HASSH
SSH Client/Server Profiling

Ben Reardon and Adel Karimi
Detection Cloud
Security Operations

March 16 2019, BSides Canberra

About us

Ben Reardon
breardon@salesforce.com
@benreardon

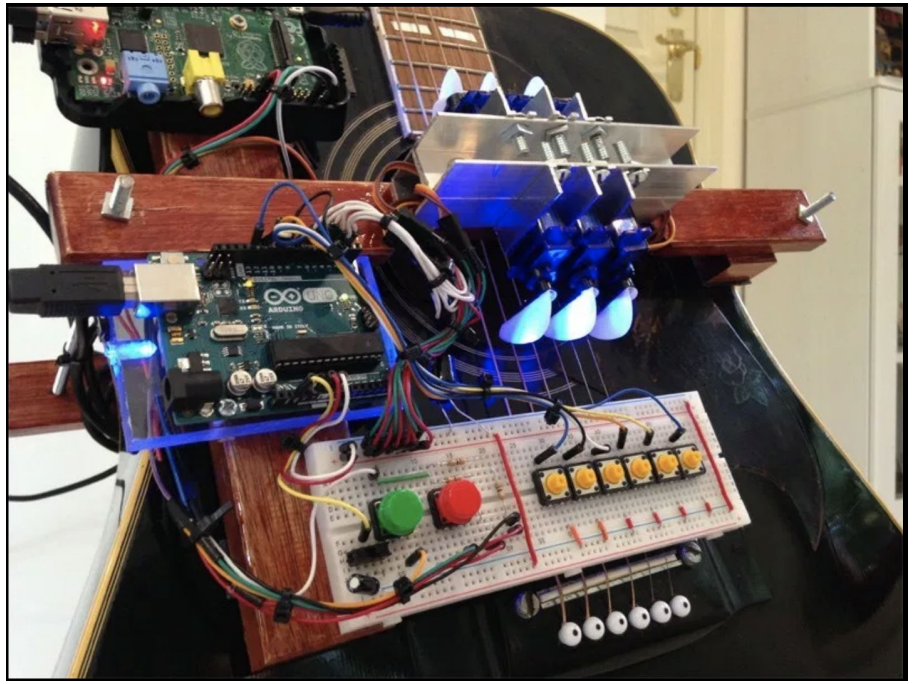
Engineer

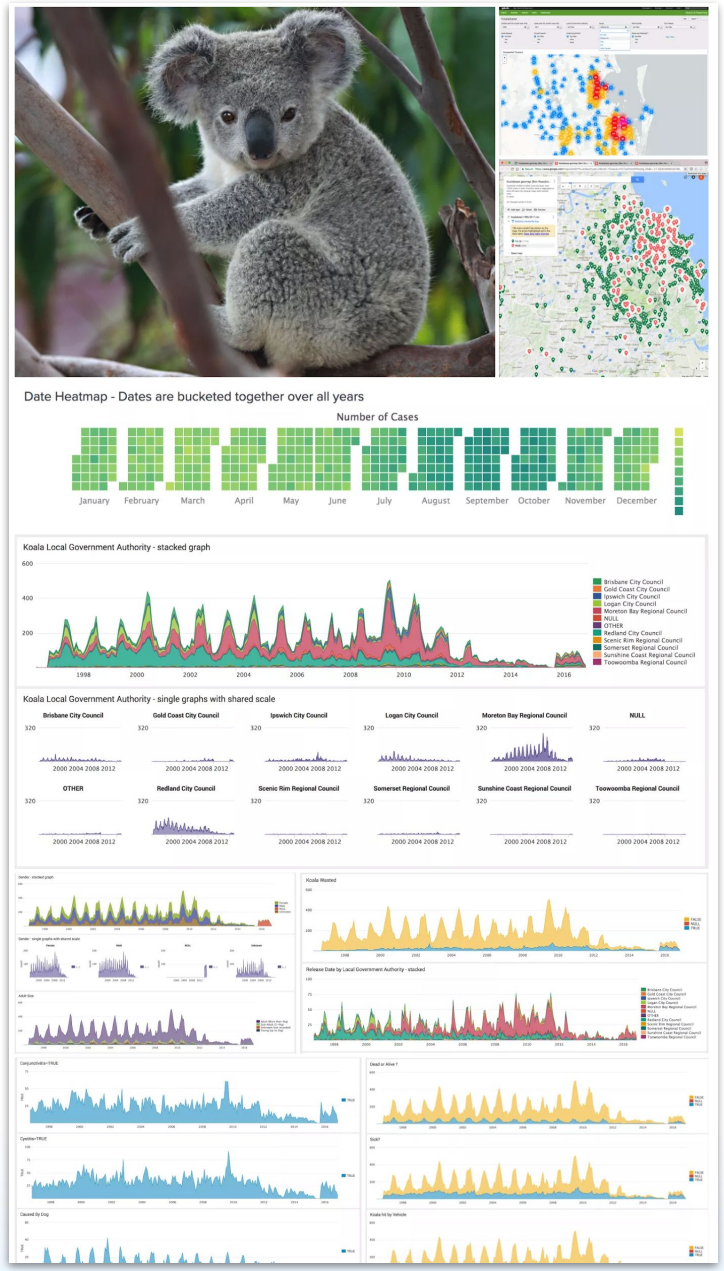
Detection Team

BrisVegas

Data Visualization, ML

Robotics / Electronics





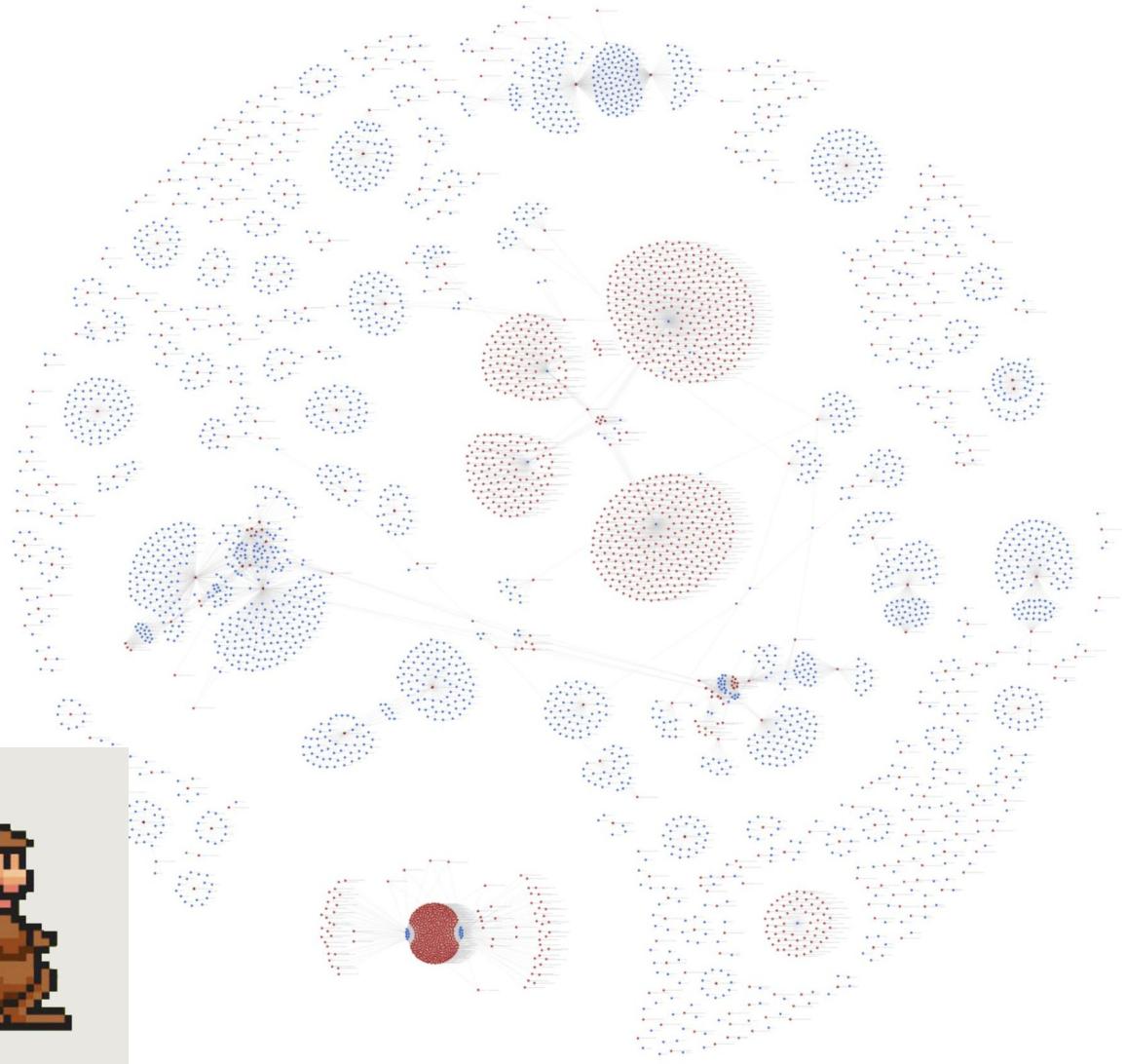
https://dataviz.com.au/
Robot Guitar

https://dataviz.com.au/ KoalaBase

About us

Adel “0x4D31” Karimi

- Lead Detection Engineer
- Honeynet Project
- Developer of honeyλ, and a couple more open-source projects
- github.com/0x4D31



Summary

What is it ?

How it works

What problems it helps solve

Show some use cases

Supporting tools that we've open sourced

Standing on the shoulders of giants

Move the needle, you should

Credit: EVR !

<next>

bro-symon

bro-OSQUERY

HASSH

JA3

bro-TLS client fingerprinting

FingerprinTLS

….

nmap -A

p0f

“Silence on the Wire”: Chapter 9

Footprints, in the dawn of time..

What is HASSH

Encrypted protocols need to negotiate many parameters

This negotiation is necessarily done in clear text

Certain Compositions of these parameters tend to be app unique

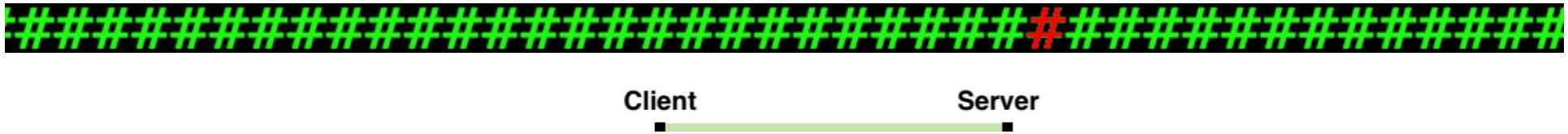
Build this composition to fingerprint client and server apps

HASSH does this with SSH

FingerprinTLS, JA3 do this with TLS

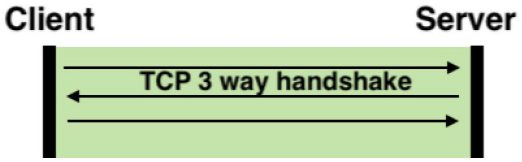
https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c

HASSH - how it works



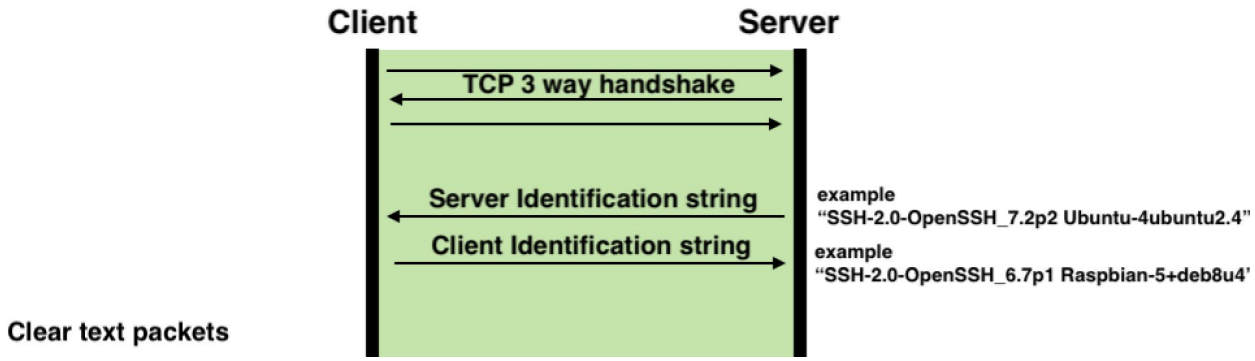
https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c

HASSH - how it works



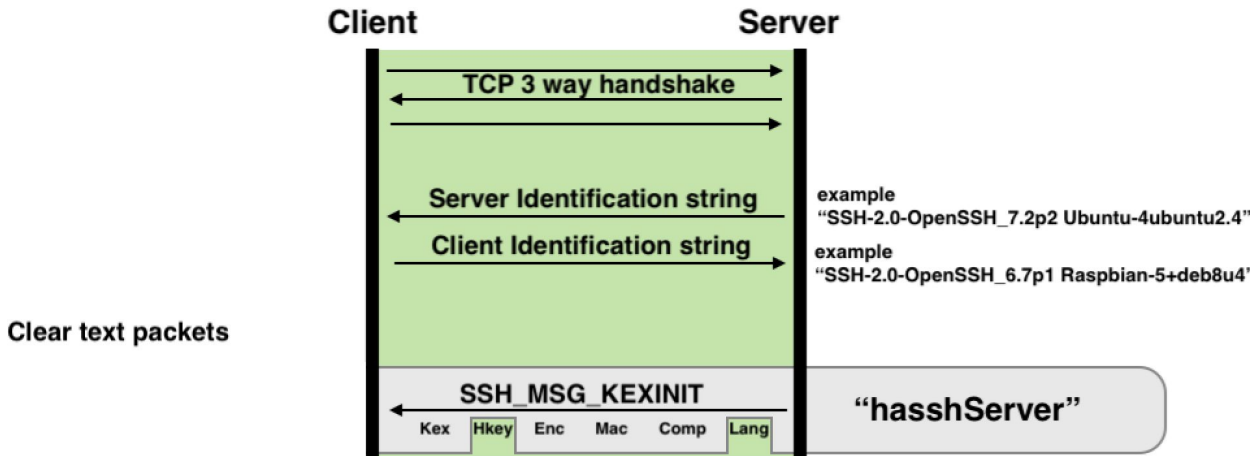
https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c

HASSH - how it works



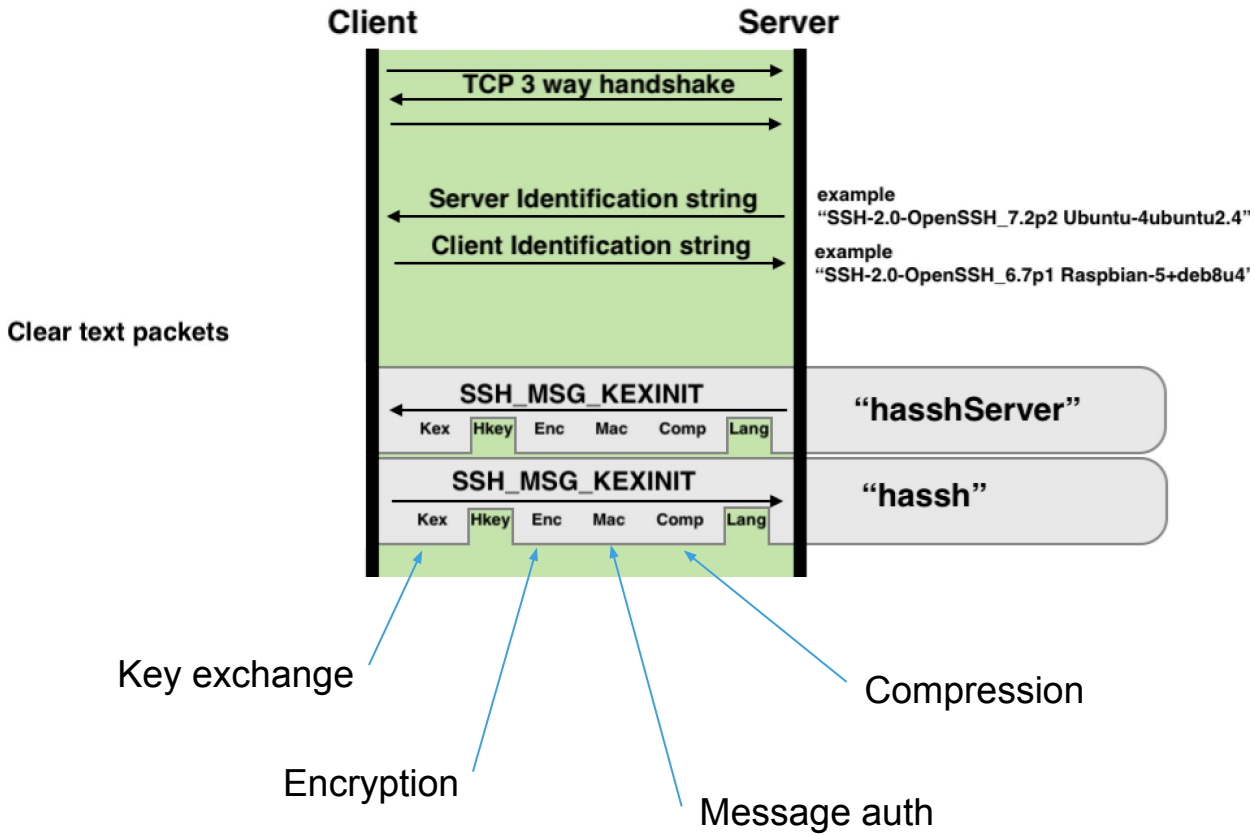
https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c

HASSH - how it works



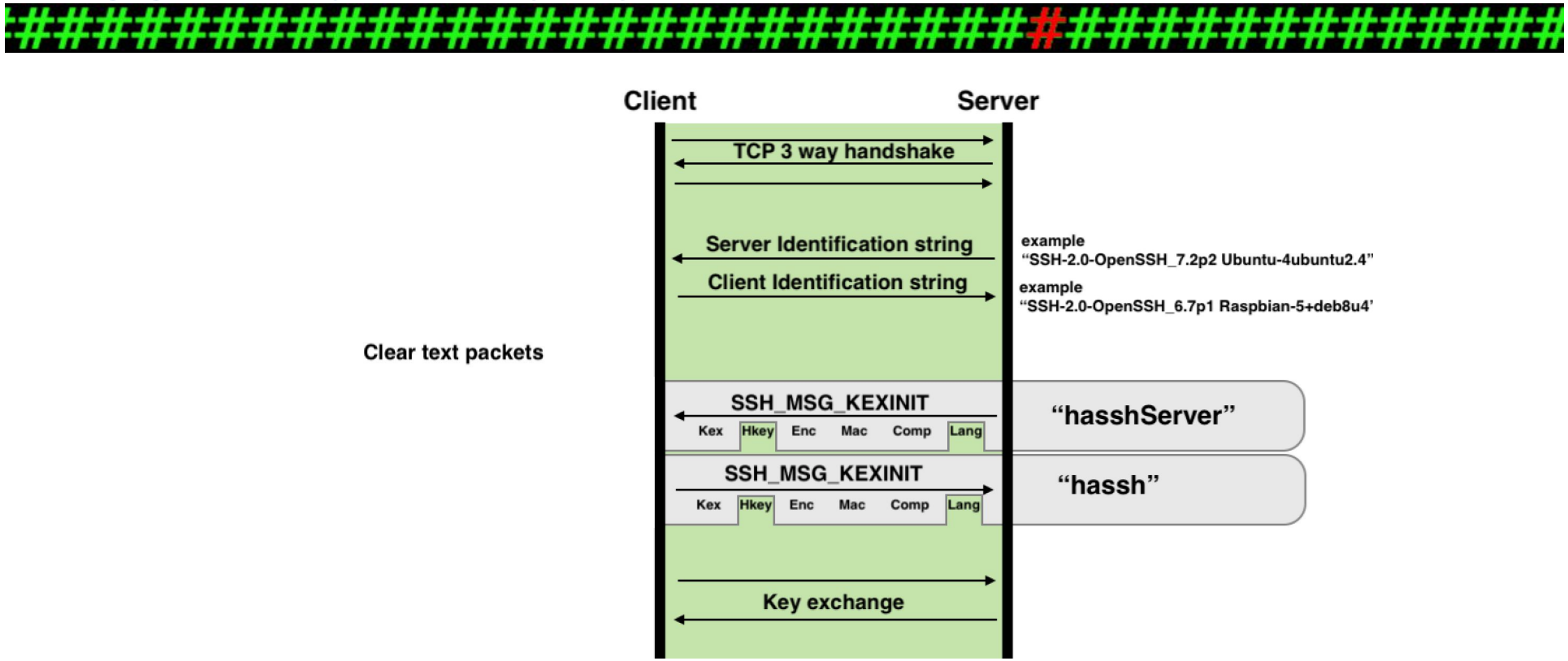
https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c

HASSH - how it works



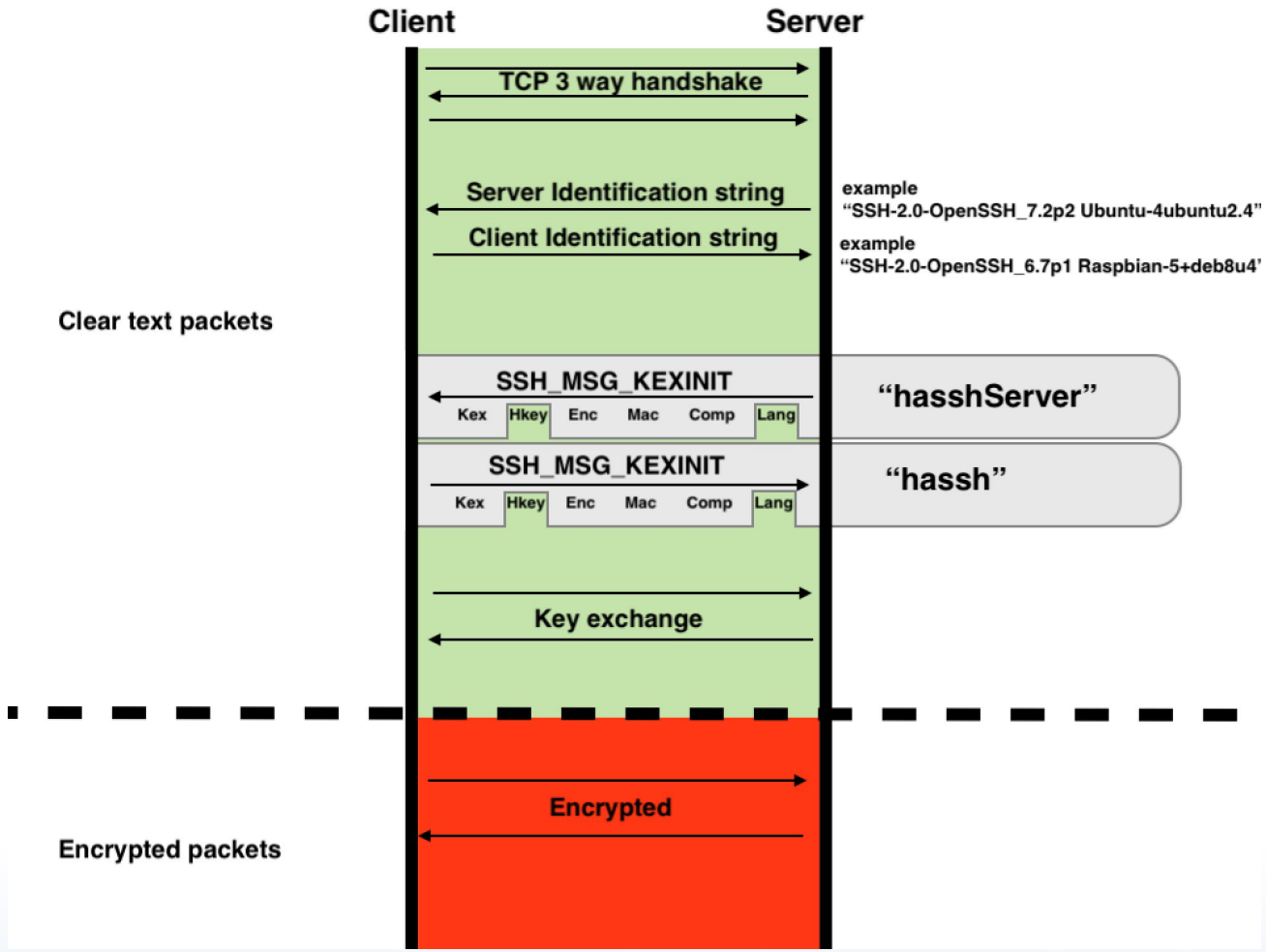
https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c

HASSH - how it works



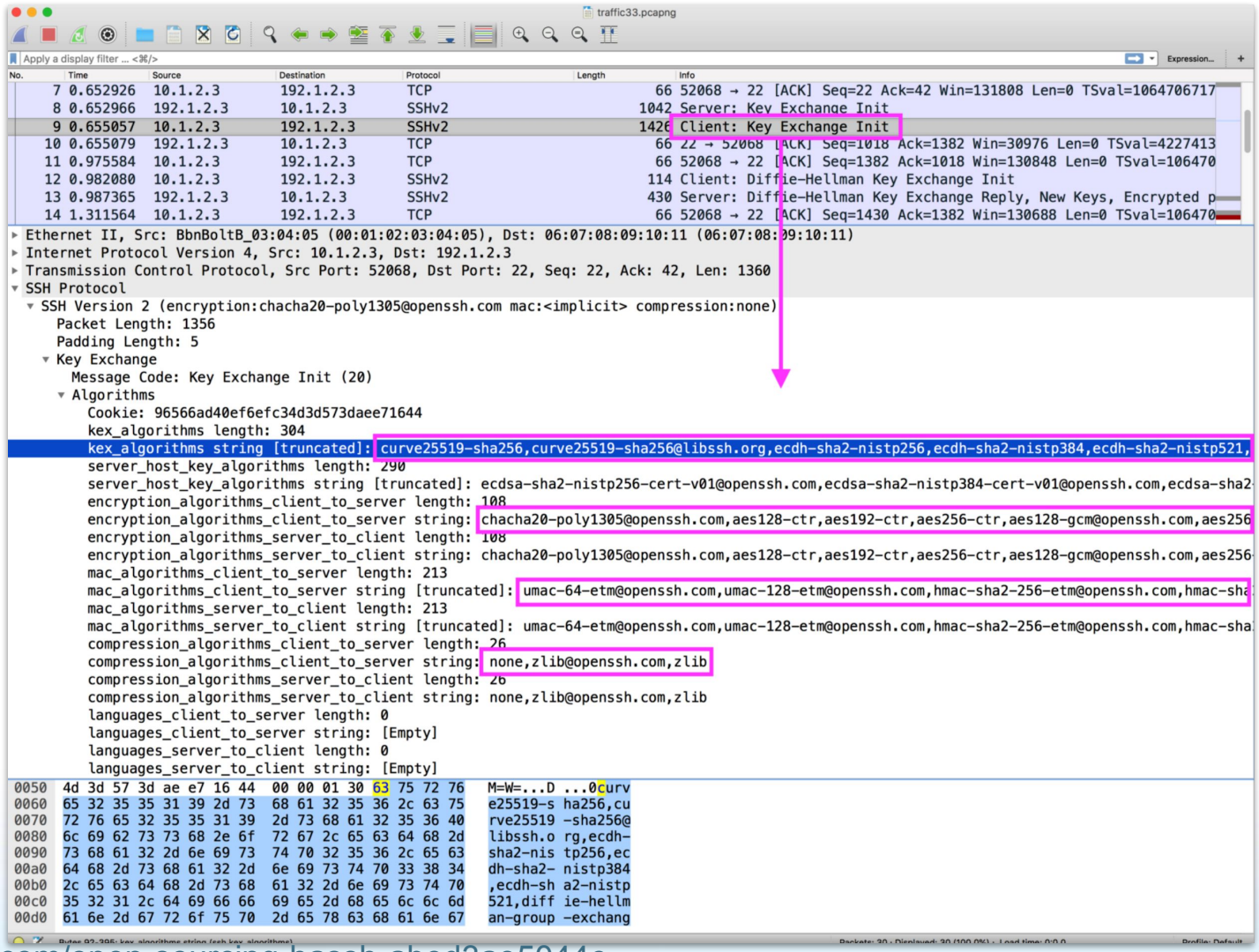
https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c

HASSH - how it works



https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c

HASSH - how it works



Key exchange algos

Encryption algos

Message auth algos

Compression algos

https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c

HASSH - how it works

Key Exchange methods

curve25519-sha256@libssh.org,diffie-hellman-group-exchange-sha256,ecdh-sha2-nistp521,ecdh-sha2-nistp384,ecdh-sha2-nistp256,diffie-hellman-group-exchange-sha1,diffie-hellman-group1-sha1,diffie-hellman-group14-sha1,diffie-hellman-group14-sha256,diffie-hellman-group15-sha512,diffie-hellman-group16-sha512,diffie-hellman-group17-sha512,diffie-hellman-group18-sha512,diffie-hellman-group14-sha256@ssh.com,diffie-hellman-group15-sha256,diffie-hellman-group15-sha256@ssh.com,diffie-hellman-group15-sha384@ssh.com,diffie-hellman-group16-sha256,diffie-hellman-group16-sha384@ssh.com,diffie-hellman-group16-sha512@ssh.com,diffie-hellman-group18-sha512@ssh.com

Encryption

aes128-cbc,aes128-ctr,aes192-cbc,aes192-ctr,aes256-cbc,aes256-ctr,blowfish-cbc,blowfish-ctr,cast128-cbc,cast128-ctr,idea-cbc,idea-ctr,serpent128-cbc,serpent128-ctr,serpent192-cbc,serpent192-ctr,serpent256-cbc,serpent256-ctr,3des-cbc,3des-ctr,twofish128-cbc,twofish128-ctr,twofish192-cbc,twofish192-ctr,twofish256-cbc,twofish256-ctr,twofish-cbc,arcfour,arcfour128,arcfour256

Message Auth

hmac-sha1,hmac-sha1-96,hmac-md5,hmac-md5-96,hmac-sha2-256,hmac-sha2-512

Compression zlib@openssh.com,zlib,none

CyberDuck Version 6.7.1

hasshAlgorithms (all the things, concatenated with a ; delim)

curve25519-sha256@libssh.org,diffie-hellman-group-exchange-sha256,ecdh-sha2-nistp521,ecdh-sha2-nistp384,ecdh-sha2-nistp256,diffie-hellman-group-exchange-sha1,diffie-hellman-group1-sha1,diffie-hellman-group14-sha1,diffie-hellman-group14-sha256,diffie-hellman-group15-sha512,diffie-hellman-group16-sha512,diffie-hellman-group17-sha512,diffie-hellman-group18-sha512,diffie-hellman-group14-sha256@ssh.com,diffie-hellman-group15-sha256,diffie-hellman-group15-sha256@ssh.com,diffie-hellman-group15-sha384@ssh.com,diffie-hellman-group16-sha256,diffie-hellman-group16-sha384@ssh.com,diffie-hellman-group16-sha512@ssh.com,diffie-hellman-group18-sha512@ssh.com;aes128-cbc,aes128-ctr,aes192-cbc,aes192-ctr,aes256-cbc,aes256-ctr,blowfish-cbc,blowfish-ctr,cast128-cbc,cast128-ctr,idea-cbc,idea-ctr,serpent128-cbc,serpent128-ctr,serpent192-cbc,serpent192-ctr,serpent256-cbc,serpent256-ctr,3des-cbc,3des-ctr,twofish128-cbc,twofish128-ctr,twofish192-cbc,twofish192-ctr,twofish256-cbc,twofish256-ctr,twofish-cbc,arcfour,arcfour128,arcfour256;hmac-sha1,hmac-sha1-96,hmac-md5,hmac-md5-96,hmac-sha2-256,hmac-sha2-512;zlib@openssh.com,zlib,none

hassh = 8a8ae540028bf433cd68356c1b9e8d5b

HASSH - how it works

Key Exchange methods

curve25519-sha256@libssh.org,ecdh-sha2-nistp256,ecdh-sha2-nistp384,ecdh-sha2-nistp521,diffie-hellman-group-exchange-sha256,diffie-hellman-group-exchange-sha1,diffie-hellman-group14-sha1

Encryption

chacha20-poly1305@openssh.com,aes128-ctr,aes192-ctr,aes256-ctr,aes128-gcm@openssh.com,aes256-gcm@openssh.com,arcfour256,arcfour128,aes128-cbc,3des-cbc,blowfish-cbc,cast128-cbc,aes192-cbc,aes256-cbc,arcfour,rijndael-cbc@lysator.liu.se

Message Auth

umac-64-etm@openssh.com,umac-128-etm@openssh.com,hmac-sha2-256-etm@openssh.com,hmac-sha2-512-etm@openssh.com,hmac-sha1-etm@openssh.com,umac-64@openssh.com,umac-128@openssh.com,hmac-sha2-256,hmac-sha2-512,hmac-sha1,hmac-md5-etm@openssh.com,hmac-ripemd160-etm@openssh.com,hmac-sha1-96-etm@openssh.com,hmac-md5-96-etm@openssh.com,hmac-md5,hmac-ripemd160,hmac-ripemd160@openssh.com,hmac-sha1-96,hmac-md5-96

Compression none,zlib@openssh.com,zlib

hasshAlgorithms

curve25519-sha256@libssh.org,ecdh-sha2-nistp256,ecdh-sha2-nistp384,ecdh-sha2-nistp521,diffie-hellman-group-exchange-sha256,diffie-hellman-group-exchange-sha1,diffie-hellman-group14-sha1;chacha20-poly1305@openssh.com,aes128-ctr,aes192-ctr,aes256-ctr,aes128-gcm@openssh.com,aes256-gcm@openssh.com,arcfour256,arcfour128,aes128-cbc,3des-cbc,blowfish-cbc,cast128-cbc,aes192-cbc,aes256-cbc,arcfour,rijndael-cbc@lysator.liu.se;umac-64-etm@openssh.com,umac-128-etm@openssh.com,hmac-sha2-256-etm@openssh.com,hmac-sha2-512-etm@openssh.com,hmac-sha1-etm@openssh.com,umac-64@openssh.com,umac-128@openssh.com,hmac-sha2-256,hmac-sha2-512,hmac-sha1,hmac-md5-etm@openssh.com,hmac-ripemd160-etm@openssh.com,hmac-sha1-96-etm@openssh.com,hmac-md5-96-etm@openssh.com,hmac-md5,hmac-ripemd160,hmac-ripemd160@openssh.com,hmac-sha1-96,hmac-md5-96;none,zlib@openssh.com,zlib

hassh = 55a77ae9728654f1d4240a29287dc296

HASSH - how it works

ncrack

ncrack poses as OpenSSH 7.1.., but

`hassh(ncrack)=55a77ae9728654f1d4240a29287dc296`

```
152
153 #define SSH_TIMEOUT 20000
154 #define CLIENT_VERSION "SSH-2.0-OpenSSH_7.1\n"
155
156
157 extern NcrackOps o;
158
159 extern void ncrack_read_handler(nsock_pool nsp, nsock_event nse, void *mydata);
160 extern void ncrack_write_handler(nsock_pool nsp, nsock_event nse, void *mydata);
161 extern void ncrack_module_end(nsock_pool nsp, void *mydata);
162
163 enum states { SSH_INIT, SSH_ID_EX, SSH_KEY, SSH_KEY2, SSH_KEY3, SSH_KEY4,
164   SSH_AUTH, SSH_AUTH2, SSH_AUTH3, SSH_AUTH4, SSH_FINI };
165
166 static void ssh_free(Connection *con);
167
168
169
170 static inline int
171 ssh_loop_read(nsock_pool nsp, Connection *con, ncrack_ssh_state *info)
```

HASSH - Notes during Development

MD5 vs SHA256

	MD5	**SHA-256**
Collisions?	Not for our use case	No
Short	Yes, 32 chars	No, 64 chars
Tweetable	Yes	Not friendly at all
Swamps surrounding text	No	Yes
Supported	By everything	Probably by everything
Provides Nett Benefit	Yes	No

https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c

HASSH - Notes during Development

Language Field

compression_algorithms
A name-list of acceptable compression algorithms in order of preference. The chosen compression algorithm MUST be the first algorithm on the client's name-list that is also on the server's name-list. If there is no such algorithm, both sides MUST disconnect.

Note that "none" must be explicitly listed if it is to be acceptable. The compression algorithm names are listed in Section 6.2.

languages
This is a name-list of language tags in order of preference [RFC3066]. Both parties MAY ignore this name-list. If there are no language preferences, this name-list SHOULD be empty as defined in Section 5 of [SSH-ARCH]. Language tags SHOULD NOT be present unless they are known to be needed by the sending party.

first_kex_packet_follows
Indicates whether a guessed key exchange packet follows. If a

https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c

HASSH - how it works

hassh examples

8a8ae540028bf433cd68356c1b9e8d5b **CyberDuck** Version 6.7.1 (28683)

b5752e36ba6c5979a575e43178908adf SSH-2.0-**paramiko**_2.4.1 (E.g. used by Metasploit exploit kit modules)

16f898dd8ed8279e1055350b4e20666c SSH-2.0-**dropbear**_2012.55 (popular embedded/IOT library)

06046964c022c6407d15a27b12a6a4fb SSH-2.0-**OpenSSH_7.6**

de30354b88bae4c2810426614e1b6976 SSH-2.0-**Renci**.SshNet.SshClient.0.0.1 (Powershell, E.g Empire)

fafc45381bfde997b6305c4e1600f1bf SSH-2.0-**Ruby**/Net::SSH_5.0.2 x86_64-linux (Ruby, E.g. used by Metasploit)

hasshServer examples

c1c596caaeb93c566b8ecf3cae9b5a9e SSH-2.0-**dropbear_2016.74**

d93f46d063c4382b6232a4d77db532b2 SSH-2.0-**dropbear_2016.72**

2dd9a9b3dbebfaeec8b8aabd689e75d2 SSH-2.0-**AWSCodeCommit**

a0fd4bcb0e72b4b21232a486825b6742 **Cowrie** SSH Honeypot

HASSH - implementation

Bro/Zeek

```
# bro-pkg install hassh
```

hasshVersion	string	1.0
hassh	string	d43d91bc39d5aaed819ad9f6b57b7348
hasshServer	string	a7a87fbe86774c2e40cc4a7ea2ab1b3c
cshka	string	ssh-rsa,rsa-sha2-512,rsa-sha2-256,ecdsa-sha2-nistp256,ssh-ed25519
hasshAlgorithms	string	curve25519-sha256@libssh.org,ecdh-sha2-nistp256,ecdh-sha2-nistp384,ecdh-sha2-nistp521,diffie-hellman-group-exchange-sha256,diffie-hellman-group14-sha1;chacha20-poly1305@openssh.com,aes128-ctr,aes192-ctr,aes256-ctr,aes128-gcm@openssh.com,aes256-gcm@openssh.com;umac-64-etm@openssh.com,umac-128-etm@openssh.com,hmac-sha2-256-etm@openssh.com,hmac-sha2-512-etm@openssh.com,hmac-sha1-etm@openssh.com,umac-64@openssh.com,umac-128@openssh.com,hmac-sha2-256,hmac-sha2-512,hmac-sha1;none,zlib@openssh.com
sshka	string	ssh-rsa,ssh-dss
hasshServerAlgorithms	string	diffie-hellman-group-exchange-sha256,diffie-hellman-group-exchange-sha1,diffie-hellman-group14-sha1,diffie-hellman-group1-sha1;aes128-ctr,aes192-ctr,aes256-ctr,aes256-cbc,rijndael-cbc@lysator.liu.se,aes192-cbc,aes128-cbc,blowfish-cbc,arcfour128,arcfour,cast128-cbc,3des-cbc;hmac-sha2-256,hmac-sha2-512,hmac-sha1,hmac-sha1-96,hmac-md5,hmac-md5-96,hmac-ripemd160,hmac-ripemd160@openssh.com;none

https://github.com/salesforce/hassh

HASSH - implementation

Python

Mode 1: Reads PCAP files

Mode 2: Live traffic capture

JSON Output:


```
{
 "timestamp": "2018-09-04T18:57:03.644663",
 "sourceIp": "10.1.2.3",
 "destinationIp": "192.1.2.3",
 "sourcePort": "52068",
 "destinationPort": "22",
 "client": "SSH-2.0-OpenSSH_7.6",
 "hassh": "06046964c022c6407d15a27b12a6a4fb",
 "hasshAlgorithms": "curve25519-sha256,curve25519-sha256@
 "hasshVersion": "1.0",
 "ckex": "curve25519-sha256,curve25519-sha256@libssh.org,
 "ceacts": "chacha20-poly1305@openssh.com,aes128-ctr,aes1
 "cmacts": "umac-64-etm@openssh.com,umac-128-etm@openssh.
 "ccacts": "none,zlib@openssh.com,zlib",
 "clcts": "[Empty]",
 "clstc": "[Empty]",
 "ceastc": "chacha20-poly1305@openssh.com,aes128-ctr,aes1
 "cmastc": "umac-64-etm@openssh.com,umac-128-etm@openssh.com,hmac-sha2-256-etm@openssh.com,hmac-sha2-512-etm
 "ccastc": "none,zlib@openssh.com,zlib",
 "cshka": "ecdsa-sha2-nistp256-cert-v01@openssh.com,ecdsa-sha2-nistp384-cert-v01@openssh.com,ecdsa-sha2-nist
}
```


Usage

- Live network traffic capture:

```
$ python3 hassh.py -i eth0 -l json -o hassh.json --print
```

Output:

```
[+] Server SSH_MSG_KEXINIT detected
    [ 192.1.2.3:22 -> 10.1.2.3:52068 ]
        [-] Identification String: SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.4
        [-] hasshServer: d43d91bc39d5aaed819ad9f6b57b7348
        [-] hasshServer Algorithms: curve25519-sha256@libssh.org,ecdh-sha2-n
[+] Client SSH_MSG_KEXINIT detected
    [ 10.1.2.3:52068 -> 192.1.2.3:22 ]
        [-] Identification String: SSH-2.0-OpenSSH_7.6
        [-] hassh: 06046964c022c6407d15a27b12a6a4fb
        [-] hassh Algorithms: curve25519-sha256,curve25519-sha256@libssh.org
```

https://github.com/salesforce/hassh

HASSH - implementation

Dockerized hassh.py

A dockerized version of hassh.py can be used to extract HASSH fingerprints from input PCAP files and live network traffic.

Build the docker image using Dockerfile:

```
$ docker build -t hassh:latest .
```

0x4d31/hassh ☆

Pulls 30

By 0x4d31 • Updated 5 months ago

A dockerized hassh.py to extract HASSH fingerprints from input PCAP files and live network traffic.

Container

Docker Pull Command

```
docker pull 0x4d31/hassh
```

https://github.com/salesforce/hassh

HASSH - implementation

hasshGen.py

License BSD 3-Clause

Sample python script and Dockerfiles to automate building docker images with different SSH clients/versions for generating HASSH fingerprints. As a demonstration we created a list (sshClient_list) containing 49 different version of OpenSSH, Python's paramiko and Dropbear SSH clients and generated a database of HASSH fingerprints in JSON and CSV formats.

Getting Started

1. Install Docker CE:

   > Download Docker Engine

2. Install the Docker's Python library:

   > `pipenv install docker`

3. Test:

   > `pipenv run python3 hasshgen.py -h`

Output:

```
usage: hasshgen.py [-h] [-i IMAGE] [-iV IMAGE_VER] [-c SSHCLIENT]
                   [-cV SSHCLIENT_VER] [-d DOCKER_FILE] -s SERVER
                   [-iF INPUT_FILE] [-f] [--cmd CMD]

A python script to automate building docker images with different SSH
clients/versions.

optional arguments:
  -h, --help            show this help message and exit
  -i IMAGE, --image IMAGE
                        Docker image name. e.g. alpine, ubuntu
  -iV IMAGE_VER, --image_ver IMAGE_VER
                        Docker image version. e.g. 18.04, latest
  -c SSHCLIENT, --sshclient SSHCLIENT
                        SSH client name
  -cV SSHCLIENT_VER, --sshclient_ver SSHCLIENT_VER
                        SSH client version
  -d DOCKER_FILE, --docker_file DOCKER_FILE
                        Specify the Dockerfile
  -s SERVER, --server SERVER
                        Specify the server address to test the SSH connection
  -iF INPUT_FILE, --input_file INPUT_FILE
                        Bulk mode; Specify an input file containing a list of
                        docker image, image version, sshclient and sshclient
                        version
  -f, --fingerprint     Set this option to automatically run hassh.py for
                        capturing SSH client fingerprints (HASSH). Specify the
                        command for running hassh.py using --cmd arg.
  --cmd CMD             Enter the command for running hassh.py. Use with
                        -f/--fingerprint arg
```

Branch: master ▾ **hassh / python / hasshGen / hassh_fingerprints.csv** Find file | Copy path

0x4d31-sfdc Updated hasshVersion 44f1d6f on 26 Sep

2 contributors

51 lines (50 sloc) | 28.6 KB Raw | Blame | History

Search this file...

	image	imageVersion	sshClient	sshClientVersion	clientIdentificationString	hassh
2	debian	sid-slim	dropbear	2018.76-4	SSH-2.0-dropbear_2018.76	e22efe3cde8b396b874c3f13fdb6c61a
3	debian	buster-slim	dropbear	2018.76-4	SSH-2.0-dropbear_2018.76	e22efe3cde8b396b874c3f13fdb6c61a
4	debian	stretch-slim	dropbear	2016.74-5	SSH-2.0-dropbear_2016.74	7742887e2a57712bdb91a772093f54ce
5	debian	jessie-slim	dropbear	2014.65-1+deb8u2	SSH-2.0-dropbear_2014.65	ad00edb0c2a031d9884826ba7b7ba41e
6	debian	jessie-slim	dropbear	2014.65-1+deb8u3	SSH-2.0-dropbear_2014.65	ad00edb0c2a031d9884826ba7b7ba41e
7	debian	wheezy-slim	dropbear	2012.55-1.3	SSH-2.0-dropbear_2012.55	16f898dd8ed8279e1055350b4e20666c
8	debian	wheezy-slim	dropbear	2012.55-1.3+deb7u2	SSH-2.0-dropbear_2012.55	16f898dd8ed8279e1055350b4e20666c
9	ubuntu	18.04	dropbear	2017.75-3build1	SSH-2.0-dropbear_2017.75	7742887e2a57712bdb91a772093f54ce
10	ubuntu	16.04	dropbear	2016.72-1	SSH-2.0-dropbear_2016.72	7742887e2a57712bdb91a772093f54ce
11	ubuntu	14.04	dropbear	2013.60-1ubuntu2	SSH-2.0-dropbear_2013.60	22865d7159a8dedcb091cd4fc5fd2841
12	alpine	3.8	dropbear-dbclient	2018.76-r2	SSH-2.0-dropbear_2018.76	e22efe3cde8b396b874c3f13fdb6c61a
13	alpine	3.7	dropbear-dbclient	2017.75-r1	SSH-2.0-dropbear_2017.75	7742887e2a57712bdb91a772093f54ce
14	alpine	3.5	dropbear-dbclient	2017.75-r0	SSH-2.0-dropbear_2017.75	7742887e2a57712bdb91a772093f54ce
15	python	3.6-alpine	paramiko	1.13.0	SSH-2.0-paramiko_1.13.0	b05dfbbf26090c7792d4aa6b76cd1f1a
16	python	3.6-alpine	paramiko	1.14.0	SSH-2.0-paramiko_1.14.0	b05dfbbf26090c7792d4aa6b76cd1f1a
17	python	3.6-alpine	paramiko	1.15.0	SSH-2.0-paramiko_1.15.0	d72f74b08466652d162ca02ad197b9ad
18	python	3.6-alpine	paramiko	1.16.0	SSH-2.0-paramiko_1.16.0	c6f5e6d54285a11b9f02fef7fc77bd6f
19	python	3.6-alpine	paramiko	1.17.0	SSH-2.0-paramiko_1.17.0	c6f5e6d54285a11b9f02fef7fc77bd6f
20	python	3.6-alpine	paramiko	1.18.0	SSH-2.0-paramiko_1.18.0	c6f5e6d54285a11b9f02fef7fc77bd6f
21	python	3.6-alpine	paramiko	2.0.0	SSH-2.0-paramiko_2.0.0	c6f5e6d54285a11b9f02fef7fc77bd6f
22	python	3.6-alpine	paramiko	2.1.0	SSH-2.0-paramiko_2.1.0	c6f5e6d54285a11b9f02fef7fc77bd6f
23	python	3.6-alpine	paramiko	2.2.0	SSH-2.0-paramiko_2.2.0	b5752e36ba6c5979a575e43178908adf
24	python	3.6-alpine	paramiko	2.3.0	SSH-2.0-paramiko_2.3.0	b5752e36ba6c5979a575e43178908adf
25	python	3.6-alpine	paramiko	2.4.1	SSH-2.0-paramiko_2.4.1	b5752e36ba6c5979a575e43178908adf
26	centos	7	openssh-clients	7.4p1-16.el7	SSH-2.0-OpenSSH_7.4	ec9ea89c70f5fc71cf61061bff5e4740
27	centos	6	openssh-clients	5.3p1-123.el6_9	SSH-2.0-OpenSSH_5.3	3646f4d62498cd92721bd242ce988a42
28	fedora	rawhide	openssh-clients	7.8p1-2.fc30	SSH-2.0-OpenSSH_7.8	4eea4239dc591cbf6e661a656ae9d58c
29	fedora	29	openssh-clients	7.8p1-1.fc29	SSH-2.0-OpenSSH_7.8	4eea4239dc591cbf6e661a656ae9d58c
30	fedora	28	openssh-clients	7.7p1-2.fc28	SSH-2.0-OpenSSH_7.7	4eea4239dc591cbf6e661a656ae9d58c
31	fedora	27	openssh-clients	7.5p1-5.fc27	SSH-2.0-OpenSSH_7.5	82a9e17b13a816a13b8c8bfe1b5cf030

https://github.com/salesforce/hassh

HASSH - implementation

Nmap script

ssh-hassh nse script reports hasshServer (i.e. SSH Server Fingerprint) and hasshServerAlgorithms for the target SSH server.

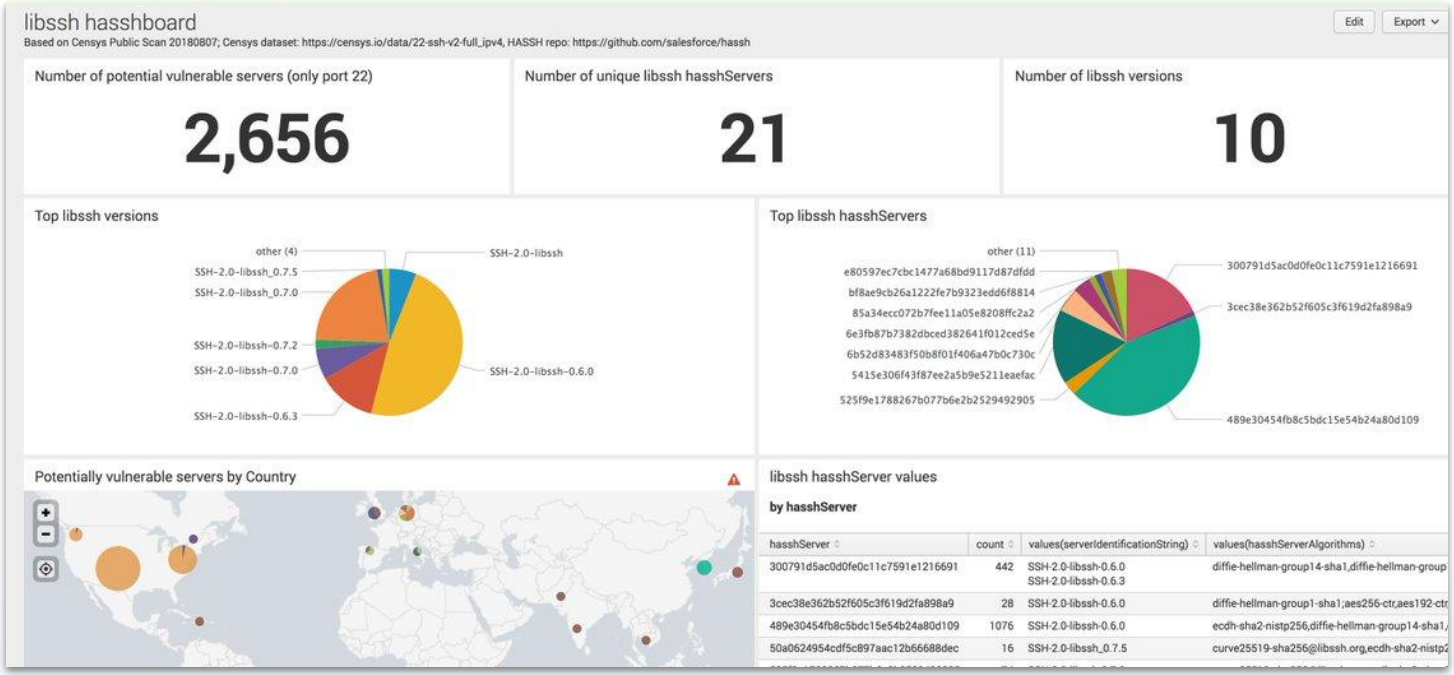
```
adel$ nmap -p 22 --script ssh-hassh github.com -oX hassh.xml

Starting Nmap 7.00 ( https://nmap.org ) at 2018-09-26 03:13 AEST
Nmap scan report for github.com (192.30.255.112)
Host is up (0.18s latency).
Other addresses for github.com (not scanned): 192.30.255.113
rDNS record for 192.30.255.112: lb-192-30-255-112-sea.github.com
PORT   STATE SERVICE
22/tcp open  ssh
| ssh-hassh:
|   Server Identification String: SSH-2.0-libssh_0.7.0
|   hasshServer: 85a34ecc072b7fee11a05e8208ffc2a2
|_  hasshServer Algorithms: curve25519-sha256@libssh.org,ecdh-sha2-nistp256,ecdh-sha2-nistp384,ecdh-sha2-nis

Nmap done: 1 IP address (1 host up) scanned in 1.47 seconds
```

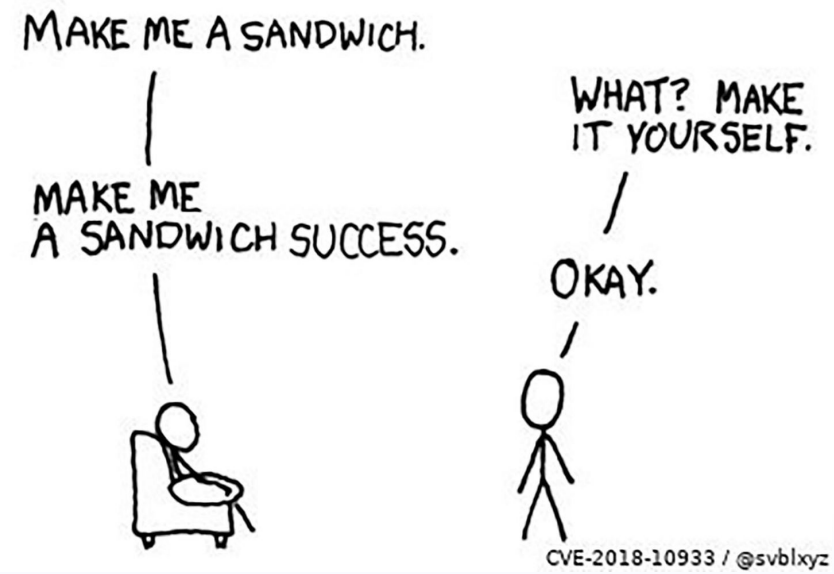
https://github.com/0x4D31/hassh-utils

HASSH - Use cases

CVE-2018-10933 libssh auth bypass bug

The vulnerability, which was introduced in libssh version 0.6, makes it possible to log in by presenting a server with a **SSH2_MSG_USERAUTH_SUCCESS** message rather than SSH2_MSG_USERAUTH_REQUEST message the server was expecting



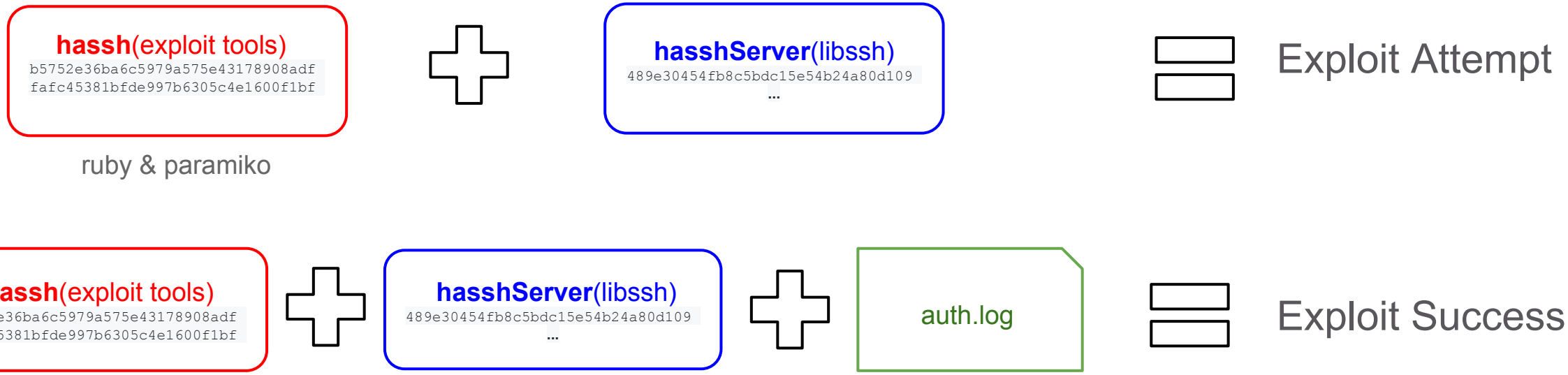


https://www.libssh.org/security/advisories/CVE-2018-10933.txt

HASSH - Use cases

libssh Auth bypass bug

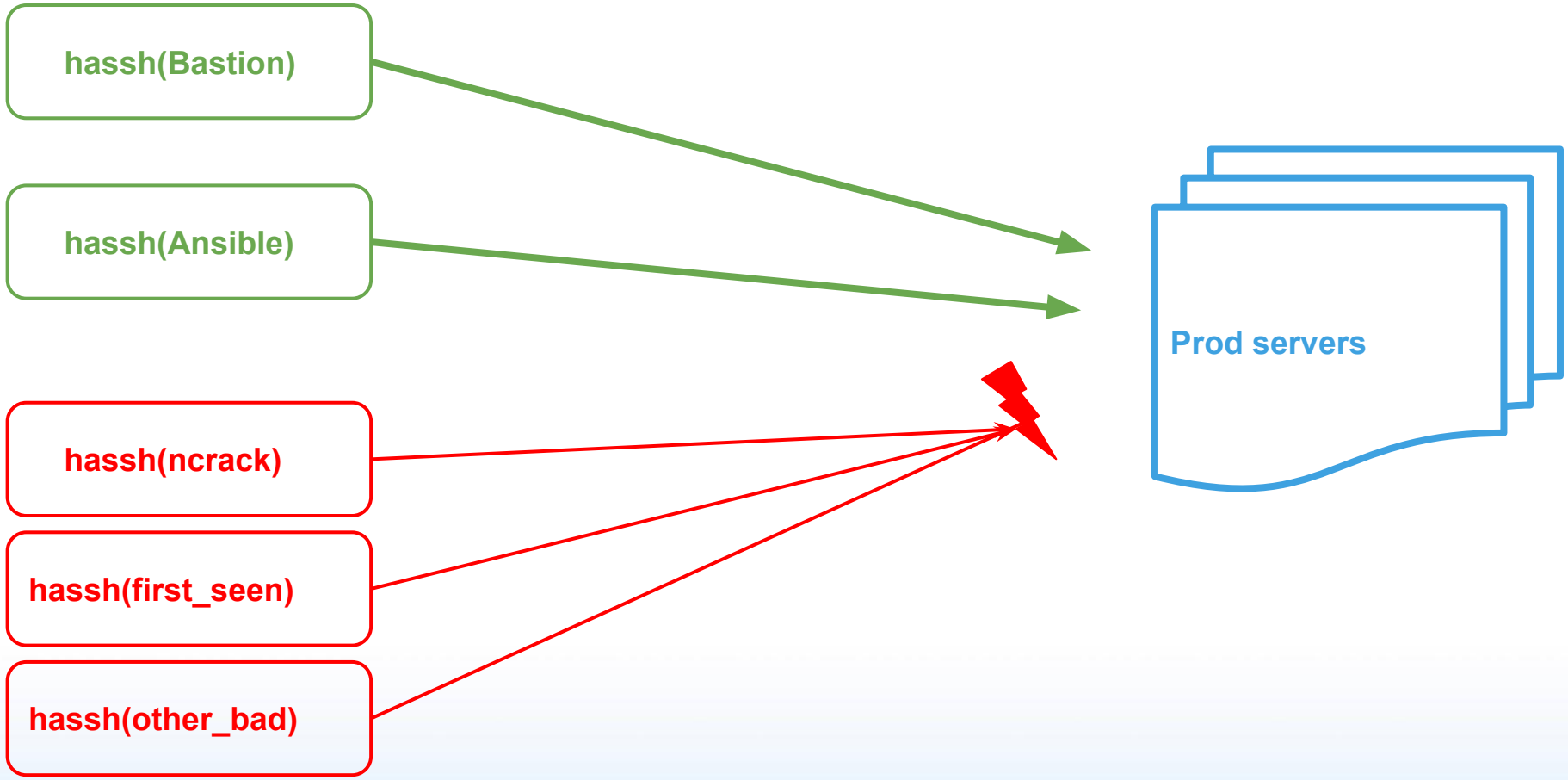
Building a detection case



https://www.libssh.org/security/advisories/CVE-2018-10933.txt

HASSH - Use cases

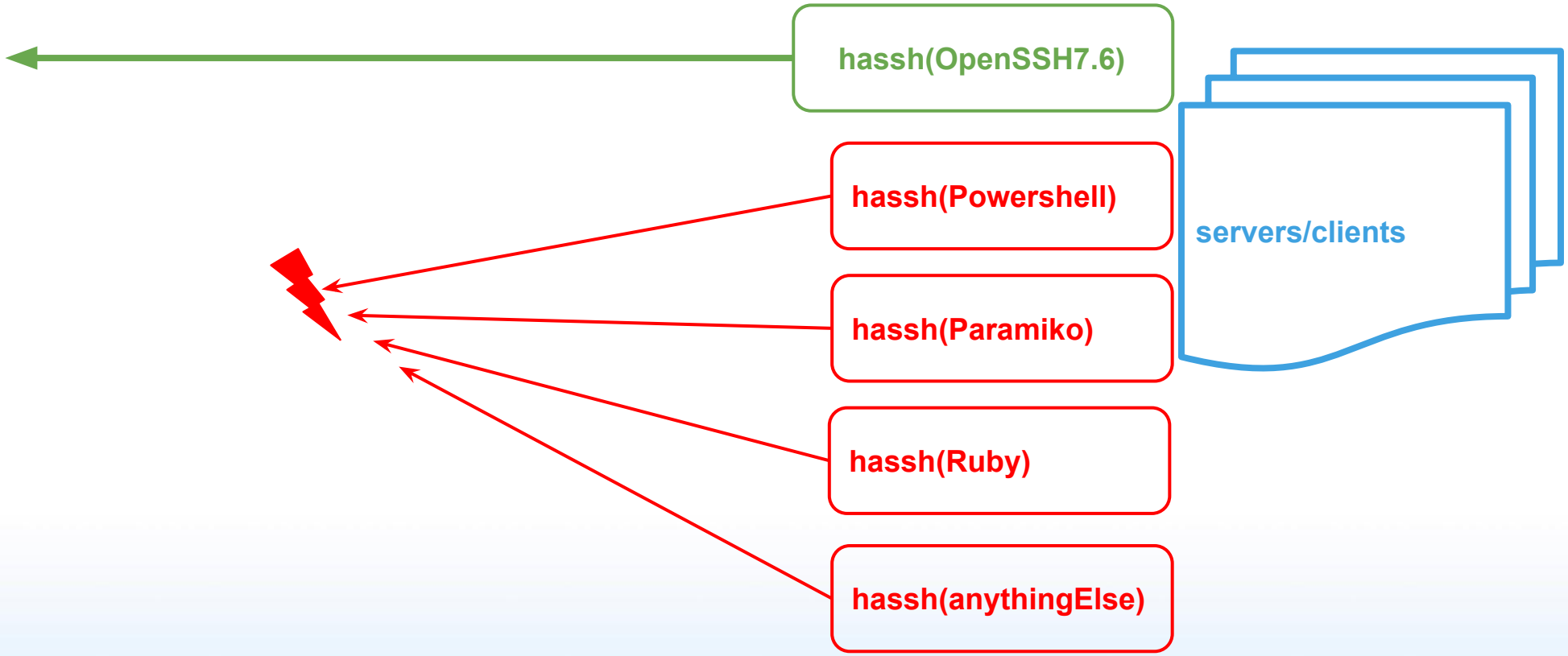
Control on whitelist/blacklist TO servers



https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c

HASSH - Use cases

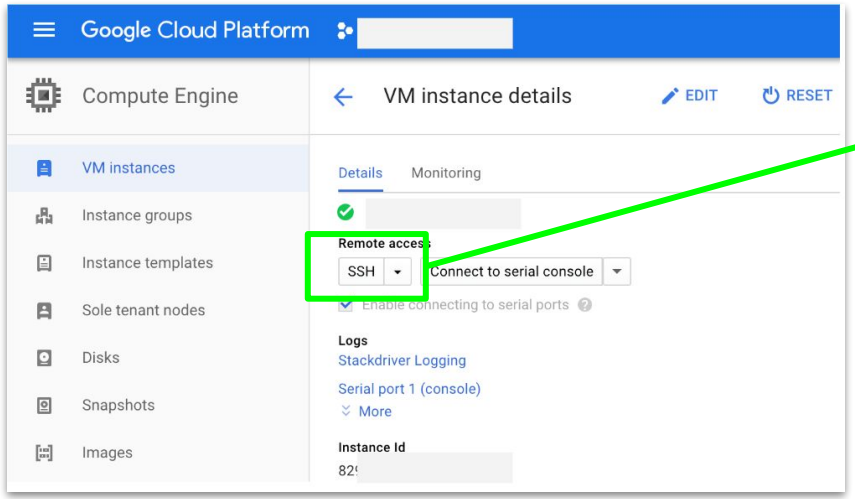
Control on whitelist/blacklist FROM systems

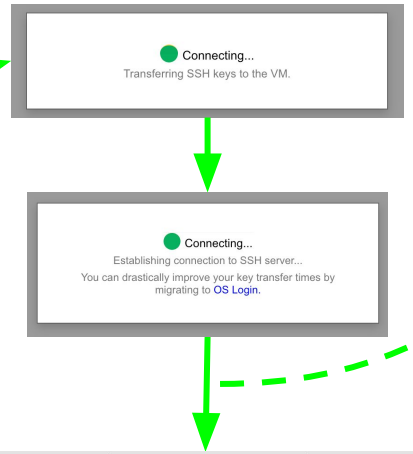


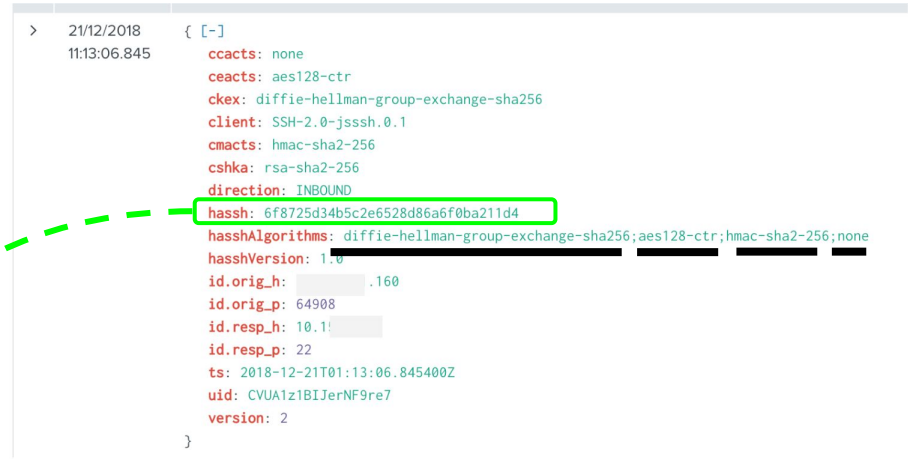
https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c

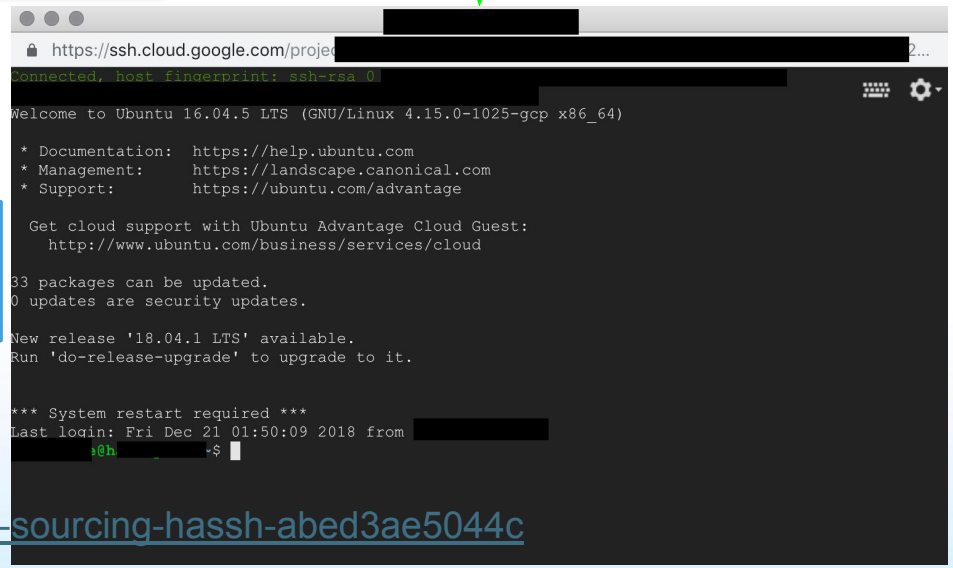
HASSH - Use cases

Highly defined environment - eg GCP control









https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c

HASSH - Use cases

Lateral Movement Detection

- Attackers use systems' default SSH client for lateral movement - **But not always**!
  - Empire's **`Invoke-SSHCommand.ps1`**
  - Renci SSH.NET Library
  - Specific Client ID and HASSH

```
 # This is the base 64 encoded Renci.SshNet.dll You may find it at https://sshnet.codeplex.com/downloads/get/944156
$Base64 = 'TVqQAAMAAAAEAAAA//8AALgAAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAgAAAAA4fug4AtAnNIbgBTM0hVGhp
$Content = [System.Convert]::FromBase64String($Base64)
```

- How about Cobalt Strike?

HASSH - Use cases

Lateral Movement Detection - Cobalt Strike

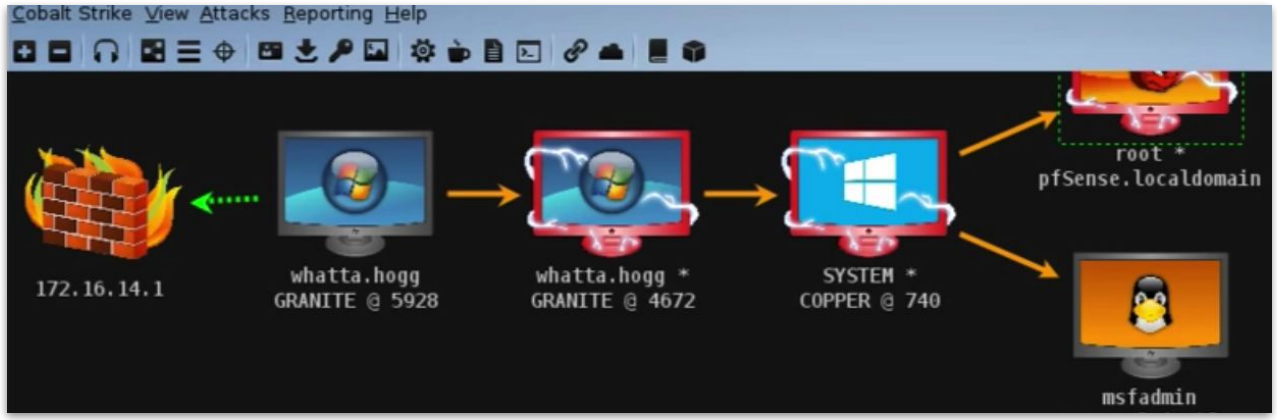
- A popular adversary simulation and post-exploitation tool
- In 2016 an SSH client added to Cobalt Strike (version 3.5+) that allows you to conduct post-exploitation actions against UNIX targets!
- Built-in SSH client


```
beacon> ssh 172.16.20.159 root pfsense
[*] Tasked beacon to SSH to 172.16.20.159:22 as root
[+] host called home, sent: 848967 bytes
[+] host called home, sent: 45 bytes
[+] established link to child session: 172.16.20.159
[COPPER] SYSTEM */740
beacon>
```

HASSH - Use cases

Lateral Movement Detection - Cobalt Strike

- Windows → Linux



- Detection:
  - Internal SSH connections from a **Windows** host using the CS built-in SSH Client
  - Client ID String: `SSH-2.0-libssh2_1.7.0`
  - HASSH: `a7a87fbe86774c2e40cc4a7ea2ab1b3c`

HASSH - Use cases

Detect Evasion techniques: SSH Honeypots

- Cowrie hasshServer:

`a0fd4bcb0e72b4b21232a486825b6742` and `06046964c022c6407d15a27b12a6a4fb`

hasshServer	string	06046964c022c6407d15a27b12a6a4fb
hasshServerAlgorithms	string	curve25519-sha256,curve25519-sha256@libssh.org,ecdh-s nistp384,ecdh-sha2-nistp521,diffie-hellman-group-exchang sha512,diffie-hellman-group18-sha512,diffie-hellman-group group14-sha256,diffie-hellman-group14-sha1,ext-info-c;cha poly1305@openssh.com,aes128-ctr,aes192-ctr,aes256-ctr,a gcm@openssh.com;umac-64-etm@openssh.com,umac-128 256-etm@openssh.com,hmac-sha2-512-etm@openssh.com etm@openssh.com,umac-64@openssh.com,umac-128@op sha2-512,hmac-sha1;none,zlib@openssh.com,zlib

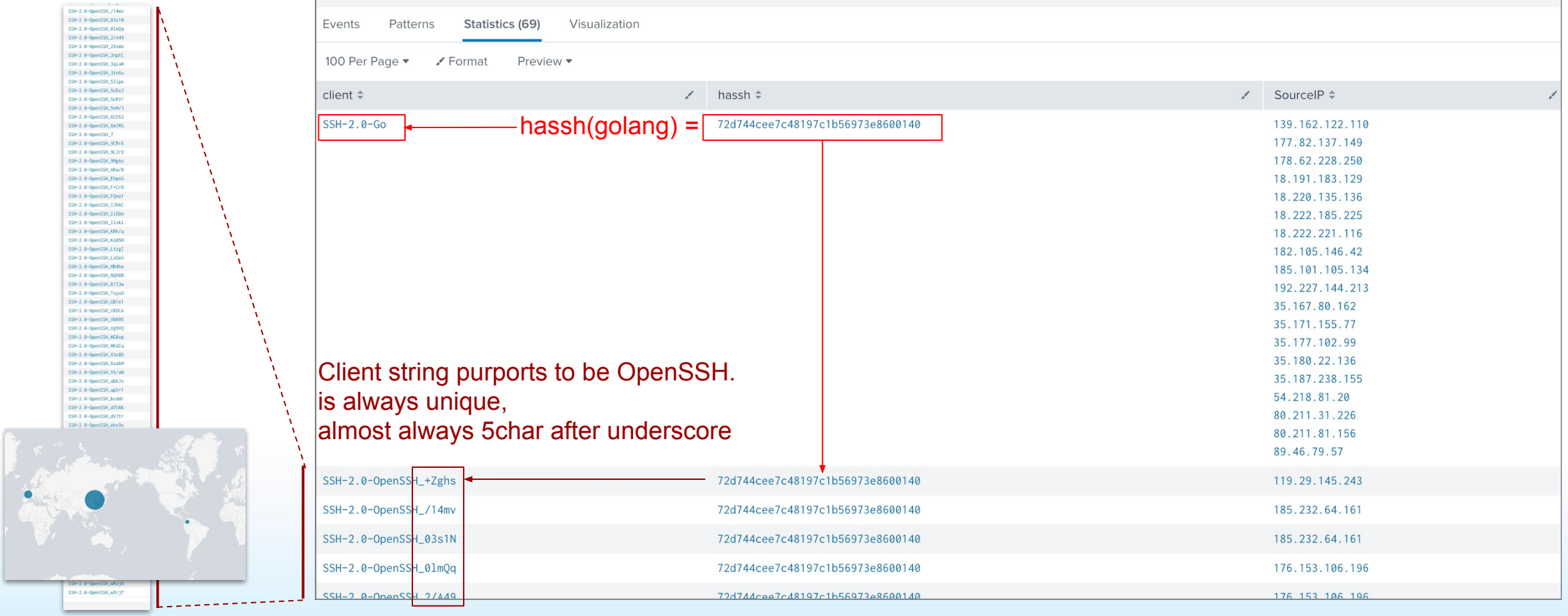
```
t.supportedCiphers = [
    b'aes128-ctr',
    b'aes192-ctr',
    b'aes256-ctr',
    b'aes128-cbc',
    b'3des-cbc',
    b'blowfish-cbc',
    b'cast128-cbc',
    b'aes192-cbc',
    b'aes256-cbc'
]
t.supportedPublicKeys = [b'ssh-rsa', b'ssh-dss']
t.supportedMACs = [b'hmac-md5', b'hmac-sha1']
t.supportedCompressions = [b'zlib@openssh.com', b'zlib', b'none']
```

- Cymmetria honeycomb, libssh module:

`b5752e36ba6c5979a575e43178908adf`

HASSH - Use cases

Detect Evasion techniques used by cred stuffers and brute forcers



HASSH - Use cases

1) All scans from this one IP address

2) One hassh value,
So it must be only one client application

3) Client Strings are being spoofed and are cycled through

4) Accounts attempted (all root)

Because defenders can shun connections based on Client Protocol string

HASSH - Industry support

- Grey Noise
- Binary Edge
- Cowrie
- Shodan
- Trisul NSM
- MISP
- Security Onion
- ExtraHop (soon)
- Suricata (soon)


```
{
  "timestamp":"2018-09-29T15:43:09.300627Z",
  "sensor":"0x4d31-nyc3-01",
  "src_ip":"10.1.2.3",
  "eventid":"cowrie.client.kex",
  "message":"SSH client hassh fingerprint: 68e0ba85e1a818f7c49ea3f4b849bd15",
  "hassh":"68e0ba85e1a818f7c49ea3f4b849bd15",
  "hasshAlgorithms":"curve25519-sha256@libssh.org,ecdh-sha2-nistp256,ecdh-sha2-nistp3
   diffie-hellman-group-exchange-sha1,diffie-hellman-group14-sha1,ext-info-c;chacha20
   128-gcm@openssh.com,aes256-gcm@openssh.com,aes128-cbc,aes192-cbc,aes256-cbc,3des-c
   6-etm@openssh.com,hmac-sha2-512-etm@openssh.com,hmac-sha1-etm@openssh.com,umac-64@
   c-sha1;none,zlib@openssh.com,zlib",
  "session":"3fd36ed0e459",
  "keyAlgs":[ ⊞ ],
  "kexAlgs":[ ⊞ ],
  "macCS":[ ⊞ ],
  "langCS":[ ⊞ ],
  "compCS":[ ⊞ ],
  "encCS":[ ⊞ ]
}
```

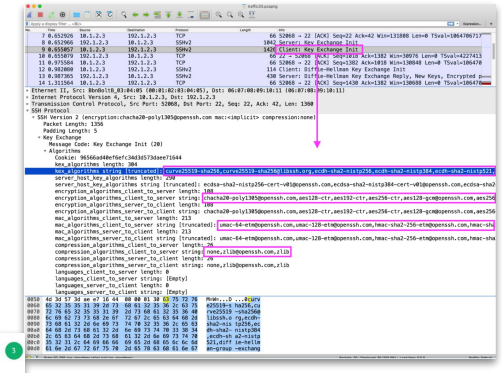

HASSH (i.e. SSH Client Fingerprint)
MD5(kexAlgs;encryptionAlgs;macAlgs;compressionAlgs)

Cowrie logs - hassh support

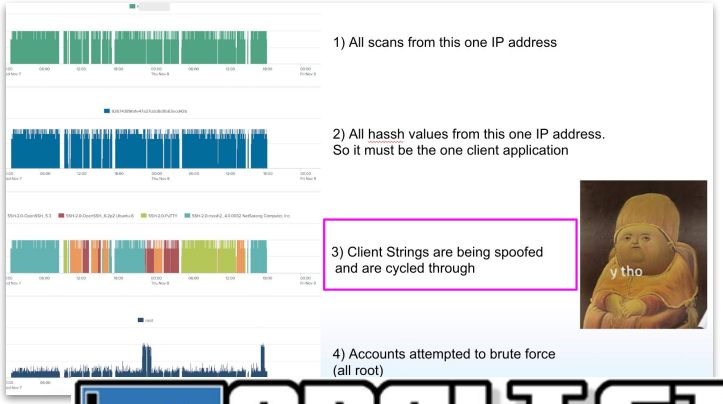
@benreardon
breardon@salesforce.com

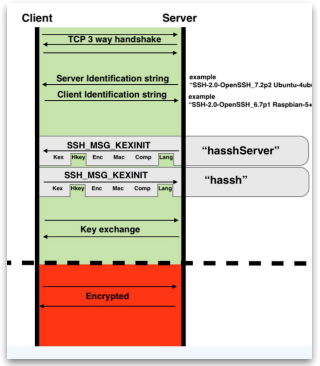
HASSH

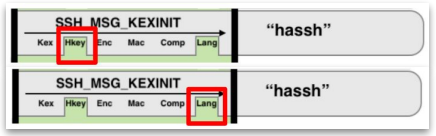
Questions, you will ask



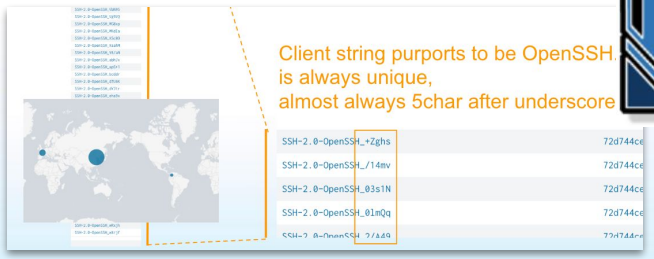
	MD5	SHA-256
Collisions?	Not for our use case	No
Short	Yes, 32 chars	No, 64 chars
Tweetable	Yes	Not friendly at all
Swamps surrounding text	No	Yes
Supported	By everything	Probably by everything
Provides Nett Benefit	Yes	No

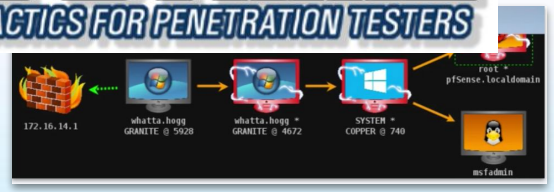






https://github.com/salesforce/hassh





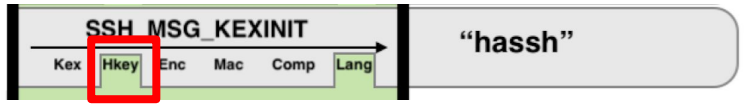

THANK YOU

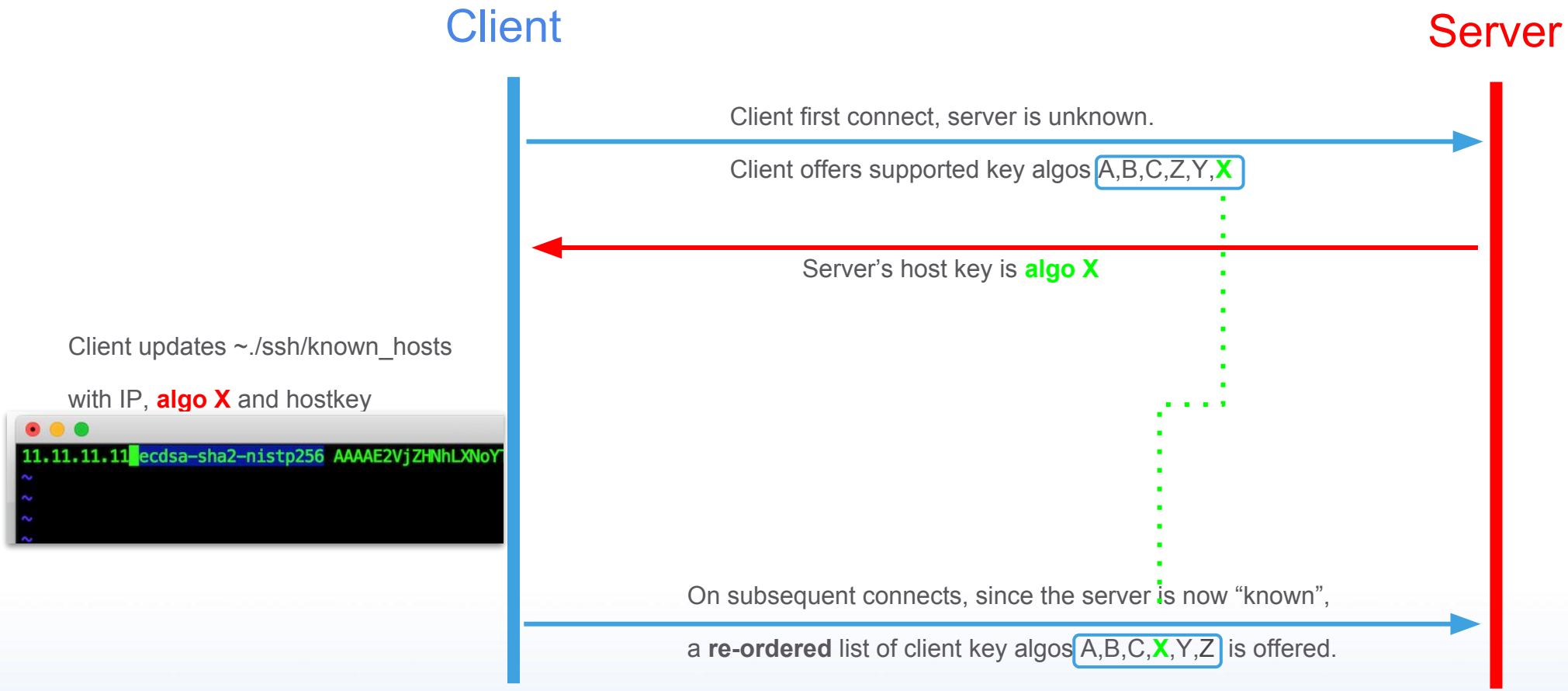
HASSH

Time Permitting Slides

HASSH - Notes during Development

Host Key Algos





https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c

HASSH - Use cases

libssh Auth bypass bug

Building a detection case

Step 1: Find **hasshServer**(vuln libssh server) AND/OR use the Server Identification strings

Even knowing you have libssh servers may be of interest.

Libssh authentication bypass vulnerability (CVE-2018-10933)

An analysis of Censys Public Scan 20180807 (only port 22) to estimate the number of servers {potentially} vulnerable to the recent Libssh bug.

- CVE-2018-10933 Advisory
- Exploit - Patched Libssh Client
- CVE-2018-10933-test
- libSSH-Authentication-Bypass
- HASSH - an SSH client/server profiling method
- ssh-hassh nmap script
- Censys data
- Hunt for and Exploit the libSSH Authentication Bypass
- A sample Shodan filter: `product:libssh port:22,2222 ssh.hassh:489e30454fb8c5bdc15e54b24a80d109`

The most common hasshServer values for each libssh version:

hasshServer	Server Identification String	Count
489e30454fb8c5bdc15e54b24a80d109	SSH-2.0-libssh-0.6.0	1076
5415e306f43f87ee2a5b9e5211eaefac	SSH-2.0-libssh_0.7.0	406
300791d5ac0d0fe0c11c7591e1216691	SSH-2.0-libssh-0.6.3	321
e80597ec7cbc1477a68bd9117d87dfdd	SSH-2.0-libssh-0.7.2	51
e5c1da26cdde67ec7b2a7759b13b6d28	SSH-2.0-libssh-0.6.5	22
50a0624954cdf5c897aac12b66688dec	SSH-2.0-libssh_0.7.5	16
e80597ec7cbc1477a68bd9117d87dfdd	SSH-2.0-libssh-0.7.3	8
c251cb842064997a986c1bc145aec3bd	SSH-2.0-libssh-0.7.1	6

hasshServer values of all observed libssh versions:

hasshServer	Server Identification String	Count
489e30454fb8c5bdc15e54b24a80d109	SSH-2.0-libssh-0.6.0	1076
300791d5ac0d0fe0c11c7591e1216691	SSH-2.0-libssh-0.6.0, SSH-2.0-libssh-0.6.3	442
5415e306f43f87ee2a5b9e5211eaefac	SSH-2.0-libssh_0.7.0	406
6b52d83483f50b8f01f406a47b0c730c	SSH-2.0-libssh_0.7.0	131
6e3fb87b7382dbced382641f012ced5e	SSH-2.0-libssh-0.7.0	98

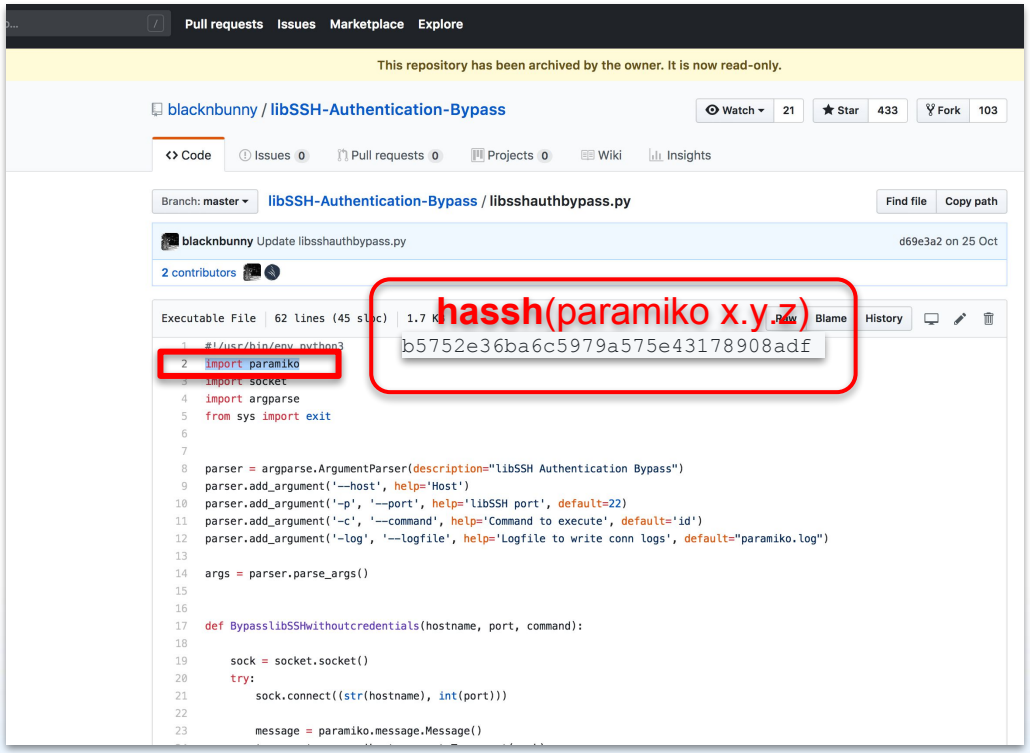
https://www.libssh.org/security/advisories/CVE-2018-10933.txt

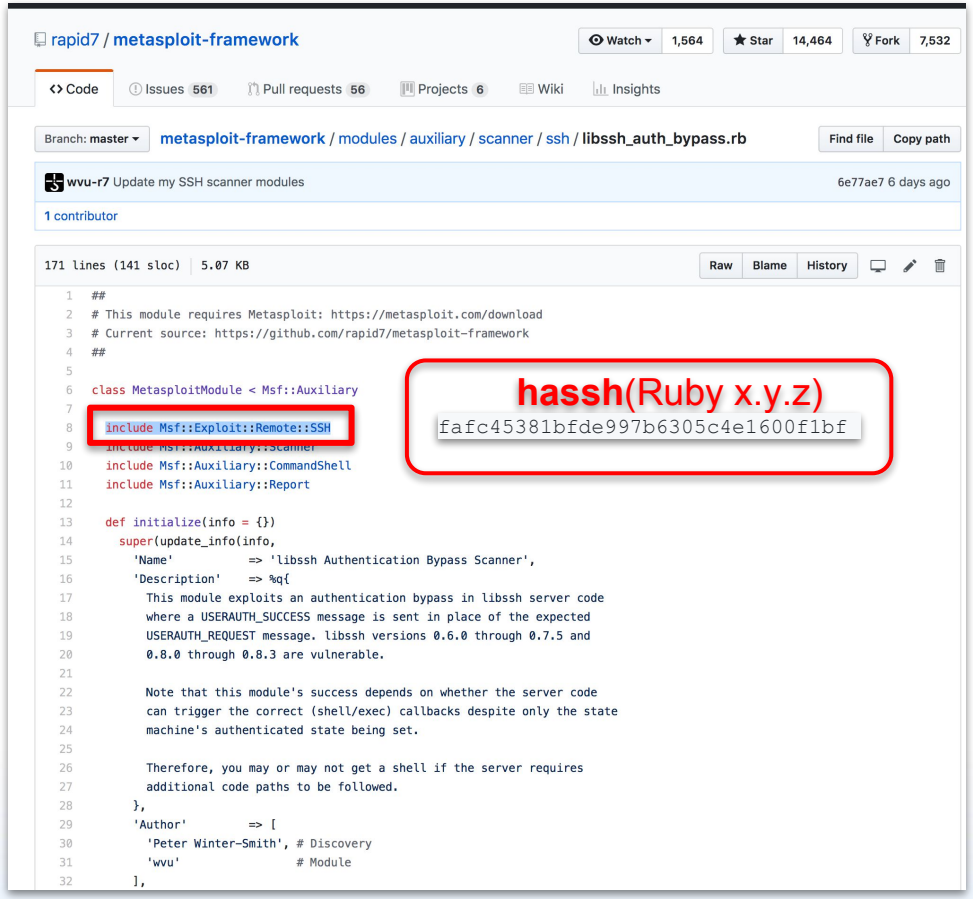
HASSH - Use cases

libssh Auth bypass bug

Building a detection case

Step 2 : find **hassh**(exploit tools)





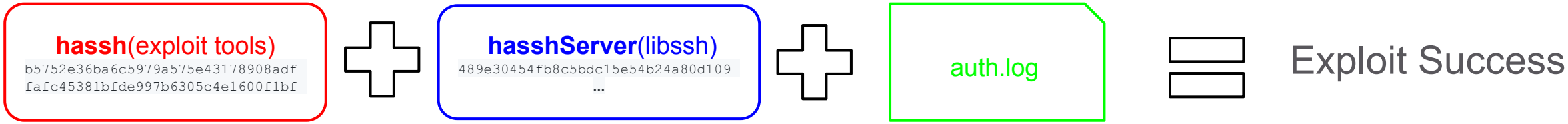
https://www.libssh.org/security/advisories/CVE-2018-10933.txt

HASSH - Use cases

libssh Auth bypass bug

Building a detection case

Step 3 : Apply logic (example)

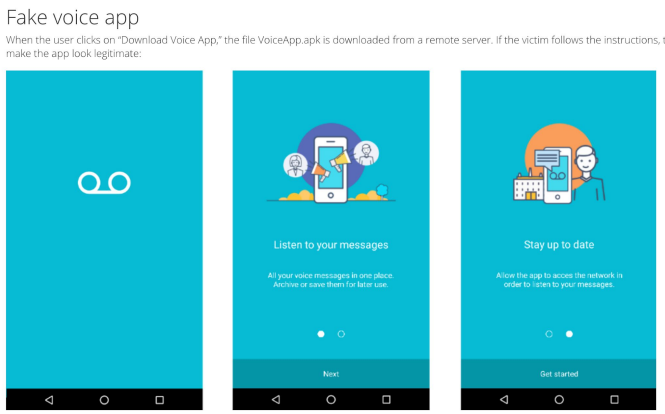


https://www.libssh.org/security/advisories/CVE-2018-10933.txt

HASSH - use cases

Detect Trojaned Applications

TimpDoor/Milkdoor uses SSH to tunnel



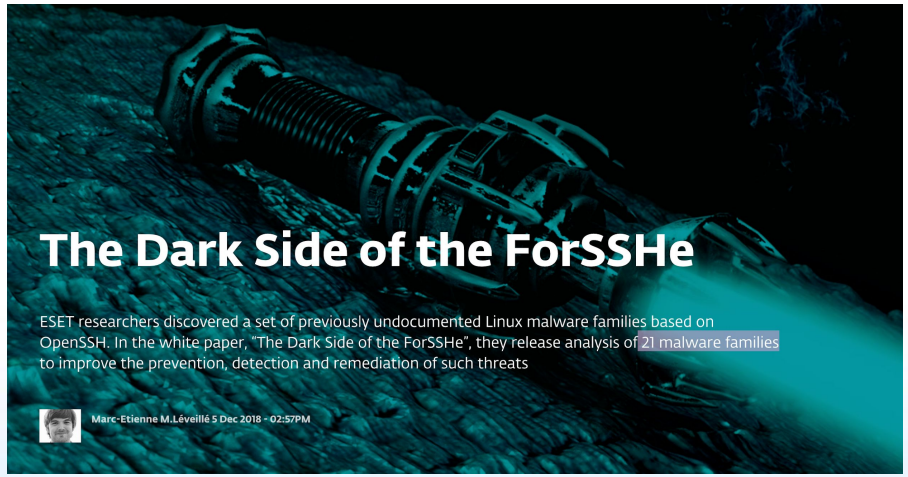
https://securingtomorrow.mcafee.com/other-blogs/mcafee-labs/android-timpdoor-turns-mobile-devices-into-hidden-proxies/

YARA sig for private keys in PE files.

```
rule adversary_methods_pe_with_openssh_key {
meta:
author="smiller"
description="Looking for PE files with default OpenSSH private key strings"
strings:
$a1= "[-----BEGIN OPENSSH PRIVATE KEY-----"
$a2= {0A2D2D2D2D2D454E44204F50454E5353482050524956415445204B45592D2D2D2D2D0A257373682D}
condition:
uint16(0) == 0x5A4D and uint32(uint32(0x3C)) == 0x00004550 and all of them
}
```

https://gist.github.com/stvemillertime/f518d598097eb657a215795950ff8b8d

21 Malware families in OpenSSH



https://www.welivesecurity.com/2018/12/05/dark-side-of-the-forsshe/

Recap - what is HASSH

Looking for signals in the initialization of encrypted communication channels is not a new concept. There are many examples of fingerprinting both unencrypted and encrypted protocols such as TLS. However somewhat surprisingly, no open source scalable fingerprinting method has been developed for one of our most common and relied upon encrypted protocols SSH—an integral component of the internet. Enter, the HASSH.

https://engineering.salesforce.com/open-sourcing-hassh-abed3ae5044c

Summary

- HASSH *augments*, provides another datapoint that *can* be insightful
- Not a silver bullet
- Value depends on the maturity of the consuming org.
- Knowledge of “what is normal” still critical
- Detection/Hunting/Forensics/Control?
- Contributes to non-repudiation in a Forensic context