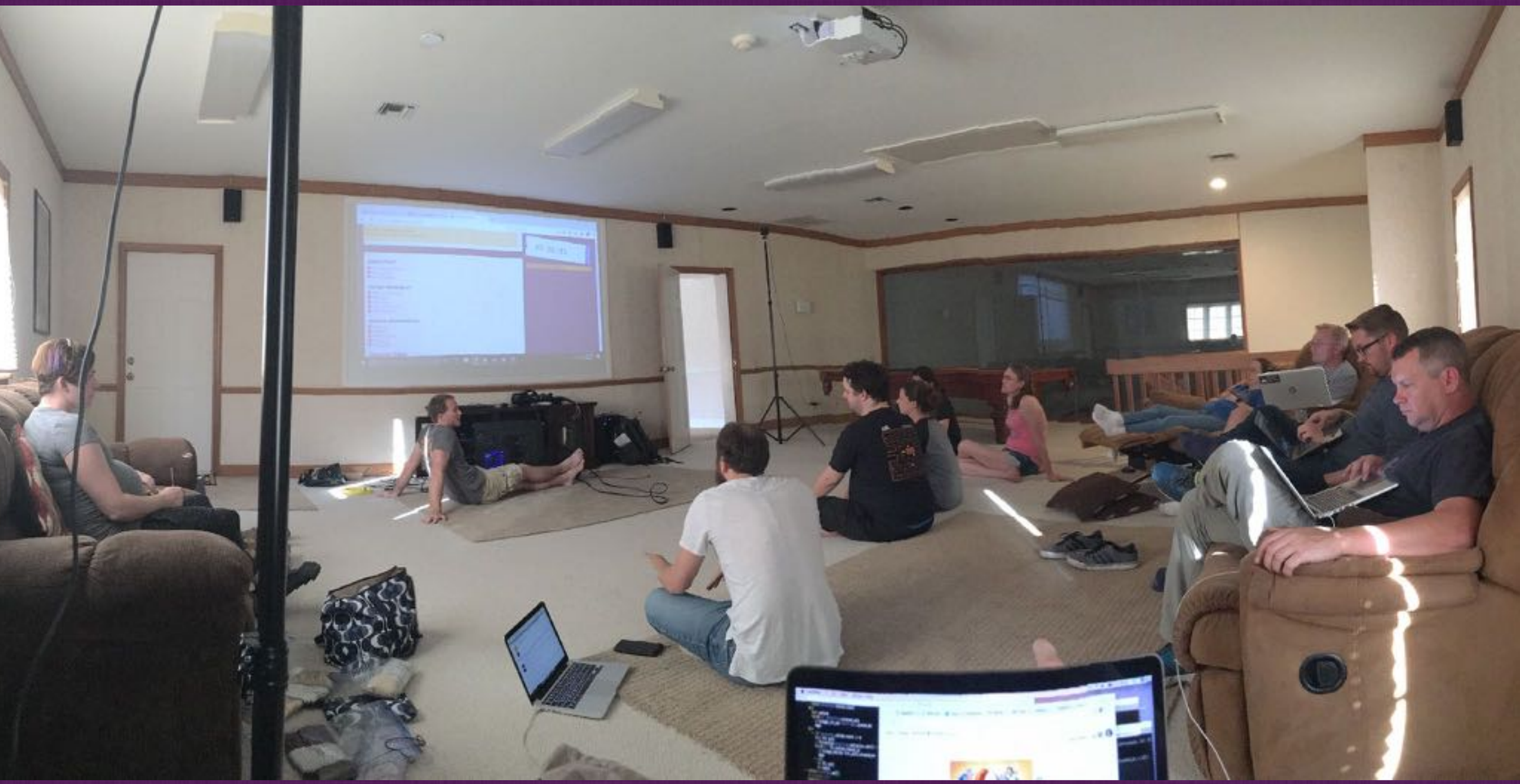
COMMUNICATION

"It's good."

COMMUNICATION

async (chat) is great

weekly meetings are great



Smooth Operation

Support your team

Smooth Operation

CTF software is software

Smooth Operation

Automate testing and deployment

CTF

- Running Smoothly
- Fair Contest
- Fun Challenges

FAIR CONTEST

Fair Contest

CTF is computer hacking

Fair Contest

CTF is computer system

Fair Contest

Hack the right thing the wrong way

Fair Contest

Hack the wrong thing

Fair Contest

Fix a thing the "wrong" way

Fair Contest

Restrict players more

Qualifiers

- Services on separate hosts
  - Multiple hosts in different locations
- Connections get separate container
  - `xinetd` and `runc`
- Limit system calls
  - `seccomp`

FINALS

More complex game

More complex problems

Finals

- Keep the game about reverse engineering
- (Not OS administration)

Finals

- 2013: unprivileged team account, unprivileged service accounts
- 2014: understood "Superman defense" better

Superman Defense

- Block opponent IPs
- Prevent reading the flag

Cyber Grand Challenge

US Defense Advanced Research Projects Agency (DARPA) project starting in 2014

Cyber Grand Challenge

CTF for autonomous computers

Cyber Grand Challenge

Extremely formalized

Challenge Binaries

- "CBs"
- 32-bit i386
- Special CGCEF executable format
- Limited system calls
- No retained state

Proof of Vulnerability

- "PoVs"
- 32-bit i386 CGCEF
- Demonstrate a vulnerability:
  - Register control
  - Memory disclosure
- Run by scoring system

Offline Evaluation

- Team interface gives out binaries
- Team interface collects replacement CBs, PoVs
- Runs availability checks and PoVs in isolation
- Designed for reproducibility and audibility

Finals

- 2015: restrict system calls
- 2016: use CGC game format
- 2017: everything in limited emulator

Fair Contest

Release scoring information

Fair Contest

Think about accessibility

CTF

- Running Smoothly
- Fair Contest
- Fun Challenges

Fun Challenges

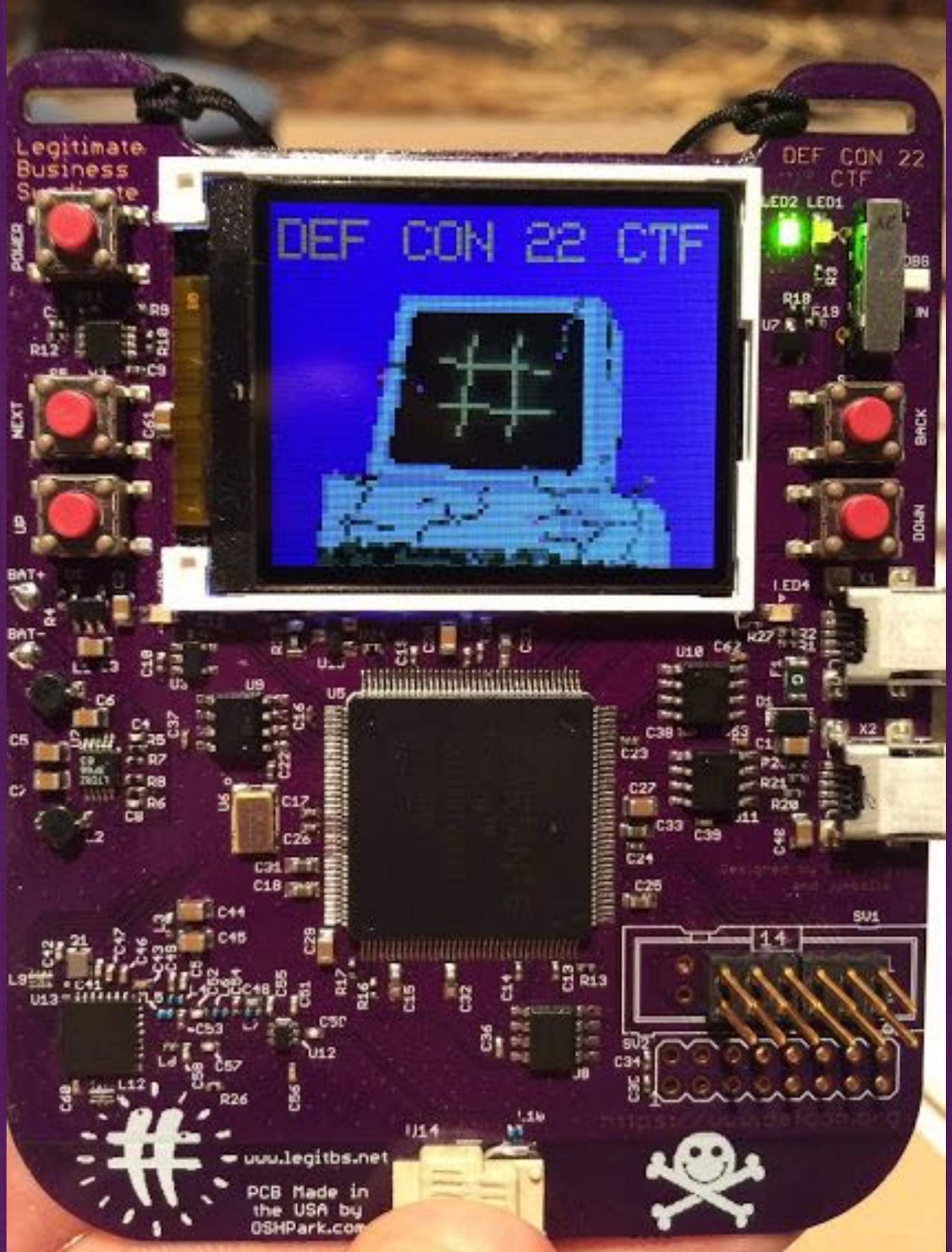
Break expectations

DOSFUNKY

- Discover that it's a DOS binary
- Debug and patch IDA Pro
- Start actual reverse engineering

BADGER

- MSP-430 on physical hardware
- custom CDMA radio network


DEF CON 22 CTF
Legitimate Business
DEF CON 22 CTF
LED2 LED1
POWER
NEXT
UP
BACK
DOWN
IN
BAT+
BAT-
LED4
SV1
14
www.legitbs.net
PCB Made in the USA by OSHPark.com

Consensus Evaluation

- CGC's big attack-defense innovation
- Everyone sees everyone else's patched binaries
- Explosion in number of binaries that need reversing

1000 cuts / crackme2000

Push teams into
automated analysis
Hundreds of binaries

Consensus Evaluation in 2016

Player asks about losing points

Service being attacked, that's why

"But we're using the same binariess as the winning team"

Consensus Evaluation in 2017

Rubix expected shellcode to work in availability checks

Defenders would add checks to block "evil" or allow "good" shellcode

Attackers would build new shellcode to pass checks

"Felt like a multiplayer game against humans"

CTF

- Running Smoothly
- Fair Contest
- Fun Challenges

CTF

Still more to learn!

CTF

More work ahead of us

CTF

Opportunity to grow for more players

CTF

Best way to learn is to do

Five years with the best group of people I've ever worked with

Five years building a contest for the friendliest and smartest community I know

Thanks for making it amazing!

www.oooverflow.io

Welcome, young hacker.

Who are we?

We are the Order of the Overflow. You can read about our philosophy here or reach us at team@oooverflow.io.

When are the quals?

The qualifying event for DEF CON CTF 2018 Finals will start on 00:00 UTC on May 12th and go for 48 hours.

What are the pre-qualifying events?

The winners of the following events will be automatically pre-qualified for finals:

DEF CON CTF - 28 July 2017 - prequalified: PPP

vito@legitbs.net

@vito_lbs