Web Server Security

with Apache/PHP

Adam Arrowood (adam.arrowood@oit.gatech.edu)

John Douglass (john.douglass@oit.gatech.edu)

Michael Mealling (michael.mealling@oit.gatech.edu)

Assumptions

- Familiarity with Apache
- Class with focus on Apache on Unix systems, most commonly on RedHat Linux
- Familiarity with PHP

Overview

- Patching 1, 2, 3
- Protecting
- Knowing
- Install/Configure Apache, SSL
- ModSecurity
- Coping with DOS
- Install/Configure PHP
- Secure PHP programming tips
- Suhosin

Patching I

- **Patch your Operating System**
  - automate alerts
  - have a schedule
  - prepare users, authors, etc. for OS patch-related downtime

Patching 2

- **Patch your product(s)**
  - out-of-date software can be full of known, advertised holes
  - follow product communities, automate checking for new versions
  - prepare users, authors, etc. for software-related downtime

Patching 3

- **Patch your product's libraries/plug-ins**
  - A secure up-to-date product can be undone by out-of-date modules
  - Custom written code? all of it? if not, patch your included libraries (e.g. jQuery, YUI, etc.)
  - follow product communities, automate checking for new versions

Protecting

- Firewalls
  - network-based (PIX)
  - host-based (iptables)
  - server-based (Apache config, ModSecurity, Suhosin)
  - application-based
- Limit dev, test servers to on-campus/VPN
- Walk the tightrope of convenience vs. security when granting access
- Allowing anonymous/self-registered authoring without moderation or anti-spam tools will lead to comment spam.

Knowing

- *Know* your site
- *Google* your site
  - What does [crawler] know about your site?
    - Is your site selling Viagra? Does Google/Bing/etc. think it does?
- *Clean* your site... remove tar files, sql dumps, etc.
- Watch your logs and stats (you are running a web statistics package, right?)

Apache

- Why Apache? Why not?
- Apache installation/configuration is a part of your web site/application's security
- Basic (stock) configuration will get you far, but that "far" is getting shorter
- Many hardening guides available (e.g. cisecurity.org Apache security benchmark)

Apache: Installing

Apache 2.2 can be installed via two methods:

- Vendor supplied packages
  - works, all in one, including modules
  - customized for your OS
  - tested, QA'd
  - vendor is "on the hook" for monitoring, fixing security issues, might be automatic, easy
- Compiling apache src
  - out-of-the-box often less "full-featured" than src dist
  - "only if you have to"

Apache: Modules

- Unused modules
  - comment out “LoadModule” line in httpd.conf, restart; some modules may be a package that can be uninstalled
  - Better to disable what you don’t use
  - Can be trial-and-error, helped by product, modules “requirements”
  - Disabling some modules means editing many more parts of the default httpd.conf

Apache: Modules

- Unused modules... popular candidates:
  - DAV
  - mod_info
  - Autoindex
  - Proxy
  - UserDir
  - mod_include

Apache: Configuration

- Logging
  - Make sure log directory is root access only
  - Set LogLevel to notice
  - Use CustomLog (combined) to log UserAgent
  - Rotate your logs (/etc/logrotate.d/httpd file)
  - Process your logs (e.g. stats: awstats, IDS: OSSEC)

Apache: Configuration

- Logging to syslog
  - ErrorLog syslog:local7
  - AccessLog is a little more complex:
    CustomLog "|/usr/bin/logger -p local7.info access_syslog
  - Advantages
    - centralized
    - off the box
      - can’t be modified
      - won’t fill up webserver
  - Disadvantages
    - could be unreliable
    - can be faked, network must exert security
    - clear-text across the network
  - Explore alternatives to standard syslog (e.g. syslog-ng)

Apache: Configuration

- Permissions, Content
  - Run apache as separate, non-nobody user & group, no shell, locked password
  - Root ownership of apache directories (except content directories of DocRoot, cgi-bin)
  - Deny / directory, allow only DocRoot, cgi-bin

    <Directory />
    Options None
    AllowOverride None
    </Directory>
  - Delete default DocRoot, cgi-bin content
  - Don’t serve apache icons (not needed if not using autoindex)
    # Alias /icons/ "/var/www/icons/"

Apache: Configuration

Network/HTTP

- multiple interfaces on the box? Listen to only those you need
- Turn off HTTP Trace:
  TraceEnable off
- Only serve correctly named files:

```
# Block all files by default, unless specifically allowed.
<FilesMatch "^.*$">
   Order Deny,Allow
   Deny from all
</FilesMatch>

# Allow files with specifically approved file extensions
<FilesMatch "^.*\.(php|css|html?|js|pdf|txt|xml|xsl|gif|ico|jpe?g|png)$">
   Order Deny,Allow
   Allow from all
</FilesMatch>
```

Apache: Configuration

Network/HTTP (con't)

- Limit HTTP Request methods:

```
<Directory "/var/www/html">
  ...
  Order allow,deny
  <LimitExcept GET POST OPTIONS>
    deny from all
  </LimitExcept>
</Directory>
```

- Deny all non-HTTP 1.1 traffic:

```
RewriteEngine On
RewriteCond %{THE_REQUEST} !HTTP/1\.1$
RewriteRule .* - [F]

# in each vhost section
RewriteEngine On
RewriteOptions Inherit
```

- Set ServerTokens to Prod, ServerSignature to Off

Apache: SSL

Why do you need SSL?

- secure information submission/distribution
- authentication ... (consider GT Login?)
- authorized session protection (think FireSheep)
- assertion of identity

Apache: SSL

- Use a "real" certificate (Verisign, GeoTrust, etc.) or a GTCA signed cert (request via https://ca.gatech.edu/server)
- use SSLv3 and TLSv1 only:
  SSLProtocol all -SSLv2
  SSLCipherSuite ALL:!EXP:!NULL:!ADH:!LOW:!SSLv2:!MD5:!RC4
- Make sure SSLInsecureRenegotiation is not set or is set to off
- SSL key file
  - make sure permissions are correct (root owned, 0400)
  - 2048-bit key recommended

Apache: ModSecurity

ModSecurity (www.modsecurity.org) is an open source Web Application Firewall (WAF) that can be installed in Apache as a standard module and when configured with appropriate rules, can provide protection from a range of attacks against web applications and allow for HTTP traffic monitoring, logging and real-time analysis.

Apache: ModSecurity

ModSecurity is made up of four projects:

- mod_security Apache module
- ModSecurity Core Rule Set (CRS), a set of predefined general rules for mod_security that turns it into an IDS ...now part of The Open Web Application Security Project (OWASP)
- ModSecurity Console, a network-based console designed to collect logs and alerts from remote ModSecurity sensors in real-time
- ModProfiler, use logs to generate mod_security rules

Apache: ModSecurity

ModSecurity key features:

- Request filtering to reject or clean requests *before* they are processed by the target apache handler
- Output filtering to reject or clean the output of a request *after* the target apache handler
- Interception and vetting of uploaded files
- Audit logging of the full request (headers and POST content)
- Configurable via rules and actions in the apache config file(s)
- Can be run on each server or as a proxy WAF

Apache: ModSecurity

ModSecurity can be used to implement:

- **Negative Security model:** monitors requests for anomalies, unusual behavior, and common web application attacks. It keeps anomaly scores for each request, IP addresses, application sessions, and user accounts. Requests with high anomaly scores are either logged or rejected altogether.
- **Positive security model:** only requests that are known to be valid are accepted, with everything else rejected. This model requires knowledge of the web applications you are protecting.
- **Extrusion Detection model:** ModSecurity can also monitor outbound data and identify and block information disclosure issues such as leaking detailed error messages or Social Security Numbers or Credit Card Numbers.
- **Known weaknesses and vulnerabilities:** ModSecurity an ideal external patching tool. External patching (sometimes referred to as Virtual Patching) is about reducing the window of opportunity. Applications can be patched from the outside, without touching the application source code (and even without any access to it), making systems secure until a proper patch can be applied.

Apache: ModSecurity

Negative Security examples

```
SecRuleEngine On

# reject all requests with ‘viagra’ or ‘cialis’ in the HTTP Referer field of the request
SecRule REQUEST_HEADERS:Referer "viagra|\bcialis\b" deny,log,status:400',phase:1

# redirect all requests with any arguments containing matching < >’s (tags)
SecRule ARGS "<(.|\n)+>" phase:2,redirect:http://www.gatech.edu/error

# block all requests but GET, POST, and HEAD
SecRule REQUEST_METHOD "!^(?:GET|POST|HEAD)$" phase:1,log,deny,status:400

# block requests with no Host: header
SecRule &REQUEST_HEADERS:Host "@eq 0" skip:1,log,deny,status:403
SecRule REQUEST_HEADERS:Host "^$" log,deny,status:403
```

Apache: ModSecurity

Positive Security example

```
SecRuleEngine On
<Location /user_view.php >
# This script only accepts GET
SecRule REQUEST_METHOD "!^GET$" phase:2,log,deny

# Accept and require only one parameter: id
SecRule &ARGS !^1$ phase:2,log,deny,skip:1
SecRule ARGS_NAMES "!^id$" phase:2,log,deny

# Parameter id is mandatory, and it must be # a number, 4-14 digits long
SecRule ARGS:id "!^[[:digit:]]{4,14}$" phase:2,log,deny

</Location>
```

External Libraries

Client-side inclusion of external libraries (e.g. Google Analyitcs, google-hosted jQuery) is risky:

- relies on client DNS to resolve to legit host
- usually not over a secure channel
- relies on someone else's web server security

Solutions depend on your level of paranoia:

- make sure to use ssl transport for external links
- proxy back through yourself to external links
- proxy back through yourself to audited, updated copies of external libraries (http(s)://gac.gatech.edu/ga/ga.js)

Additional Host Based Security

Firewalling

Using IPtables

- You can use the "connlimit" module in iptables to limit the number of parallel TCP connections to a server per client IP (or address block)

```
iptables -A INPUT -d $ip -p tcp --dport 80 --
syn -m connlimit --connlimit-above 20 -j DROP

iptables -A INPUT -d $ip -p tcp --dport 443 --
syn -m connlimit --connlimit-above 20 -j DROP
```

Using Apache Module mod_evasive

- You can download source from:
  http://zdziarski.com/blog/?page_id=442
- Install into your Apache installation and modify your httpd.conf like so:

```
<IfModule mod_evasive20.c>
    DOSHashTableSize    3145739
    DOSPageCount        10
    DOSSiteCount        10
    DOSPageInterval     1
    DOSSiteInterval     1
    DOSBlockingPeriod   20
    DOSSystemCommand    "/usr/local/bin/ip_block.sh %s 2"
</IfModule>
```

Additional Scripts for mod_evasive

File: /usr/local/bin/ip_block.sh

```
#!/bin/bash
sudo /sbin/iptables -I INPUT -s $1 -j DROP
echo "/usr/local/bin/ip_unblock.sh $1" | at now + $2 minutes
logger -p local2.notice "ip_block: mod_evasive blocked $1 for $2 minutes"
```

File: /usr/local/bin/ip_unblock.sh

```
#!/bin/bash
sudo /sbin/iptables -D INPUT -s $1 -j DROP
logger -p local2.info "ip_block: mod_evasive unblocked $1"
```

NOTE: In order for this to work:

1. Your apache user MUST have SUDO access to “iptables” and “logger”

2. The apache user must also have a shell (not /bin/no-login in /etc/passwd)

3. In /etc/sudoers “requiretty” must be commented out

Zend Server

Who, What, Why, How?

Who is Zend?

- Founded by Andi Gutmans and Zeev Suraski
- Provides leadership for PHP
- Promotes PHP by building tools, promoting the language, offering training
- http://www.zend.com/

Why Use Zend Server instead of RHEL PHP?

- Until recently, PHP 5.3 wasn't available (PHP 5.3.1 ships with RHEL6)
- Get most often needed modules (MySQL, Oracle, LDAP, etc) by default
- Provides engine features such as Zend Optimizer+ (caching) and provides additional functions to assist IN data caching
- Web Based PHP administrator console

Installing Zend Server

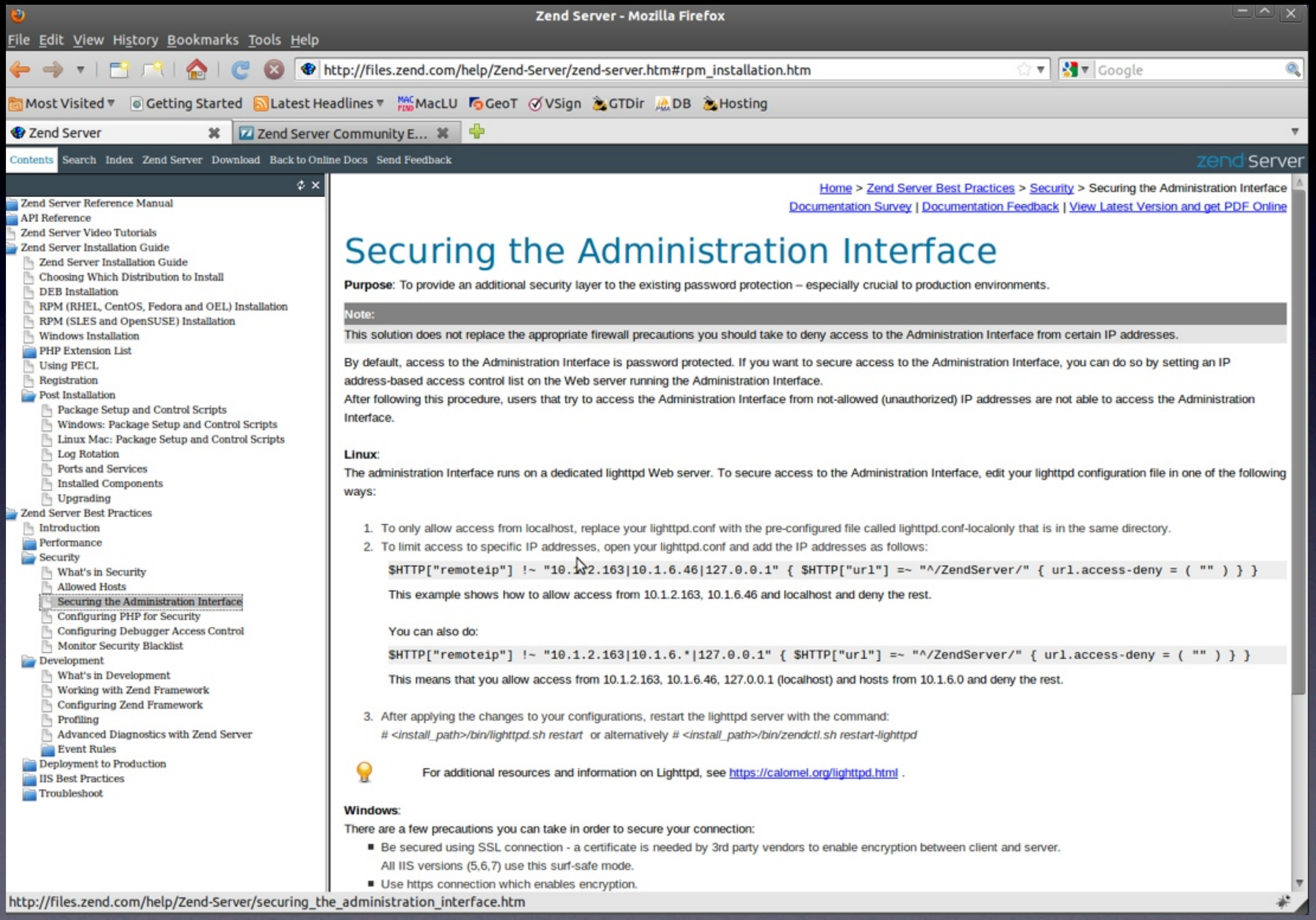
- Uninstall any RHEL PHP versions (conflict)
- Zend Server PE is fine to run without a license

- http://www.zend.com/server

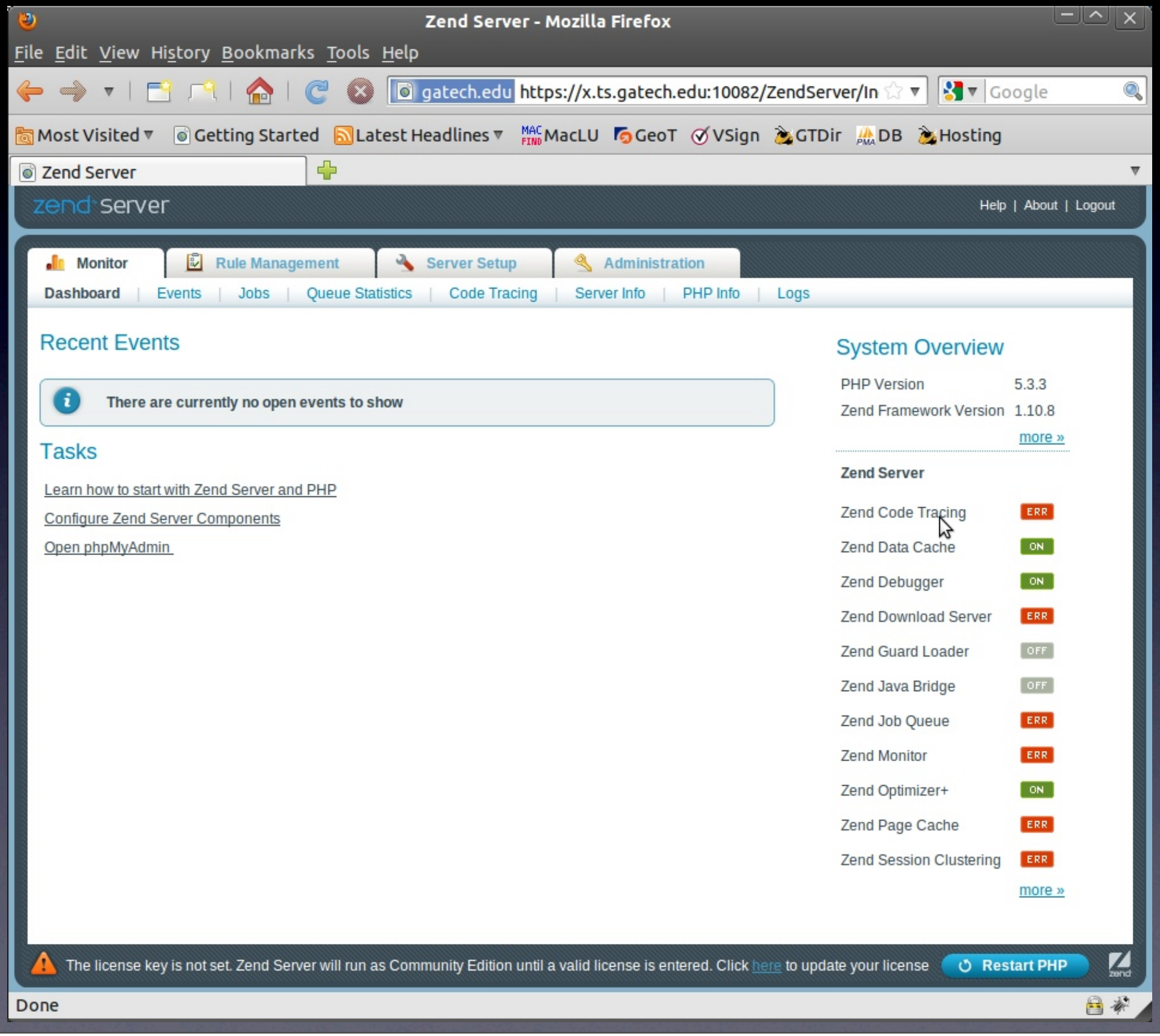
Post Installation

- Install the Zend Server Source
- Recompile any modules that require SSL (such as LDAP to GTED)
- Secure the Zend Server Console (https://servername:10082)

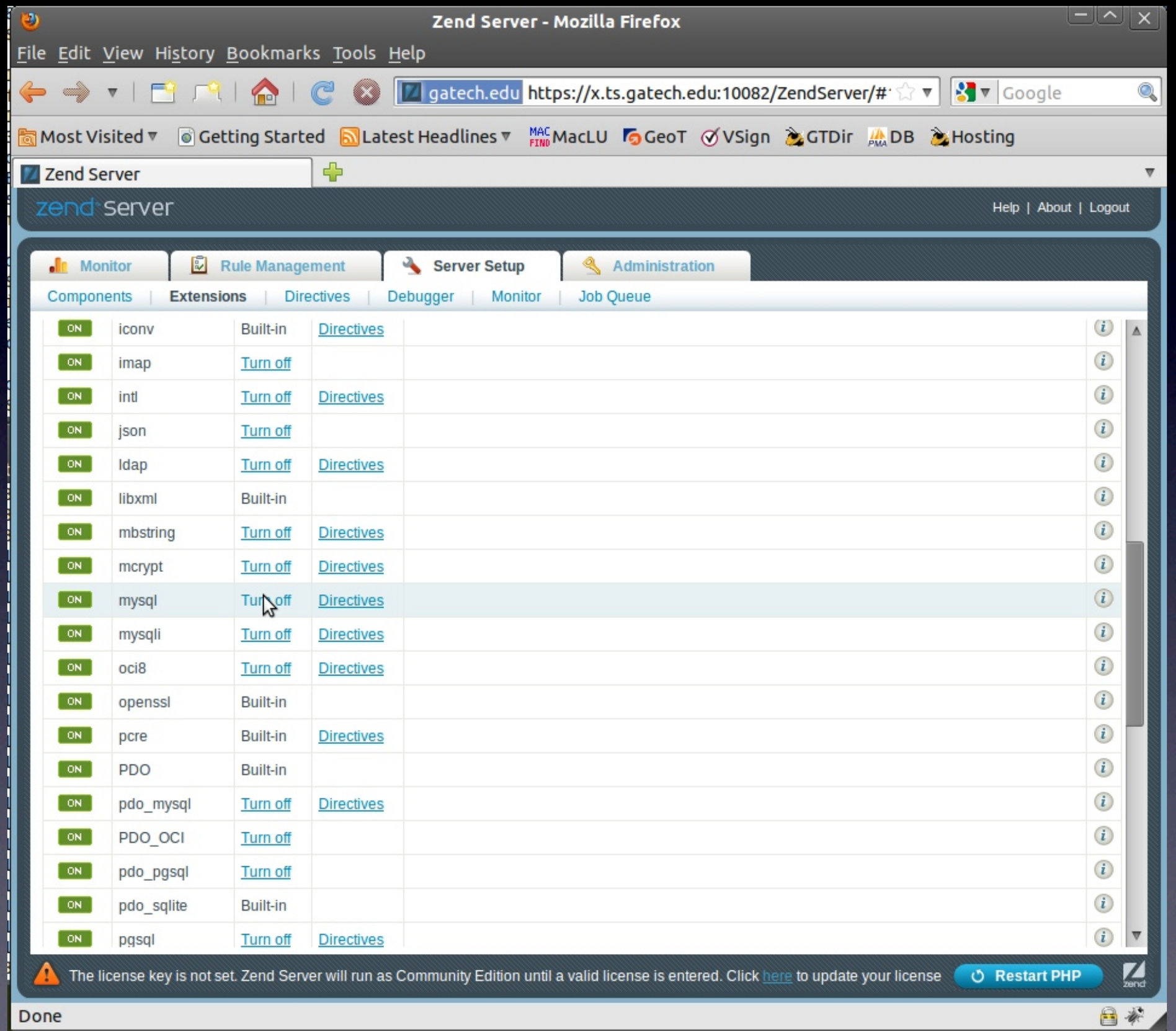
Review Security Documents on Zend



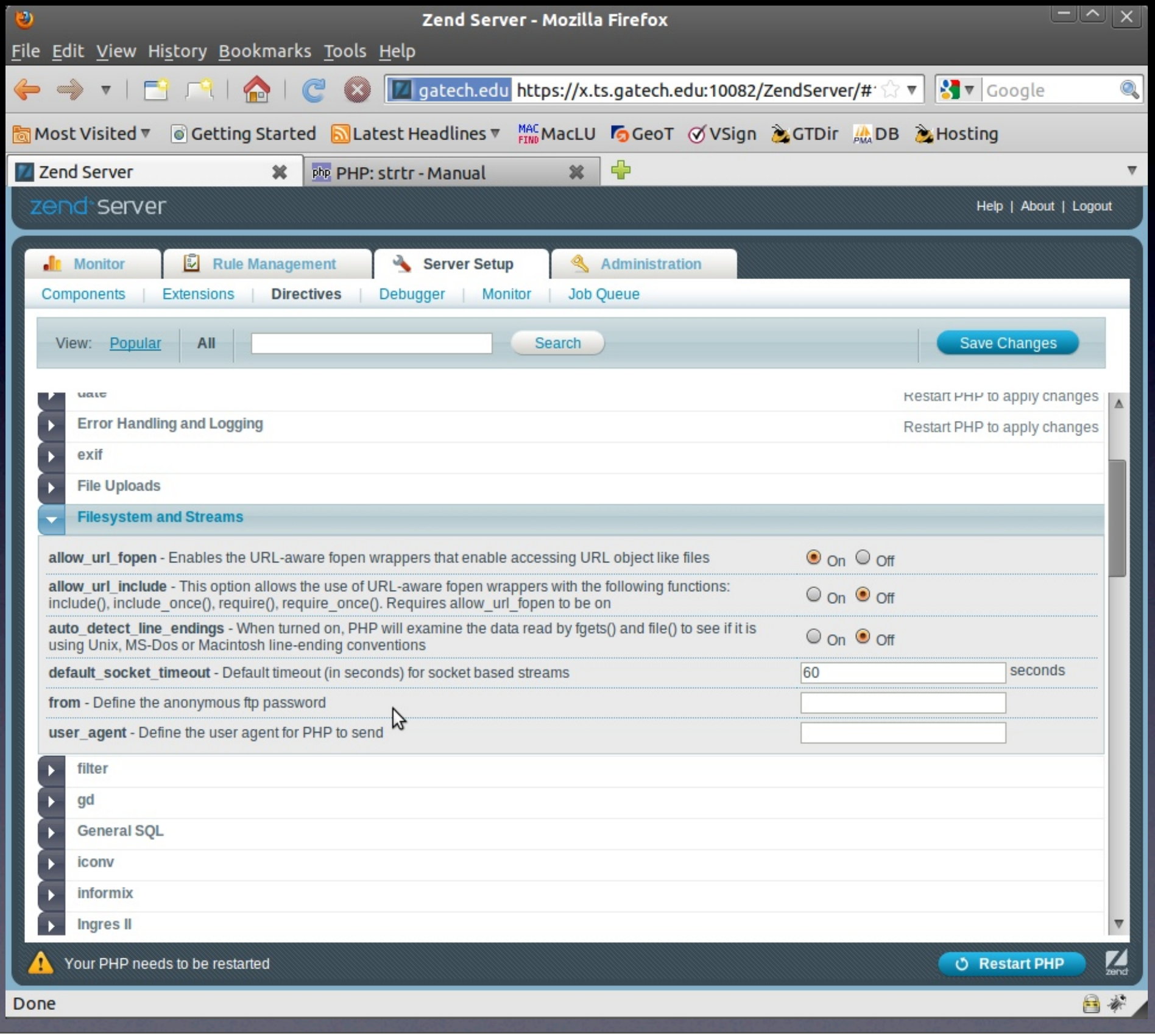
Zend Server Console



Zend Server Console



Changing PHP Settings



PHP Directives of Interest

register_globals = Off
allow_url_fopen = Off
enable_dl = Off
expose_php = Off
disable_functions = apache_get_modules,apache_get_version,apache_getenv,apache_note,apache_setenv,virtual,apache_child_terminate

file_uploads = Off
upload_max_filesize = 1M
upload_tmp_dir = /var/www/tmp

memory_limit = 8M
post_max_size = 8M
max_input_time = 60
max_execution_time = 30

session.save_path = /var/www/sessions
session.referer_check = gatech.edu

Restricting File Access

open_basedir = /var/www/:/usr/local/zend/share/ZendFramework/library/:/usr/local/zend/share/pear/

- Make sure you end the basedir with a "/" (or else you are creating a prefix (/var/www = /var/www or /var/www2)

- Be sure to add ANY external PHP library paths, such as PEAR or Zend Framework

PHP Coding Security

You can lock the server down all you want, but your software can still bite you in the butt

PHP Coding Security Basics

- Consider illegitimate uses of your application
- Educate your programmers
- If nothing else, FILTER ALL EXTERNAL DATA

register_globals is BAD

```
<?php

if (authenticated_user())
{
    $authorized = true;
}

if ($authorized)
{
    $include ‘/highly/sensitive/data.php’;
}

?>
```

Validate ALL Form Data

- If you are expecting a name, you shouldn't get strange characters like #%><;
- If you are expecting a number it should be a number
- Functions that are your friend:

bool is_numeric(mixed $var)
bool ctype_alnum(string $text)
int preg_match($pattern, $subject, $matches)
string strtr($string, $from, $to)

SQL Injection Attacks

- This method of attack had to do with crafted input that performs operations on your database that you didn't intend to occur.
- For example:

SELECT * from users where username = '$username' or 1=1#'

Frameworks

- Zend Framework
- Symphony
- CakePHP
- etc.

```
$sql = ‘SELECT * FROM messages WHERE username = ?’;
$row = $db->fetchRow($sql, $username)
```

Passwords in your Code

- Usernames/Passwords in PHP are hard to eliminate
- Store them OUTSIDE of your application where possible (some applications require them to be in their configuration files)
- Use “include /full/path/to/config.ini” or limit via .htaccess or in the least, name your files “.php” so it gets interpreted if requested

Securing PHP with Suhosin

- PHP was not built with security in mind
- Suhosin: a patch to secure PHP
  http://www.hardened-php.net/suhosin/
  - much more granular than php.ini settings
  - open_basedir + suhosin = no "safe_mode"

Important Suhosin options

suhosin.executor.eval.blacklist

- Restricts those things you can use inside of an eval()
- obfuscated code executed by an eval() is a typical script kiddie trick
- **examples**: include, curl_init, fpassthru, file, base64_encode, base64_decode, mail, exec, system, proc_open, leak, syslog, pfsockopen, shell_exec, ini_restore, symlink, stream_socket_server, proc_nice, popen, proc_get_status,dl, pcntl_exec, pcntl_fork, pcntl_signal, pcntl_waitpid, pcntl_wexitstatus, pcntl_wifexited, pcntl_wifsignaled, pcntl_wifstopped, pcntl_wstopsig, pcntl_wtermsig, socket_accept, socket_bind, socket_connect, socket_create, socket_create_listen, socket_create_pair,link, register_shutdown_function,register_tick_function

Important Suhosin options

suhosin.executor.func.blacklist

- Completely removes ability to execute a function
- Do you *really* need exec()?
- **examples**: exec,system,proc_open, pfsockopen, shell_exec, ini_restore, stream_socket_server, proc_nice, popen, proc_get_status, pcntl_exec, pcntl_fork, pcntl_signal, pcntl_waitpid, pcntl_wexitstatus, pcntl_wifexited, pcntl_wifsignaled, pcntl_wifstopped, pcntl_wstopsig, pcntl_wtermsig, socket_accept, socket_bind, socket_connect, socket_create, socket_create_listen, socket_create_pair, passthru

- suhosin.executor.disable_emodifier=On
  Even if you close off those, the \e modifier to preg_replace allows arbitrary code execution

Useful Suhosin defaults

- Even if you don't configure anything several useful defaults are set
- useful defaults
  - No nulls allowed in GET or POST variables
  - suhosin.upload.disallow_elf

Suhosin configuration

Logging Configuration

suhosin.log.syslog
suhosin.log.syslog.facility
suhosin.log.syslog.priority
suhosin.log.sapi
suhosin.log.script
suhosin.log.phpscript
suhosin.log.script.name
suhosin.log.phpscript.name
suhosin.log.use-x-forwarded-for

Executor Options

suhosin.executor.max_depth
suhosin.executor.include.max_traversal
suhosin.executor.include.whitelist
suhosin.executor.include.blacklist
suhosin.executor.func.whitelist
suhosin.executor.func.blacklist
suhosin.executor.eval.whitelist
suhosin.executor.eval.blacklist
suhosin.executor.disable_eval
suhosin.executor.disable_emodifier
suhosin.executor.allow_symlink

Filtering Options

suhosin.filter.action
suhosin.cookie.max_array_depth
suhosin.cookie.max_array_index_length
suhosin.cookie.max_name_length
suhosin.cookie.max_totalname_length
suhosin.cookie.max_value_length
suhosin.cookie.max_vars
suhosin.cookie.disallow_nul
suhosin.get.max_array_depth
suhosin.get.max_array_index_length
suhosin.get.max_name_length
suhosin.get.max_totalname_length
suhosin.get.max_value_length
suhosin.get.max_vars
suhosin.get.disallow_nul
suhosin.post.max_array_depth
suhosin.post.max_array_index_length
suhosin.post.max_name_length
suhosin.post.max_totalname_length
suhosin.post.max_value_length
suhosin.post.max_vars
suhosin.post.disallow_nul
suhosin.request.max_array_depth
suhosin.request.max_array_index_length
suhosin.request.max_totalname_length
suhosin.request.max_value_length
suhosin.request.max_vars
suhosin.request.max_varname_length
suhosin.request.disallow_nul
suhosin.upload.max_uploads
suhosin.upload.disallow_elf
suhosin.upload.disallow_binary
suhosin.upload.remove_binary
suhosin.upload.verification_script
suhosin.session.max_id_length

Transparent Encryption Options

suhosin.session.encrypt
suhosin.session.cryptkey
suhosin.session.cryptua
suhosin.session.cryptdocroot
suhosin.session.cryptraddr
suhosin.session.checkraddr
suhosin.cookie.encrypt
suhosin.cookie.cryptkey
suhosin.cookie.cryptua
suhosin.cookie.cryptdocroot
suhosin.cookie.cryptraddr
suhosin.cookie.checkraddr
suhosin.cookie.cryptlist
suhosin.cookie.plainlist

Misc Options

suhosin.simulation
suhosin.apc_bug_workaround
suhosin.sql.bailout_on_error
suhosin.sql.user_prefix
suhosin.sql.user_postfix
suhosin.multiheader
suhosin.mail.protect
suhosin.memory_limit

The End

Q & A ?