Passwords^15 @ Security BSides Las Vegas

Harvesting Passwords from Source Code, Scripts and Code Repositories

Philippe Paquet
philippe@jaegerindustries.com

Housekeeping

- We have 10 minutes reserved at the end for Q&A
- Presentation and tools are available on Github: https://github.com/jaegerindustries/passwords15

Why this talk?

- I have been stumbling on credentials in source code regularly
- I am curious by nature so I started to dig in the issue
- I found more than I was expecting, a lot more

To illustrate the issue, let’s see what we can find on Github

Some considerations

- Results are from analysis performed in the last 6 months
- Results can be replicated with other public source code repositories
- I specifically avoided to search for generic terms to restrict the sample size and work associated with the analysis

Some considerations

- When searching on Github:
    - Only the default branch is considered
    - Only files smaller than 384 KB are searchable
    - So, there is a lot more to find than straightforward searches show
- You don’t have to use github to search
    - Use your favorite search engine
    - Restrict results to raw.githubusercontent.com

authorize.net

- authorize.net is a payment processor
- Sample code that they provide:

```
$api_login_id = 'YOUR_API_LOGIN_ID';
$transaction_key = 'YOUR_TRANSACTION_KEY';
$md5_setting = 'YOUR_API_LOGIN_ID'; // Your MD5 Setting
```

- Let’s search for:

  ”transaction_key api_login_id md5_setting”

authorize.net

- 1134 code results
- 31 valid set of credentials

```
$api_login_id = ‘7E*****63’;
$transaction_key = ‘3v************pV’;
$mdc_setting = ‘7E*****63’; // Your MD5 Setting

$api_login_id = ’64********8r’; // your api login
$transaction_key = ’38*************RN’; //Your transaction Key
$md5_setting = ’64********8r’; // Your MD5 Setting - use your api login id
```

shodanhq.com

- Shodan is an HTTP header search engine
- Sample code that they provide:

  SHODAN_API_KEY = "insert your API key here"
- Let’s search for:

  ”SHODAN_API_KEY”

shodanhq.com

- 114 code results
- 15 valid keys:

```
SHODAN_API_KEY = ‘B4****************************kl’
SHODAN_API_KEY = “CU****************************fm”
SHODAN_API_KEY = “2M****************************zF”
SHODAN_API_KEY = “uA****************************iW”
SHODAN_API_KEY = “c1****************************uY” #Enter API key here
```

Amazon Web Services

- Amazon Web Services is an infrastructure provider
- Amazon Web Services keys start with ”AKIAI”
- Searching for ”AKIAI” gives us:
    - 100 code results
    - 2 valid keys

Amazon Web Services

```
private String s1 = "AKIAI";
private String s4 = “Dv*****4m”;
private String s5 = “R+*********MM”;

private String getAccessKey(){
    String s3 = "6JIQ";
    String s2 = “6T*******G6”;
    return s1 + s2 + s3;
}

private String getSecretKey(){
    String s6 = “19*****QY”;
    String s7 = “Hu*****ps”;
    return s4 + s5 + s6 + s7;
}
```

Let’s check default passwords

Default passwords

- Let’s search for “default_password extension:properties”
  - 208 code results
  - 28 default passwords including:
    111111 123456 123qwe abc123 abc123456 abcd1234
    admin admin888 demo technician timesheet web

Default passwords

- Let’s search for“default_password extension:yml”
  - 72 code results
  - 23 default passwords including:
    1234 12345678 aaaaaaaa admin Admin123 changeme password
  - Some better passwords this time:
    <3_ygriTtE andwFlxe77c2A

Let’s search for backdoor passwords

Why not?

Backdoor passwords

- Searching for “backdoor_password” gives us:
  - 16 code results
  - 4 backdoor passwords:
    backdoor 1233321 sage backdoor_password

Expect more

- It is currently possible to find credentials and API keys for every major services with simple queries
- There is a magnitude mode to find in private source code repositories

We need a systematic approach

How to find password in source code?

- Search for definitions and string assignments
    - “keyword = password”
- You can also search for particular function calls
    - function(username, password)
- So you need:
    - Regular expressions
    - Keywords

Regular expressions

Javascript

- Patterns
  - “keyword” : “password”
  - keyword : “password”
  - keyword = “password”
- Regular expressions
  - /"KEYWORD"\s*?:\s*?"(\S+?)"/i
  - /KEYWORD\s*?:\s*?"(\S+?)"/i
  - /KEYWORD\s*?=\s*?"(\S+?)"/i

PHP

- Patterns
    - ‘keyword’ => ‘password’
    - $keyword = ‘password’
    - ‘keyword’ , ‘password’
- Regular expressions
    - /((‘KEYWORD’)|(“KEYWORD”))\s*?=>\s*?(('\S+?')|("\S+?"))/i
    - /\$KEYWORD\s*?=\s*?((‘\S+?’)|(“\S+?"))/i
    - /((‘KEYWORD’)|("KEYWORD"))\s*?,\s*?(('\S+?')|("\S+?"))/i

Python

- Patterns
  - keyword = ‘password’
  - ‘keyword’ : ‘password’
- Regular expressions
  - /KEYWORD\s*?=\s*?u?(('\S+?')|("\S+?"))/i
  - /u?(('KEYWORD')|("KEYWORD"))\s*?:\s*?u?(('\S+?')|("\S+?"))/i

C#

- Pattern
  - keyword = "password"
  - System.Net.NetworkCredential("username", "password")
- Regular expression
  - /KEYWORD\s*?=\s*?”(\S+?)”/i
  - /System.Net.NetworkCredential\s*?\(\s*?”(\S+?)"\s*?,\s*?"(\S+?)"\s*?\)/i

Keywords

Harvesting keywords is easy

- Choose a service that you are interested in
- Download sample code
- Build a keywords list from the definitions and variables names
- You can also match the format of the API key

Amazon Web Services

- Keywords
    - aws_access_key_id = YOUR_AWS_ACCESS_KEY_ID
      aws_secret_access_key = YOUR_AWS_SECRET_ACCESS_KEY
    - aws_access_key_id
      aws_secret_access_key
- Function
    - Credentials(‘YOUR_ACCESS_KEY’, ‘YOUR_SECRET_KEY’);
    - /Credentials\s*?\(\s*?'(\S+?)'\s*?,\s*?'(\S+?)'\s*?\)/i

2 Tools to automate harvesting

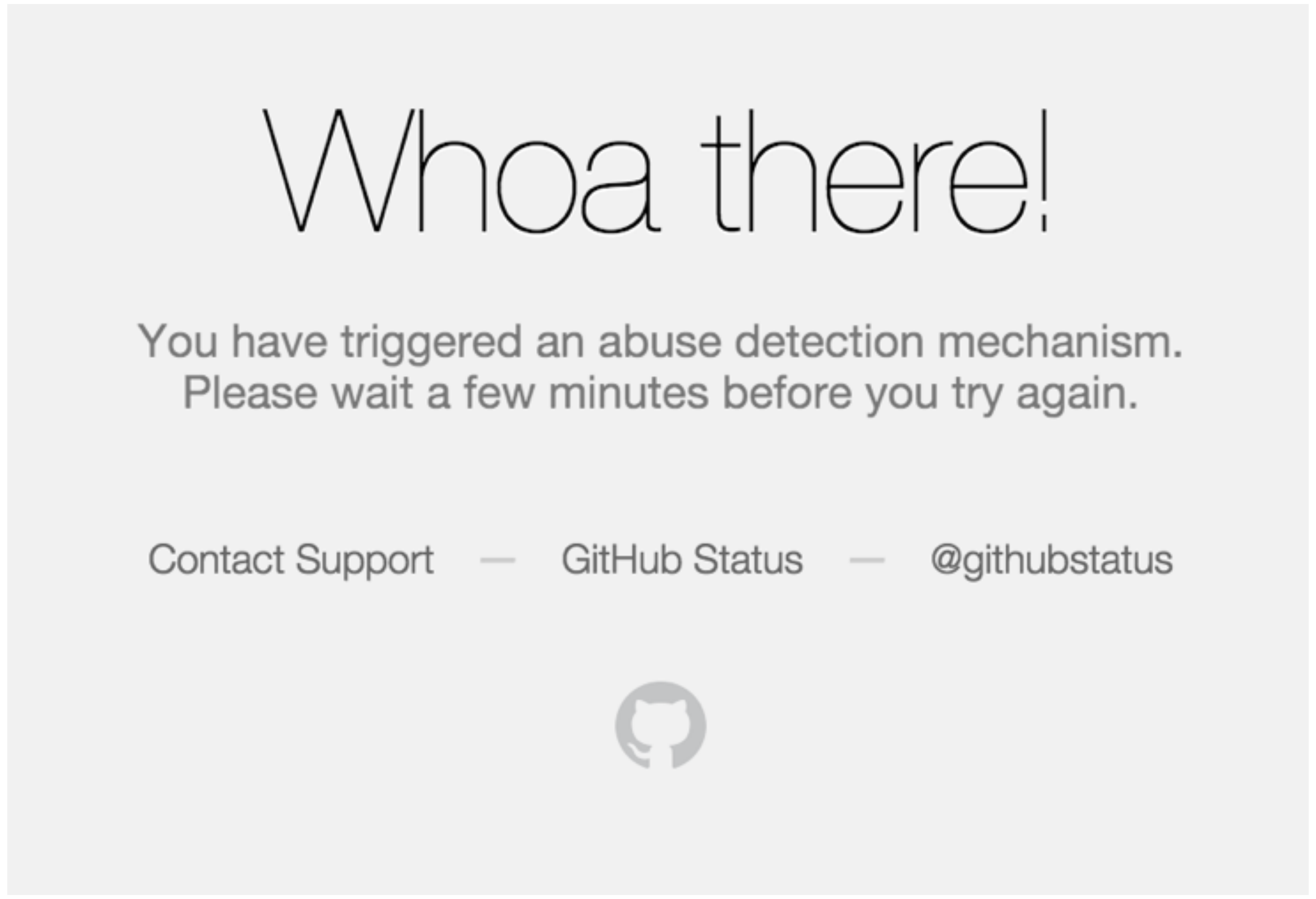
password_search

- Search for passwords on a file system
- Written in PHP
- Comes with a set of regular expressions
- Comes with a set of keywords

password_crawl

- Similar to password_search
    - Written in PHP
    - Comes with a set of regular expressions
    - Comes with a set of keywords
- Integrate an HTTP crawler
- Ideal for your enterprise instance of Github

This are <u>auditing</u> tools



Whoa there!

You have triggered an abuse detection mechanism.
Please wait a few minutes before you try again.

Contact Support — GitHub Status — @githubstatus

If you want to crawl Github, you will have to make some modifications

Now, the interesting part

- How did we get there?
- What are we going to do about it?

Credentials in source code

- Development is an iterative process
    - From simple to complex
    - Hardcoding credentials is simple
- This is the main reason that you will find credentials in source code
- But credentials should never be stored in source code

Credentials in source code

- Design a solution
    - Standardize files holding credentials
    - Excluded them from code repositories
    - Separation of duty = separate credentials
    - Deployment procedures
- Communicate

Default passwords

- Default passwords shouldn’t be a problem
  - They get changed during setup, right?
- Do not use default passwords
- Force users to create a password at setup time

Sample code

- Sample code get integrated in applications as is
- Do not hardcode credentials in sample code
- In fact, treat sample code as production code

I suppose it is tempting, if the only tool you have is a hammer, to treat everything as if it were a nail.

Abraham H. Maslow (1962)
Toward a Psychology of Being

Passwords

- Passwords are an authentication mechanism designed for humans
- Something you know
- Knowing implies safe storage

Applications

- Applications are not human
- Applications do not provide safe storage

Solution?

- We need an authentication method suited to applications
- Maybe white box cryptography? I don’t know…
- But it is up to us to come with a solution

Questions?

Contact me

Philippe Paquet
philippe@jaegerindustries.com