



**AJAX Security: Alte und neue Risiken bei Web 2.0**

Webmontag Köln | 22.1.2007 | Björn Schotte





- GF Mayflower GmbH (35 MA, Webschmiede, Premium-/High-Performance Development)
- PHP Pionier (1999: [phpcenter.de](http://phpcenter.de))
- Internet seit 1993/1994 – in BBS-Systemen und mit Usenet groß geworden
- Wurst im Portfolio: Chorizo
  
- „Freizeit“-Projekte:
  - PHP Magazin (seit 2001)
  - International PHP Conference: Head of Chair (seit 2000)

- JavaScript-Injections und Ajax (XSS)
  - Bedeutung und Verbreitung von XSS
  - Was sind JavaScript Injections?
  - Warum Web 2.0 und XSS besonders wehtut
- Ajax / Web 2.0 Probleme
  - Xml HTTP Request, Toolkits, JSON
- Ajax-Malware: XSS-Würmer und Viren
  - Kurze Virengeschichte
  - Web 2.0, Viren und Würmer
- Ajax-Malware: Web 2.0-Attacken
  - Browser Zombies
  - Intranet-Attacken
- Ajax-Applikationen absichern



## Zitate zu XSS

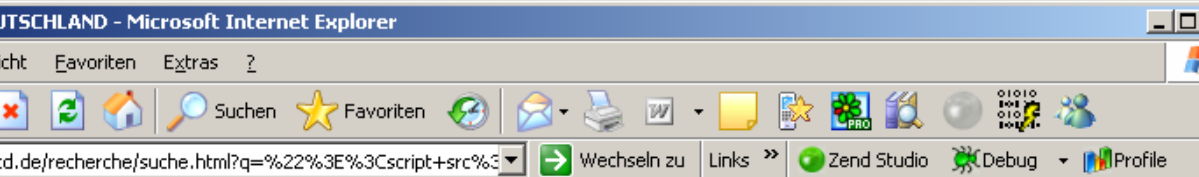
- Symantec Internet Security Threat Report:
  - 69% aller Vulnerabilities passieren in Webapplikationen
  - Web 2.0 und AJAX Sicherheitsprobleme werden wichtiger
- Mitre Corporation CVE Datenbank:
  - 21.5 % aller Lücken sind XSS-Lücken
- Von „low“ über „medium“ zu „high“ risk
- „XSS is the new hotness!“ Billy Hoffman, SPI



# Wer ist alles betroffen?

## Angela Merkels Rücktritt:

- Bundesregierung.de
- Spiegel
- Financial Times Deutschland
- Stern



Sie sind nicht eingeloggt (login)

Mo, 04. 09. 2006 18:41

### Merkel tritt zurück



Laut einer Pressemitteilung des Bundeskanzleramts trat die amtierende Kanzlerin Merkel am heutigen Nachmittag zurück. Eine Regierungserklärung liegt noch nicht vor. **mehr**

### Airbus ernennt neuen Chef für Super-Jumbo

In Toulouse ist erstmals ein Super-Jumbo zu einem Testflug mit Passagieren gestartet. **mehr**

### An der Spitze wird es eng

Während die Topschulen immer



Home > SPIEGEL Digital > Archiv

### Überraschung in Berlin

### Merkel tritt zurück

Bundeskanzlerin Angela Merkel ist überraschend zurück getreten. Über die Hintergründe ist bekannt. Die Amtsgeschäfte werden von Vizekanzler Franz Müntefering übernommen. Für eine Stellungnahme war die Regierung bis zur Stunde nicht zu erreichen.



Die Meldung wurde in Berlin mit Überraschung aufgenommen. Beobachter

Foto: REUTERS

TOP

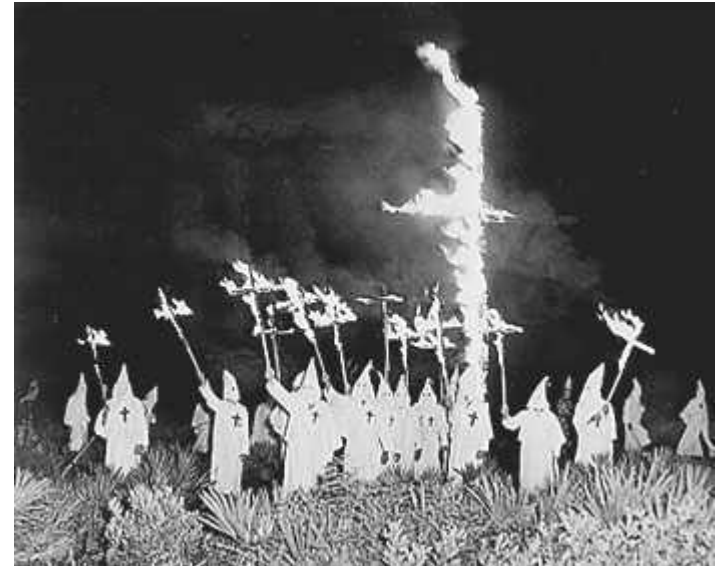
# Warum Web 2.0 und XSS besonders wehtut

- Mehr Logik im Client
  - MVC: bis zu 100% von View und Controller können im Browser stattfinden
  - Professionelle GUI-Erstellung mit JavaScript-Widgets und Komponenten
- Wachsendes JavaScript-Knowhow
  - Mehr und neue Möglichkeiten für XSS
- Neue Vektoren
  - Toolkits
  - JSON
  - RSS
  - REST / SOAP



# Wie Cross-Site-Scripting(XSS) funktioniert

- Same-Origin-Policy
  - Wenn eine Seite JavaScript mitliefert, dann ist dieses JavaScript vertrauenswürdig
  - Es darf die Seite ändern
  - Daten vom gleichen Host können gelesen werden
- XSS verletzt diese Policy
  - Es kann JavaScript in den aktuellen Seitenkontext eingeschmuggelt werden
  - Es können beliebige Requests ausgeführt werden
  - es können lokale Daten ausgelesen werden



- Password-Diebstahl
  - Firefox Password-Safe gespeichert wird (Host, Feldname, Wert)  
-> wird nach dem onLoad des Dokuments befüllt
- Datenspionage
  - CSS-History Hack per Stylesheet
  - Firefox-Plugins per chrome-img-onError



# XSS Filter und wie man sie umgeht

- Es existiert eine Vielzahl von Filtern in Anwendung und Firewall
- Problem: HTML soll möglich sein, JavaScript aber nicht
- Klassische Filter-Evasions:
  - `<IMG ""'"><SCRIPT>alert("XSS")</SCRIPT>">`
  - `<META HTTP-EQUIV="refresh" CONTENT="0;url=data:text/html;base64,PHNjcmlwdD5hbGVydCgnWFNTJyk8L3NjcmlwdD4K">`
- Code-Page-basierte Filter Evasions
  - UTF-7 (Google XSS-Hack)
  - Variable-Width Encoding Evasions
- Toolkit-basierte Filter Evasions
  - Dojo: `dojoAttachEvent`

# Exploiting Ajax



- Sicherheitskonzept hinter XmlHttpRequest:

- Same-Origin-Policy
- Ähnlich Java-Sandbox

- Umgehen der Host-Begrenzung

- DNS-Pinning
  - Wechsel der IP zwischen den Requests

- Proxy-Request-Spoofing

- Eine Seite ohne XSS:
  - <http://phprojekt.com/demo/>
- Trotzdem kann man einen XSS einschmuggeln:
  - [Web Cache Poisoning Url](#)



- Vollständiger Vertrauensverlust in JavaScript
  - Nutzer-Variablen und Funktionen
  - Systemfunktionen und Methoden
  - Formulare und (Hidden-)Variablen
  - Content, Cookie, ...
- Beispiel: Überschreiben von alert()

```
old_alert = alert;  
function myalert(str) {  
    old_alert('myalert: '+str);  
}  
alert = myalert;  
alert('Test');
```

- Prototype.js: Ajax.Request = myRequestMITM

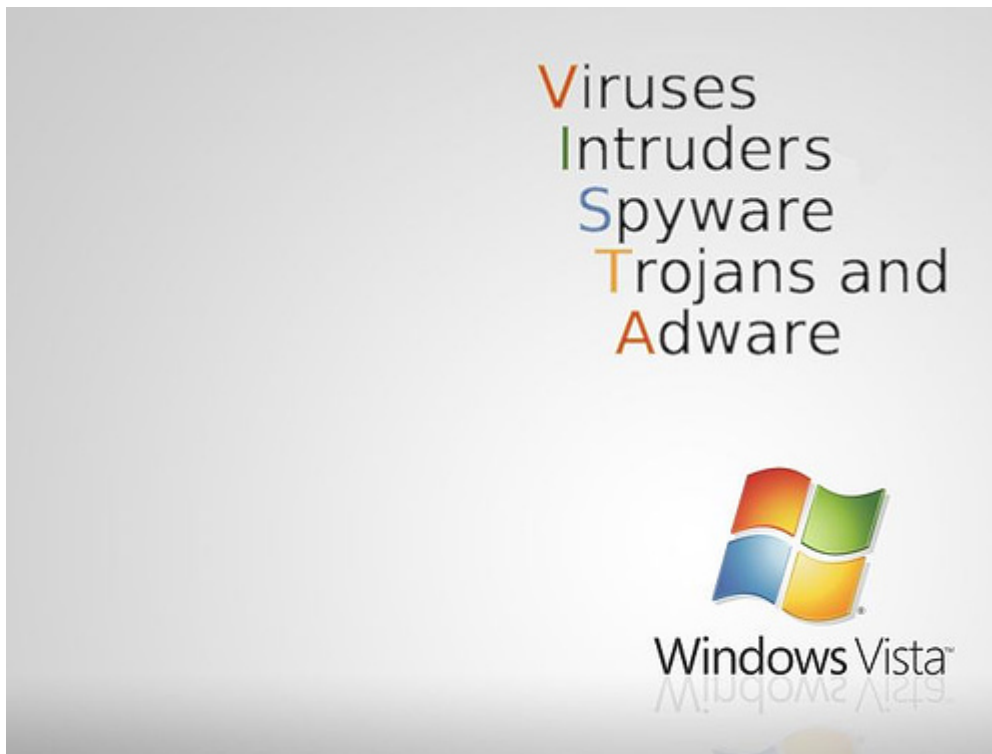


- JavaScript Object Notation
- If JSON were Food it would be Sushi
- Beispiel:  

```
{ "type": "menu", "value": "File", "items": [ {"value": "New",  
"action": "CreateNewDoc"}, {"value": "Open", "action":  
"OpenDoc"}, {"value": "Close", "action": "CloseDoc"} ] }
```
- Vorteil: kann direkt in JavaScript evaluiert werden
- Nachteil: kann direkt in JavaScript evaluiert werden
- Zusätzlicher Vektor zum Einschleusen von JavaScript

# JavaScript-Malware

- Willkommen im Browser: Viren, Intruder, Spyware und Trojaner



# Kleine Virengeschichte

- 1949: John von Neumann prophezeit sich replizierende Software
- 1986: der erste PC-Virus verbreitet sich
- 1992: Michelangelo-Virus – der erste „Medien-Virus“
- 1995: Script-Viren im Mail-Client (Outlook/Express)
- 1998: Melissa-Wurm in Outlook
- 2004: Web Application Würmer
- 2005: MySpace-Virus: Samy is my hero
- 2006: Ihre Ajax-Applikation (just FUDding)



## Was sind Web 2.0 Würmer/Viren?

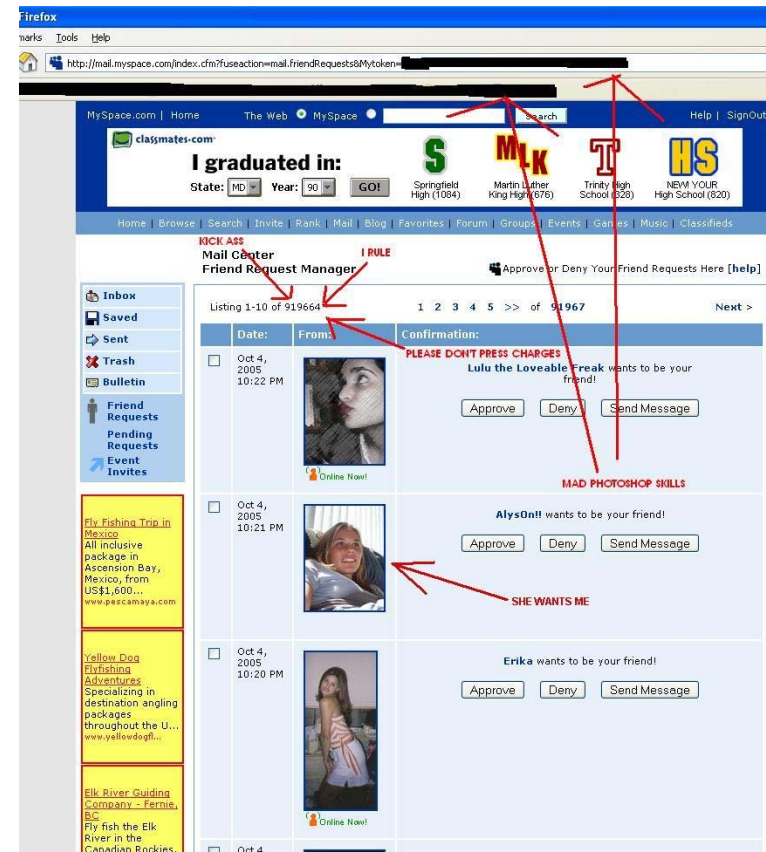
- Ort: Innerhalb einer/mehrerer Webapplikationen
- Infizierung per XSS und AJAX
- Verbreitung per AJAX, Formular, Link, RSS
- Der Browser führt den Virus aus
- Applikation ist Speicherort des Virus
- Der erste wirkliche Cross-Platform-Virus!
- Kritische Payloads sind möglich
  - Datenänderung
  - Datenspionage (Kreditkarten usw)
  - Durchführung von Transaktionen (Aktienkauf)
- Wer von Ihnen loggt sich immer aus?
- Auch bei Google Mail?
- Bei vorhandenen Logins kann im Hintergrund infiziert werden

- Warum funktionieren Viren auf Web 2.0?
  - LAW – Die Nutzer stellen Inhalte für Nutzer (lokale Vermehrung)
  - Mash-Ups – Es werden Applikationsteile von anderen Websites eingebunden (verteilte Vermehrung)
  - Neue Verbreitungswege wie RSS, SOAP, REST (verteilte Vermehrung)
  - Ajax-Möglichkeiten stehen auch Hackern offen (Infizierung und Vermehrung)
  - Es gibt mehr Schnittstellen zwischen Client und Server(n) (mehr Möglichkeiten zur Infizierung)



# Samy is my Hero – der MySpace Wurm

- MySpace, zu dem Zeitpunkt 5-wichtigste Internetseite weltweit
- Samy hatte nur 73 Freunde. Zu wenig.
- Aber er hatte JavaScript-Knowhow.
  - JavaScript im eigenen Profil, daß Samy per XMLHttpRequest zum Freund macht
- MySpace war vorbereitet:
  - Guter JavaScript-Filter
  - - aber mit Filter-Evasions!
  - CSRF-Schutz per ZufallsHash
  - - per XHR umgangen!
- Nach 20 Stunden hatte Samy mehr als 1.000.000 Freunde!

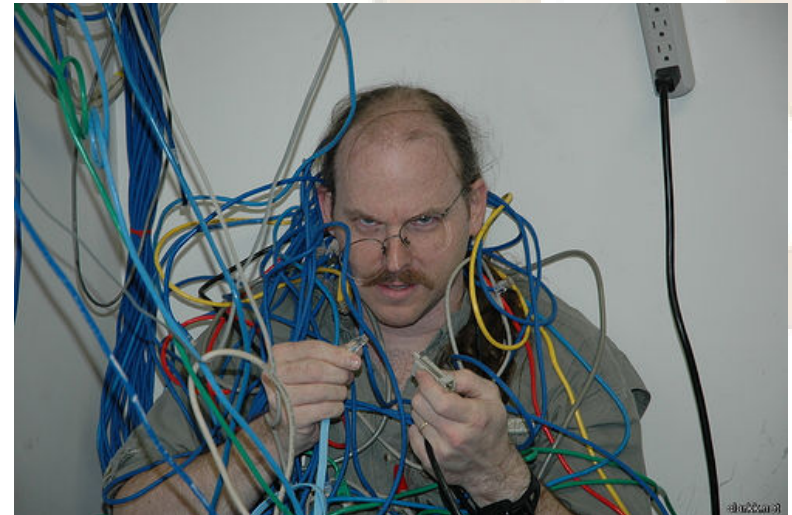


# Browser Zombies

- JavaScript bedeutet Kontrolle über den Browser
- JavaScript muss nicht statisch sein
- Es könnte auch ein Mensch auf der anderen Seite sein, oder ein intelligentes Script
- Besuchen Sie uns!  
<http://mayflower.de/>
- Wir besuchen Sie auch!  
[Browser Exploitation Framework](#) (BeEF, Browser Exploitation Framework)
- COMET!



- Hinter der Firewall ist das Paradies
  - Ungepatchte Software
  - Default-Passworte
  - Ungeschützte Dienste
- Aber: das Intranet kann gescannt werden
  - 1. die lokale Adresse ermitteln
  - 2. das lokale Class-C-Netz auf Port 80 scannen
- Erkennung der Infrastruktur
- Individuelle Attacken
  - Siehe Proxy-Exploit
  - bekannte Lücken: Linksys-WRT-Router
  - <http://fritz.box/>



# Ajax Applikationen sichern



- XSS ist kein kleines Risiko mehr
  - mehr Funktionalität im Client bedeutet mehr Potentielle Lücken
  - Um so mehr JavaScript kann, um so mehr kann auch XSS
- JavaScript- und integrierte Toolkits haben regelmässig Lücken
- Alle 2 Monate eine neue XSS-Lücke bekannt
  - „Security is a process, not a product“



- Der Weg zu 100-prozentiger Sicherheit im Web:
  - 25% Web Application Firewalls wie mod\_security
  - 25% Web Security Scanner
  - 25% Source Code Audits und Entwicklung mit Security in Mind
  - ... da fehlt doch was
- Vermeidung von kritischen Funktionalitäten
  - HTML-Eingaben
  - Vertrauen auf externe Quellen



- Dem Client nicht vertrauen:
  - Header, Cookies, Get, Post
  - Auch nicht der Applikationslogik im Client
- Daten Filtern:
  - Formate erzwingen:
    - Es sind nur bestimmte Zeichen valide (0-9,)
    - Es ist nur ein Format gültig (089/24 20 54 13)
    - Es ist nur eine bestimmte Länge möglich
  - Bei Nutzung escapen
    - Entities für HTML
    - Hochkomma für SQL
    - Slashes für JavaScript
  - Meist in einer Zeile machbar



## Fazit

- JavaScript etabliert sich als Angriffsplattform
- Die Möglichkeiten und Gefahren von JavaScript-Malware werden zur Zeit deutlich unterschätzt
- Es wird mehr Viren und Würmer auf Web-Applikationen geben
- Sie werden Applikations- und Domainübergreifend sein
- Es wird gezielte XSS-Attacken geben
- Intranet-Attacken werden über JavaScript stattfinden
- Sicherheit wird bei der Applikationsentwicklung fürs Web in der Vordergrund rücken



**Vielen Dank für Ihre Aufmerksamkeit**

Björn Schotte  
Mayflower GmbH  
Pleichertorstrasse 2  
97070 Würzburg  
+49 (931) 35965 - 15  
schotte@mayflower.de

