

SSHliders - Answers

Hello challenge fans. Sorry for the long delay, but better late than never, right? Actually this one caused a little debate, because we did not have anyone that gave a completely accurate answer on either the technical or creative sides. But in considering that these challenges are not just contests but also great ways to learn, we decided to release the answers without any winners. So although there are no signed copies of Ed Skoudis' book, Counter Hack Reloaded, a couple of you still get your name in lights as we mention some of your good thoughts. We'll just have to keep in mind the immortal words of Mike McDermott in Rounders when replying to one of the participants in the judges poker game that Professor Petrovsky is not paying him. Mike kindly replies, "Oh, well, knowledge is my reward, sir." So without further delay, here's Mr. Shewmaker with the answers to SSHliders.

del.icio.us

Discuss in Forums {mos_smf_discuss:October 2009 - SSHliders}

SANS 2010 Orlando March 6 - 15

SSHliders

Answers

By James Shewmaker

<http://www.bluenotch.com/>

We did not have any complete winners, so to keep things short, I selected interesting parts from the top submissions. The emphasis of the challenge was to highlight the subtle differences in BSD vs.

Linux/Gnu distributions and to have a little fun weaponizing commands. Let's get to it:

1) What assumptions does the timer device make if the basketball returns as described?

The first portion of the challenge was to bootstrap your hacking thought process—you can't patch bad assumptions, and good assumptions can expose workflow flaws. Since we are talking about alternate realities, this necessary step also keeps our imagination from running completely wild.

Alexis Michon established some basic assumptions for the TCP protocol to work: "The ports 1111 and 2222 must be free, no daemons should be bind on these ports."

Stacy Olivas described the assumptions on how the command and components are assumed to behave (based on behavior and man pages):

"The assumption is that the timer device will only start when something is sent through the portal and not close when what is being sent through is only partially sent. Also it assumes that the portal will open for 100 seconds the first instance something attempts to go through it, closing after 100 seconds regardless of what has been sent."

2) If Mr. Brown's Caddie travels over TCP and was coming from an arbitrary port on the host to another destination such as a stadium hosted at 10.10.10.12 port 80, what changes to the vortex and environment did Quinn make that would collect any passerby and send them through the vortex as well?

Stacy took advantage of the fact we didn't specify how organized the results came out of the vortex by replacing the "echo "basketball"" with `cat *` (which would lump all the contents of every file in the current directory into the vortex and back out, as one stream of data). This is precisely why we need to think about assumptions and how to leverage them to our advantage. But if you wanted a more practical version of Stacy's idea, you could send either with the tar command to output to STDOUT instead of a file (with a dash) or the more appropriate cpio command (and reverse it to pull it out):

```
tar -f - . > .vortex|nc 127.0.0.1 1111
```

or here's an example of combining the vortex and the sliding device:

```
find . -print | cpio -oaV | ssh -i id.dsa -p 1111 127.0.0.1 'cpio -imVd'
```

Alexis had a more powerful solution and moved the problem from Transport to the Network. The newer "w" command-line option of openssh (another assumption that we have that available...). The "w

option uses a tunnel device to tunnel IP traffic over SSH, not just a TCP tunnel.

I was surprised not a single submission included the `openssh -D` command line option, which creates a dynamic tunnel—effectively a SOCKS compatible proxy that listens locally on a TCP port. This sends all tunneled traffic out a dynamic port on the other end (excellent for quick and dirty proxy for web-browsers and instant messaging clients).

3) How can `timer.sh` be fixed to be more reliable?

Stacy adds in a sleep delay before the HUP is sent to SSHD ensuring that the sliding message is sent to all recipients:

```
echo -e "echo sliding...|wall;sleep 15;killall -HUP sshd" | at "now +$x seconds"
```

This works if you want to keep the HUP signal. The better solution is to remove it completely:

```
echo -e "echo sliding...|wall;" | at "now +$x seconds"
```

So in Stacy's interpretation of this alternate world, merely saying you are returning is sliding. If I was sliding myself, I would change the `at` command to minutes since no world I've been on schedules via `at` reliably in seconds (I was surprised the timer even accepted scheduling via seconds).

4) How can the sliders bring the timer with them on every slide automatically?

Stacy modifies the timer to allow it to be tossed into the vortex first (but any of the above transfer techniques could be made to be automatic as well):

```
#!/usr/local/bin/bash
let x=`echo -e "$RANDOM/60"|bc`
export j=`date "-v+$x"S "+%s"`
echo -e "echo sliding...|wall;cat timer.sh | nc 127.0.0.1 1111; \
nc -l -l 1111 > timer.sh" | at "now +$x seconds"
```

```
export PS1=`date -j -f '%s' $(echo -e "$j-$(date +%s)"|bc) "+%M:%S">` `
```

5) How can the timer be modified to show the correct countdown on either the original or this gnu world?

The real trick to converting the timer to work on GNU distribution would be identify the differences in command-line options of the date command. They are ususally different—but varies on each distribution so YMMV (your mileage may vary).

6) How could Quinn modify the vortex to send multiple people and objects in parallel (at the same time) to the alternate Earth?

The `-w` and `-D` options for openssh could solve this task as well when combined with either tar or cpio.

7) One of the sliders' enemies is disrupting their vortex by changing the timer. How should they enhance the timer to ensure it hasn't been tampered with?

Stacy decided to put an md5 hash verification function in the timer script for a self-check. The function would compute its own md5 hash and compare it with an embedded pre-computed string to make sure it matches. This could detect a simple change, but if we combine the verification (hash) with the data, we can still have both tampered with. The only 'practical' way would be similar to what packers do for executables: sign the timer as Stacy described but either stage the verification or obfuscate to thwart tampering. Not perfect in any way, but manageable if it is important to you.

8) How can Quinn record the coordinates (host and port) of each world as he slides, so he can arbitrarily return to any of the previous worlds?

I really liked Stacy's solution for this: Quinn can do something like the following to identify the TCP port he arrived on (combine this with the 'bring stuff with you' concepts above):

```
sockstat | awk -F: '/Quinn/ { getline; print $0}' | grep ssh > history.txt
```

Since a full terminal is used with ssh, we could use some of the user commands to identify the TTY and distinguish slides separate from others in case there is multiple simultaneous slides from another party.

So kudos to Stacy Olivas for having the 'most' technically correct submission & Alexis Michon for some good ideas. And the Random Winner is… localhost!

There are other features of ssh and shell shenanigans—including –t for TTY tricks so please experiment on your own. There is a great doc from ace1 with examples from Netwars about using ssh tunnels, proxies (proxychains FTW!) at <http://google.com/group/netwarsplayers/files/>.

This brings us to the end of yet another challenge; I hope you found something you could use in your world. If you want to experiment or learn more of these sorts of things, you may want to play a round or two of Netwars (see www.netwars.info for more information).

James Shewmaker has over 15 years experience in IT, primarily developing appliances for automation and security for broadcast radio, internet, and satellite devices. He is one of the first GIAC Platinum certified Malware (GSM) experts. James is a founder and active consultant for Bluenotch Corporation which focuses on investigations, penetration testing, and analysis. He has contributed to the courseware in various SANS courses including Security Essentials and Reverse Engineering Malware: Advanced Techniques. His focus in 2009 has been constructing and running the NetWars Capture the Flag Competition, part of the US Cyber Challenge.