

# **A tool with a fool is still a fool**

**Sinn (und Unsinn) der Anwendungen von  
Web Application Firewalls,  
Blackbox-Web-Verwundbarkeitsscannern  
und Statischer Code-Analyse-Tools**

# 1. Einleitung, alte Ziele

- **Systemsicherheit: besser**

- > **Unices:**

- Bessere Default-Installation
    - Linux: Käfigmechanismen/MAC (AppArmor, SELinux)
    - Solaris sogar: Trusted Extensions

- > **Windows (letztes Q vielleicht nicht so)**

- XP SP2: sogar mit (hostbasierter) FW! ;-)
    - Vista/2008 Server: MIC, Least Privileges (IE 7)

- > **CPU+OS: Nicht ausführbare Datenbereiche**

- > **Compiler:**

- Canaries = Stack smash protection

# 1. Einleitung, neue Ziele

- **Das Internet:**

- > **Verbreiteter**

- DSL, Kabel
    - UMTS, HSPA

- > **Schneller**

- > **Üblicher**

- Elster
    - Online Banking
    - Ebay/Amazon
    - Bald sogar Bürgerportal

- > **Webanwendungen!**

# 1. Einleitung, neue Ziele

- Verlagerung Ziele: **Webanwendung**
  - > **Anwendung:**
    - „Steht im Netz“ !
  - > **Junge Technologie —> Beherrschbarkeit**
  - > **HTTP: „missbrauchtes Protokoll“**
    - Session
    - Authentifizierung

## 2. Prominente Beispiele

- **alt:**

- > **5/2003: T-Hack (CCC @ Telekom)**

- GET [URL]/frameset.asp?ConPK=**Vertrags#**
    - GET [URL]/FileList.asp?curPt=**../../../../**
      - MS SQL DB-Backup,
      - User-Accounts, Zugriff auf 70 Intranet-Server
    - Darunter BND, Bundesbank, türkische Botschaft

- > **10/2005: Samy-Wurm**

- Stored **XSS** @ MySpace
    - In CSS eingebettetes JS, XMLHttpRequests (AJAX)
    - 10h: 560 Freunde von "Samy", 13h: 6400, 18h: 1mio  
19h: rien ne va plus

# 2. Prominente Beispiele

- **Neue Trends, I:**

- > **CSRF : Wichtig: Eingelogggt sein**

- 1/2008 @Linksys WRT54, GET verändert config, z.B.:
      - WAN-Firewall runter (z.B. IMG SRC in E-Mail):  
`https://192.168.1.1/apply.cgi?submit_button=Firewall&change_action=&action=Apply&block_wan=1&...`
    - 1/08 Mexiko: drive-by pharming (DNS verbiegen)
    - 10/2007 @ Gmail: e-mail theft, Domain-Klau
    - 1/2009 @ Novell GroupWise WebAccess

- > **SaaS**

- 2008: XSS Crossdomain:  
Gmail ← Google-Spreadsheets

## 2. Prominente Beispiele

- **Neue Trends II:**

- > **Wellen von SQL Mass Injections in 2008**

- Botnets: >750 000 Seiten gehackt (Jan, Apr, Mai, Dez)
    - Prominente:
      - UN
      - UK Government
      - DHS
    - JS auf Seite von kompromittiertem MS SQL-Backend
    - JS infiziert PC mit Malware (IE, flash, realplayer)
    - 12/2008: Zero-Day IE!

# 3. Maßnahmen

- **Reflex mancher CTOs:**

- > **WAF!** („Firewalls helfen ja immer“)

- > **Irreführendes „F“ in WAF**

- > **„Tools drauf werfen“-Strategie:**

- *If you think technology can solve your security problems then you don't understand the problems and you don't understand the technology (Bruce Schneier)*

- > **Leider:**

- WAFs können leider nicht alles
    - WAFs brauchen Einstellungen —> nicht trivial
    - WAF ist nicht gleich WAF
    - WAFs beheben Probleme nicht an der Wurzel



# 3a. Maßnahmen: WAF

## > **Stärke: Eingabefilterüberprüfungen**

- Klassiker: XSS, SQLI
- sehr gut in „known weaknesses“
  - dir traversal, file inclusion
  - welche Pattern nach draußen
- Je nach WAF unterschiedliche weitere Hilfe (Sessions)

## > **Architektur:**

- Webserver-Modul, Reverse Proxy, Bridge?
- SSL:
  - terminieren
  - private key
- SpoF/ Availability?

# 3a. Maßnahmen: WAF

## > „Constraints“ beim Filtern:

- Blacklist läuft Hackern hinterher (arms race)
- Whitelists:
  - Erstmal Aufstellen
  - Besser: Lernmodus
  - jede Applikationsänderung: Whitelist pflegen

# 3a. Maßnahmen: WAF

## > Geht gar nicht mit WAF \*)

- GET —> POST
- Persistentes/Stored XSS (Kanal)
- Applikationslogik, Designprobleme:
  - Authorisierung (Darf User X Daten v. User Y sehen?)
  - Kaputte Authentifizierung (WAF kann's nicht selbst)
    - » schwache PWs
    - » Unbegrenzte Anzahl failed logons
    - » PW-Änderung ohne altes PW
    - » PW-Sicherheitsfrage: Lieblingsfarbe

\*) Ausnahmen bestätigen die Regel ;-)

# 3a. Maßnahmen: WAF

## > **Können nicht alle verhindern bzw. schwer:**

- Session Hijacking (Cookie-Klau)
- Falls kein eigenes Session-Management:
  - Session Fixation (=SID vorgeben)
  - Schlechte Zufälligkeit der Session-ID in App
- Falls keine Page Tokens/URL bzw. Form Encryption:
  - CSRF/Session Riding
  - Forceful Browsing wie T-Hack
  - GET <URL>.<SID>
  - Hidden-Field mit Value (Preis der Ware)

# 3a. Maßnahmen: WAF

## > Ergo, WAFs sind:

- **Zusätzlicher** Schutz
- Gut, **bekannte** Probleme zu mitigieren, wenn
  - Softwareentwicklungs-Knowhow nicht (mehr) vorh.
  - Keine Sourcen mehr (—> jad, reflector)
  - Geldmangel (betrifft höchstens mod\_security)
  - Um Zeit zu gewinnen
- Compliance (PCI DSS v1.2: Abschn. 6.6)

## > **Nicht alle WAFs können alles:**

- Vorher Specs anschauen!
- Wissen, **welche** auszubügelnden Lücken man hat ;-)

# 3b. Maßnahmen, Audit (extern)

- **Nicht viel zu sagen, außer...**

- > **Audit ist immer gut, nur:**

- Die richtigen (externen) Experten
    - Selbst: Möglichkeit, aufgedeckte Probleme zu fixen (an der Wurzel, WAF)

- > **Experte:**

- Muss „Pentester“ speziell für Webapplikationen sein
    - Völlig andere Baustelle als Infrastruktursicherheit

# 3b. Maßnahmen, Audit (extern)

- **Werkzeuge: Kategorien**
  - (1) **Automatisch**
  - (2) **Manuell**

# 3b. Maßnahmen, Audit (extern)

## • Werkzeuge

### (1) Automatisch

- Kosten viiiiel Geld (bis zu 30k€ p.A. Miete)
- Können prinzipbedingt nicht alle Probleme finden
  - Semantisches Verständnis
    - » Authentication Bypass
    - » Was sind wertvolle Infos (Info Leakage)
  - Applikationslogik, Designfehler (s.a. WAFs)
    - » Absichtliche Hintertüren
    - » `if useragent_string==iPhone then freeWLAN`
    - » `if locale==CZ then buggy_function`
    - » Darf im Portal User X Daten von User Y sehen?
    - » Cross-Domain-Issues (priv. MySpace-Fotos v. Yahoo)

> **Auch hier: A fool with a tool...**



# 3b. Maßnahmen, Audit (extern)

## • **Werkzeuge**

### **(1) Automatisch**

- Kosten viiiiel Geld (bis zu 30k€ p.A. Miete)
- Können prinzipbedingt nicht alle Probleme finden
- Selbst das, was sie finden könnten, tun sie nicht immer (falsch-negative Befunde)
- Behandlung von falsch-positiven Befunden
- HTTP 200 als Fehlerseite
- Reporting sehr unterschiedlich

# 3b. Maßnahmen, Audit (extern)

## • Werkzeuge

### (1) Automatisch. Aaaber:

- Sind als Ergänzung mehr als nützlich!
  - Geldersparnis+Automatisierung
- Beispiel:
  - Anwendung mit
    - » Ø 3 Felder=Variablen pro Seite
    - » 30 URLs
    - » Annahme: 50 Möglichkeiten Eingabe: XSS/SQLI
  - $30 * 50 * 3 = 4500$
  - 0,5 Minuten: 37,5h
  - 3750 €

# 3b. Maßnahmen, Audit (extern)

- **Werkzeuge, cont'd**

- **(2) Manuell**

- a) Intercepting Proxies
    - b) Firefox Plugins

- **> Nachteil:**

- Umfassende Test XSS/SQLI fast unmöglich

- **> Vorteil:**

- Semantisches Verständnis der Applikation
      - Geübtes Auge sieht potenzielle Angriffspunkte

- **Kombination manuell/automatisch**

# 3b. Maßnahmen, Audit (extern)

- **Werkzeuge, Aufstellung:**

## **automatisch**

HP WebInspect (SPI Dyn.)  
IBM Rat. Appscan(Watchfire)  
Cenzic Hailstrom  
NTOSpider  
Acunetix Web Vuln. Scanner  
hyperscan

Windows, Java, Unix

## **manuell**

Paros Proxy  
Burpsuite\*  
Webscarab

Intercepting Proxys

w3af  
Grendel-Scan

# 3c. Maßnahmen, safer programming

- **Das Übel an der Wurzel packen :-)**
- **Wann**
  - > **Währenddessen / Danach?**
- **Wie**
  - > **Automatisch / Manuelle (SCA / Code Review)**

# 3c. Maßnahmen, safer programming

- **Automatisch / Manuell ?**

- > **Manueller Code Review + große Applikationen: Aufwand**

- > **Rein automatisch mit SCA-Tool**

- A fool with a tool...
    - Erfordert Verständnis! => „halbautomatisch“

# 3c. Maßnahmen, SCA

## • **Währendessen/Danach**

### > **Vorsorge besser als Nachsorge:**

- Von vornherein sicher Programmieren (ggf. schulen)
  - Einsatz von Bibliotheken zur Eingabeüberprüfung (AntiXSS, ESAPI,...)
  - SCA als Unterstützung in IDE benutzen
- Externe:
  - Nachweise (PCI DSS v1.2: Abschn.: 6.3.7+6.5)
  - Code + Code-Doku!

### > **Tick später: Im QA-Prozess**

- SCA-Tool + Mensch
- (Interner) Web-Vuln.-Scan

# 3c. Maßnahmen, SCA

- **Währendessen/Danach**

- > **Code-Sicherheit posthum:**

- Teurer, hoher Aufwand
    - Technisch (je nach Geld/Zeit): „Sicherheit obendrauf“



# 4. Zusammenfassung/Ausblick

- **Heute: Webanwendungen sind das Ziel**

- > **Schutz nicht allein durch Technik**

- Die Axt im Hause ersetzt den Zimmermann nicht:  
Knowhow gefragt!
      - inhouse wo sinnvoll
      - Audit besser extern

- > **Bündel von Maßnahmen, Best Practice:**

- Sicher Programmieren von Anfang an!
    - Überprüfen durch
      - SCA + Mensch
      - Externer Audit/Scan
      - WAF, wo nötig+erfolgversprechend

• **Danke für die Aufmerksamkeit!**

> **Fragen?**

**[dirk.wetter@sogeti.de](mailto:dirk.wetter@sogeti.de)**