# Objective 5: Strange USB Device

Our next quest requires us to determine what an evil USB flash drive is doing. The strange USB device can be found with Marcel Nougat in the Speaker Unpreparedness Room. Before we can do the objective, we need to get hints from Jewel Loggins on the talks floor.

A screenshot of a computer

Description automatically generated with medium confidenceGraphical user interface

Description automatically generated

## Terminal: IPv6 Sandbox

Before you can get Jewel’s hints for the USB objective, you’ll need help Jewel with the IPv6 Sandbox. IP version 6 was developed to remedy the exhaustion of the IP version 4 address space. There are 232 possible addresses in IPv4, but IPv6 has 2128. Large IPv4 address assignments for the US and Europe, along with the use of Network Address Translation (NAT) have allowed us to largely ignore IPv6 so far. However, Windows and Linux operating systems both enable IPv6 on the local network even when it is not routed to the Internet. A good InfoSec professional should have at least a basic knowledge of IPv6.

The link that Jewel gives you to get started can make this challenge much easier for you, so please read it carefully.

|  |  |
| --- | --- |
| A screenshot of a video game  Description automatically generated with medium confidence | Text  Description automatically generated |
| Graphical user interface, table  Description automatically generated  <https://gist.github.com/chriselgee/c1c69756e527f649d0a95b6f20337c2f> |

## Assignment

If you have no knowledge of IPv6 addressing tutorial may be helpful. <http://www.steves-internet-guide.com/ipv6-guide/>. Just as IPv4 has public (routed to the Internet) and private (internal, not routed to the Internet) IPv6 has global (like IPv4 public) and two flavors like IPv4 private addresses, Link Local and Unique Local. You may see all three types in this terminal, but we will only be concerned with Link Local addresses, which all start with fe80.

A major difference between IPv4 and IPv6 is how hosts that are on the same network locate each other. An IPv4 host sends ARP broadcasts asking, “Who has IP address 192.168.1.1?” An IPv6 host uses multicast packets to find routers and other hosts.

### Step 1 question: Who’s there?

Use the three commands in the bottom part of Chris Elgee’s Gist to find any other hosts on the same network as the terminal host. Jewel has given you the link to the Gist, but here it is again: <https://gist.github.com/chriselgee/c1c69756e527f649d0a95b6f20337c2f> . You should be able to locate two hosts; what are their IPv6 addresses?

### Step 1 answer

Here is the result of using Chris’ commands.

Graphical user interface

Description automatically generated with medium confidence

The first two commands were IPv6 ping commands. Both addresses are multicast addresses. The first is fe80::1 which all nodes (hosts) monitor, and the second fe80::2 is monitored by routers. <https://menandmice.com/blog/ipv6-reference-multicast> . So, we basically asked our network, “Who’s there?” The answers are used to fill our Network Discovery cache. Here is the first in the cache.  
Our router answered using fe80::1 (router multicast). The response was received on our eth0 interface, it is a link local address, and the MAC address of the router is 02:42:80:6b:38:e5.

Here is the second entry.  
  
A host with the address fe80::42:coff:fea8:a002 answered us on our eth0. This host’s MAC address is 02:42:c0:a8:a0:02. If you look carefully you can see most of the MAC address in the IPv6 address, with ff:fe added to the middle of the MAC.

Jewel said she had forgotten the IP address of the other host; it is probably the second entry.

### Step 2 question: What ports are open on the host? What is the password?

Play with the commands in the top half of the Gist to see what you can learn about the other host. Note that some of the commands require you to specify your network interface at the end of the command. What is the password Jewel needs?

### Step 2 answer

Nmap tells us that the host is listening on ports 80 and 9000.

Text

Description automatically generated

We can use cURL to connect to the http port.  
Text

Description automatically generated

That’s handy, we need to connect to port 9000 to get the answer. For fun we will use both wget and netcat (nc).

Text

Description automatically generated

It turns out that wget (and telnet) are not installed on the terminal, but cURL and nc both work and give us the activation phrase that Jewel needs. Enter it into the top of the terminal to claim your credit.



## Hints from the IPv6 Sandbox

Jewel has several interesting hints for us.

|  |  |
| --- | --- |
| A picture containing table  Description automatically generated  <https://attack.mitre.org/techniques/T1098/004/> | A picture containing table  Description automatically generated  <https://docs.hak5.org/hc/en-us/articles/360010471234-Writing-your-first-USB-Rubber-Ducky-Payload> |
| Table  Description automatically generated <https://docs.hak5.org/hc/en-us/articles/360010555153-Ducky-Script-the-USB-Rubber-Ducky-language> | A green screen with white text  Description automatically generated with medium confidence  <https://github.com/dagonis/Mallard> |

## Objective 5: Strange USB Device

A [USB Rubber Ducky](https://shop.hak5.org/products/usb-rubber-ducky-deluxe) looks like a USB flash drive, but it impersonates a keyboard and enters preprogrammed keystrokes as soon as it is connected to a computer. This could be devastating if a user finds one left by an attacker and plugs it into their computer. The idea of a USB device impersonating a keyboard is not entirely evil, however; a YubiKey Multifactor Authentication (MFA) device uses the same technique so that its user does not have to type in a one-time password.

Marcel Nougat is standing beside the Strange USB Device terminal in the Speaker Un\_Preparedness Room, and he has found a Rubber Ducky that has been programmed with evil intent.

This talk by Kevin Tyers will help you with this challenge.  
Text

Description automatically generated with low confidence  
<https://www.youtube.com/watch?v=tkAYncGF-zw>

|  |  |
| --- | --- |
| Graphical user interface  Description automatically generated | Text  Description automatically generated |

## Assignment

Read the link you received from Jewel in the Mitre ATT&CKtm hint. Mitre has developed a table of common attacker techniques, and it is a powerful tool that helps defenders learn about attacks that could be used against them. The attack the link describes is one where the attacker inserts their own key into a computer’s list of keys that it will allow to log in using SSH. If you have not used SSH before, it is a powerful tool that administrators use to connect to remote systems. This tutorial gives some background knowledge. <https://www.digitalocean.com/community/tutorials/ssh-essentials-working-with-ssh-servers-clients-and-keys> This link gives you the format of the authorized\_keys file. <https://www.ibm.com/docs/en/zos/2.2.0?topic=daemon-format-authorized-keys-file>

Jewel’s other three hints describe the language used to program a Rubber Ducky, how to write and encode your own Ducky script, and how to Reverse Engineer (RE) an existing Ducky script. The last hint is the most helpful; Mallard is even available on the terminal!

Examine Marcel’s Strange USB Device and discover the username of the troll who created it.

### Step 1 question

Use Mallard to decode the contents of the USB binary.

### Step 1 answer

You can run mallard.py with either ./mallard.py or python3 mallard.py, and use the help to see the correct syntax. Enter the name of the file to decode as  
./mallard.py –file /mnt/USBDEVICE/inject.bin.

Text

Description automatically generated

### Step 2 question

The inject.bin file you decoded is a bash script. The first three quarters of it is an interesting attempt to steal the user’s password by using sudo, to save the password in a file called /home/elf/.config/sudo, and to exfiltrate it to trollfun.jackfrosttower.com on port 1337. We are asked to find the attacker’s username, and not the URL of their server, so this is no help.

Try to decode the long, encoded string at the end of the script. Maybe the answer you need is in there. Remember that the Mitre ATT&CKtm hint talks about the SSH authorized\_keys file.

Note: it is difficult to copy and paste *outside* of the terminal, but you can copy and paste *within* the terminal, and the terminal has the tools you need.

What is the attacker’s username?

### Step 2 answer

The encoded string is a base64 string, sort of. It has two ‘=’s at the beginning instead of the end of the string though. If you treat the string as a glob, you can see that the command in the entire line is:  
echo <base64 glob> | rev | base64 -d | bash.

The code uses rev to reverse the string so the ‘=’s are at the end where they belong. Then it decodes the base64 and sends it to bash to be executed. So, we can use the same command, except remove bash from the end so we do not execute it.

A screenshot of a computer

Description automatically generated with medium confidence

The blob decodes to a command to copy a public key (the new base64 blob) to the user’s ssh/authorized\_keys file.

Text

Description automatically generated

Now, whenever someone with a username of ickymcgoop from trollfun.jackfrosttower.com presents this public key, they will be allowed to login via SSH.

Text

Description automatically generated