Confessions of a lactose intolerant

late night coding, why cheese hates me and vulnerabilities!

Things to talk about ...

Shell Scripts	Docker	SQL
API	Ruby	Scan Tools

The story of fon-diddly-do

Once upon a time...

The charge of developers

Awareness

Educate

Ownership

The tool
fon-diddly-do

App Runner

Ruby on Rails API

PostgreSQL

The Cloner

Clone your targets

Github API

curl

wget

… others

The Gatherer

Resolve Application Services

Language	Framework	Database
Caching	Background Processing	Versions

The Gatherer

Generate a Dockerfile

```
1  FROM ruby: <<RUBY_VERSION>>
2
3  RUN apt-get update -yqq  \
4    && apt-get install -yqq –no-install-recommends \
5      build-essential \
6      libqp-dev \
7      Nodejs
8
9  WORKDIR /demo_app
10 COPY Gemfile* ./
11 RUN bundle install
12 RUN gem install brakeman
13 COPY . .
14
15 CMD bundle exec unicorn -p 8081 -c   .config/.unicorn.rb
```

The Gatherer

Generate docker-compose.yml

```
1  version: “2”
2  volumes:
3    database-data:
4      external: false
5  services:
6    cache:
7      image: <<CACHE_IMAGE>>
8    redis:
9      image: <<REDIS_IMAGE>>
10   database:
11     image: <<DATABASE_IMAGE>>
12     volumes:
13       - database-data:/demo_data
```

```
14 demo_app:
15   env_file:  .demo_env
16   build:  .
17   volumes:
18     - . :/demo_app
19   ports:
20     - “8081:8081”
21   depends_on:
22     - database
23     - redis
24     - cache
```

The Gatherer
Copy generic configs

```
1  default: &default
2    adaptor: <<DB_ADAPTOR>>
3    host: <%= ENV[“DB_HOST”]%>
4    username: <=% ENV[“DB_USER”]%>
5    password: <=% ENV[“DB_PWD”]%>
6    encoding: utf8
7    min_message: warning
8    tool: 2
9    timeout: 5000
10
11  development:
12    <<: *default
13    database: <<DB_NAME>>
```

```
1  DB_HOST=localhost
2  DB_PORT=5432
3  DB_USER=db_user
4  DB_PASSWORD=db_pwd
5  REDIS_USER=rd_user
6  REDIS_PASSWORD=rd_pwd
7  SECRET_KEY_BASE=dev_secret_key
8  etc ...
```

The Gatherer

Build it up, scan it, tear it down

```
docker-compose up -d ...
```

```
scan-target (with opts ...)
```

```
docker-compose down
```

```
rm -rf ~/repos/demo_app
```

The Scanner
Burp

Automated

Driven from the command line

Run scanner with arguments

Generate well formed reports

The Scanner
Extending Burp

```
1  class BurpExtender
2    Include IBurpExtender, IHttpListner, IScannerListner
3
4    def registerExtenderCallbacks(callbacks)
5      # implement IBurpExtender
6      @callbacks = callbacks
7    end
8
9    def cmd_line_args
10     params = @callbacks.getCommandLineArguments
11
12     @target = params[0]
13     @format = params[1]
14     @name = params[2]
15   end
16
17   def processHttpMessage(flag, request, message)
18     # implement IHTTPListener
19   end
20
21   def newScanIssue(issue)
22     # implement IScannerListener
23   end
```

The Scanner
Scripting Burp

```
java -Xmx4g -Djava.awt.headless=true -jar ~/lib/burpsuite.jar
```

```
vim ~/bin/scan-target
#!/bin/sh
umask 022
PATH="/bin:/sbin:/usr/bin:/usr/sbin:$HOME/bin"
java -Xmx4g -Djava.awt.headless=true -jar ~/lib/burpsuite.jar $1 $2 $3
# add other script tasks ...
```

The Scanner
Scripting Burp

```
scan-target http://demo_app:8081 xml /tmp/demo_app.xml
```

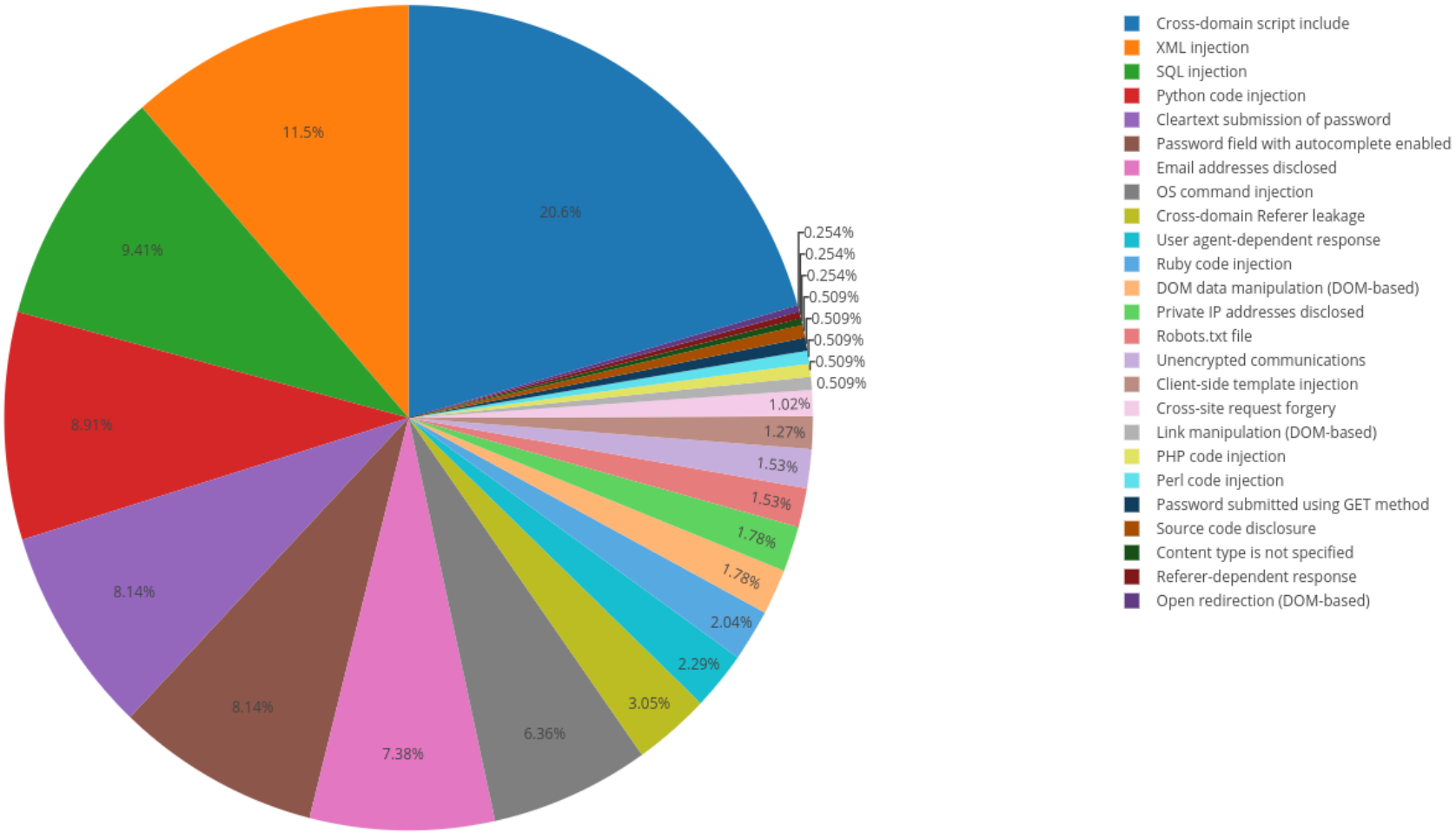
The Importer

Rake tasks for data import

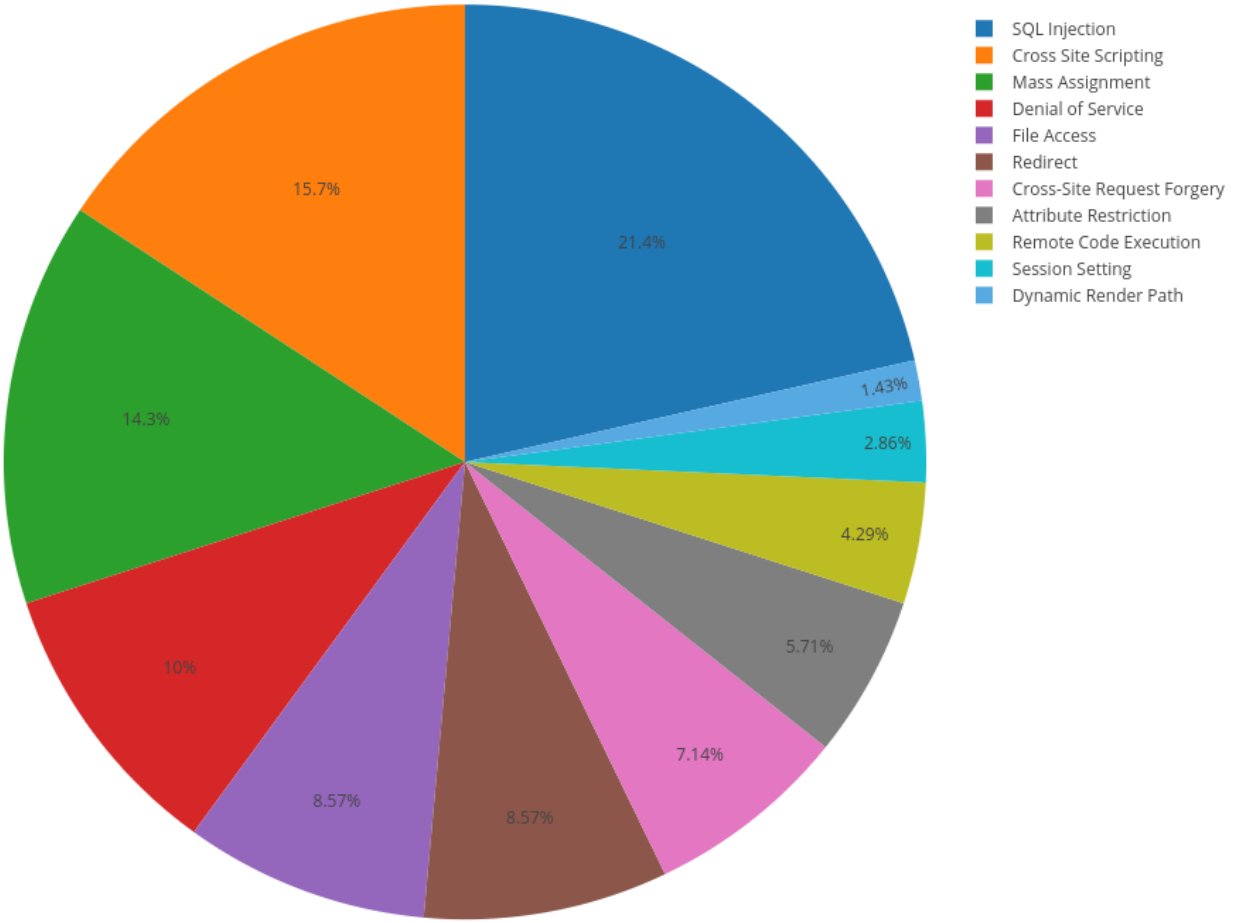
```
bundle exec rake "import:burp[demo_app.xml, demo_app,
      rails, 4.2.8, https://demo, demo@demo]"
```

```
bundle exec rake "import:brakeman[demo_app.xml,
demo_app, rails, 4.2.8, https://demo, demo@demo]"
```

The Analysis
Organise, prioritize and fix



The Analysis
Organise, prioritize and fix



The Analysis
Materialized Views

Latest web scan

Previous web scan

Latest static scan

Previous stat scan

Further thoughts
Future development

- Improve gathering process
- Gatherers for different frameworks
- Interface for different scan-tools
- Javascript reporting server
- Scanning git history
- Vulnerability mitigation

The End

https://github.com/redshieldsecurityltd

Questions & Suggestions

appreggios.from.the.kitchen@gmail.com