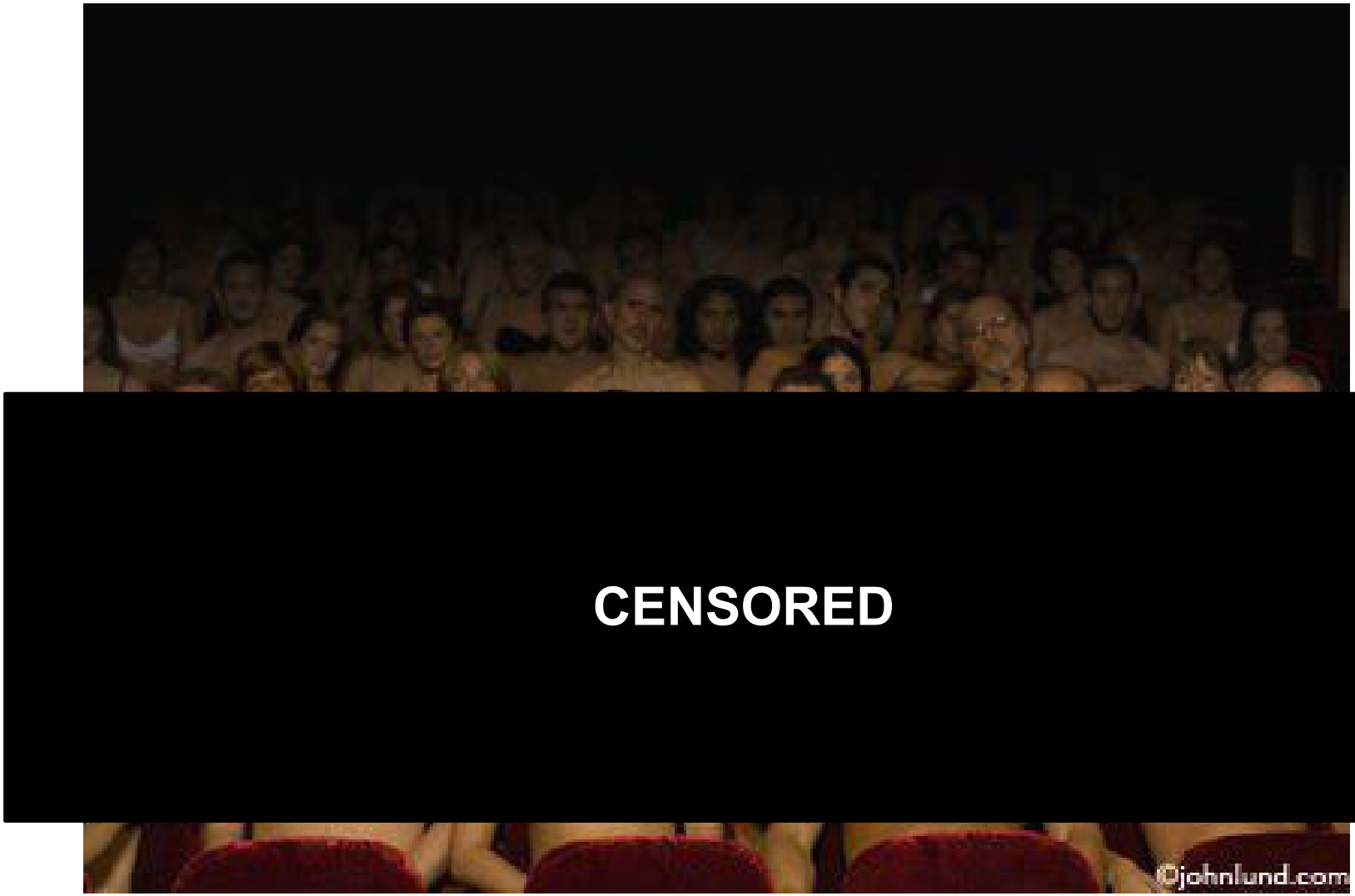
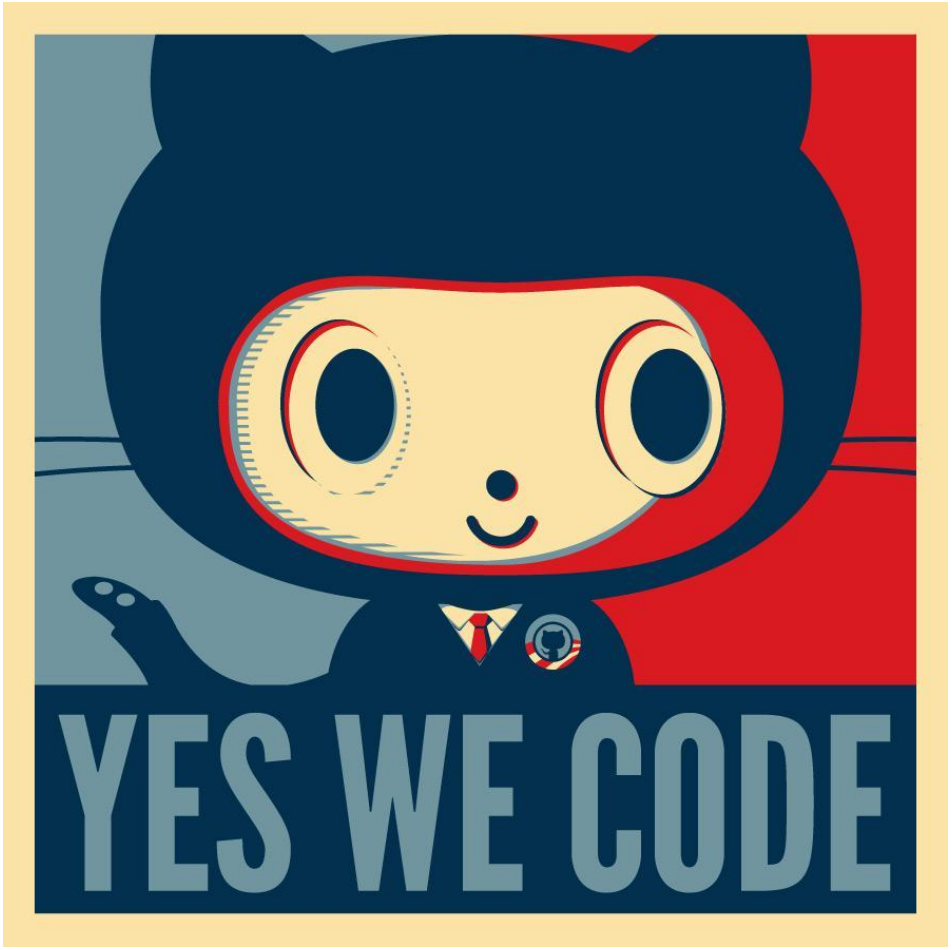
gitDigger

Creating useful wordlists from GitHub

By: WiK & Mubix

We suck at “picturing” things, so in order for this presentation to be successful you must all strip down to your underwear





The Researcher – WiK

@jaimefilson

We weren't the first to go digging…

RSS

SVN Digger - Better Wordlists for Forced Browsing

Forced browsing / finding hidden resources is one of the crucial part of any black-box web application security assessment. There are great tools to accomplish this task, but our favorite is DirBuster. Simple, fast & smart.

DirBuster ships with several wordlists, these wordlists generated via one big crawler which visited tons of websites, collected links and created most common directory / file names on the Internet. This is a really nice approach and DirBuster's wordlists worked much better than any other wordlists out there.

However there is one fundamental problem with these wordlists. Whilst the purpose of these wordlists is finding hidden and not linked resources, ironically they are generated **only** from **known** and **linked resources**. To address this problem we came up with the idea of generating wordlists from open source code repositories. This way it would be possible to see all file/directory names and create much more useful wordlists.

We have extracted the directory structure and file names of many projects from Google Code and SourceForge to prepare a good wordlist for discovering hidden files/folders on a targeted web application.

Numbers

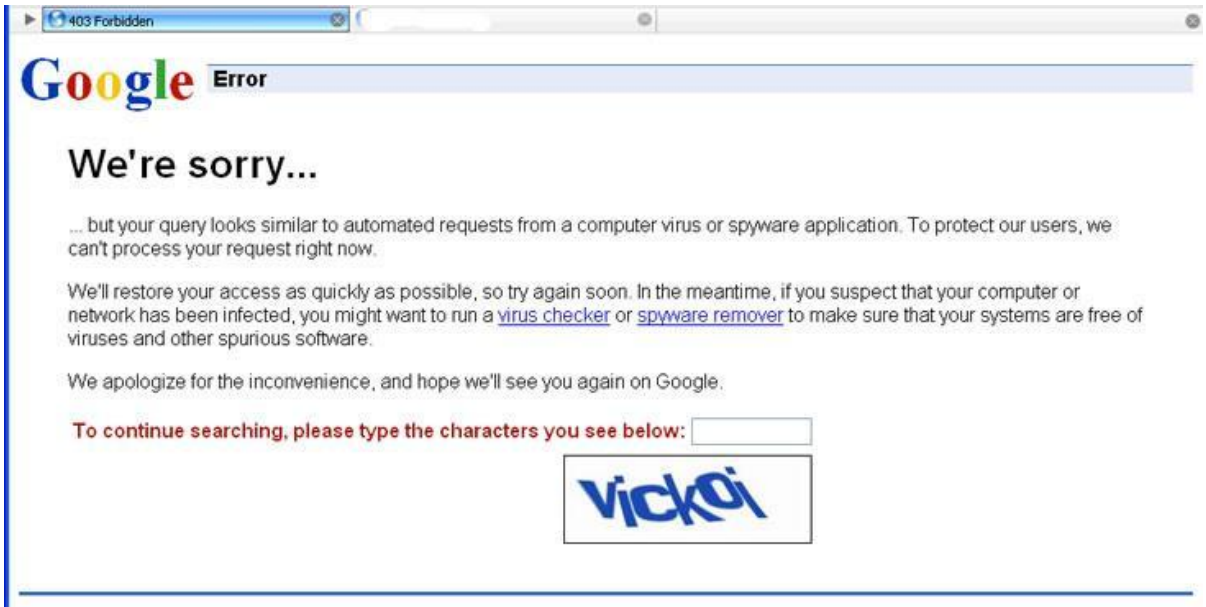
- We have processed over 5000 projects.
- We have more than 400k words at our database.

We have sorted the words according to the their frequency count and prepared some lists based on this data.

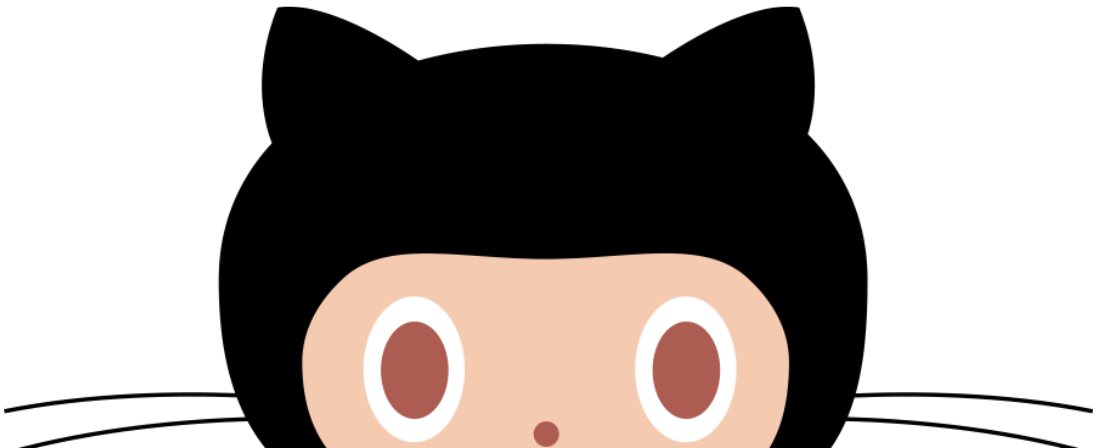
Link to blog post

Only problem is you need to find a service that is “friendly” to “research”

Google™ code



02:09 < pasv> http://www.mavitunasecurity.com/blog/svn-digger-better-lists-for-forced-browsing/
02:11 <@WiK> nice find
02:11 < pasv> i wish i had thought of it
02:16 < mubix> Thats awesome
02:16 <@WiK> ive done similar stuff, now i have a font collection that **10gb of unique fonts**
02:17 < mubix> wish they would add **bitbucket**, and **github** to their searches
02:19 <@WiK> ive looked at scrapin github.. theres no real **good** way to do it



The 30 minute

I CAN DO THIS!

Solution

but there is no “all repos” list…

Most Starred Today

- WordPress / **secure-swfupload**
- pachilo / **TMX_Starling**
- CadetEditor / **CadetEngine-as**
- Esri / **arcgis-viewer-flex**
- gimite / **web-socket-js**

Most Forked Today

- Esri / **arcgis-viewer-flex**
- CadetEditor / **CadetEngine-as**
- mikechambers / **as3corelib**
- nyfelix / **wikimindmap**
- PrimaryFeather / **Starling-Framework**

Most Starred This Week

- CadetEditor / **CadetEngine-as**
- WordPress / **secure-swfupload**
- pachilo / **TMX_Starling**
- gimite / **web-socket-js**
- Esri / **arcgis-viewer-flex**

Most Forked This Week

- CadetEditor / **CadetEngine-as**
- Esri / **arcgis-viewer-flex**
- Esri / **arcgis-samples-flex**
- alamboley / **Citrus-Engine**
- mikechambers / **as3corelib**

All Languages

ABAP

ActionScript

Ada

Apex

AppleScript

Arc

Arduino

ASP

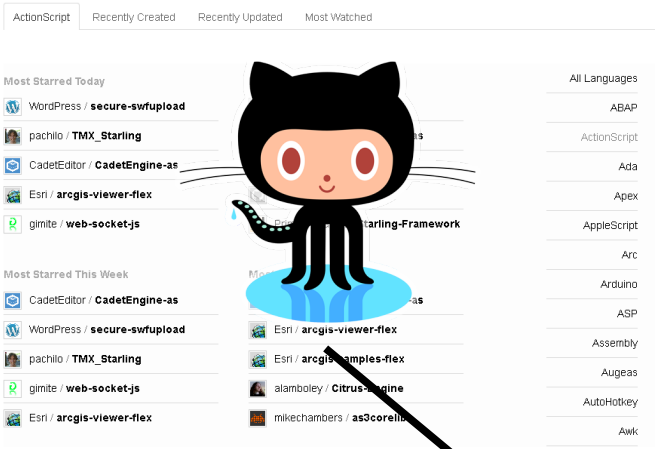
Assembly

Augeas

AutoHotkey

Awk

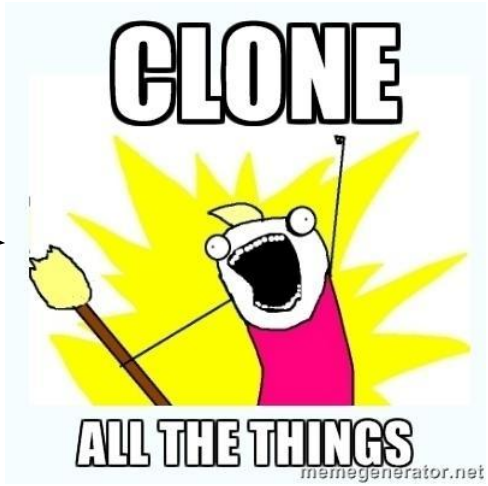
Enter Python WGET



Import os
Import urllib
Import urllib2
Import sqlite3

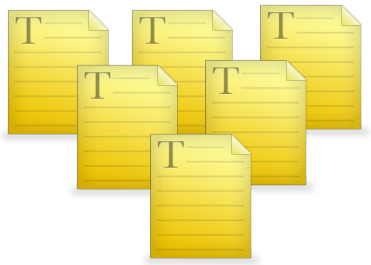
SQLite

Usernames & their repositories



Got it… now what?

Repositories



os.walk()

sort
grep
awk

Lots of manual review and headaches

But finally resulted in

Wordlists

Only the “TOP” repositories

17 hours

of manual kung-fu to process into wordlists

Betterwalk.walk() vs os.walk()

Background

Python's built-in `os.walk()` is significantly slower than it needs to be, because -- in addition to calling `listdir()` on each directory -- it calls `stat()` on each file to determine whether the filename is a directory or not. But both `FindFirstFile` / `FindNextFile` on Windows and `readdir` on Linux/OS X/BSD already tell you whether the files returned are directories or not, so no further `stat` system calls are needed. In short, you can reduce the number of system calls from about 2N to N, where N is the total number of files and directories in the tree.

Benchmarks

Below are results showing how many times as fast `betterwalk.walk()` is than `os.walk()` on various systems, found by running `benchmark.py` with no arguments as well as with the `-s` argument (which totals the directory size).

System version	Python version	Speed ratio	With -s
Windows 7 64 bit	2.6 64 bit	2.5	4.5
Windows 7 64 bit	2.7 64 bit	2.2	4.2
Windows 7 64 bit	3.2 64 bit	3.0	6.2
Windows XP 32 bit	2.7 32 bit	1.3	2.4
Windows XP 32 bit	3.3 32 bit	2.0	4.8
Debian 2.6.32 32 bit	2.6 32 bit	1.6	1.5
Ubuntu 12.04 64 bit VBox	2.7 64 bit	1.5	1.3
Ubuntu 12.04 64 bit VBox	3.2 64 bit	1.7	1.4
Mac OS X 10.7.5	2.7 64 bit	1.6	1.3

Link to project

Initial Thoughts

The Good News

I got the wordlists that I wanted and they were useful

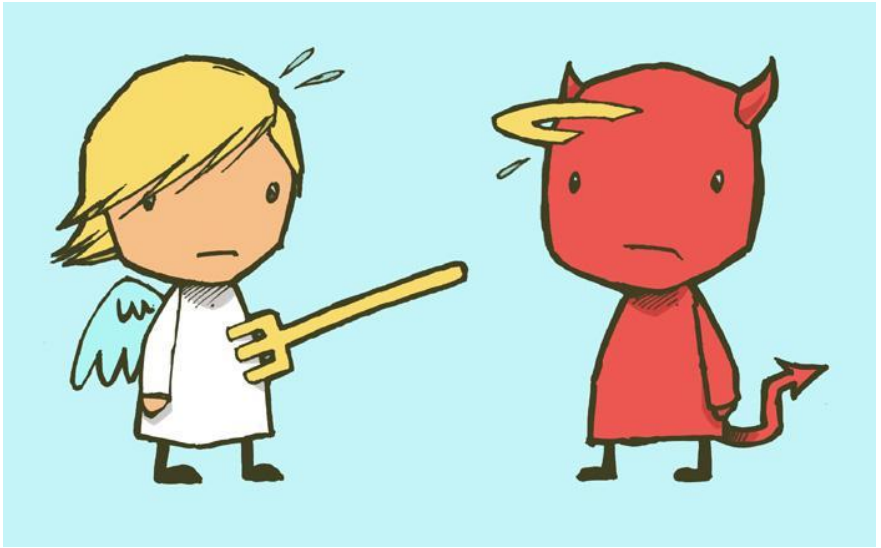
The Bad News

Only the “TOP” repositories

Sqlite3 transactions were **slow**

17 hours of manual labor sucks

My HDD was now full



Lets Get Serious
(well, kinda...)

FIRST PROBLEM: STORAGE

Storage Options

bitcasa
INFINITE STORAGE

Pros
- Cheap ($99 USD a year)
- Built-in “indexing”

Cons
- Windows Only
- Crashes OFTEN
- Encryption == SLOW

Pros
- Central, local, fast storage

Cons
- Expensive

Remember… I’m already at 3 TB…

Solution

SECOND PROBLEM: PYTHON WGET

GITHUB API


develop.github
home
login
signup!
Welcome to the GitHub Developer site
Resources on using the official GitHub API. This site contains documentation on the major API sections and libraries you can use to make use of GitHub with your programs and scripts.
General API Info
How to authenticate as a user, URL schema, secure and unsecured access and access limitations.
Repository API
Searching repositories, getting repository information and managing repository information for authenticated users.
User API
Searching users, getting user information and managing authenticated user account information.
Commit API
Getting information on specific commits, the diffs they introduce, the files they've changed.
Issues API
Listing issues, searching, editing and closing your projects issues.
Object API
Getting full versions of specific files and trees in your Git repositories.
Network API
Listing all the data needed to draw the network graph, heads of every fork with new changes and all relevant commits.
GitHub Libraries
Libraries in various languages that make use of the GitHub API (ruby, python, perl, php).
T-Shirts | Blog | Support | Git Training | Contact | Google Group | Status
GitHub is Logical Awesome ©2009 | Terms of Service | Privacy Policy
This website is open source. Please help us by forking the project and adding to it.

```
"id": 27,
"name": "rubinius",
"full_name": "rubinius/rubinius",
"owner": {
  "login": "rubinius",
  "id": 317747,
  "avatar_url": "https://secure.gravatar.com/avatar/8a664b7c5ca834af3e7e49d3a6160082?d=https://a248.e
  "gravatar_id": "8a664b7c5ca834af3e7e49d3a6160082",
  "url": "https://api.github.com/users/rubinius",
  "html_url": "https://github.com/rubinius",
  "followers_url": "https://api.github.com/users/rubinius/followers",
  "following_url": "https://api.github.com/users/rubinius/following{/other_user}",
  "gists_url": "https://api.github.com/users/rubinius/gists{/gist_id}",
  "starred_url": "https://api.github.com/users/rubinius/starred{/owner}{/repo}",
  "subscriptions_url": "https://api.github.com/users/rubinius/subscriptions",
  "organizations_url": "https://api.github.com/users/rubinius/orgs",
  "repos_url": "https://api.github.com/users/rubinius/repos",
  "events_url": "https://api.github.com/users/rubinius/events{/privacy}",
  "received_events_url": "https://api.github.com/users/rubinius/received_events",
  "type": "Organization"
},
"private": false,
"html_url": "https://github.com/rubinius/rubinius",
"description": "Rubinius, the Ruby Environment",
"fork": false,
"url": "https://api.github.com/repos/rubinius/rubinius",
"forks_url": "https://api.github.com/repos/rubinius/rubinius/forks",
"keys_url": "https://api.github.com/repos/rubinius/rubinius/keys{/key_id}",
"collaborators_url": "https://api.github.com/repos/rubinius/rubinius/collaborators{/collaborator}",
"teams_url": "https://api.github.com/repos/rubinius/rubinius/teams",
"hooks_url": "https://api.github.com/repos/rubinius/rubinius/hooks",
"issue_events_url": "https://api.github.com/repos/rubinius/rubinius/issues/events{/number}",
"events_url": "https://api.github.com/repos/rubinius/rubinius/events",
"assignees_url": "https://api.github.com/repos/rubinius/rubinius/assignees{/user}",
"branches_url": "https://api.github.com/repos/rubinius/rubinius/branches{/branch}",
```

THIRD PROBLEM: SQLITE SUCKS

Solution

PUTTING IT ALL TOGETHER

GitHub Developer

Upgrades

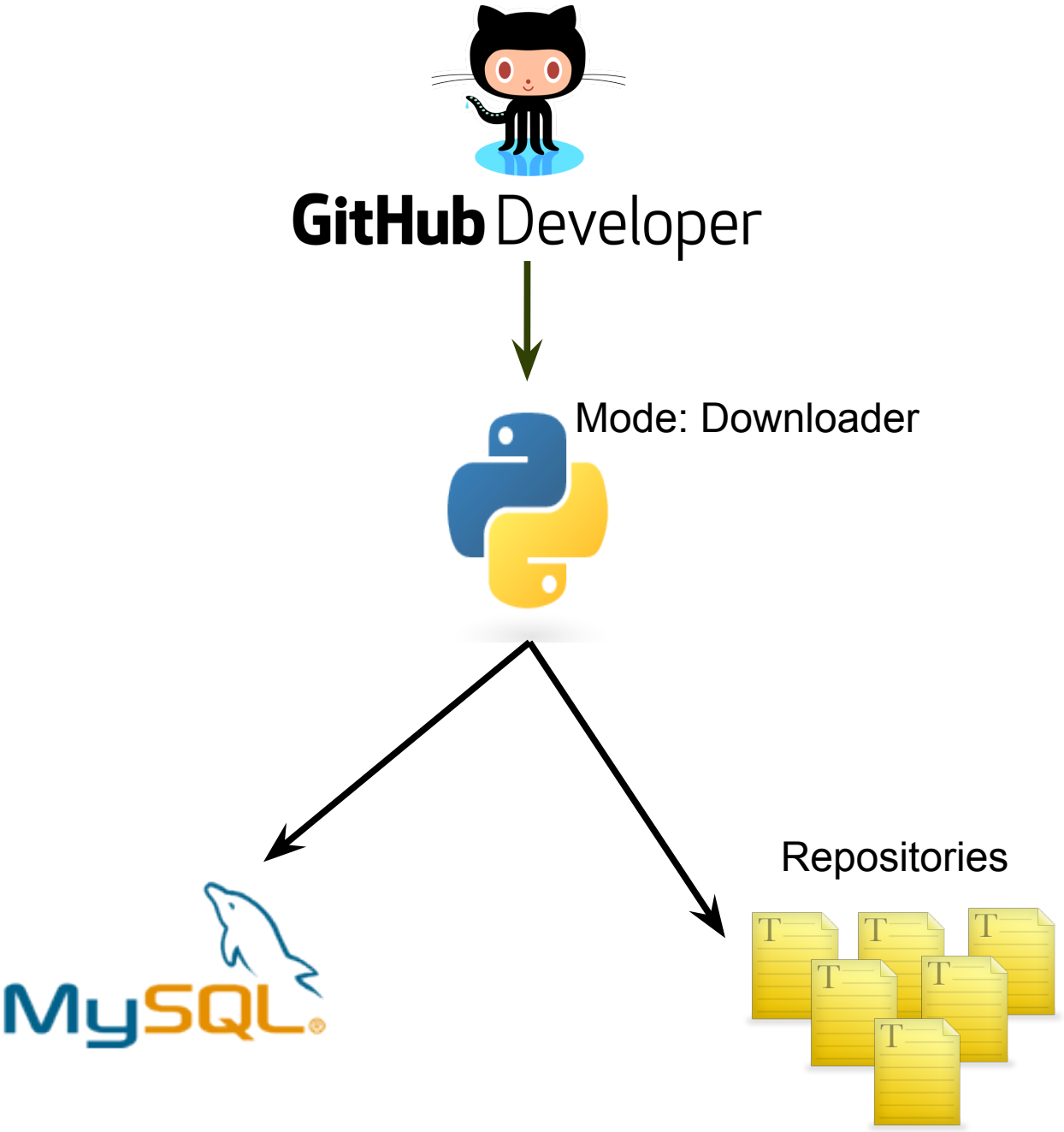
Added 2 modes
Downloader
Processor
Added Threading
Replaced sqlite3 with mysql

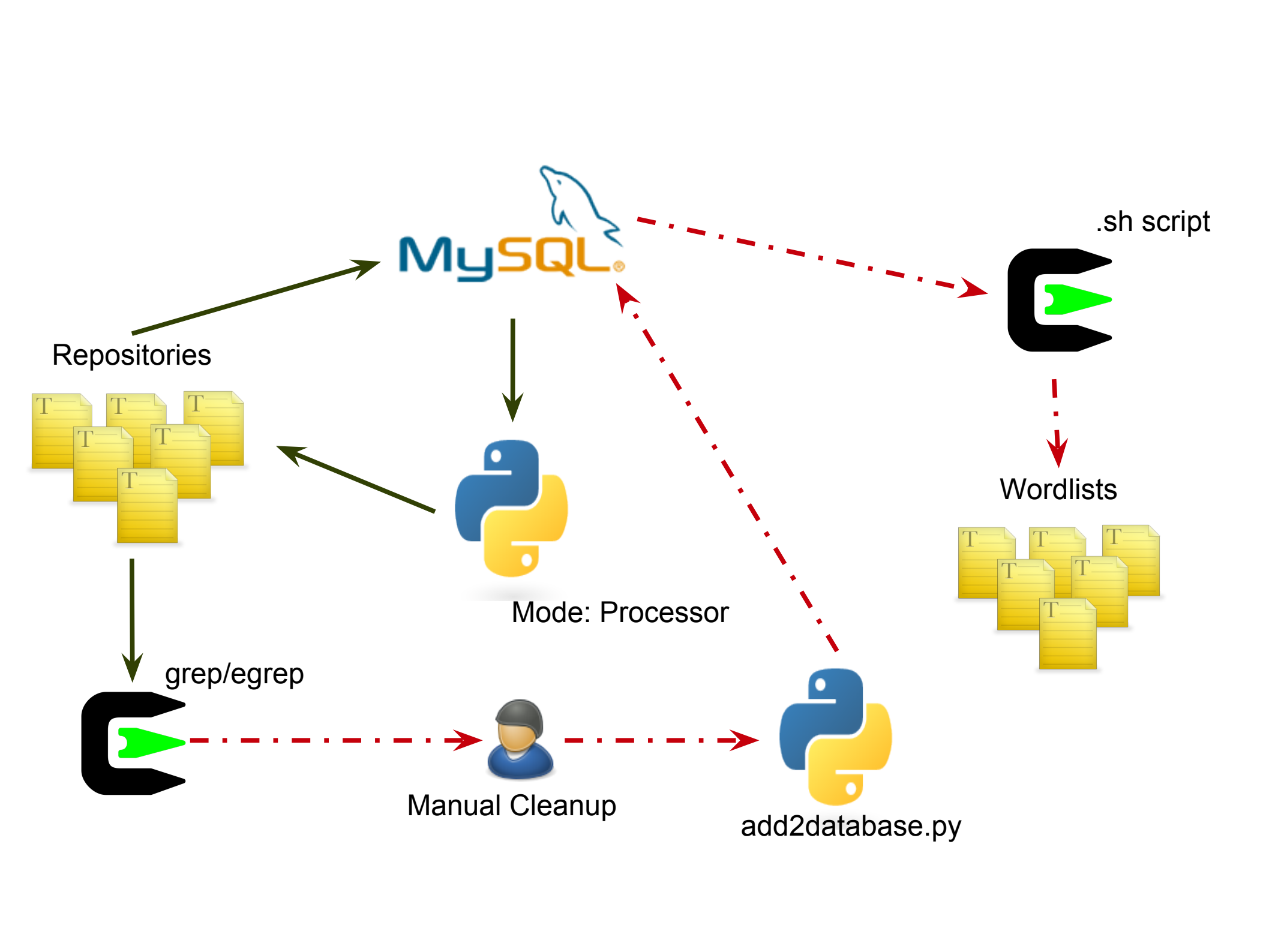
Added

Script to add items to database

Upgrades

- Password Table
  - count
  - name
- Username Table
  - count
  - name
- Email Table
  - count
  - name
- Projects Table
  - name
  - project
  - processed
  - grepped
- Directories Table
- Files Table
- Last Seen ID Table


GitHub Developer
Mode: Downloader
MySQL
Repositories


MySQL
.sh script
Repositories
Wordlists
Mode: Processor
grep/egrep
Manual Cleanup
add2database.py

Updated Results

The Good News

I'm now getting **ALL** public repositories

Generating wordlists now takes automated **minutes** instead of manual hours

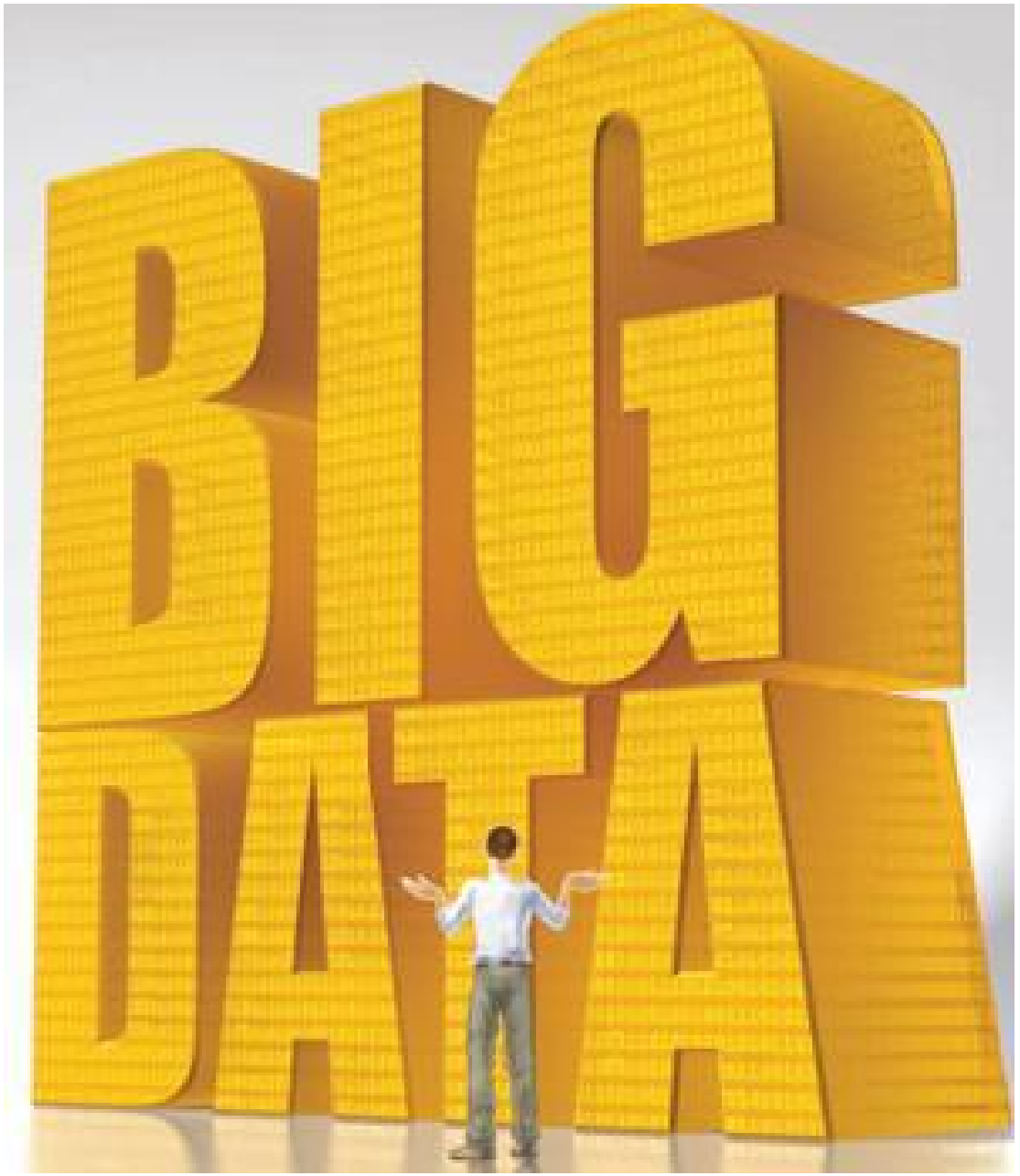
I'm able to store the data over multiple USB HDDs

The Bad News

Carving out data such as usernames, passwords, and emails still requires some manual work which takes up a bunch of time

Huge amount of storage needed
An estimated **30TB uncompressed**

CUE BIG DATA DRINKING GAME



You can buy our product for the low low price of 19.95 per MB, maxing at 1 TB, each additional TB will cost one child or goat. Prices and participation may vary, see your BIG DATA representative at the door for a list of vendors who want to take your money.


THE WORDLISTS
DUN DUN DAHHHHHHHH!
utor.net

all_dirs.txt

751,991	info
686,812	logs
645,023	lib
555,954	src
490,724	test

all_files.txt

846,524	README
683,848	index.html
408,574	ChangeLog
307,197	README.txt
132,053	license.txt

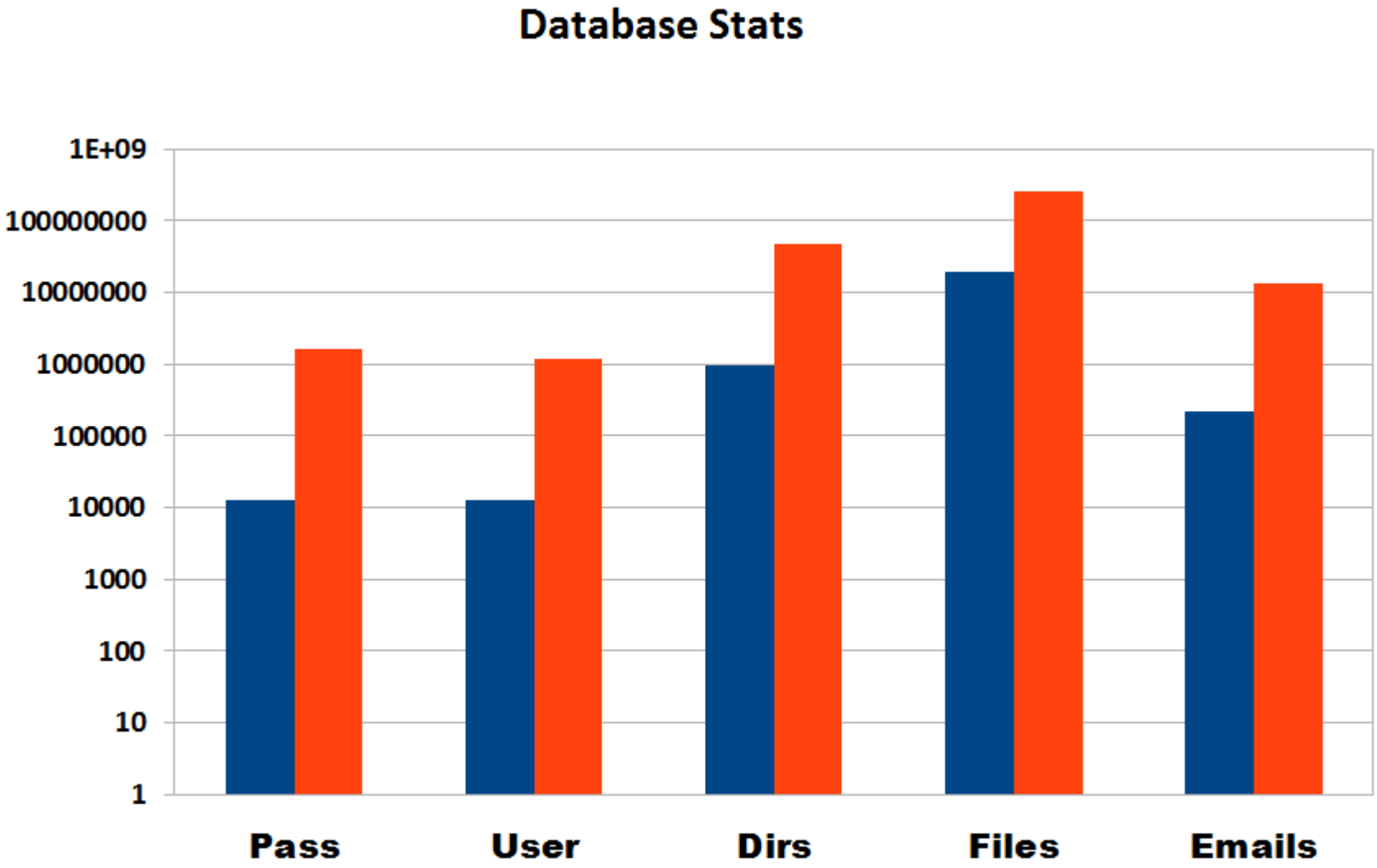
usernames.txt

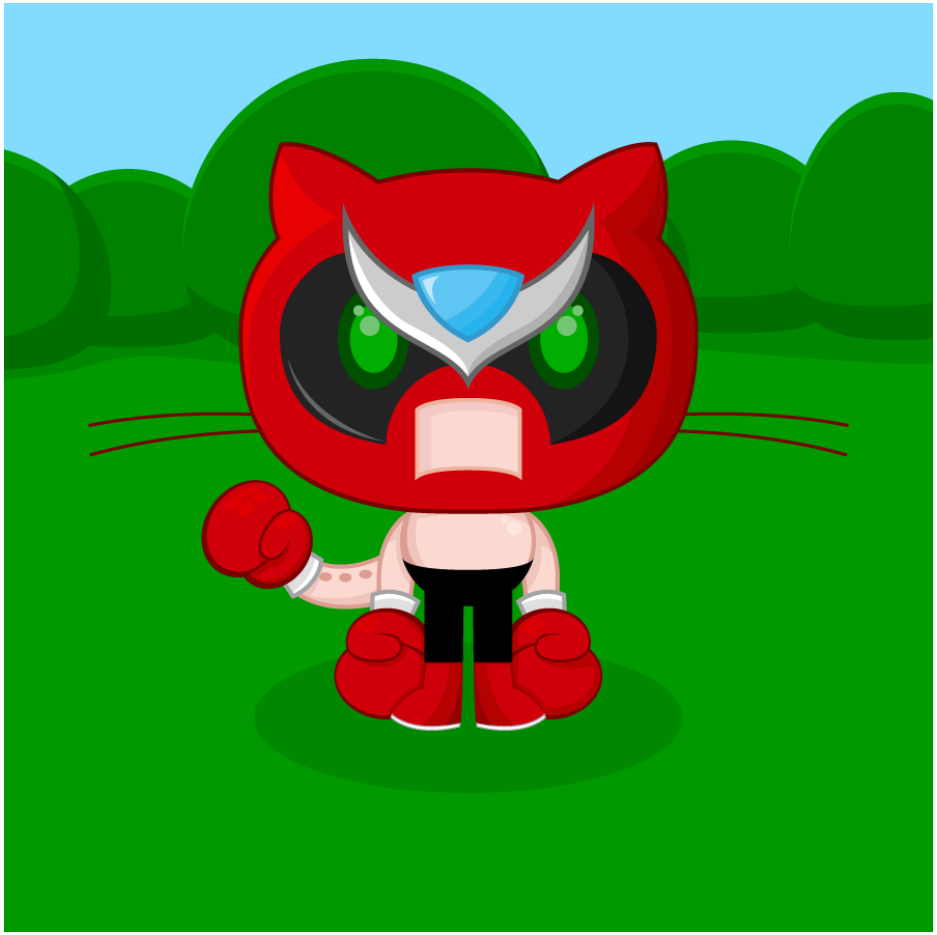
166,997	username
75,794	bob
72,360	users
59,595	admin
45,522	user
38,024	name
29,799	rails
25,853	sa
22,981	root
21,293	test

passwords.txt

358,949	password
118,287	foobar
75,567	test
53,238	secret
35,842	user

Link to wordlists


Database Stats
1E+09
100000000
10000000
1000000
100000
10000
1000
100
10
1
Pass
User
Dirs
Files
Emails



The Attacker – Mubix

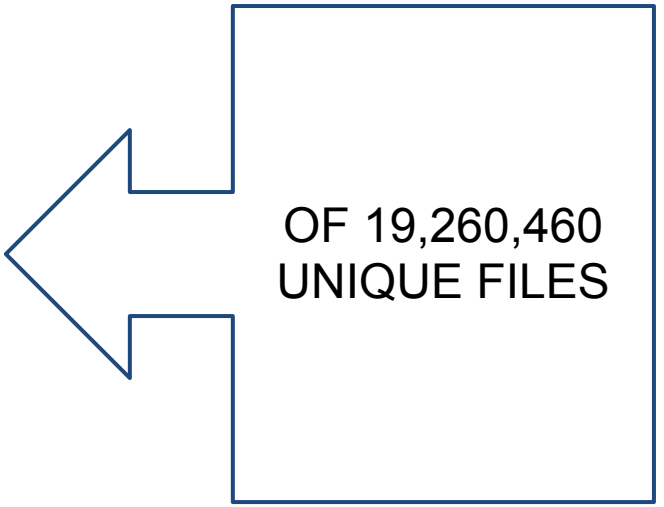
@mubix

the Strongbadtocat by 

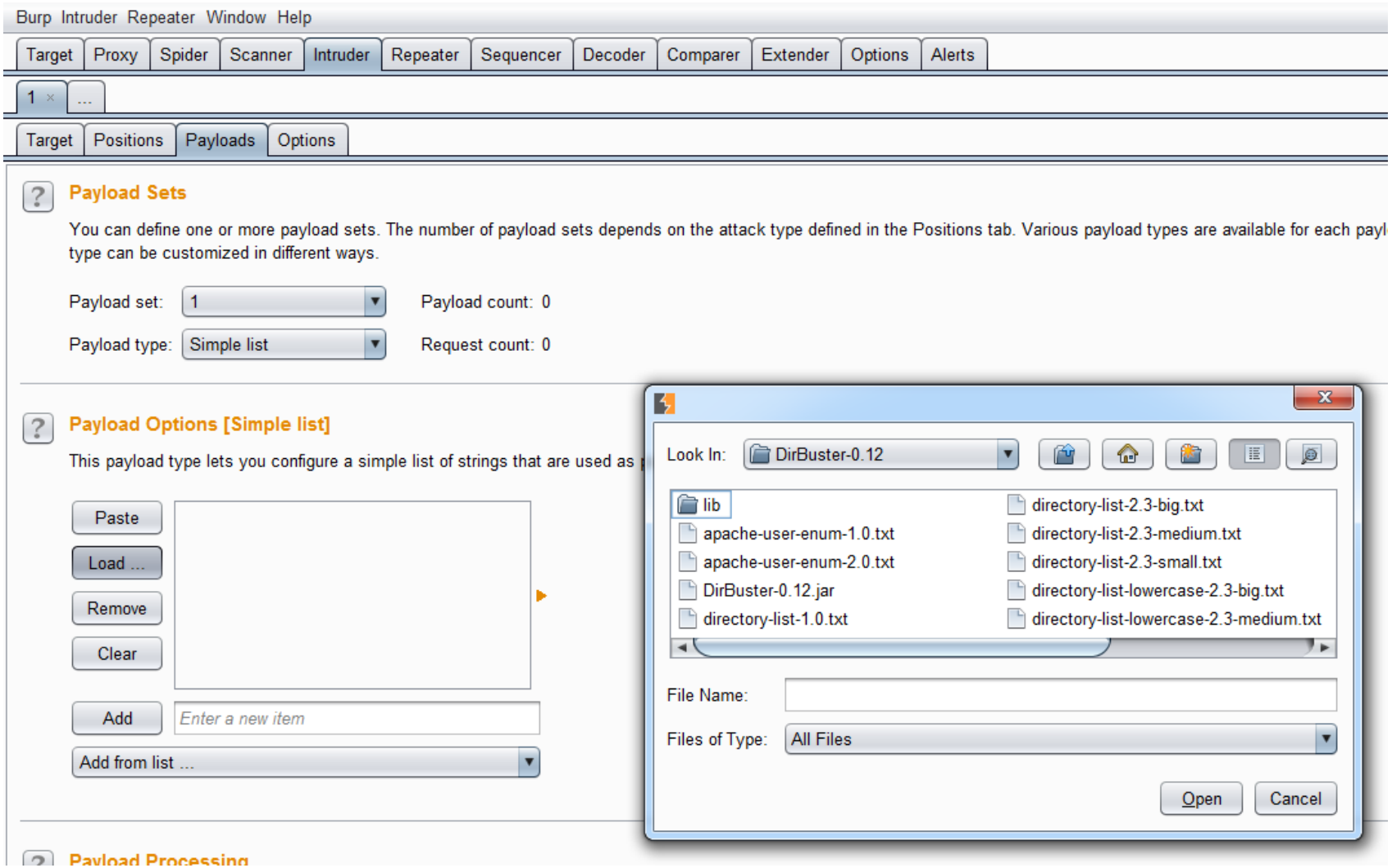

This is relevant to my interests.

The obvious stuff...

- Wordlists for "forced" browsing as with the SVN digger project
- Small default passwords list
- static_salts.txt: static salts found within github projects.
- #22 file "Exception.php"
- #323 is "file.php"
- #4819 is password.txt (wtf?)



Burp


Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts
1 ...
Target Positions Payloads Options
Payload Sets
You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payl
type can be customized in different ways.
Payload set: 1
Payload count: 0
Payload type: Simple list
Request count: 0
Payload Options [Simple list]
This payload type lets you configure a simple list of strings that are used as
Paste
Load ...
Remove
Clear
Add
Enter a new item
Add from list ...
Payload Processing
Look In: DirBuster-0.12
lib
apache-user-enum-1.0.txt
apache-user-enum-2.0.txt
DirBuster-0.12.jar
directory-list-1.0.txt
directory-list-2.3-big.txt
directory-list-2.3-medium.txt
directory-list-2.3-small.txt
directory-list-lowercase-2.3-big.txt
directory-list-lowercase-2.3-medium.txt
File Name:
Files of Type: All Files
Open
Cancel

The obvious stuff...

- #370848 ssh1_auth_keys
- #185308 ntlmsso_magic.php

keeps going... too much fun.. but how real world is this stuff?

coulda woulda shoulda.. SO WHAT!

http://www.phenoelit.org/stuff/tokenfaq.html

phenoelit.org

Oh no my "secret_token" is on GitHub!

What does this mean?

The "secret_token" is used as a key to cryptographically ensure that no one tampers with your Rails apps' session cookies. So if this token is exposed on the public Internet, attackers are able to forge session cookies which are valid within your Rails application. This does not only mean that likely an attacker can impersonate any user of your application. If the bad guy is a bit more fancy she (or he) can craft some session cookies which will make the Rails app execute arbitrary code. This attack applies if you use the standard Ruby on Rails session cookies. If you are using ActiveRecord sessions, you are most likely fine =).

What should I do?

You should replace the "secret_token", obviously. In order to keep the new token secret this time, you could do the following (code stolen from Gitlabhq's config/initializers/secret_token.rb):

```
# Be sure to restart your server when you modify this file.

require 'securerandom'

# Your secret key for verifying the integrity of signed cookies.
# If you change this key, all old signed cookies will become invalid!
# Make sure the secret is at least 30 characters and all random,
# no regular words or you'll be exposed to dictionary attacks.

def find_secure_token
  token_file = Rails.root.join('.secret')
  if File.exist? token_file
    # Use the existing token.
    File.read(token_file).chomp
  else
    # Generate a new token of 64 random hexadecimal characters and store it in token_file.
    token = SecureRandom.hex(64)
    File.write(token_file, token)
    token
  end
end

YOUR_RAILS_APP::Application.config.secret_token = find_secure_token
```

Where YOUR_RAILS_APP should be adjusted accordingly. Second step would be to exclude the file ".secret" form your Git repository by listing it in your ".gitignore" file. If your Rails app has a larger user base, you should think about issuing a security advisory in order to alert your users about this issue.

coulda woulda shoulda.. SO WHAT!

The not so obvious

- Starting to parse every file from the git revision history (*thought you removed that default password did ya?*)
- Mass static code analysis for vulnerabilities
- One of the top directories is ".svn", another is ".settings" ;-)
- Parsing .gitignore of production targets
- Verify directories w/ HTTP 403 on . empty_directory and .DS_Store files

The not so obvious

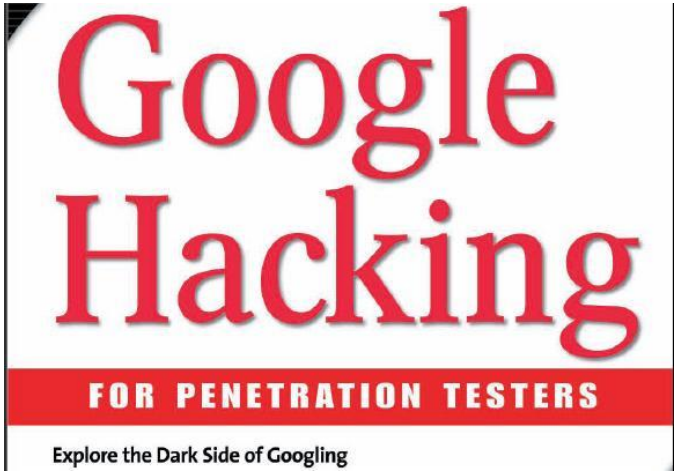
- Run OCR on all image files
- Using list of .txt files for intelligence gathering
- Grep out ALL email addresses

STOP you're just giving them ideas

iOS

INTERNET ARCHIVE
WayBackMachine

Google™




THE
END

Thank You!

http://github.com/wick2o/gitdigger

wick2o@gmail.com
@jaimefilson

Link to wordlists