Web Intrusion Detection with ModSecurity

Ivan Ristic <ivanr@webkreator.com>

Aim of This Talk

- Discuss the state of Web Intrusion Detection
- Introduce ModSecurity
- Introduce an open source web application firewall, consisting of Apache and ModSecurity
- Discuss what can be done to detect and prevent application attacks

Who Am I?

- Developer / architect / administrator, spent a great deal of time looking at web security issues from different points of view.
- Author of **ModSecurity**, an open source web firewall / IDS.
- Author of **Apache Security**, published by O'Reilly in March 2005.
- Founder of **Thinking Stone**, a web security company.

Talk Overview

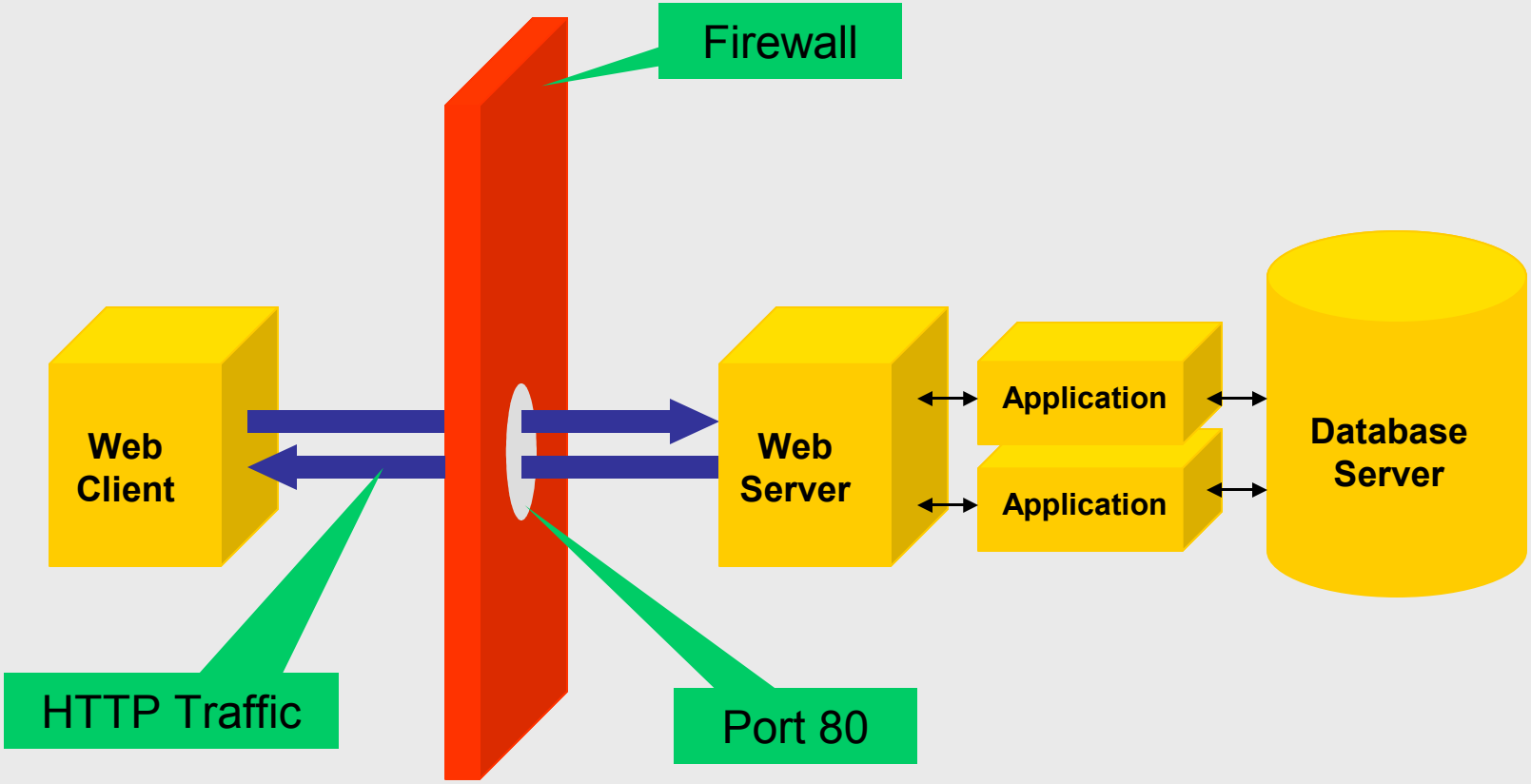
1. What is the problem?
2. Web intrusion detection approaches
3. Web application firewalls
4. ModSecurity
5. Application-based IDS

1. What Is the Problem?

What is the Problem? (1)

- The world is going Web, companies must open their systems to their customers and partners.
- Port 80 is used for everything now.
- Web applications, web services.
- Classic firewall architectures do not help any more.

Firewalls Do Not Work



What is the problem? (2)

- Web development is a mess.
- Web applications are not secure.
- Web application security field is getting there, but it's still young.
- Web servers do not provide the correct tools (e.g. auditing).
- The awareness is rising but we have a long way to go.

In the Ideal World

- Security thought out at the **beginning** of the project and **throughout**.
- Security requirements exist, **security policy** is defined.
- **Threat modelling** is used to discover threats.
- **Developers trained** in application security, a **security specialist** is on board.
- **Code reviews** are performed.

Back In the Real World

- Applications are insecure.
- Trivial vulnerabilities demonstrate serious lack of understanding of the web programming model.
- Users want features; security is an afterthought.
- Anyone with a browser can break in.

Where We Stand (1)

- Doing it right from the start is better: developers should design and develop secure software.
- But: it is not possible nor feasible to achieve 100% security. Even getting close is difficult.
- But: you have to use third-party products which are of unknown quality.
- But: you have to live with the existing systems.

Where We Stand (2)

- The application security community will work to increase awareness and educate developers.
- You can do this within your organisation.
- It will take a while.
- In the meantime, do anything you can to increase security.

What Can You Do? (1)

- By all means, if you can improve the software – do it!
- But it is more likely that you will have to attempt to increase security from the outside.
- It is not easy.
- You'll have to put insecure applications into secure environments.

What Can You Do? (2)

- Use threat modelling for deployment to determine the threats.
- Then correct architectural issues that can be corrected.
- Use network design tools to increase security by limiting exposure.

What Can You Do? (3)

- At this point many organizations stop, and prefer to keep their fingers crossed: "**It will not happen to us**".
- Intrusions are **always possible**, it is the **probability** you need to worry about.
- It depends on your circumstances – are you a high-profile, high-risk case?

What Can You Do? (4)

- **Monitoring**: know what happened.
- **Detection**: know when you are being attacked.
- **Prevention**: stop attacks before they succeed.
- **Assessment**: discover problems before the attackers do.

2. Web Intrusion Detection Approaches

What is Intrusion Detection?

- Intrusion Detection is a method of detecting attacks by monitoring traffic or system events.
- Most people mean N(etwork) IDS when they say IDS.
- But there is also Host-based IDS, and other hybrid approaches.

NIDS Applied to Web

- Traffic can be overwhelming.
- Encryption (SSL) makes data invisible.
- Compression makes data hard to see.
- Designed to work at the TCP/IP level, not as effective for HTTP.
- Evasion is a problem.
- Bottom line: NIDS is not suitable for application-level protection.

Evolution of NIDS

- Deep-inspection Firewalls: vendors are building HTTP extensions and making improvements.
- Application Firewall (a.k.a Application Gateway) is born.
- **Web Application Firewall** (WAF) is a reverse proxy with additional security-related features.

Batch Web Intrusion Detection

- Collect logs at a single location:
  - Manual collection (**cron** + **scp**)
  - Syslog
  - Spread toolkit (**mod_log_spread**)
- Run a script periodically to check the logs.
- Prevention not possible.
- **Can go back in time!**

Log-based IDS in Real-time

- Collect logs at a single location using some real time method (**syslog**, **mod_log_spread**).
- Tail and analyse the central log file in real-time.
- **SEC** (Simple Event Correlator, http://kodu.neti.ee/~risto/sec/) may be of help.
- Prevention still not possible.

3. Web Application Firewalls

Web Application Firewalls

- They understand HTTP very well.
- Can be applied selectively to parts of the traffic.
- They work after traffic is decrypted, or can otherwise terminate SSL.
- Prevention is possible.

Web IDS Strategies (1)

- Network-based:
  - Protects any web server
  - Works with many servers at once
- Web server-based:
  - Closer to the application
  - Limited by the web server API

Web IDS Strategies (2)

- Simple defence:
  - Supports a limited number of pre-defined defences
- Rule-based:
  - Uses rules to look for known vulnerabilities
  - Or rules to look for classes of attack
  - Rely on rule databases
- Anomaly-based:
  - Attempts to figure out what normal operation means

Web IDS Strategies (3)

- Negative security model:
  - Deny what might be dangerous.
  - Do you always know what is dangerous?
- Positive security model:
  - Allow what is known to be safe.
  - **Positive security model is better.**

Features (1)

- Audit logging.
- Defend from specific attacks.
- Defend from general attacks.
- Defend from brute-force attacks.

Features (2)

- Enforce client-side validation. (Excellent idea!)
- Introduce per-session restrictions.
- Learn how application works over time, then create a white list.

Evasion Issues

- Most IDS systems are watching for patterns and attackers know that.
- There are many ways to obfuscate attack content to prevent detection and still make it work.
- "**DROP/**/TABLE xyz**" is a valid SQL query in MySQL.

Evasion Techniques

- Mixed case: **DeleTe From**
- Whitespace: **DELETE FROM**
- Self-referencing filenames: **/etc/./passwd**
- Directory backreferences: **/etc/xyz/../passwd**
- Double slashes: **/etc//passwd**
- Escaping: **/etc/passw\d**
- …and many others

Impedance mismatch

- Web application firewalls parse HTTP independently from the application – that's where the protection comes from
- But often the way parsing is done is **slightly different**
- Examples:
  - PHP will ignore spaces at the beginning of variable names
  - It will also convert all subsequent spaces to underscores
  - Under some circumstances PHP treats cookies as requests parameters
- Such problems make it more difficult for web application firewalls to work out of the box
- Customisation is necessary

OSS vs. Commercial (1)

- Commercial:
  - There are many mature offerings.
  - Appliance black-boxes.
  - Can be added to network easily.
  - Very expensive.

OSS vs. Commercial (2)

- Open Source:
  - Do not have all the features of commercial offerings, but have the ones that are really important.
  - No nice GUIs yet - you have to get your hands dirty, understand how it works, and know the components well.

4. ModSecurity

ModSecurity

- Open source: **http://www.modsecurity.org**.
- GPL and commercial licensing.
- Free and commercial support available.
- 3500 downloads per month in a quiet season; growing steadily.
- Apache version (1.x and 2.x).
- Java version (Servlet Filter) at some point in the future.

Embed Into Web Server

- Inexpensive and easy to use since no changes to the network design are required.
- But works only for one web server.
- No practical impact on performance.

Apache-based Web Application Firewall

- It is a reverse proxy.
- Easy to install and configure.
- Created out of default and third-party modules:
  - mod_proxy
  - mod_proxy_html
  - mod_security

ModSecurity Features (1)

- Audit logging.
- Provides access to any part of the request (request body included) and the response.
- Flexible regular expression-based rule engine.
- Rules can be combined.
- External logic can be invoked.
- Supports unlimited number of different policies (per virtual host, folder, even a single file).

ModSecurity Features (2)

- Supports file upload interception and real-time validation (e.g. anti-virus integration).
- Anti-evasion built in.
- Encoding validation built in.
- Buffer overflow protection.
- A variety of things to do upon attack detection.

Simple Rule Examples

- Prevent JavaScript injection:

  SecFilter "<script"

- Prevent SQL injection:

  SecFilter "DELETE[[:space:]]+FROM"

Another Example

- Well-known problem in many PHP applications: **register_globals**.
- Prevent with:

```
SecFilterSelective ARG_authorised "!^$"
SecFilterSelective COOKIE_authorised "!^$"
```

Advanced Rule Example

- Prevent the “admin” user from logging from computers other than his workstation:

```
SecFilterSelective ARG_username "^admin$" chain
SecFilterSelective REMOTE_ADDR "!^192.168.0.99$"
```

Beware of False Positives!

- Some people do this:

  SecFilter bin/

- But that prevents this:

  http://www.xyz.com/cgi-bin/innocent.cgi

- **You do not have to use it in prevention mode!**
- Use detection mode only, until you are sure the rules are correct.

5. Application-based intrusion detection

Application IDS (1)

- Use the application as an IDS.
- Applications view data in context.
- The closer IDS gets to application logic – the better.
- Each software error is a potential attack.
- Log events to the application event log.
- At the very least use the response codes (**500** – error, **403** – permission problem).

Application IDS (2)

- In Java, create a security **Servlet Filter**.
- In .Net, create a **HttpModule**.
- In PHP, use **auto_prepend** to execute security code before the application begins processing.
- PHP5 (and PHP4 with the Hardened-PHP patch applied) has a special hook that allows an extension to access the parameters before script is started.

Application IDS (3)

- It is easy and fast to change libraries.
- For example, change the database abstraction library to detect SQL comments and multiple queries in a single call.

Questions?

Thank you!

Download this presentation from
http://www.thinkingstone.com/talks/