

Telekom - BROKEN W3B CHALLENGE 2.0

Final Pitch

von Andreas Zinkl

Köln – CMD+O – 15. November 2018

Agenda

1. Informationsgewinnung
2. Eindringen in das System
3. Schwachstellen- & Risikobewertung
4. Fazit und Handlungsempfehlungen

1. Informationsgewinnung

1.1 Analyse verfügbarer Services

- Analyse aller offener Ports unter der Adresse <http://80.158.6.18> mit **nmap**
 1. Offene Ports finden
 2. Services hinter den Ports analysieren
- Resultat der Analyse:
 - Port 80: NodeJS (ExpressJS) Server
 - Port 8888: Apache Server httpd (Version 2.4.18)



1.2 Analyse der Homepage

- Analyse des Source-Codes der Homepage unter der Adresse <http://80.158.6.18>
- Resultat der Analyse - Port 80:
 - Kommentare innerhalb des Source-Codes
 - Informationen über Infrastruktur → NoSQL-Datenbank → MongoDB?
 - Informationen über Mitarbeiternamen
- Resultat der Analyse - Port 8888:
 - Zugang zur Seite durch .htaccess-Authentifizierung gesperrt

2. Eindringen in das System

(inkl. maximaler Informationsgewinnung)

2.1 NoSQL-Injection

- Mit Hilfe von NoSQL-Injection wird versucht Zugang zu erhalten
- Analyse eines Anmeldeversuchs zeigt einen JSON-Body mit den Attributten:
 - user
 - pass
- NoSQL-Injection:

```
{  
    "user": {"$not": {"$in": ["user", "guest"]}},  
    "pass": {"$gte": ""}  
}
```



2.2 Local-File-Intrusion (LFI)

- Analyse des “Secure-Project-Assistant“
 - Download über ein PHP-Skript „grabfile.php“
 - Datei-Upload-Service „Secure Image Uploader“ mit der Adresse <http://160.44.192.138/>
- Analyse des PHP-Skriptes ergibt LFI-Schwachstelle über ein URL-Attribut

`http://80.158.6.18:8881/grabfile.php?
key=5f39e90d93baad53fd7288e8a6fa9586&
download=../../../../etc/passwd`



2.3 Schadcode-Upload

- Erlaubt Upload von „kleinen“ Dateien im Format PNG, JPG, JPEG
- Upload eines **PHP-Meterpreter-Reverse-TCP** Schadcodes
 1. Erstellen der Reverse-TCP Payload mit Metasploit
 2. Erstellen des Fotos mit „`cat foto.png exploit.php > exploit.png`“



2.4 Meterpreter-Shell-Session

1. TCP-Listener auf dem eigenen Rechner wartet auf TCP-Verbindung
2. Ausführen des PHP-Schadcodes über die LFI-Schwachstelle
3. Herstellen einer Meterpreter-Shell-Session

- Ergebnis:
 - Zugriff auf das System als Nutzer mit Rolle “www”
 - Shell-Zugriff
 - Ermitteln aller Informationen über das System
(verfügbare Services & Infrastruktur)



2.5 Versuch: Privilege Escalation

- Versuch zum erreichen von Administrationsrechten schlägt fehl
- Kernel-Exploits sind nicht ausführbar
- Einzig „main“-Datei ist ausführbar
(Pfad: /home/flagmaster/final_step/main)

2.6 Analyse des Programms „main“

- Verwendung von Shell-Tools
 - GNU-DEBUGGER (GDB)
 - HEXDUMP
- Ergebnisse
 - Buffer-Overflow nicht umsetzbar
 - Ermittlung des Admin-Users „*admin*“ durch debuggen
 - Keine weiteren Informationen aufgrund von *Segmentation-Faults* beim Laden der *password.txt*

3. Schwachstellen- und Risikobewertung

3. Schwachstellen und Risikobewertung

Bewertungsskala:

- 0 = kein Risiko
- 1-2 = geringes Risiko
- 3-4 = mittleres Risiko
- 5-6 = hohes Risiko
- 7 = sehr hohes Risiko

| Nr. | Schwachstelle | Eintrittswahrscheinlichkeit | Schadensrisiko | Gesamtrisiko |
|-----|---|-----------------------------|----------------|--------------|
| 1. | Kommentare in den Website-Quellcodes über die interne Infrastruktur | 3 | 3 | 3 |
| 2. | Verwendung der Standard-“.htaccess“ Authentifizierung von Apache | 4 | 4 | 4 |
| 3. | Veraltete Service Version (z.B. Apache) | 5 | 6 | 6 |
| 4. | Unzureichende Validierung beim Upload von Bildern | 5 | 6 | 6 |
| 5. | Keine bzw. unzureichende Validierung der Eingaben – NoSQL Injection | 7 | 7 | 7 |
| 6. | Keine Verwendung von HTTPS | 7 | 7 | 7 |
| 7. | Local-File-Intrusion (LFI) Schwachstelle | 7 | 7 | 7 |

4. Fazit und Handlungsempfehlungen

5. Fazit und Handlungsempfehlungen

- Aktueller Status zur Sicherheit: **Extrem unsicher**
- Schwerwiegende Schwachstellen ermöglichen einen raschen Zugang in das System
- **Handlungsempfehlung:**
 - Updates der Services (Apache)
 - Verstecken von Service-Informationen (Port-Scanning)
 - Keine Verwendung von Standard-Authentifizierungsmechanismen wie .htaccess
 - Prüfen von zulässigen Eingaben
 - NoSQL-Injection vermeiden
 - LFI-Schwachstelle vermeiden
 - Verwendung aktueller Sicherheitsstandards durch Anwendung des HTTPS-Protokolls

Vielen Dank für Ihre Aufmerksamkeit!

Noch Fragen?

- Kontakt:
- Email:
 - andreas.zinkl@st.oth-regensburg.de
 - andreas.zinkl@trinovative.de
- Twitter: @zinklandi



OSTBAYERISCHE
TECHNISCHE HOCHSCHULE
REGENSBURG



LABORATORY FOR
INFORMATION SECURITY



trinovative

Anhang: Ermittelte Flags

- Ermittelte Flags:
 - Flag 1: TSEC{5f39e90d93baad53fd7288e8a6fa9586} (NoSQL-Injection)
 - Flag 2: TSEC{24c2dc999e66bba7c356b91334a25308} (Local-File-Intrusion)
 - Flag 3: TSEC{aa791a74d55bbd786833b6e490dd6677} (Upload-Tool)
 - Flag 4: TSEC{1d55c399453b885892ea51820b460c40} (Eindringen über Reverse-TCP-Verbindung)
 - Flag 5: Finales Flag ist unbekannt (Pfad im System: /home/flagmaster/final_step/flag.txt)
 - Dieses konnte nicht ermittelt werden.
 - Hierbei stellte das „main“-Executable ein Hindernis dar, da das Passwort nicht ermittelt werden konnte
 - Benutzer „adm1n“ konnte mit Hilfe des verfügbaren gdb durch debuggen ermittelt werden