



BlackHat®

USA • EUROPE • ASIA

USA 2006

<bhtalk>

***Defending Black Box
Web Applications:
Building a Web Security Gateway***

</bhtalk>

***Building an Open Source
Web Security Gateway***

shawn . moyer : CISO
:: agura digital security ::
blackhat@agurasec.com



Black Hat Briefings

The Black Box App Defined

Black Box Web App:

You own the security, but not the code
Canned apps, “appliances”, abandonware
Legacy apps with minimal dev time available

Required by business, but can't be properly secured
Vulns are there, but app must remain in production

Examples: home-baked stuff, webmail, HRMSes, CMSes, insert your crapware here



Bad news / Disclaimers

Front-ending security is not a panacea

Reusable input val, secure coding standards, aggressive code auditing, are the optimal path

Sorry, no free lunch - unless the vendor pays =)

Sooner or later... you will have to fix this stuff.



Good news / Nuggets of hope

Still... You can do more than you're doing now

Relatively simple to stop known (CVE / OSVDB, etc) attacks
Skiddies / anklebiters are mostly preventable
Net benefit is high with minimal cost / effort

Better than waiting to get Ownzored



Meet the WSGW

Web Security Gateway:

HTTP(s) proxy... on crack

OSS version of what a lot of vendors are doing

Bundle web security modules and tools in a hardened frontend proxy

Central enforcement / auth / auditing point for all web traffic



Meet the WSGW (cont.)

More features:

Hide / rewrite generated content

Scan and validate POSTs / GETs / params

Wrap SSL, authentication, other goodies

Bonus points: XML “firewalling”

Bonus points: Content caching and acceleration

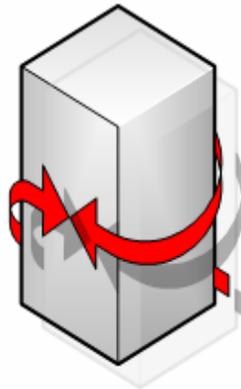


WSGW Architecture: Placement

GET /badstuff.asp?stuff=bad&bad=stuff HTTP/1.1



:: Client Request
:: Inspect
:: Normalize
:: Scan
:: Rewrite



GET /goodstuff.jsp?stuff=good&good=stuff HTTP/1.1

:: Server Response
:: Inspect
:: Normalize
:: Scan
:: Rewrite



```
HTTP/1.1 200 OK
Server: UnsuckWeb/777.77
Date: Mon, 01 May 2006 21:26:41 GMT
X-Powered-By: Starbucks
Connection: Keep-Alive
Content-Length: 50514
Content-Type: text/html
Set-Cookie: SESSIONID=sp0rkl3f0rk398!@m00;
           path=/
Cache-control: private
```

```
HTTP/1.1 200 OK
Server: SuckWeb/1.0
Date: Mon, 01 May 2006 21:26:41 GMT
X-Powered-By: Craptastic.NET
Connection: Keep-Alive
Content-Length: 50514
Content-Type: text/html
Set-Cookie: CRAPSESSIONID=123456789;
           path=/
Cache-control: private
```



Getting Started: Platform

Apache proxy on extensively hardened host with minimal install

OpenBSD, Trustix, Engarde,
Bastille-ified generic distro

Minimal services / users

No compilers, all apps built with stack protection

SSH with two-factor auth from specific hosts

Log to remote syslog(-ng)

Note: load up on proc / mem: we are parsing all www traffic!



Why Apache?

Arguably not secure by default, but highly configurable...

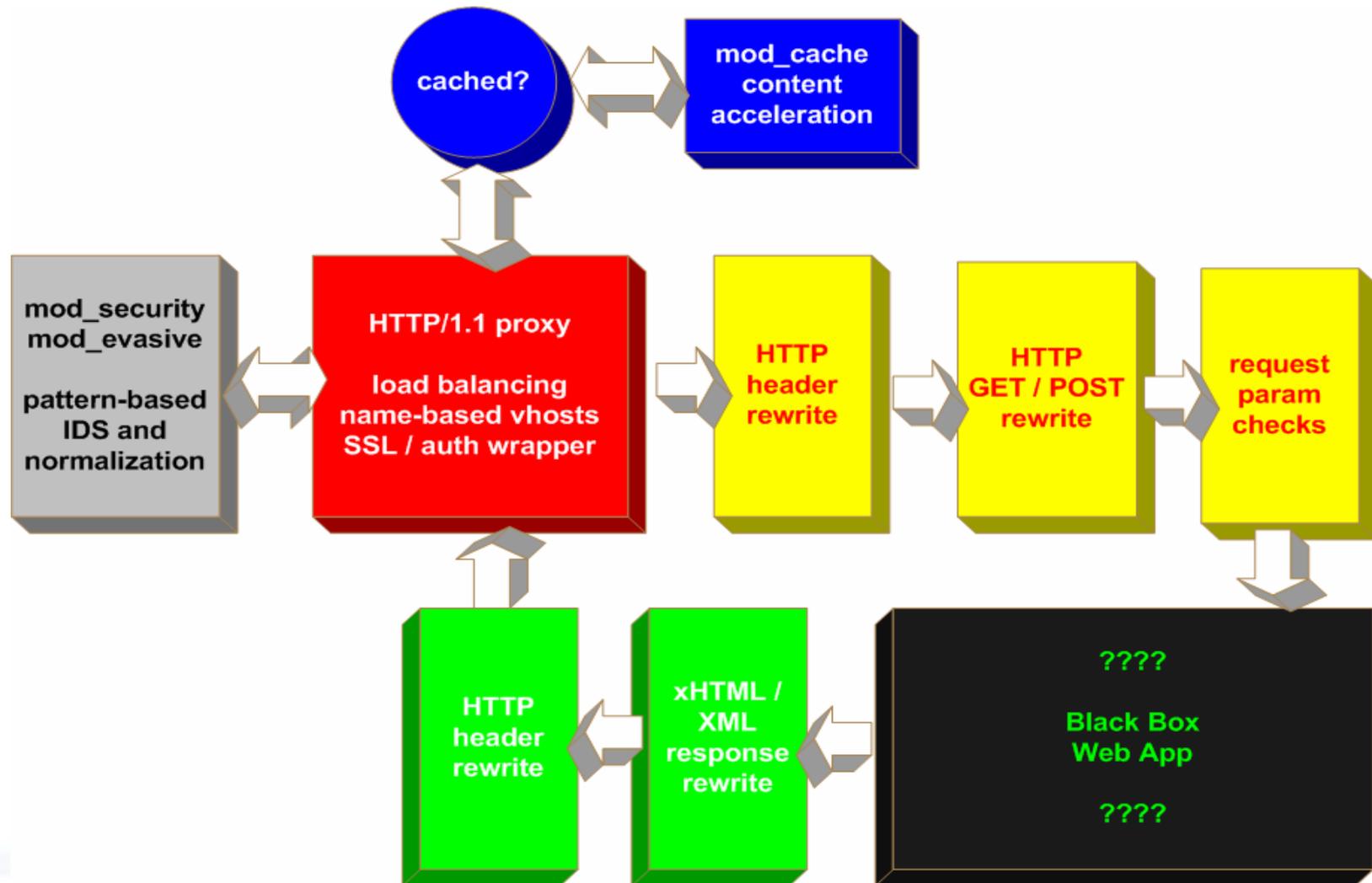
We need an HTTP proxy that is content-aware. Why reinvent the wheel?

mod_filter and the FilterChain directive == efficient inline content inspection with multiple filters, especially w/ 2.x and PCRE

Substantial number of free modules and tools to aggregate into an “app firewall”



WSGW Architecture: Data flow



Building Apache

Modules we need (disable everything else):

mod_proxy, mod_(disk|mem)cache, mod_log_forensic,
mod_log_config, mod_auth(dbm|ldap), mod_headers,
mod_rewrite

Static compile of built-in and external modules is a good idea.

Note: Off-by-one in mod_rewrite < 2.3 / 2.0.59...



Building Apache

Chroot / Jail!

Typically a PitA, but...mod_security does internal, pre-fork chroot for Apache! (Similar to Postfix / QMail, etc)

SecChrootDir /var/www

mod_security should be first module loaded in conf for this to behave



External Modules

*mod_security**: multipurpose IPS / security toolkit for Apache
<http://www.modsecurity.org>

mod_proxy_html: content rewriting of proxied (x)HTML

mod_line_edit: sed-fu for *any* proxied data

mod_publisher: rewrite proxied XML

<http://apache.webthing.com>

mod_evasive: limit per-IP requests to a reasonable number

http://www.nuclearelephant.com/projects/mod_evasive/

* Go home and put ModSec on everything you can. Now. And send Ivan money and cookies.



Headers / Cookies

Header rewrites:

*Header set Server "Cern-HTTP/3.0a" (no ErrorHandler w/ 2.x! Eeep!)
SecServer SecServerSignature "CernHTTP/3.0A" (via ModSec)*

*Header set X-Pad "Free the mallocs"
Header set X-Powered-By "Hamsters.NET"*

Header set Public "REBOOT, PEEK, POKE, OPTIONS, TRACE, GET, HEAD, DELETE, PUT, POST, COPY, MOVE, MKCOL, PROPFIND, PROPPATCH, LOCK, UNLOCK"

Header add set-Cookie "ASPSESSIONI%d34db33fc4f3; path=/"

*Header set Via "1.1 squidcache.sporkworks.com (Squid/2.2.STABLE3)"
Header set X-Cache "MISS from squishymiddle.sporkworks.com"*

Mod_headers and mod_usertrack should be capable of cookie rewriting with minimal effort. Map PHPSESSIONID / ASPSESSIONID to new values and track state. In progress...



Headers / Cookies (Cont.)

Header Kung-Fu:

Modify `http_protocol.c` (re-order Date / Server) and `httpd.h` (error codes and messages) in Apache src to further obfuscate.*

Not foolproof, but sufficiently random / anomalous headers will fool HTTPPrint / scanners / etc.

Experiment with HTTPPrint and various header / error codes until you get the result you want.

* Thanks much to the WASC (webappsec.org) folks for this idea.



Rip out meta tags / info leaks

***mod_line_edit* rewrites content with PCRE:**

Strip generator and author tag:

```
LeRewriteRule <meta(.*?)name=\"Generator\"(.*?)content=(.*)> <> Ri
```

```
LeRewriteRule <meta(.*?)name=\"Author\"(.*?)content=(.*)> <> Ri
```

Strip comments. Be careful with Javascript.

ProxyHTMLStripComments On

XML “firewalling”: Strip info leakage, fault codes, etc from XML with mod_line_edit. Or, use mod_transform / mod_publisher.



Impersonate static content

mod_rewrite and mod_proxy_html:

RewriteRule ^/(.*)/(.*)/(.*)/(.*) /index.aspx?\$1=\$2&\$3=\$4 [P,L]

Normally requires modification to site code,
but with mod_proxy_html...

ProxyHTMLUrlMap (.*)/index.aspx\?(.*)=(.*)&(.*)=(.*) \$1/\$2/\$3/\$4/\$5/

<http://www.sporkworks.com/index.aspx?spork=foon&fork=spoon>

Presents as:

<http://www.sporkworks.com/spork/foon/fork/spoon/>



And... With input val!

Same example, with input validation via mod_rewrite:

```
RewriteRule ^([a-z]{4})/([a-z]{4,5})/([a-z]{2,7})/([a-zA-Z0-9]{5,6}) /index.aspx?$1=$2&$3=$4 [P,L]
```

Create regexp for valid param contents, with a max param length for your app. Anything not matching your patterns will be stripped at proxy.

Cool, huh?

Now...add a clean-up rule as the last rule to send any non-matches to the main page, or a canned error page. This is key.

```
RewriteCond !%{REMOTE_HOST} Your.Proxy.IP.ADDY  
RewriteRule ^(.*) /error.aspx? [P,L]
```

Note we exclude the proxy itself, otherwise we go into flailing death loop. Also note “?” to strip query string.

Also need rules for any .js, .css, graphics, etc.



Form / POST validation

Validate form posts with Mod_Security:

SecFilterScanPost On

SecFilterSelective "POST_PAYLOAD" !({a-zA-Z0-9}+) deny,status:403

SecFilterSelective "POST_PAYLOAD" !CCNUM={0-9} deny,status:403

SecFilterSelective "POST_PAYLOAD" !LAUNCH_CODE=0s4m42006 deny,status:403

... etc.



Belt and suspenders / extras

Lots of great ModSec rules at <http://www.gotroot.com>

Write ModSec sigs for any error leakage in your environment (IIS / .NET errors, anything with a path, version info, etc. Redirect to a canned error.

Stunnel in client mode allows you to proxy / filter for SSL-only sites.

Mod_auth_(ldap|dbm|vas) allows you to wrap auth here.

Mod_evasive can throttle scans, DoS, etc

WSGW is a good place to run snort_inline, too. Why not? :)



Thanks! You guys rawk!

Thanks much!

Updated version of this talk, plus configs, examples, build scripts, recipes, other stuff at SourceForge:

<http://wsgw.sf.net>

