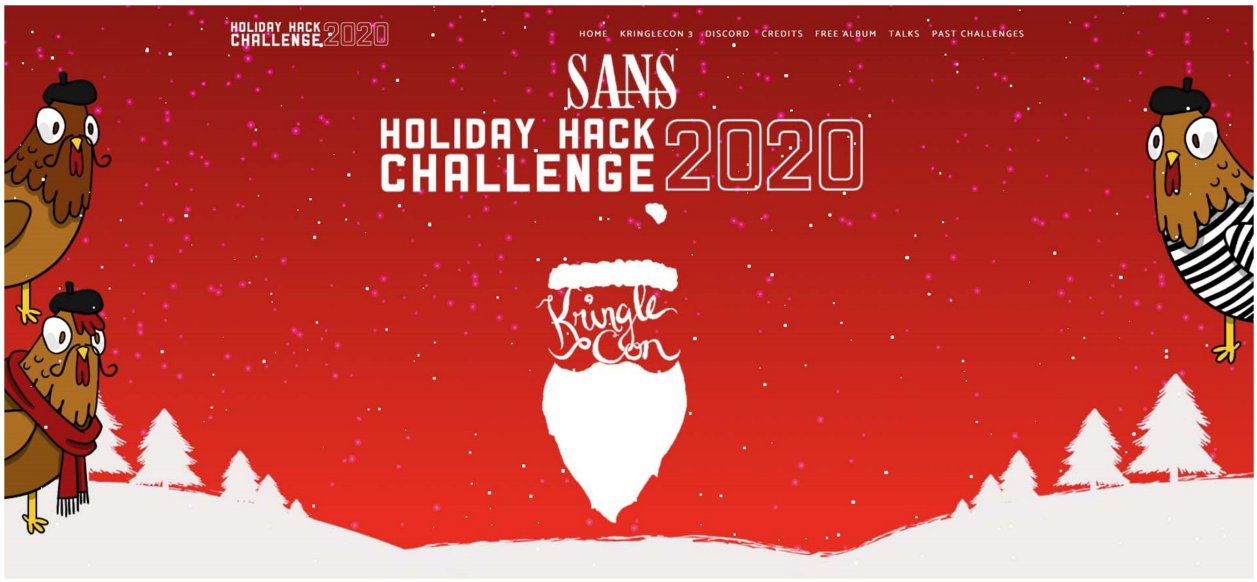
SANS Holiday Hack 2020 – Extra



Authors:

Father and son team: Matt (son) and Jim Kirn (father)

(**jj**) in game, @infosecjim in Discord, @JimKirn on Twitter

Introduction

Below are several items that didn't make it into our report submitted on 12/26/2001. We were limited to 50 pages as per the guidelines. So, we created this supplemental addition to cover topics we wanted to document but that weren't needed for the primary submission.

Talks

All the talks are available via youtube.com – KringlecCon2020
https://www.youtube.com/playlist?list=PLjLd1hNA7YVwqXqaBJfbXqkFb7LKw3r31

Kringle Con 2020

***SANS Holiday Hack Challenge 2020**, Ed Skoudis, Track 1*
https://www.youtube.com/watch?v=8e0SZrbWFuU
In this presentation, Ed welcomes you to the 2020 SANS Holiday Hack Challenge, orienting you to the environment, the characters, the storyline, and the super useful KringleCon 3 snowflake badge. He gives tips for navigating Santa's castle and its interface, as well as ways to chat with other participants and get hints. In 18 short minutes, Ed provides you all the information you need to get rolling in this year's super exciting Holiday Hack extravaganza!

***Open S3 Buckets: Still a Problem in 2020**, Joshua Wright, Track 2*
https://www.youtube.com/watch?v=t4UzXx5JHk0
SANS Senior Instructor Joshua Wright delivers a lightning talk about what you need to know about insecure cloud storage discovery, enumeration, and the opportunities to make money through creative assessment of cloud resources.

***CAN Bus Can-Can**, Chris Elgee, Track 2*
https://www.youtube.com/watch?v=96u-uHRBI0I
Riddle: what connects your steering wheel to your door locks and your radio? It's the CAN bus! Let's examine what this low-level network does and finally find out what our cars are thinking!

***Working with the Official Naughty/Nice Blockchain**, Prof. Qwerty Petabyte, Track 3*
https://www.youtube.com/watch?v=reKsZ8E44vw
In this talk, part of Elf U's "Assessing and Evaluating Human Behavior for Naughty/Niceness" curriculum, Professor Petabyte outlines blockchain technology, and specifically outlines how the North Pole uses blockchains to ensure the integrity of the Naughty/Nice list, gives an overview of the data stored on the Naughty/Nice blockchain, and talks about the current two year project to update the North Pole's blockchain code.

***Adversary Emulation and Automation**, Dave Herrald, Track 3*
https://www.youtube.com/watch?v=RxVgEFt08kU
Learn a quick, easy, and free way to emulate adversary techniques selected from MITRE ATT&CK® and the Atomic Red Team project. We'll show how the resulting telemetry can be collected for analysis and detection engineering using Splunk.

***HID Card Hacking**, Larry Pesce, Track 4*
https://www.youtube.com/watch?v=647U85Phxgo

The HID ProxCard II RFID cards are arguably the most deployed physical access control systems. In this talk we'll give you the quick technical run down on the technology and how we can interact with them for shenanigans with a Proxmark 3.

Rudolph the Red-Nosed Raptor: Acquiring Triage Images via EDR*, *Dan Banker, Track 4

https://www.youtube.com/watch?v=VWDiA6WspSM

Velocidex has created a fantastic (free!) tool called Velociraptor; in their own words, it "...provides the next generation in endpoint monitoring, digital forensic investigations and cyber incident response." One capability is that the user can specify an array of forensic artifacts to be collected, and Velociraptor will produce an executable. The executable is run on the target machine, and the artifacts are collected and a report is generated. This is a quick way of generating a triage image, i.e. grabbing the juicy system artifacts without copying an entire drive. I will explore the deployment of velociraptor.exe with the live response capability of EDR (specifically Carbon Black). I will demonstrate how this is done and point out a couple of pitfalls. I will also talk about how to automate this process.

Give Yourself a Blog for Christmas*, *Jack Rhysider, Track 5

https://www.youtube.com/watch?v=NKHF5VZmCig

Everyone in IT has notes they've written for things they should remember. Commands that are hard to remember, tips for how to configure something, or troubleshooting techniques. The best place to put those notes is into a blog. This talk will cover the reasons why everyone in IT should be writing a blog, and what to put in it. Even if you're just starting your career or haven't yet started it. The beginners mind is a beautiful thing and can sometimes explain things better than expert can.

Red Teaming: Why Organizations Hack Themselves*, *David Tomaschik, Track 5

https://www.youtube.com/watch?v=2ejR8ITe_uA

Penetration testing and red teaming are popular, high-visibility specialties in the information security space, but why do organizations do these, and how are they executed? We'll discuss the phases and execution of a Red Team exercise and how the results help the organization's overall security posture.

IOMs vs IOAs in AWS*, *Spencer Gietzen, Track 6

https://www.youtube.com/watch?v=KliCQbJT6YQ

There is a lot of buzz in the public cloud industry around indicators of misconfigurations, detecting them, and responding to them, but there is one important area that is lacking the same support, indicators of attack. It is important to know when there is a potential for a breach in your cloud environment, but you also need to know what malicious activity may look like after a breach. This talk will cover what Indicators of Misconfigurations (IOMs) and Indicators of Attack (IOAs) are, why they are important to differentiate, and the differences between them using Amazon Web Services (AWS) as an example.

Attacking and Hardening Kubernetes*, *Jay Beale, Track 6

https://www.youtube.com/watch?v=S4ySed0k7uE

Come see a Kubernetes attack demonstration, where a hostile developer must escalate privilege to steal data from a GKE Kubernetes cluster and its cloud environment. Whether you're completely new to Kubernetes or you've used it, but not yet attacked it, you'll enjoy and learn something useful from this talk. Afterwards, download the slides from InGuardians.com and learn about using admission controllers to block the attack!

Random Facts About Mersenne Twisters*, *Tom Liston, Track 7

https://www.youtube.com/watch?v=Jo5Nlbqd-Vg

https://github.com/tliston/mt19937

An introduction to the properties and pitfalls of one of the most widely deployed pseudo-random number generators (PRNGs), the Mersenne Twister, MT19937. Along with this presentation, Tom is releasing some Python code, demonstrating how to clone the PRNG used by the Random module in both Python 2 and Python 3.

***Offensive Security Tools: Providing Value with the C2 Matrix**, Jorge Orchilles, Track 7*
https://www.youtube.com/watch?v=CcteG3Z2nCU
Offensive Security has always been about providing value. This talk goes through the history of ethical hacking through red team, purple team, and adversary emulation. Choosing the correct tools for the job has always been an important preparation step; with the C2 Matrix, you can quickly choose the best one for your needs. Lastly, we release an update for the SANS Slingshot C2 Matrix Edition virtual machine which includes multiple C2s preinstalled to get you up and running quickly. It also includes VECTR to measure, track, and show the progress made in your Red Team and Purple Team programs.

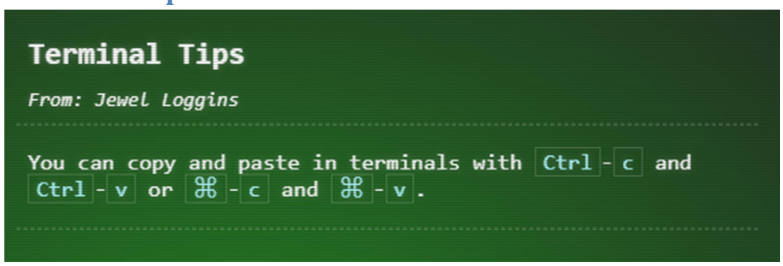
Other Previous Talks that may be of use

***Analyzing PowerShell Malware (2018)**, Chris Davis*
https://www.youtube.com/watch?v=wd12XRq2DNk
In this talk we discuss how to properly reverse engineering many types of powershell malware from analyzing dropper downloads to powershell memory analysis.

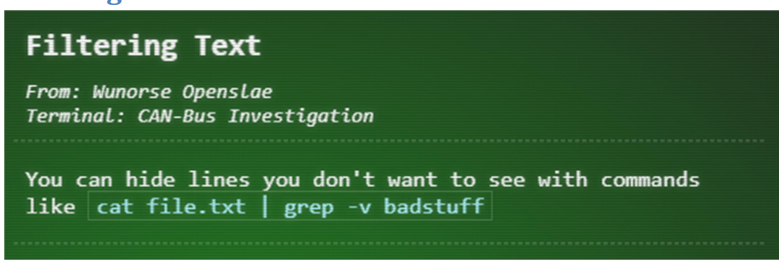
***Dashing Through the Logs (2019)**, James Brodsky*
https://www.youtube.com/watch?v=qbIhHhRKQCw
If you want your hunt to be successful, you need to look where the threats are. In modern environments, that means collecting endpoint and email logs and knowing what to search for in it. In this talk, we will cover critical Windows-based security event log sources like Sysmon, PowerShell, and process launch events. Additionally, we will introduce the stoQ automation framework for analyzing email. We'll show you how to use this data to pragmatically hunt for threats operating in your environment.

Hints

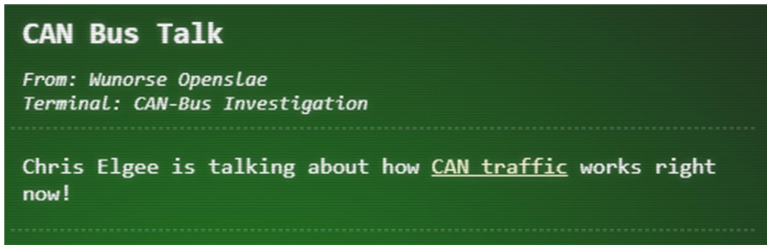
Terminal Tips



Filtering Text

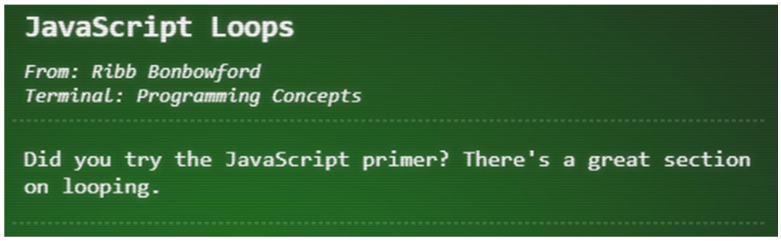


CAN Bus Talk

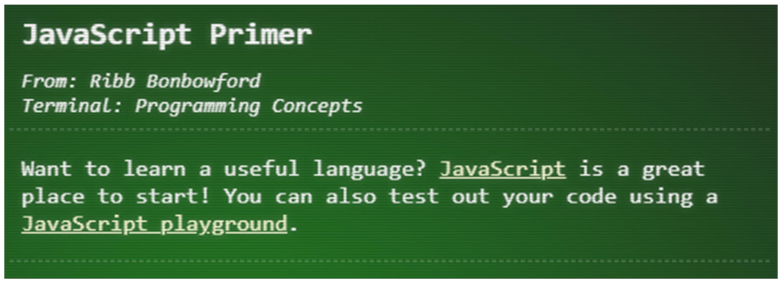


https://www.youtube.com/watch?v=96u-uHRBI0I

JavaScript Loops

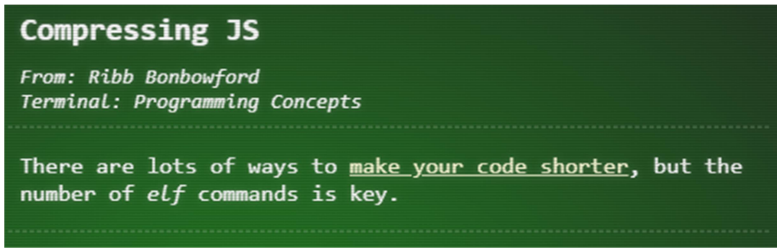


JavaScrip Primer



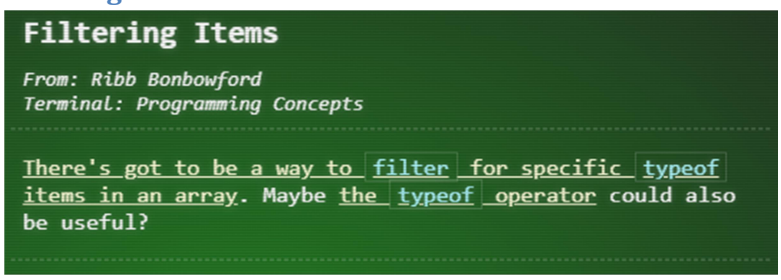
https://jgthms.com/javascript-in-14-minutes/
https://playcode.io/

Compressing JS



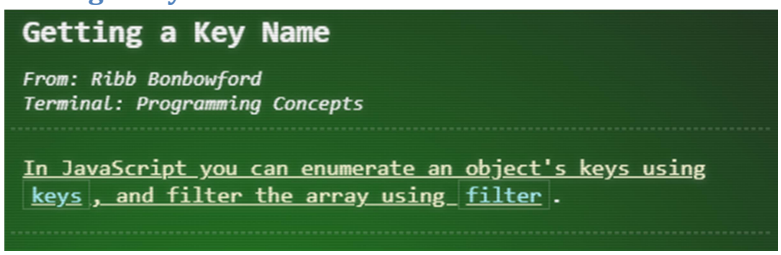
https://jscompress.com/

Filtering Items



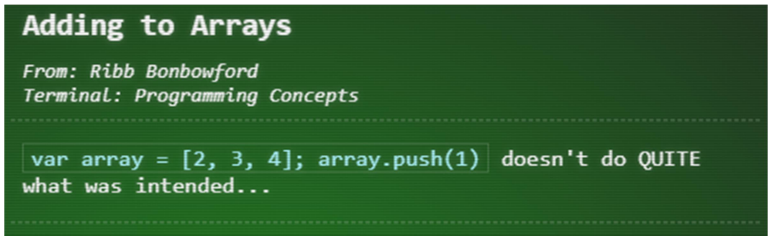
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/TypedArray/filter
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/typeof

Getting a Key Name

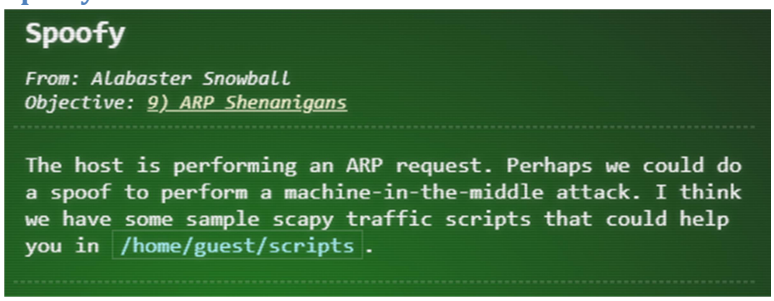


https://stackoverflow.com/questions/9907419/how-to-get-a-key-in-a-javascript-object-by-its-value

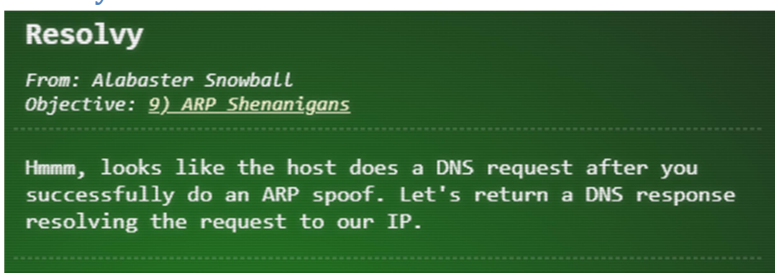
Adding to Arrays



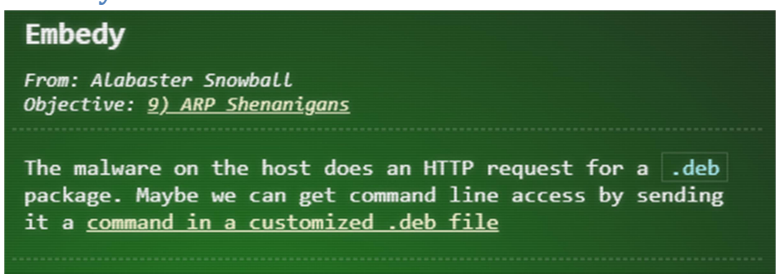
Spoofy



Resolvy

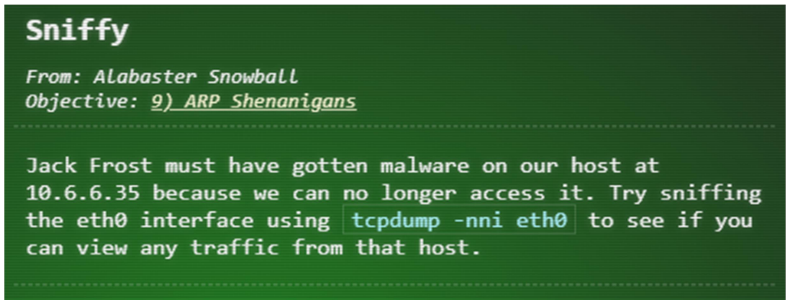


Embedy



http://www.wannescolman.be/?p=98

Sniffy



MD5 Hash Collisions

https://github.com/corkami/collisions

Blockchain Talk

https://www.youtube.com/watch?v=7rLMl88p-ec

Block Investigation

Block Investigation

From: Tangle Coalbox
Objective: 11b) Naughty/Nice List with Blockchain Investigation Part 2

The idea that Jack could somehow change the data in a block without invalidating the whole chain just collides with the concept of hashes and blockchains. While there's no way it could happen, maybe if you look at the block that seems like it got changed, it might help.

Minimal Changes

Minimal Changes

From: Tangle Coalbox
Objective: 11b) Naughty/Nice List with Blockchain Investigation Part 2

Apparently Jack was able to change just 4 bytes in the block to completely change everything about it. It's like some sort of evil game to him.

https://speakerdeck.com/ange/colltris

Blockchain Chaining

Blockchain ... Chaining

From: Tangle Coalbox
Objective: 11b) Naughty/Nice List with Blockchain Investigation Part 2

A blockchain works by "chaining" blocks together - each new block includes a hash of the previous block. That previous hash value is included in the data that is hashed - and that hash value will be in the next block. So there's no way that Jack could change an existing block without it messing up the chain...

Unique Hash Collision

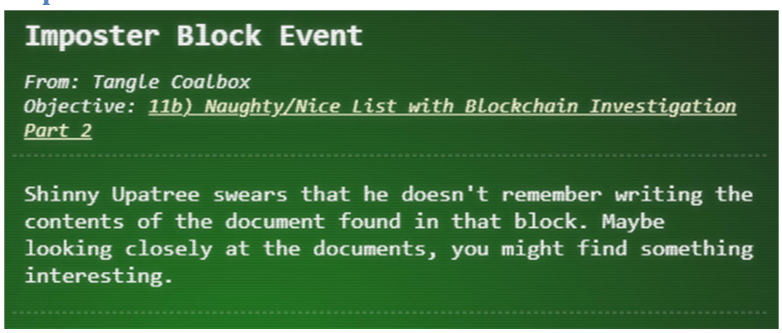
Unique Hash Collision

From: Tangle Coalbox
Objective: 11b) Naughty/Nice List with Blockchain Investigation Part 2

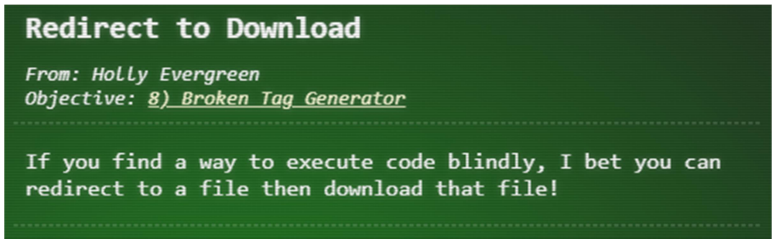
If Jack was somehow able to change the contents of the block AND the document without changing the hash... that would require a very UNIque hash COLLision.

https://github.com/cr-marcstevens/hashclash

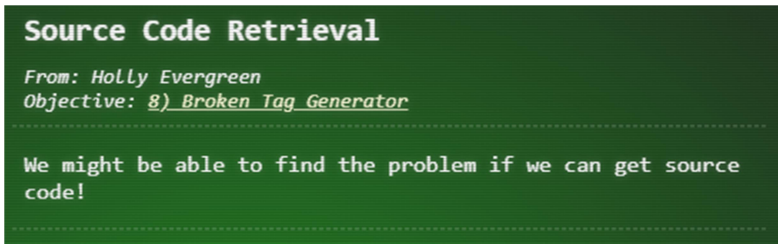
Imposter Block Event



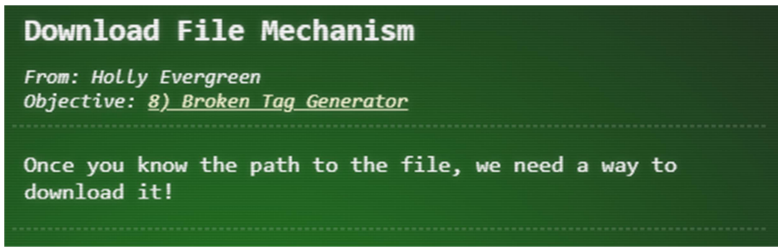
Redirect to Download



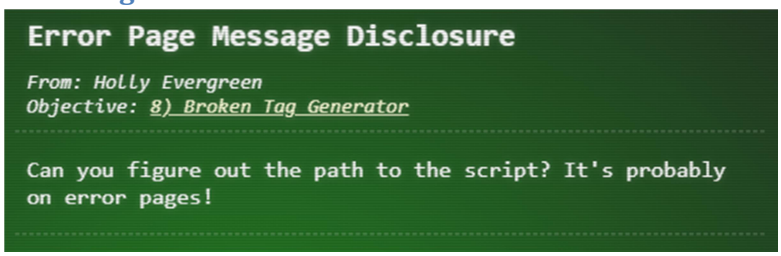
Source Code Retrieval



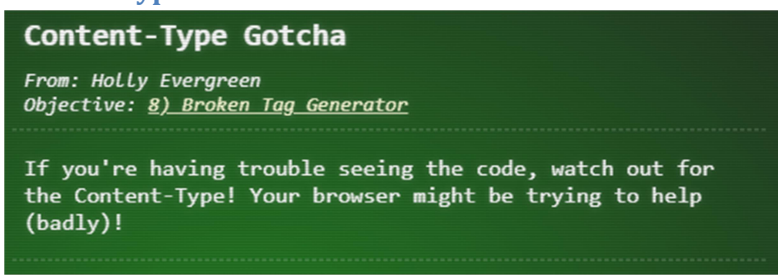
Download File Mechanism



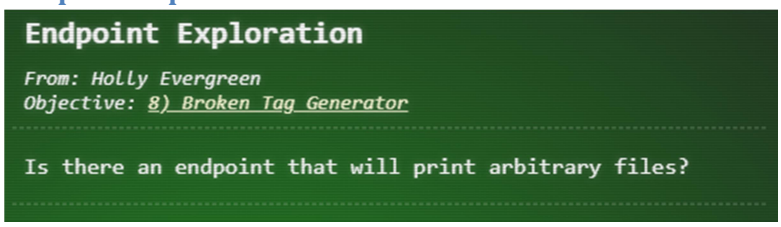
Error Page Disclosure



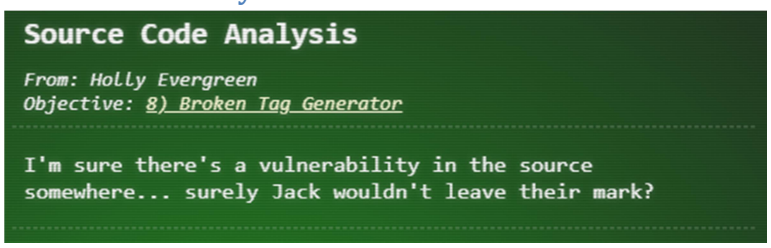
Content Type Gotcha



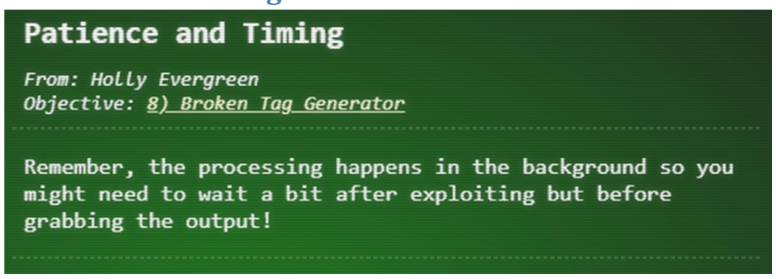
Endpoint Exploration



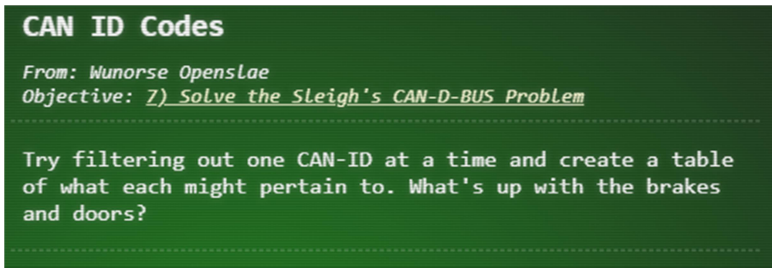
Source Code Analysis



Patience and Timing



CAN ID Codes



Twirl Area

Twirl Area

From: Jingle Ringford
Objective: 1) Uncover Santa's Gift List

Make sure you Lasso the correct twirly area.

Image Edit Tool

Image Edit Tool

From: Jingle Ringford
Objective: 1) Uncover Santa's Gift List

There are tools out there that could help Filter the Distortion that is this Twirl.

https://www.photopea.com/

Santavator Operations

Santavator Operations

From: Pepper Minstix
Objective: 4) Operate the Santavator

It's really more art than science. The goal is to put the right colored light into the receivers on the left and top of the panel.

Santavator Bypass

Santavator Bypass

From: Ribb Bonbowford
Objective: 4) Operate the Santavator

There *may* be a way to bypass the Santavator S4 game with the browser console...

Electron Applications

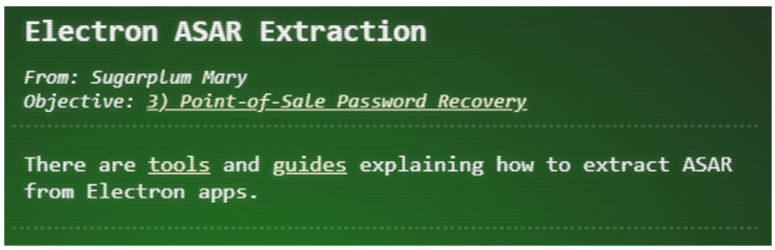
Electron Applications

From: Sugarplum Mary
Objective: 3) Point-of-Sale Password Recovery

It's possible to extract the source code from an Electron app.

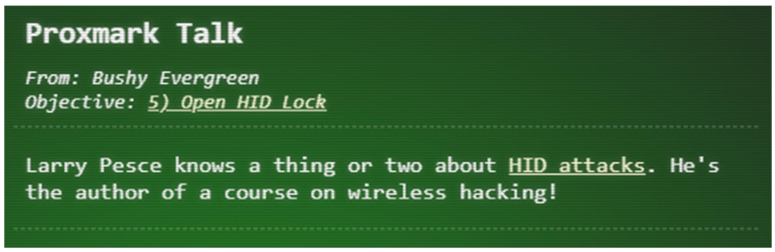
https://www.electronjs.org/

Electron ASAR Extraction



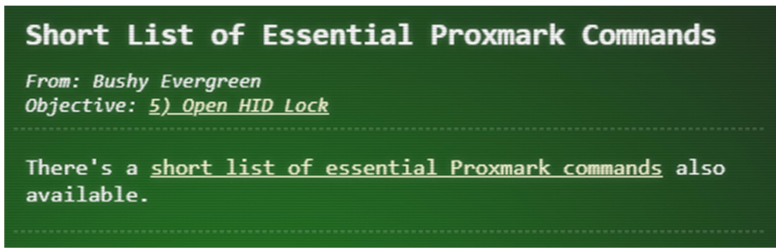
https://www.npmjs.com/package/asar
https://medium.com/how-to-electron/how-to-get-source-code-of-any-electron-application-cbb5c7726c37

Proxmark Talk



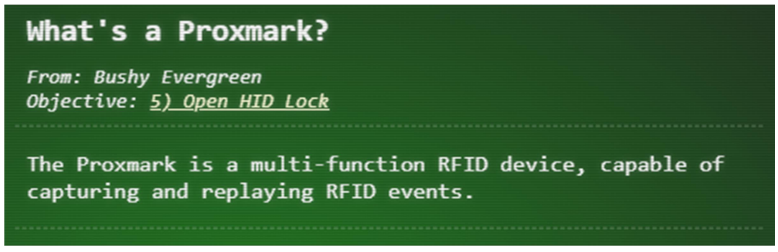
https://www.youtube.com/watch?v=647U85Phxgo

Short List of Essential Proxmark Commands



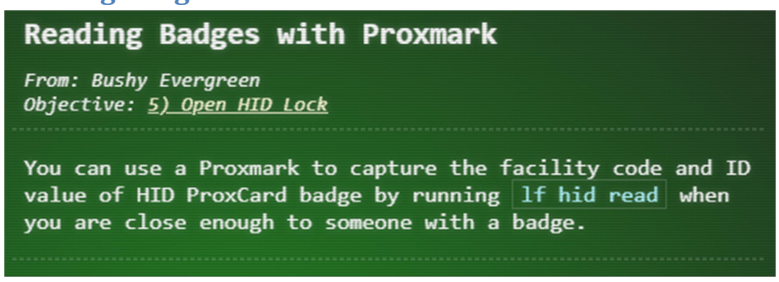
https://gist.github.com/joswr1ght/efdb669d2f3feb018a22650ddc01f5f2

What's a Proxmark

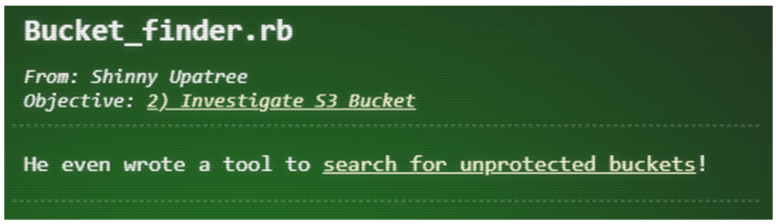


Impersonating Badges with Proxmark

Reading Badges with Proxmark



Bucket_finder.rb

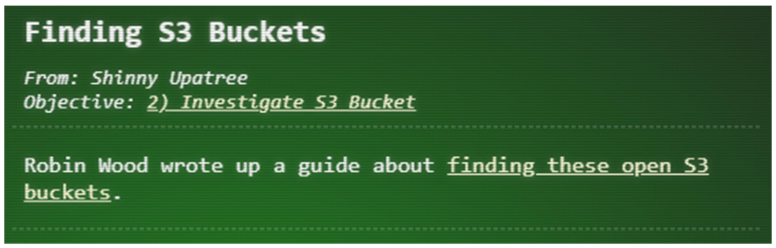


https://digi.ninja/projects/bucket_finder.php

Leaky AWS S3 Buckets

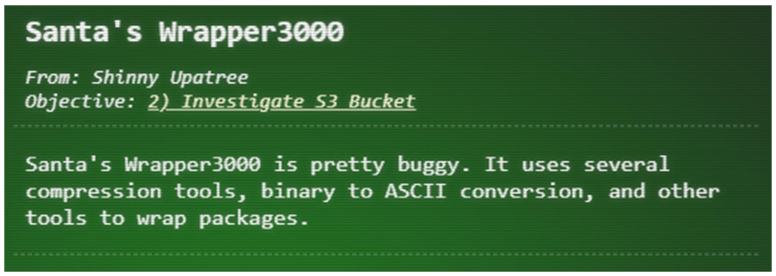
https://www.computerweekly.com/news/252491842/Leaky-AWS-S3-bucket-once-again-at-centre-of-data-breach

Finding S3 Buckets

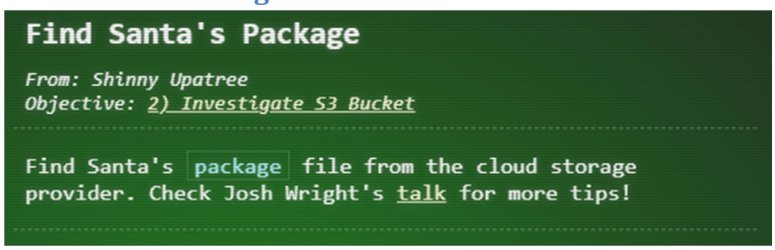


https://digi.ninja/blog/whats_in_amazons_buckets.php

Santa's Wrapper3000

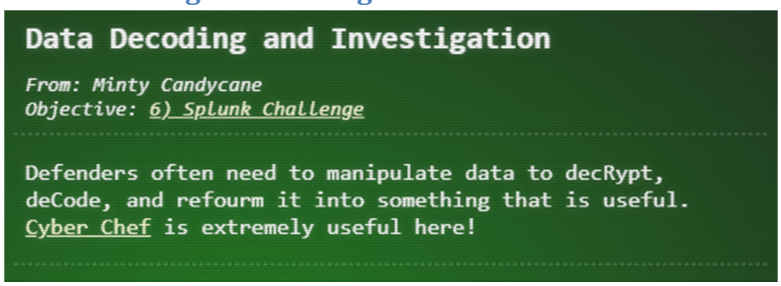


Find Santa's Package



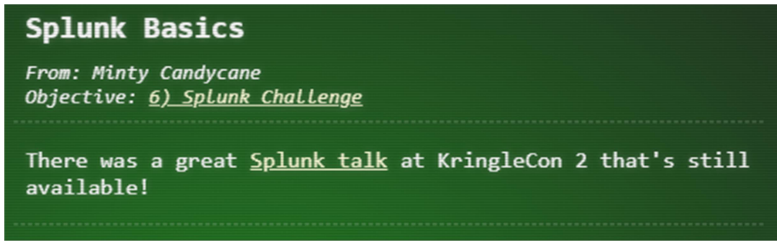
https://www.youtube.com/watch?v=t4UzXx5JHk0

Data Decoding and Investigation



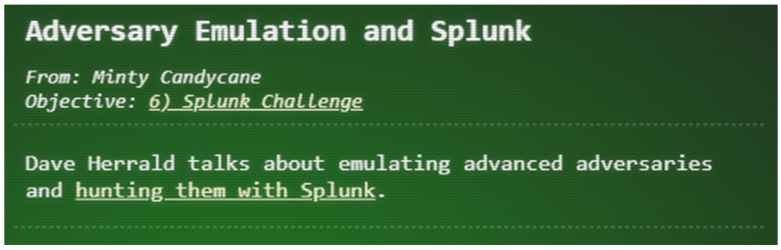
https://gchq.github.io/CyberChef/

Splunk Basics



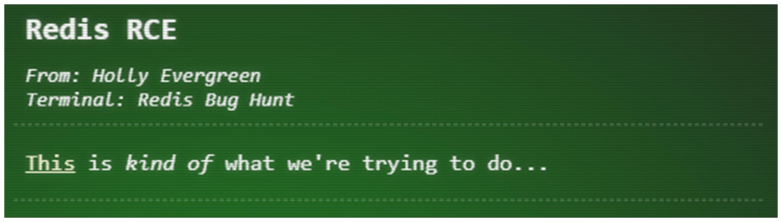
https://www.youtube.com/watch?v=qbIhHhRKQCw

Adversary Emulation with Splunk



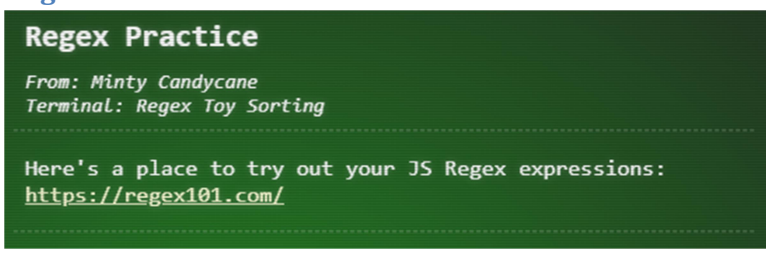
https://www.youtube.com/watch?v=RxVgEFt08kU

Redis RCE



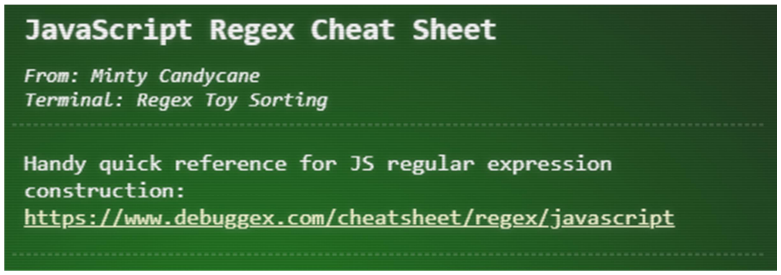
https://book.hacktricks.xyz/pentesting/6379-pentesting-redis

Regex Practice



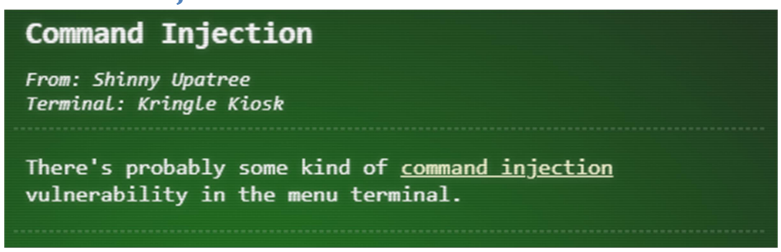
https://regex101.com/

JavaScript Regex Cheat Sheet



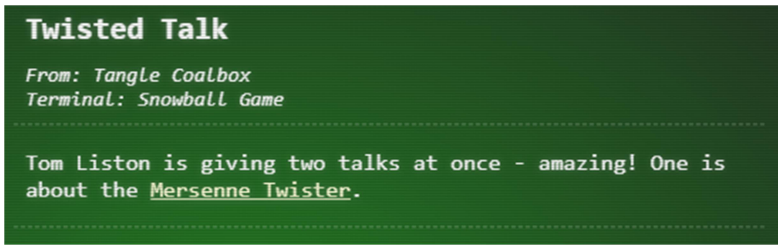
https://www.debuggex.com/cheatsheet/regex/javascript

Command Injection



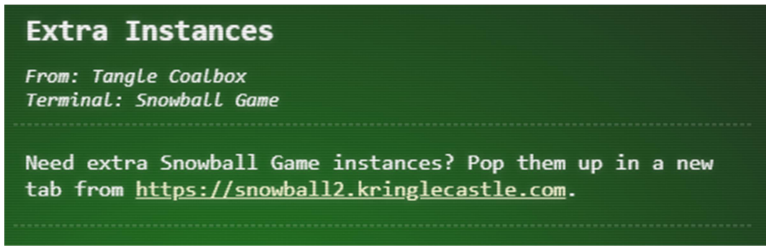
https://owasp.org/www-community/attacks/Command_Injection

Twisted Talk



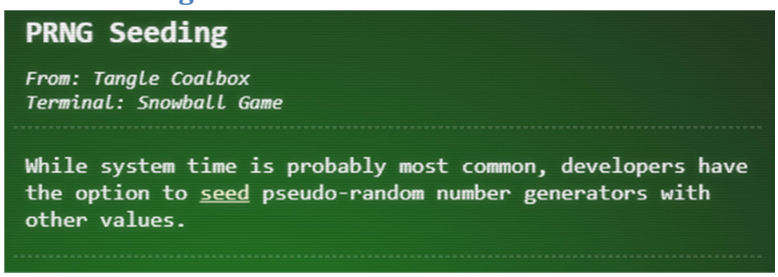
https://www.youtube.com/watch?v=Jo5Nlbqd-Vg

Extra Instances



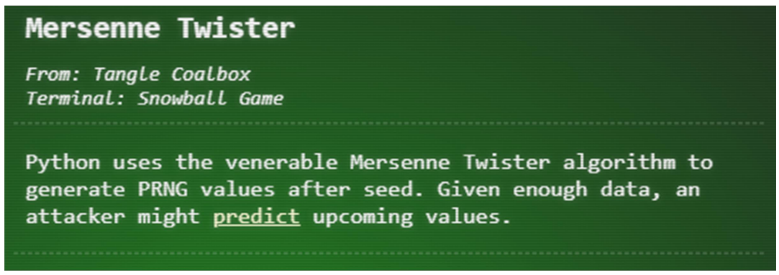
https://snowball2.kringlecastle.com/

PRNG Seeding



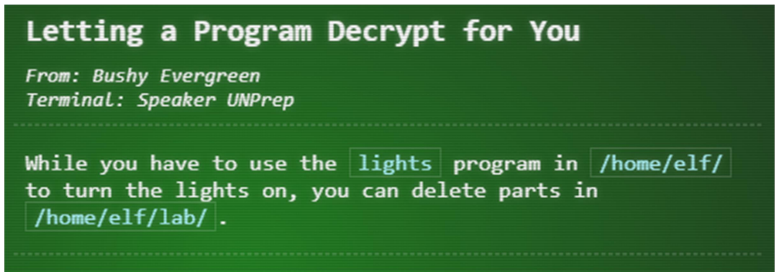
https://docs.python.org/3/library/random.html

Mersenne Twister

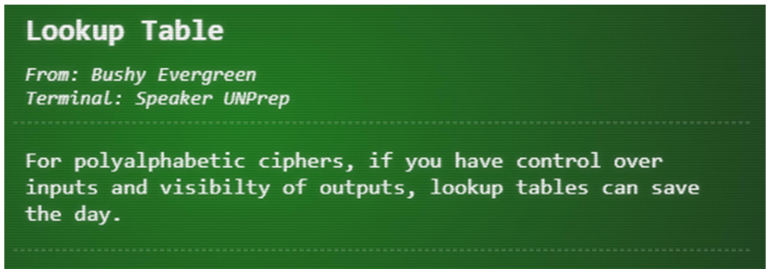


https://github.com/kmyk/mersenne-twister-predictor/blob/master/readme.md

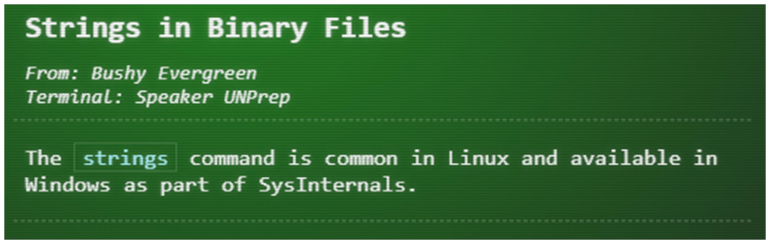
Letting a Program Decrypt for You



Lookup Table



Strings in Binary Files



Tmux Cheat Sheet

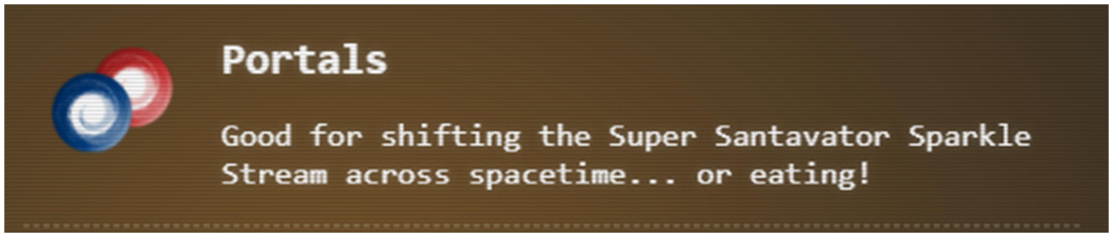
https://tmuxcheatsheet.com

Items – Needed to Complete Challenges

Large Marble

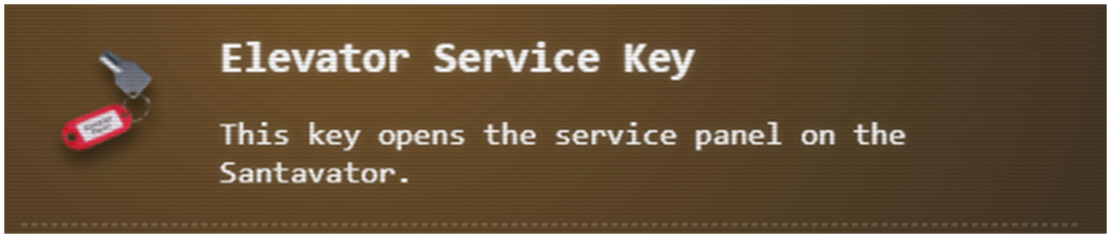
Location: Wrapping Room?
Use: In the Santavator service panel, it is used to attract streams.

Portals



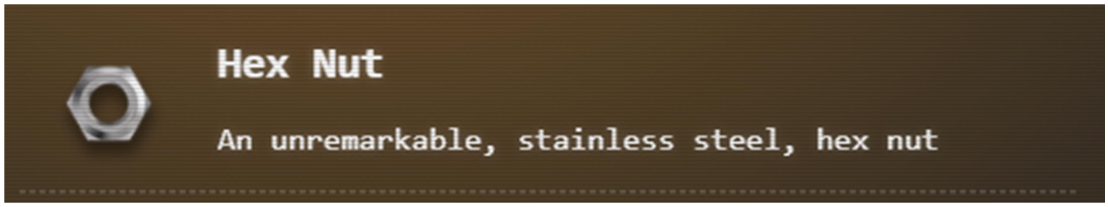
Location: SpeakerUnPrepareness Room -Vending Machine. However, you need to completely solve the Speaker UNPrep terminal (open door, lights on, and power on vending machine).
Use: You can now use the transporter function on your badge.

Elevator Service Key



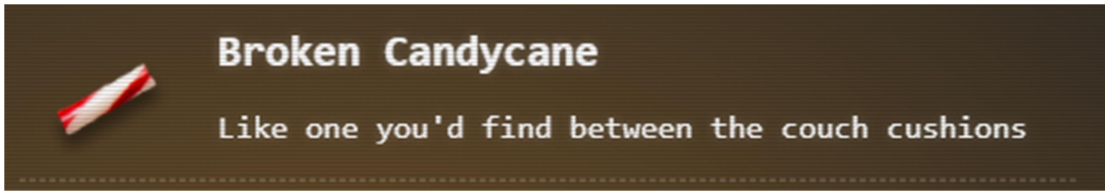
Location: Entry Area, talk to Sparkle Redberry – she gives you the key.
Use: Santavator – allows you to have access to the service panel.

Hex Nut (1)



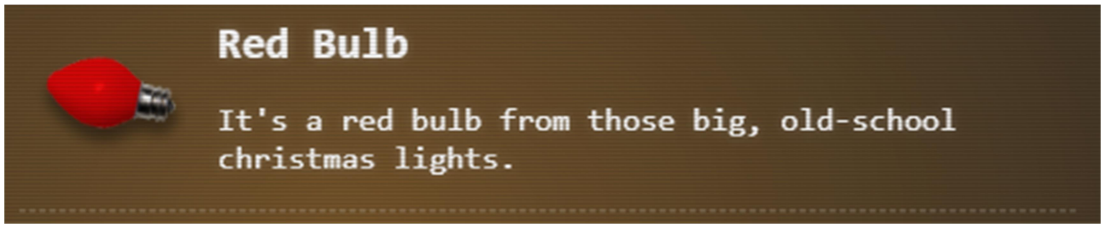
Location: Entry – just to the right of the Santavator.
Use: Santavator service panel – deflects the streams.

Broken Candycane



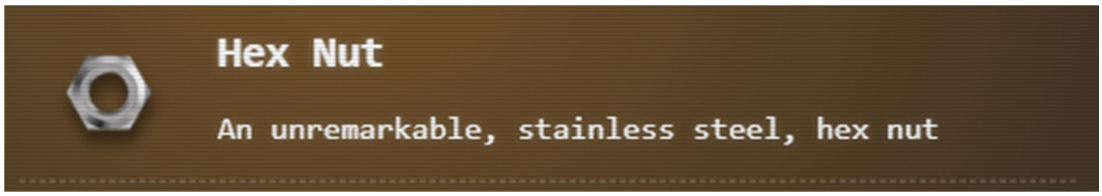
Location: Castle Approach – by the door.
Use: Santavator service panel – split a stream.

Red Bulb



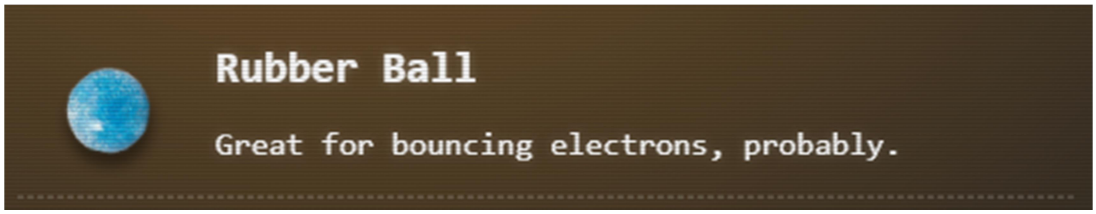
Location: Talks Lobby – to the right of Track 7
Use: Santavator service panel – color stream red. This is needed to get access to get to the Workshop (Level 1 ½).

Hex Nut (2)



Location: Dining Room – move to top of table.
Use: Santavator service panel – deflects the streams.

Rubber Ball



Location: didn't find this
Use: Santavator service panel - deflect streams

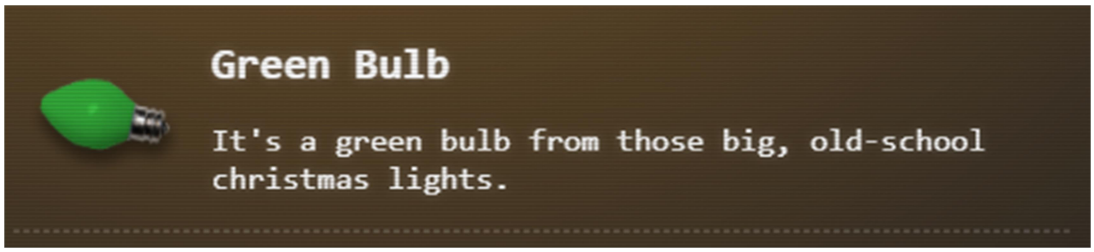
Proxmark3

Location: Go to Workshop, then to the Wrapping Room – the proxmark device is on the floor to the right of the table.
Use: The device is used to read badges (think copy info) and then simulate that badge. It is used to gain access the ??? Room via the HID reader next to the its door.

Yellow Bulb

Location: Roof (NetWars) – to the left of the sleigh.
Use: Santavator service panel – color stream yellow.

Green Bulb



Location: Courtyard – upper left corner.
Use: Santavator service panel – color a stream green.

Elevator 1.5 Button

Location: Speaker UNPreparedness Room – button is just to the bottom by the door. However, you need to gain access to this room by solving the Speaker UNPrep terminal to open the door.
Use: Gain access to Workshop and Wrapping Room (Level 1 ½)

Splunk Challenge - Details

You must be Santa to complete it!

Hint: Watch "**Adversary Emulation and Automation**" by Dave Herrald, https://www.youtube.com/watch?app=desktop&v=RxVgEFt08kU

Training Question 1

How many distinct MITRE ATT&CK techniques did Alice emulate?
Answer: 13

Enter into Splunk window:
| tstats count where index=* by index
| search index=T*-win OR T*-main
| rex field=index "(?<technique>t\d+)[\.\-].0*"
| stats dc(technique)
t1033
t1057
t1059
t1071
t1082
t1105
t1106
t1123
t1204
t1547
t1548
t1559
t1566

Training Question 2

What are the names of the two indexes that contain the results of emulating Enterprise ATT&CK 1059.003? (Put them in alphabetical order and separate them with a space)

Answer: t1059.003-main t1059.003-win

Enter into Splunk window:
index=* | search index=t1059* and look at the index field

(or | tstats count where index=* by index)

Training Question 3

One technique that Santa had us simulate deals with 'system information discovery'. What is the full name of the registry key that is queried to determine the MachineGuid?

Answer: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography

System information discovery -> T1082-win

A quick google (https://social.msdn.microsoft.com/Forums/sqlserver/en-US/d617a579-278d-4e77-812e-99fa68135d0d/windows-installation-guid?forum=windowscompatibility) revealed that the MachineGuid is located at the above location

(or index=*win reg AND MachineGuid)

Training Question 4

According to events recorded by the Splunk Attack Range, when was the first OSTAP related atomic test executed? (Please provide the alphanumeric UTC timestamp.)

Answer: 2020-11-30T17:44:15Z

Enter into Splunk window:
index=attack | search OSTAP

Click on Execution Time_UTC 5

(or index=attack OSTap | sort _time asc)

Training Question 5

One Atomic Red Team test executed by the Attack Range makes use of an open source package authored by frgnca on GitHub. According to Sysmon (Event Code 1) events in Splunk, what was the ProcessId associated with the first use of this component?

Hint: https://github.com/frgnca , AudioDeviceCmdlets.dll

Answer: 3648

Enter into Splunk window:
index=* | search EventCode="1" | search process_name="*powershell*" CommandLine="*Audio*"

(or index=t1123* EventCode=1 AND NOT splunk cmdlet | table _time, CommandLine, process_id)

Training Question 6

Alice ran a simulation of an attacker abusing Windows registry run keys. This technique leveraged a multi-line batch file that was also used by a few other techniques. What is the final command of this multi-line batch file used as part of this simulation?

Hint: https://raw.githubusercontent.com/redcanaryco/atomic-red-team/master/ARTifacts/Misc/Discovery.bat

Answer: quser

The contents of the “Discovery.bat” file:
net user Administrator /domain
net Accounts
net localgroup administrators
net use

net share
net group "domain admins" /domain
net config workstation
net accounts
net accounts /domain
net view
sc.exe query
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows"
reg query HKLM\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce
reg query HKCU\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce
reg query HKLM\Software\Microsoft\Windows\CurrentVersion\RunServices
reg query HKCU\Software\Microsoft\Windows\CurrentVersion\RunServices
reg query HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Notify
reg query HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\Userinit
reg query HKCU\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\\Shell
reg query HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\\Shell
reg query HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\ShellServiceObjectDelayLoad
reg query HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnce
reg query HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnceEx
reg query HKLM\Software\Microsoft\Windows\CurrentVersion\Run
reg query HKCU\Software\Microsoft\Windows\CurrentVersion\Run
reg query HKCU\Software\Microsoft\Windows\CurrentVersion\RunOnce
reg query HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run
reg query HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run
wmic useraccount list
wmic useraccount get /ALL
wmic startup list brief
wmic share list
wmic service get name,displayname,pathname,startmode
wmic process list brief
wmic process get caption,executablepath,commandline
wmic qfe get description,installedOn /format:csv
arp -a
whoami
ipconfig /displaydns
route print
netsh advfirewall show allprofiles
systeminfo
qwinsta
quser

Training Question 7

According to x509 certificate events captured by Zeek (formerly Bro), what is the serial number of the TLS certificate assigned to the Windows domain controller in the attack range?

Answer: 55FCEEBB21270D9249E86F4B9DC7AA60

Enter into Splunk window:

index=* | search *cert*
Clicked on certificate.serial

55FCEEBB21270D9249E86F4B9DC7AA60	1,288	47.318%
64B382753C36278241A352307F4351F5	1,029	37.803%
97E0C7A2510B45EF	212	7.788%
7C11E2F05FFF0B994DFF6DD5ECAFFB6F	91	3.343%
1C00145F23036BBCE62F3F2C56000000145F23	26	0.955%
2D000B030BC26D976CDF4092340000000B030B	26	0.955%
02493E07FA9E375A2DBBC61D94430FCF	21	0.771%
0557C80B282683A17B0A114493296B79	13	0.478%
0C073B676F674578F999814852844651	13	0.478%
07DAEF63022EB33897C692D28A6D3BF6	1	0.037%

Challenge question

What is the name of the adversary group that Santa feared would attack KringleCon?

This last one is encrypted using your favorite phrase! The base64 encoded ciphertext is:

7FXjP1lyfKbyDK/MChyf36h7

It's encrypted with an old algorithm that uses a key. We don't care about RFC7465 up here! I leave it to the elves to determine which one!

Hint 1: RFC7465 -> https://tools.ietf.org/html/rfc7465 -> Prohibiting RC4 Cipher Suites

Hint 2: in video mentioned above: "**Adversary Emulation and Automation**" by Dave Herrald

Answer: The Lollipop Guild

Used CyberChef online - https://gchq.github.io/CyberChef
Input: 7FXjP1lyfKbyDK/MChyf36h7
Recipe(s): From Base64, RC4 (passphrase= Stay Frosty)
Output: The Lollipop Guild

Scappy Prepper - Details

The answers to this challenge are too long to list – below is an edited version removing intermediate steps but showing most answers and the final success!

```
 ___ ___ ___ ___ ___ _  _ _____  ___  _   ___ _ _____ _____
| _ \ _ \ __/ __| __| \| |_   _| | _ \/_\ / __| |/ / __|_   _|
|  _/   / _|\__ \ _|| .` | | |   |  _/ _ \ (__| ' <| _|  | |
|_| |_|_\___|___/___|_|\_| |_|   |_|/_/ \_\___|_|\_\___| |_|
              ___
             | _ \_ _ ___ _ __ _ __  ___ _ _
             |  _/ '_/ -_) '_ \ '_ \/ -_) '_|
             |_| |_| \___| .__/ .__/\___|_|
                         |_|  |_|
              (Packets prepared with scapy)
```

Type "yes" to begin. yes

```
Start by running the task.submit() function passing in a string argument of 'start'.
>>> task.submit("start")
Correct! adding a () to a function or class will execute it. Ex - FunctionExecuted()

Submit the class object of the scapy module that sends packets at layer 3 of the OSI
model.
>>> task.submit(send)
Correct! The "send" scapy class will send a crafted scapy packet out of a network
interface.

Submit the class object of the scapy module that sniffs network packets and returns
those packets in a list.
>>> task.submit(scapy.sendrecv.sniff)
Correct! the "sniff" scapy class will sniff network traffic and return these packets
in a list.

Submit the NUMBER only from the choices below that would successfully send a TCP
packet and then return the first sniffed response packet to be stored in a variable
named "pkt":
1. pkt = sr1(IP(dst="127.0.0.1")/TCP(dport=20))
2. pkt = sniff(IP(dst="127.0.0.1")/TCP(dport=20))
3. pkt = sendp(IP(dst="127.0.0.1")/TCP(dport=20))
>>> task.submit(1)
Correct! sr1 will send a packet, then immediately sniff for a response packet.

Submit the class object of the scapy module that can read pcap or pcapng files and
return a list of packets.
>>> task.submit(scapy.utils.rdpcap)
Correct! the "rdpcap" scapy class can read pcap files.

The variable UDP_PACKETS contains a list of UDP packets. Submit the NUMBER only from
the choices below that correctly prints a summary of UDP_PACKETS:
1. UDP_PACKETS.print()
2. UDP_PACKETS.show()
3. UDP_PACKETS.list()
>>> task.submit(2)
Correct! .show() can be used on lists of packets AND on an individual packet.

Submit only the first packet found in UDP_PACKETS.
>>> task.submit(UDP_PACKETS[0])
Correct! Scapy packet lists work just like regular python lists so packets can be
accessed by their position in the list starting at offset 0.

Submit only the entire TCP layer of the second packet in TCP_PACKETS.
>>> task.submit(TCP_PACKETS[1][TCP])
Correct! Most of the major fields like Ether, IP, TCP, UDP, ICMP, DNS, DNSQR, DNSRR,
Raw, etc... can be accessed this way. Ex - pkt[IP][TCP]

Change the source IP address of the first packet found in UDP_PACKETS to 127.0.0.1
and then submit this modified packet
>>> task.submit(IP(src="127.0.0.1")/ICMP()/"test")
Correct! You can change ALL scapy packet attributes using this method.
```

```
Submit the password "task.submit('elf_password')" of the user alabaster as found in
the packet list TCP_PACKETS.
>>> task.submit('echo')
Correct! Here is some really nice list comprehension that will grab all the raw
payloads from tcp packets:
[pkt[Raw].load for pkt in TCP_PACKETS if Raw in pkt]

The ICMP_PACKETS variable contains a packet list of several icmp echo-request and
icmp echo-reply packets. Submit only the ICMP chksum value from the second packet in
the ICMP_PACKETS list.

>>> task.submit(0x4c44)
Correct! You can access the ICMP chksum value from the second packet using
ICMP_PACKETS[1][ICMP].chksum .

Submit the number of the choice below that would correctly create a ICMP echo request
packet with a destination IP of 127.0.0.1 stored in the variable named "pkt"
1. pkt = Ether(src='127.0.0.1')/ICMP(type="echo-request")
2. pkt = IP(src='127.0.0.1')/ICMP(type="echo-reply")
3. pkt = IP(dst='127.0.0.1')/ICMP(type="echo-request")
>>> task.submit(3)
Correct! Once you assign the packet to a variable named "pkt" you can then use that
variable to send or manipulate your created packet.

Create and then submit a UDP packet with a dport of 5000 and a dst IP of
127.127.127.127. (all other packet attributes can be unspecified)
>>> task.submit(Ether()/IP(dst="127.127.127.127")/UDP(dport=5000))
Correct! Your UDP packet creation should look something like this:
pkt = IP(dst="127.127.127.127")/UDP(dport=5000)
task.submit(pkt)

Create and then submit a UDP packet with a dport of 53, a dst IP of 127.2.3.4, and is
a DNS query with a qname of "elveslove.santa". (all other packet attributes can be
unspecified)
>>>
task.submit(Ether()/IP(dst="127.2.3.4")/UDP(dport=53)/DNS(qd=DNSQR(qname="elveslove.s
anta")))
Correct! Your UDP packet creation should look something like this:
pkt = IP(dst="127.2.3.4")/UDP(dport=53)/DNS(rd=1,qd=DNSQR(qname="elveslove.santa"))
task.submit(pkt)
```

The variable ARP_PACKETS contains an ARP request and response packets. The ARP response (the second packet) has 3 incorrect fields in the ARP layer. Correct the second packet in ARP_PACKETS to be a proper ARP response and then task.submit(ARP_PACKETS) for inspection.

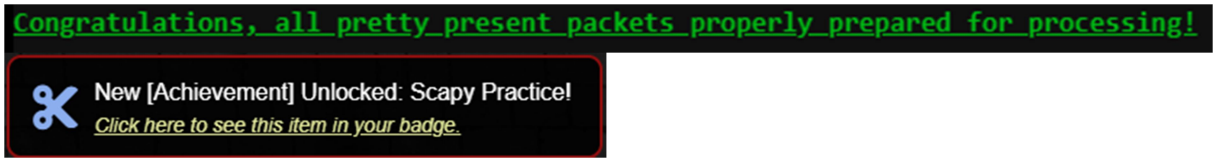
```
>>> ARP_PACKETS[0].show()
###[ Ethernet ]###
  dst       = ff:ff:ff:ff:ff:ff
  src       = 00:16:ce:6e:8b:24
  type      = ARP
###[ ARP ]###
     hwtype    = 0x1
     ptype     = IPv4
```

```
     hwlen     = 6
     plen      = 4
     op        = who-has
     hwsrc     = 00:16:ce:6e:8b:24
     psrc      = 192.168.0.114
     hwdst     = 00:00:00:00:00:00
     pdst      = 192.168.0.1

>>> ARP_PACKETS[1].show()
###[ Ethernet ]###
  dst       = 00:16:ce:6e:8b:24
  src       = 00:13:46:0b:22:ba
  type      = ARP
###[ ARP ]###
     hwtype    = 0x1
     ptype     = IPv4
     hwlen     = 6
     plen      = 4
     op        = None
     hwsrc     = ff:ff:ff:ff:ff:ff
     psrc      = 192.168.0.1
     hwdst     = ff:ff:ff:ff:ff:ff
     pdst      = 192.168.0.114
###[ Padding ]###
        load      = '\xc0\xa8\x00r'

>>> ARP_PACKETS[1].op="2"
>>> ARP_PACKETS[1].hwsrc="00:13:46:0b:22:ba"
>>> ARP_PACKETS[1].hwdst="00:16:ce:6e:8b:24"
>>> ARP_PACKETS[1].show()
###[ Ethernet ]###
  dst       = 00:16:ce:6e:8b:24
  src       = 00:13:46:0b:22:ba
  type      = ARP
###[ ARP ]###
     hwtype    = 0x1
     ptype     = IPv4
     hwlen     = 6
     plen      = 4
     op        = is-at
     hwsrc     = 00:13:46:0b:22:ba
     psrc      = 192.168.0.1
     hwdst     = 00:16:ce:6e:8b:24
     pdst      = 192.168.0.114
###[ Padding ]###
        load      = '\xc0\xa8\x00r'
>>> task.submit(ARP_PACKETS)

Great, you prepared all the present packets!
```



Notes Taken Along the Way

Modem Sounds:

https://www.youtube.com/watch?v=Yn2MVeu0XdY

Splunk Fundamentals

https://www.splunk.com/en_us/training/free-courses/splunk-fundamentals-1.html

DNS and scapy

1. https://www.cs.dartmouth.edu/~sergey/netreads/local/reliable-dns-spoofing-with-python-scapy-nfqueue.html
2. https://www2.cs.duke.edu/courses/fall16/compsci356/DNS/DNS-primer.pdf

ARP and Scapy

https://medium.com/datadriveninvestor/arp-cache-poisoning-using-scapy-d6711ecbe112

Chrome Dev Tools

https://developers.google.com/web/tools/chrome-devtools/javascript

MT19937 Mersenne Twister

1. https://github.com/tliston/mt19937/blob/main/mt19937.py (from talk -Tom Liston, Random Facts About Mersenne Twisters | KringleCon 2020)
2. https://github.com/kmyk/mersenne-twister-predictor (from elf Tangle Coalbox after Snowball game on impossible)

VScode Hex Editor

Just look for Microsoft Extension in vscode.

11b Summary of ToDos

In total you have three tasks:

1. Change the PDF that the naughty statements are visible
2. Make changes to show that Jack is considered "naughty" in the blockchain
3. Make further changes to account for the above two changes so that the MD5 hash of the block does not change

8 LFI

https://highon.coffee/blog/lfi-cheat-sheet/

Steps:

1. Figure out how to download your uploads
2. Figure out where those uploads go (locally)
3. Go somewhere else (locally) to download app.rb

4. Figure out what the app.rb says
5. Figure out what else you can download (and how)

9 Reverse Shell

https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Reverse%20Shell%20Cheatsheet.md