What’s In This Talk?

• Web Application Firewalls (WAFs)

• PCI Data Security Standard
  ▪ Requirement 6.6
  ▪ Why should you care?

• ModSecurity
  ▪ Concepts
  ▪ Core Rules

• ModSecurity WAF Live Demos
  ▪ Reverse Proxy
  ▪ Embedded
  ▪ Content Injection in Response to Attack
Web Application Firewalls (WAFs)
The Basics

• Firewall operating at the web application layer
• Hardware or Software
• Fluent in many tongues
  - HTTP / HTTPS
  - XML
  - SOAP
  - XML-RPC
  - WS-*
• Performs normalization / de-obfuscation
• Detects attacks
• Blocks attacks
• Rewrites / Modifies requests and responses
ModSecurity Web App Firewalls
The Basics

- Apache dynamically loadable module
- Dual License
  - Community Edition - GPLv2
  - Commercial License
- Created by Ivan Ristic
- Commercialized by Breach Security
- Core Rules released under GPLv2
Meet the Payment Card Industry!
Why Should You Care?

• I apologize in advance if I bore you

• PCI is driving adoption of WAF (and Web App Vulnerability Scanning)

• Pentesters (i.e. QSAs): PCI will drive your business. You will need to be familiar with WAFs to evaluate and subvert them.

• IT Security: You will be deploying WAFs because of PCI

• Blackhats: You will be subverting WAFs for fun and profit!
Meet the Payment Card Industry!

Terminology

- Lots of new acronyms, yea!
- PCI Security Standards Council (PCI)
- PCI Data Security Standard (PCI DSS)
- Other PCI security standards exist
  - PIN Entry Devices (PEDs)
  - Payment Application Data (PA-DSS)
- Qualified Security Assessors (QSAs)
- Approved Scanning Vendors (ASVs)
PCI Data Security Standard v1.1
The Basics

- Build and Maintain a Secure Network
- Protect Cardholder Data
- Maintain a Vulnerability Management Program
- Implement Strong Access Control Measures
- Regularly Monitor and Test Networks
- Maintain an Information Security Policy
PCI DSS Application / System Security
Requirement 6

- R6. "Develop & maintain secure systems and applications"

- R6.6. "Ensure that all web-facing applications are protected against known attacks by applying **either** of the following methods:
  - Having all custom application **code reviewed** for common vulnerabilities by an organization that specializes in application security
  - Installing an **application layer firewall** in front of web-facing applications.

- Note: This method is considered a best practice until **June 30, 2008**, after which it becomes a requirement."
PCI Data Security Standard
What Does All This Really Mean?

- A way to reassign legal liability
  - QSA assumes unlimited liability? (IANAL)

- Compliance rationale for bigger IT security budgets

- An economically dictated race to the bottom for ASVs?
PCI Data Security Standard
An economically dictated race to the bottom for ASVs?

- Cost of a PCI ASV's Solution
- vs. that Solution's Ability to Find Issues (its Quality)
- vs. Cost of Remediating the Identified Findings
- vs. Loss Expectancy Due to Unidentified Issues
- vs. Loss Expectancy Due to Unremediated Issues
- No market differentiator between a PCI stamp of approval granted by ASVs of varying quality!
ModSecurity Concepts

- Virtual Patching / Just-In-Time Patching
- Positive Security Model
  - Input Validation Envelope
- Negative Security Model
  - Enumerate the bad stuff

- Difficult to achieve the "positive input validation envelope" in the real-world!

- "When you know nothing, permit-all is the only option. When you know something, default-permit is what you can and should do. When you know everything, default-deny becomes possible, and only then." – Richard Bejtlich(?), quoted in WhiteHat Security WP WAF061708
More ModSecurity Concepts

Processing Phases

- Request Headers
- Request Body
- Response Headers
- Response Body
- Logging / Action
More ModSecurity Concepts
Transformations

• Can be nested / run in serial

• Replace Comments
  ▪ SQLi

• URL Encode / Decode
• Hex Encode / Decode
• JavaScript Decode
• HTML Entity Decode
• Uppercase / Lowercase
• MD5 / SHA1
• Normalize Paths
ModSecurity Core Rules

• HTTP protocol protection
  ▪ RFCs
  ▪ Defined policy
• Common Web Attack Protections
  ▪ XSS, SQLi, CSRF, HTTP Response Splitting
• Automation Detection
  ▪ Bots, web crawlers, web scanners
• Trojan Protection
• Server Error Hiding / DLP
  ▪ Mask errors sent by the server
  ▪ Data Loss Prevention
ModSecurity Rule Language Keywords

- Request (a few important keywords)
  - REQUEST_METHOD
  - REQUEST_URI
  - REQUEST_FILENAME
  - QUERY_STRING
  - REQUEST_HEADERS
  - REQUEST_BODY

- Response (a few important keywords)
  - RESPONSE_STATUS
  - RESPONSE_BODY
  - RESPONSE_HEADERS
  - RESPONSE_CONTENT_TYPE
  - RESPONSE_CONTENT_LENGTH
ModSecurity v2.5 Highlights

- Content Injection
  - "prepend" and "append"
  - Embed one of Billy Hoffman's JS payloads in response to attack?

- Aho-Corasick pattern matching algorithm
  - Improved performance when matching on large sets of patterns

- Cached transformations

- GeoIP lookup
  - Use as matching criteria in rules
More ModSecurity v2.5 Highlights

- Crediet Card verification on the wire
  - @verifyCC rule operator, takes a regexp argument
  - Luhn checksum algorithm

- PDF Universal XSS Protection
  - http://www.example.com/file.pdf#a=javascript:alert('p0wn3d')
  - All PDFs on protected site get a one-time use URI
  - Redirects visitors to the PDF
  - Flushes any malicious JS in client's browser session

- Full Lua scripting w/ SecRuleScript directive
  - Used to create more complex rules in Lua
ModSecurity Web App Firewall

Deployment Scenarios

- **Embedded**
  - Installed within Apache instance hosting site

- **Reverse Proxy**
  - Use Apache mod_proxy
  - Traffic is redirected to flow through WAF
    - DNS configuration
    - Network-layer redirection
  - Could be hosted "in the cloud"
  - Supports use of Apache Virtual Hosts
Embedded Deployment

Apache Web Server
www.example.com
VirtualHost

ModSecurity
DocumentRoot

Site Visitor
Live Demo
Reverse Proxy Deployment

Site Visitor ➔ HTTP Request ➔ WAF ➔ HTTP Response ➔ Web Server
ModSecurity Content Injection

- Credit to David (DKZA) Kierznowski, GNUCITIZEN
  - [http://www.gnucitizen.org/blog/content-injection-hack-the-hacker/](http://www.gnucitizen.org/blog/content-injection-hack-the-hacker/)

- Inject code snippets
  - Defense
  - Hijacking JS functions
    - Redefine alert() with a Logger
    - A looking glass into client's browser
Content Injection
An Example

- The following rule will inject a Java Applet

- MyAddress will force attacker's browser to callback to us

- Reveals IP of the attacking host
  - Works despite NAT
  - Good, we might be only seeing IP of WAF in server log
Content Injection
The Rules

SecContentInjection On

SecDefaultAction
"log,deny,phase:2,status:500,t:none,setvar:tx.alert=1"

SecRule TX:ALERT "@eq 1" \
"phase:3,nolog,pass,chain,prepend:'<applet\n code="MyAddress.class" width=0 height=0> \ 
<param name="url" value="grab_ip.php?IP="> \ 
<param name="action" value="auto"> \ 
</applet>\""

SecRule RESPONSE_CONTENT_TYPE "^text/html"
Content Injection
Apache Access Log

- Below is a snippet from Apache access_log on the server
- Reverse Proxy WAF has IP address is 10.0.0.10
- Attacker IP is 172.16.0.20


10.0.0.10 - - [30/May/2008:13:47:11 -0400] "GET /cgi-bin/grab_ip.php?IP=172.16.0.20 HTTP/1.1" 404 207
VA + WAF
= ??? + Profit !!!

• "The Dream"
  ▪ Automated webapp vulnerability assessment (i.e., scanning) instantly mitigates identified flaws through automagic deployment of rules to WAFs.

• Until recently, not really feasible
  ▪ Web App VA generated too many false positives
  ▪ Web App VA generated too many duplicates
  ▪ WAFs suffered under too many FPs and duplicates

• Vendors are trying again
Limitations of WAFs
It’s Just A Tool, Not A Silver Bullet

- Insecure Session Handling / Potential Cookie Tampering
  - WAF can perform transparent cookie encryption/decryption

- Flaws in Business Logic
  - Reliance on a predictable "random" number in URL to provide authentication and authorization
    - Can be solved with a WAF performing "URL encryption"
    - Similar to ModSecurity protection against Universal PDF XSS
  - Many flaws in business logic are very difficult to detect w/ automated tools...
  - ...and difficult to mitigate with a tool like a WAF
The Future
Some Closing Thoughts

• Vendors will continue to add WAF-like functionality
  ▪ Load Balancers
  ▪ Firewalls
  ▪ IPS and UTM devices

• WAF-like functionality being wrapped into malware
  ▪ Many already contain SOCKS proxy functionality

• Rogue / Malicious WAF Attacks
  ▪ WPAD-like attack vectors?
  ▪ WAF Poisoning?

• More WAF Bypass Vulnerabilities
• WAFs from an attacker's perspective

• "Playing with Web Application Firewalls"
  ▪ Wendel Guglielmetti Henrique
  ▪ Fri 15:00-15:50, Track 2

• How to detect, fingerprint, and evade WAFs

• I learned some stuff just reading the slides
Thanks to DT, the Goons and everyone who made DEFCON a reality this year!
Greetz to DC404, Atlanta's DC Group!
Speakers: dr.kaos, Carric, David Maynor, Scott Moulton & Adam Bregenzer
And our very own Goon, dc0de!
Questions?

bfeinstein@secureworks.com