"The Core Rule Set": Generic detection of application layer attacks

Ofer Shezaf
OWASP IL Chapter leader
CTO, Breach Security
Breach & the Community

- ModSecurity – open source WAF
  - Recently purchased and kept as open source
  - Most popular Web Application Firewall on the globe
  - Ivan Ristic who wrote it and Ryan Barnett community leader joined us

- Web Application Security Consortium:
  - Web Application Firewall Evaluation Criteria - Ivan
  - Web Attacks Honeypot Project - Ryan
  - Web Hacking incidents Database – Ofer
  - Member of the board of directors - Ofer

- OWASP IL chapter leadership
**ModSecurity 2.0**
- Long awaited update to ModSecurity
- Significantly enhanced analysis engine
- XML parsing

**ModSecurity Console**
- Provides GUI event viewing
- Consolidation from multiple ModSecurity sensors

**ModSecurity Core Rules**
- Package of signatures certified to be efficient and accurate by Breach Labs
- Coverage for most common web application threats
Web Application Firewalls vs.
Intrusion Prevention Systems
Deployment - Network-level device

Does not require network re-configuration.
Deployment - Embedded

Does not require network re-configuration.
Three Protection Strategies for WAFs

1. **External patching**
   - Also known as "just-in-time patching" or "virtual patching".

2. **Positive security model**
   - An independent input validation envelope.
   - Rules must be adjusted to the application.
   - Automated and continuous learning (to adjust for changes) is the key.

3. **Negative security model**
   - Looking for bad stuff,
   - Mostly signatures based.
   - Generic but requires some tweaking for each application.

IPS?
Virtual Patching

- Testing reveals that the login field is vulnerable to SQL injection.
- Login names cannot include characters beside alphanumerical characters.
- The following rule will help:

```
<LocationMatch "^/app/login.asp$">
    SecRule ARGS:username "!\w+$" "deny,log"
</LocationMatch>
```
The same, but for every field in every application

Very hard to create, requires learning by:

- Monitoring outbound traffic (match input to web server request)
  - Caveats: JavaScript, Web Services

- Monitoring inbound traffic (normal behavior):
  - Caveats: Statistics, attacks in learning period.
Positive Security

Site

Site Map

URLs

Parameters

Site Status

Parameter Types
Negative Security

An IPS, but:

- Full parsing & validation of HTTP:
  - Request, Headers, Content
  - Validation to individual fields (field content, length, field count, etc).
  - both request and response.
  - Uploaded files.

- Anti Evasion features:
  - Decoding
  - Path canonizations
  - Robust parsing (apache request line delimiters...)
Rules instead of signatures

- **Signatures**
  - Simple text strings or regular expression patterns matched against input data.
  - Not very flexible.

- **Rules**
  - Flexible.
  - Multiple operators.
  - Rule groups.
  - Anti-evasion functions.
  - Logical expressions.
  - Custom variables.
The Core Rule Set

- modsecurity-core-rules_2.0-1.1.1_blocking.zip
- modsecurity_crs_10_config.conf
- modsecurity_crs_20_protocol_violations.conf
- modsecurity_crs_30_http_policy.conf
- modsecurity_crs_35_bad_robots.conf
- modsecurity_crs_40_generic_attacks.conf
- modsecurity_crs_45_trojans.conf
- modsecurity_crs_50_outbound.conf
- modsecurity_crs_55_marketing.conf
Detection of generic app layer attacks

- Core Rule Set available for ModSecurity at:
  - Probably translatable to any App Firewall

- Benefits from ModSecurity features:
  - Anti Evasion
  - Granular Parsing

- Detection Mechanisms:
  - Protocol Violations
  - Protocol Policy
  - Generic Attack Signatures
  - Known Vulnerabilities
  - Bad Robots
  - Trojans & Anti-Virus
  - Error conditions
Protocol Violations

- **Headers:**
  - All required headers are there: Host, Accept, User-Agent
  - Host is not an IP address
  - Content length a must for none GET/HEAD methods

- **Characters:**
  - Valid encoding
  - Only printable for headers
  - Printable and formatting for parameters
  - Only NULL not allowed in international applications

- **Requires minimal tweaking**
  - Exceptions for automated software used by the application
Protocol Policy

- Allowed and blocked:
  - HTTP versions
  - Methods
  - File extensions
  - Content-Types (request AND reply)

- Global limitations:
  - Request size, Upload size,
  - # of parameters, length of parameter.

- Requires setting, but easy to set:
  - We offer tailored settings for common development environments.

- An easy (not generic) addition: envelope on valid URLs.
Signatures for generic attacks

- Signatures require knowing the attack vectors and therefore are usually used for known vulnerabilities.
- Web applications are custom, and attacks may be targeted.
- Variations on attack vectors are very easy.
- Hence, normal signatures are not suitable for application layer protection.
- In many cases few exceptions can make signatures vary effective:
  - substring
Case study: 1=1

- Classic example of an SQL injection attacks.
- Used many times as a signature.
- But, can be avoided easily using:
  - Encoding: 1%3D1
  - White Space: 1  =%091
  - Comments 1 /* This is a comment */ = 1
  - All of the above
And is actually not required at all. Any true expression would work:
  - $2 > 1$

An not necessarily a comparison or even an expression. In MS-Access all the following are true: 1, “1”, “a89”, 4-4
Rules instead of signatures

- **All these are attack indicators:**
  - `xp_cmdshell`
  - "<" valid but stinks
  - `select`, `union`, `delete`, `drop` & `script` are valid English words
  - Single quote is very much needed to type O'Brien
  - "1"

- The following rules can help:
  - Sequence: `union` .... `Select`
  - Amount: `script`, `cookie` and `document` appear in the same input field
  - Learning: `select` and a `single quote (')` in a field it never appeared in.
  - Amount & learning: three `triangular brackets (< or >)` appear in a field leaned as free text.
A recent snort rule - bugtraq 9349

**Exploit:** http://www.example.com/athenareg.php?pass=%20;whoami

**Snort Rule:**
alert tcp
$EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS
(
 msg: "BLEEDING-EDGE WEB Athena Web Registration Remote Command Execution Attempt";
 flow: to_server,established;
 uricontent:"/athenareg.php?pass=%20\"; nocase;
 reference:cve,CAN-2004-1782;
 reference:bugtraq,9349;
 classtype: web-application-attack;
 sid: 2001949; rev:4;
)
The Core Rule Set: generic detection

# Command injection
SecRule REQUEST_FILENAME|ARGS|ARGS_NAMES|REQUEST_HEADERS "(?:([\;\|/])\W*?\b(?:c(?:c?:h(?:?:grp|mod|own|sh|md|pp|c)|p(?:asswd|ython|erl|ing|s)|n(?:asm|map|c)|f(?:inger|tp)|\(?:kil|mai)1\|g(?:\:+\|cc)|(?::xte)?rm|ls(?::of)?|telnet|uname|echo|id)\|):(?:c(?:h(?:?:grp|mod|own|sh)|pp|c)|p(?:asswd|ython|erl|ing|s)|n(?:asm|map|c)|f(?:inger|tp)|\(?:kil|mai)1\|g(?:\:+\|cc )|(?::xte)?rm|ls(?::of)?|telnet|uname|echo|id))\b\b(?::(?:n(?:et(?::\b\W*?\bblocalgroup|\.exe)|\(?=map|c\)\.exe)\|t(?::racer(?:oute|t)|elnet\.exe|clsh8?|ftp)|w(?::g(?:uest\.exe|et)|sh\.exe)\|(?::rcmd|ftp)\.exe|echo\b\W*?\by+)\b|c(?:md(?::\(?=32)?\.|exe\b|\b\W*?\///c)|hmod\b\{1,100}\?\+\{1,3\}x\b\b(?:\W*?\///|\W*\b..))\" 
  "deny,log,id:950006,severity:2,msg:'System Command Injection'"
The Core Rule Set: Virtual Patching

```xml
<LocationMatch :"/athenareg.php$">
    SecRule ARGS:pass "\;" \n    "deny,log,t:urlDecodeUni,t:htmlEntityDecode, \n    t:lowercase,t:removeWhitespace,t:removeComments"
</LocationMatch>

Or:

<LocationMatch :"/athenareg.php$">
    SecRule ARGS:pass "!/\w+" \n    "deny,log,t:urlDecodeUni,t:htmlEntityDecode, \n    t:lowercase,t:removeWhitespace,t:removeComments"
</LocationMatch>
```
Bad robots

- Based on modifiable elements of the request:
  - User-Agent header
  - URL
  - Generic headers

- Therefore:
  - Not a real security measurement
  - Offloads a lot of cyberspace junk & noise
  - Effective against comment spam

- Can use RBL:
  - Potential for FPs.
Trojans and Anti-Virus

- Check uploaded for Trojans:
- Check for access to Trojans:
  - Known signatures (x_key header)
  - Generic file management output (gid, uid, drwx, c:\)
- Major problem at hosting environments
  - Uploading is allowed.
Error conditions

- If all else fails
- Important for customer experience
- Makes life for the hacker harder
Thank You!

Ofer Shezaf
ofers@breach.com