Grundpraktikum Netz- und Datensicherheit

Thema:

Angriffe in geswitchten Netzwerken

Lehrstuhl für Netz- und Datensicherheit
Ruhr-Universität Bochum

Versuchdurchführung: Raum ID 2/168

Betreuung: Marcus Niemietz
Zusammengestellt von: Michael Psarros, Hannes Oberender, Florian Bache, Patrick Meier, Endres Puschner, Dominik Noß, Klaus Meyer, Alexandra Dubovic
Stand: 25. November 2021

Inhaltsverzeichnis

1 ARP - Adress Resolution Protocol

ARP ist ein Netzwerkprotokoll, welches die Zuordnung von Internet (IP) zu Hardware-Adressen (MAC) ermöglicht. Obwohl der Einsatz von ARP in anderen Umgebungen möglich ist, beschränken wir uns im Folgenden auf den Einsatz von ARP in einer reinen IPv4/Ethernet-basierten Netzwerkumgebung.

1.1 Grundlagen

1.1.1 Netzwerkschichten

In unserer Anwendung sind die Schichten 2 (Sicherung/Netzzugang) und 3 (Vermittlung/Internet) des ISO/OSI bzw. TCP/IP Referenzmodells relevant.[**?**]

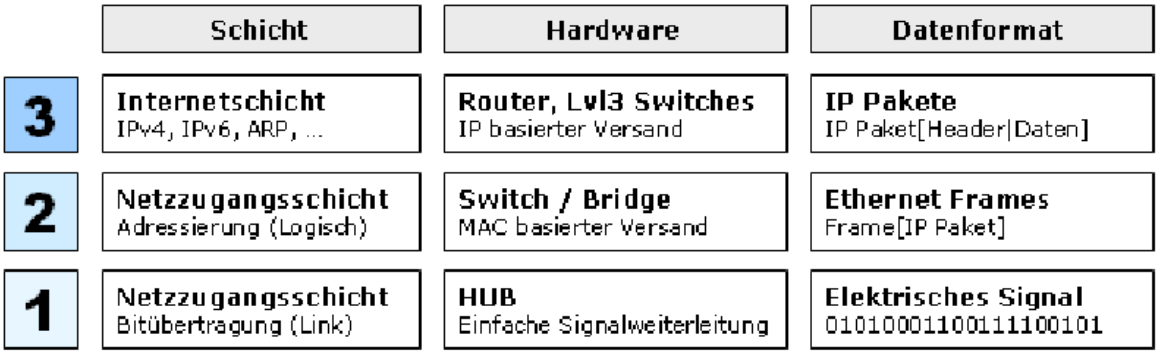

Abbildung 1: ISO/OSI-Schichtenmodell

1.1.2 MAC - Media Access Control

Ethernetfähige Netzwerkgeräte erhalten bei der Produktion eine weltweit eindeutige Hardwarekennung, die als MAC-Adresse bezeichnet wird. MAC-Adressen lassen sich sehr gut als Folge von sechs Zahlen im Bereich 0-255 in hexadezimaler Notation darstellen, z.B. 00:04:23:63:AF:03. Dabei wird in den ersten drei Stellen der jeweilige Hersteller kodiert, die restlichen drei Stellen dienen der eindeutigen Identifizierung der Hardwarekomponente. Dadurch soll sichergestellt werden, dass unter normalen Betriebsbedingungen jedes Gerät in einem Netzwerkabschnitt von allen anderen unterschieden werden kann.

MAC Adressen stehen in den von uns verwendeten Netzwerken, welche TCP/IP auf der Protokollebene verwenden, in keinem direkten Bezug zu den hier verwendeten IP-Adressen. Es lassen sich also ohne weitere Hilfsmittel keine Rückschlüsse von der IP- auf die MAC-Adresse eines Geräts oder umgekehrt treffen.

1.1.3 Ethernet - Adressierung im lokalen Netzwerk

Zum Datenversand in unserem Ethernet basierten Netzwerk werden alle IP Pakete in sogenannte Ethernet-Frames verpackt.

Abbildung 2: Ethernet-Frame

Dabei wird die Zieladresse (IP) des IP Pakets ausgelesen, entsprechend aufgelöst, und das dafür vorgesehene Adressfeld des Ethernet-Frame mit der Hardwareadresse (MAC) des Empfängers gefüllt. Außerdem wird die Hardwareadresse des Absenders in ein weiteres Feld eingetragen.

Befindet sich der Empfänger außerhalb des lokalen Subnetzes, so wird das Frame an einen Router im lokalen Netz adressiert. Dieser übernimmt dann die Weiterleitung der Daten.

Das so adressierte Ethernet-Frame wird nun auf Schicht 1 umgewandelt und auf das Übertragungsmedium (Kabel) geschickt. Abhängig von der Netzwerkstruktur müssen wir nun zwei mögliche Versandwege betrachten:

1. **Einfaches Netzwerk mit gemeinsamen Übertragungsmedium**

   Handelt es sich um ein einfaches Ringnetzwerk oder werden HUBs als Kopplungselemente eingesetzt, so wird das Ethernet-Frame, welches nun als Folge elektrischer Signale im Kabel unterwegs ist, alle im Netz vorhandenen Geräte erreichen.

2. **Durch Switches unterteiltes Netzwerk**

   Wird das Netzwerk durch Switches gekoppelt, wird das Ethernet-Frame zuerst den Switch erreichen. Dieser wird das empfangene Frame auslesen, und die Zieladresse bestimmen. Ist das Zielgerät an einem der Netzwerkports des Switches angeschlossen, oder soll das Frame weitergeleitet werden, so schickt der Switch das Ethernet-Frame nur an den entsprechenden Port. Alle anderen im Netz angeschlossenen Geräte erhalten somit das Ethernet-Frame nicht.

Ist das Ethernet-Frame nun an einem Gerät angekommen so wird der Header des Frames ausgelesen. Stimmt die Zieladresse des Frames mit der eigenen Hardwareadresse des Geräts überein, oder ist die Zieladresse eine so genannte Broadcast-Adresse, wird das Frame akzeptiert und weiterverarbeitet. Stimmen die Adressen nicht überein, so wird das Paket (im normalen Betriebsmodus) verworfen.

1.2 ARP

Die oben beschriebene Form der Adressierung stellt uns nun vor einige Probleme. So muss ein Gerät A neben seiner eigenen Hardwareadresse auch die Adresse eines Geräts B kennen, um ein Ethernet-Frame für den Datenversand zu B erstellen zu können. Da sich aus IP Adressen, die z.B. in einer der oberen Protokollschichten durch einen Benutzer eingegeben wurden, keine MAC Adressen herleiten lassen, muss ein Weg gefunden werden, dieses Problem zu lösen.

Zwar besteht theoretisch die Möglichkeit, A mit einer Liste alle Geräte im selben Netzbereich und deren IP—MAC Adresszuordnungen auszustatten, in der Praxis stoßen wir dabei aber, z.B. bei großen Netzwerken oder Netzwerken mit starker Gerätefluktuation, schnell an aufwandsbedingte Grenzen. Es ist daher notwendig, die Zuordnungen möglichst ohne Benutzerinteraktion dynamisch zu erstellen. An dieser Stelle setzt ARP an.

ARP wird im RFC826: „Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware“ der Internet Engeneering Task Force (November 1982) spezifiziert, und existiert heute in verschiedenen Implementierungen.

Wir betrachten nun ein Beispiel: PC15 (192.168.0.15) möchte ein IP Paket an PC12 (192.168.0.12) senden, die IP Adresse von PC12 ist PC15 bekannt.

1. PC15 erzeugt ein ARP Paket, das eine Anfrage nach der Hardwareadresse von PC12 und seine eigene IP—MAC Zuordnung enthält. Dieses Paket nennt man einen ARP request (Who has?).
2. PC15 erzeugt ein Ethernet-Frame, das an die Broadcast Adresse (FF:FF:FF:FF:FF:FF) gesendet wird und das in 1. erzeugte ARP Paket als Nutzlast enthält.
3. Alle Hosts im lokalen Subnetz erhalten das Ethernet-Frame und werten die Adressen aus.

4. PC12 erkennt anhand der Anfrage im ARP Paket, dass diese an ihn adressiert wurde.
5. PC12 erstellt seinerseits ein ARP Paket, das seine eigene IP—MAC Zuordnung enthält. Dieses Paket nennt man eine ARP response (x is at). Dieses Paket sendet PC12 an PC15.
6. PC15 erhält das Paket und kennt nun die Hardwareadresse von PC12. PC15 kann nun die Frames direkt an PC12 adressieren.

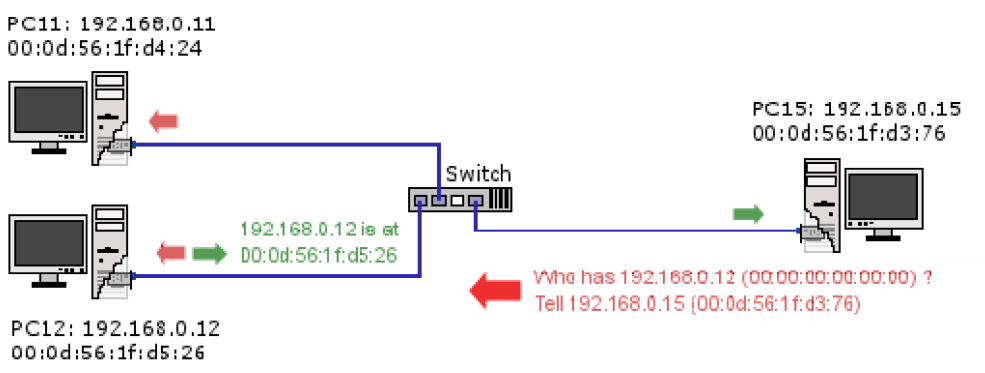

Abbildung 3: ARP Protokoll

1.3 ARP Caching

Um das oben beschriebene Verfahren möglichst effizient zu gestalten, werden einmal erfragte Adresszuordnungen lokal gespeichert, damit im Bedarfsfall keine neuen Anfragen gestellt werden müssen.

Die Speicherung erfolgt in so genannten ARP-Cache Tabellen, die sowohl dynamisch erzeugte, also durch Anfragen erhaltene, als auch statische, also von Hand eingetragene, Zuordnungen enthalten können.

Auszug einer ARP-Cache Tabelle:

```
Schnittstelle: 192.168.2.4 --- 0x10004
  Internetadresse      Physikal. Adresse   Typ
  192.168.2.1          00-30-f1-fc-5b-4f   dynamisch
```

Bevor nun das Protokoll wie oben beschrieben durchlaufen wird, erfolgt zuerst ein ARP-Cache lookup. Wird die gesuchte Zuordnung in der Tabelle gefunden, so kann diese direkt verwendet werden.

In der Praxis werden beim ARP Caching Einträge, welche innerhalb der letzten 20 Minuten erstellt wurden, verwendet. Ist ein entsprechender Eintrag zu alt, so wird ein neues ARP Request erstellt und die Tabelle entsprechend der Response aktualisiert. Dadurch soll verhindert werden, dass Geräte, die im Netz nicht mehr vorhanden sind (z.B. ausgeschaltet), weiter in den Tabellen geführt werden. Außerdem wird so verhindert, dass fehlerhafte Einträge die Kommunikation dauerhaft beeinträchtigen.

1.4 Schwächen von ARP

Das ARP Protokoll hat in den meisten verfügbaren Implementierungen Schwächen, welche sich hervorragend für Angriffe auf das lokale Netzwerk ausnutzen lassen:

- Eingehende ARP Response Nachrichten werden akzeptiert und der ARP-Cache entsprechend modifiziert, ohne dass ein ARP Request gesendet wurde.
- Es ist nicht möglich, die Identität der Gegenseite zu überprüfen. ARP Nachrichten werden im Vertrauen auf ein korrektes Verhalten der Gegenseite akzeptiert und ausgewertet.

Im Folgenden befassen wir uns mit einem der möglichen Angriffe, der sich diese Schwächen zunutze macht.

2 ARP Spoofing

Bei ARP Spoofing handelt es sich um einen „Man in the middle“ Angriff mit dem Ziel, die Kommunikation zwischen zwei (oder beliebig vielen) Parteien im lokalen Netzwerk abzufangen und entsprechend auszuwerten, oder zu manipulieren. Da die Kommunikation zwischen einer Partei im lokalen Subnetz und einer Partei in einem entfernten Netz über einen Router/Gateway, der sich ebenfalls im lokalen Subnetz befindet, stattfindet, ist der Angriff nicht auf Kommunikationsverbindungen im lokalen Subnetz beschränkt.

Um den Angriff auf eine Kommunikation zwischen zwei Parteien – Alice und Bob – erfolgreich durchführen und daraus einen Nutzen ziehen zu können, wird der Angreifer – Oscar – drei verschiedene Abläufe koordinieren müssen:

1. **ARP Spoofing** Beim ARP Spoofing selbst versucht Oscar jeweils seine eigene Hardwareadresse (MAC) in den ARP-Cache von Alice und Bob eintragen zu lassen, und zwar so, dass Oscars Hardwareadresse der jeweiligen IP von Alice oder Bob zugeordnet wird. Im Erfolgsfall sollte dies also so aussehen:

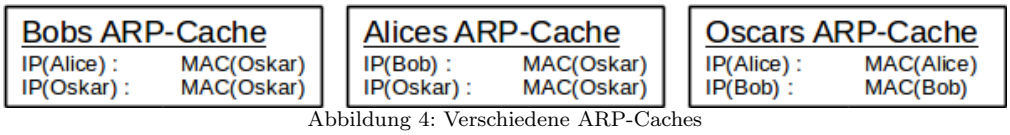

Abbildung 4: Verschiedene ARP-Caches

   Ausgehend von der vorherigen Konfiguration des ARP Cache hat Oscar unterschiedliche Möglichkeiten dies zu erreichen, worauf wir später noch eingehen werden. Wichtig ist an dieser Stelle, dass Bob nun alle Pakete an Alice mit der Hardwareadresse von Oscar adressieren wird. Erreicht ein solches Ethernet-Frame einen Switch, wird dieser das Paket direkt an Oscar weiterleiten, Alice erhält das Paket nicht. (Dasselbe trifft in umgekehrter Richtung ebenfalls zu).

   Letzteres trifft gleichfalls auch für Netzwerke ohne Switch zu, da Alice das Paket zwar erhalten aber aufgrund der falschen Hardwareadresse verwerfen wird. Somit kann Oscar auch in solchen Netzen – die er zwar im Promiscuous mode seiner Netzwerkkarte auch ohne ARP Spoofing Angriff abhören könnte – verhindern, dass Bob und Alice direkt kommunizieren.

2. **Paketweiterleitung** Oscar erhält nun alle Pakete, die von Alice an Bob und von Bob an Alice gesendet werden. Weder Alice noch Bob erhalten irgendwelche Pakete voneinander. Es ist also schon abzusehen, dass die abhörbare Kommunikation, die Oscar erreicht, recht dürftig ausfallen wird, da beide wahrscheinlich über ein Protokoll kommunizieren werden, welches zuerst versuchen wird eine Verbindung zu etablieren, bevor interessante Daten (z.B. Passwörter) gesendet werden.

   Oscar muss also dafür sorge tragen, dass alle abgefangenen Pakete weitergeleitet werden und die Kommunikation so aufrechterhalten wird. Das kann Oscar auf verschiedene Arten erreichen (IP-Forward, Proxy, Bridging). Hauptsache ist, dass die Verzögerung im Transportweg entsprechend kurz ausfällt, da Oscar sonst riskiert, die Verbindung durch Timeouts zu unterbrechen.

3. **Auswerten der gesammelten Daten** Um einen Nutzen aus den gesammelten Kommunikationsdaten zu ziehen, muss Oscar eine Software einsetzen, welche die Daten anhand von genau definierten Filtern nach interessanten Inhalten durchsucht, oder zumindest alle gesammelten Daten in einfach lesbarer Weise ausgeben kann.

   Dies kann, muss aber nicht sofort beim Abfangen der Kommunikation geschehen, da Oscar

einfach alle anfallenden Daten zur späteren Analyse speichern kann.

3 Angreifer-Tools für ARP-Spoofing

3.1 BetterCAP

Bettercap ist eine umfangreiche Sammlung von Tools für verschiedene Angriff-Szenarien, geschrieben in der GO Programmiersprache. Mit Bettercap ist es möglich Angriffe in WI-Fi Netzwerken, auf Bluetooth Low Energy Geräte, auf kabellose HID Geräte und in Ethernet Netzwerken durchzuführen. In diesem Versuch geht es um "Man in the middle"(MITM) Angriffe. Alle benötigten Funktionen um ARP Spoofing MITM Angriffe durchzuführen sind bereits in Bettercap integriert, darüber hinaus noch weitere Arten der Spoofings auf DNS und DHCPv6.

Außerdem enthalten sind ein leistungsstarker Portscanner, Netzwerksniffer und vieles mehr. Alternativ zu Wireshark kann mithilfe des Befehls **net.sniff on** der eingebaute Netzwersniffer benutzt werden, um Daten beispielsweise nach Passwörtern zu filtrieren. Um eine Übersicht über die Features vom Bettercap zu bekommen, können sie Referenzen dieses Versuchs verwendet werden. Weitere Funktionen von Bettercap beziehen sich auf die Manipulation von HTTP, HTTPS und TCP. Das Programm ist verfügbar unter Linux, Mac OS, Windows und Android. Es ist bereits auf allen Rechner im Netzlabor installiert.

In vielen MITM Szenarien ist es notwendig, on-the-fly kryptographische Werte zu manipulieren, wie beispielsweise die Generation von gefälschten Server-Zertifikaten. Dafür stellt Bettercap verschiedene Arten von Digest Authentications, Injections, Zertifikatenfälschung zur Verfügung. Bettercap unterstützt, sofern SSL Support beim Kompilieren aktiviert wurde, das Sniffen von durch SSL/TLS gesicherten Verbindungen. Natürlich ist es Bettercap dabei nicht möglich eine sichere Verschlüsselung der gesendeten Daten in Echtzeit zu brechen, oder gültige Zertifikate zu erstellen, weshalb auf einen Trick zurückgegriffen werden muss:
Dem Client wird ein gefälschtes Zertifikat angeboten, das auf den ersten Blick die Daten des Zertifikats des Originalservers enthält, aber mit neuem Schlüsselmaterial versehen wurde und von der Bettercap-eigenen Certificate Authority (CA) signiert wurde. Es handelt sich also um ein valides Zertifikat, dessen CA der Client aber nicht vertraut. Wird die dadurch auftretende Warnmeldung vom Benutzer ignoriert, kann der Angreifer sich erfolgreich in die Kommunikation zwischen Client und Server einbinden und dem Client eine durch SSL/TLS gesicherte Verbindung vortäuschen. Dieses Fenster ist in der Abbildung 5 zu sehen.

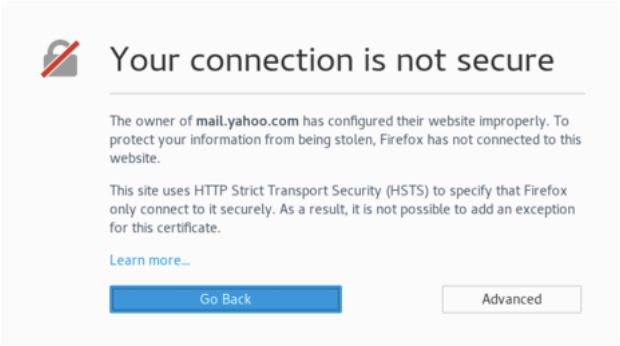

Abbildung 5: Fehlermeldung des Browsers, da er dem Zertifikat nicht vertraut.

Das verwendete Zertifikat für die CA wird normalerweise von Bettercap bei Programmstart automatisch erstellt, kann aber auch vom Anwender vorgegeben werden. Um dieses Feature anzuwenden, sollte der Befehl aus der Abbildung 6 ausgeführt werden.

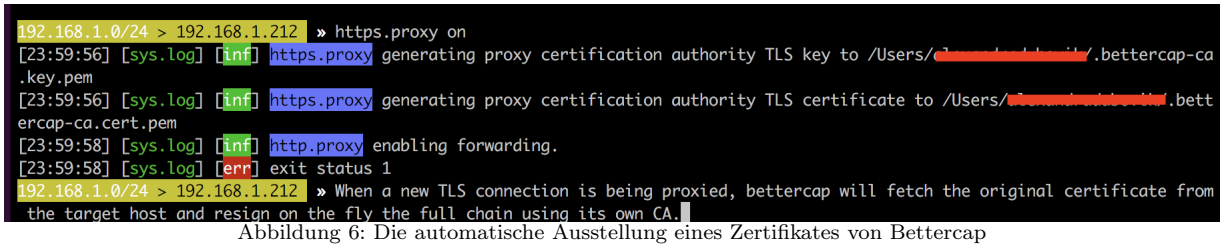

Abbildung 6: Die automatische Ausstellung eines Zertifikates von Bettercap

Es ist empfehlenswert alle angehängte Links aufmerksam durchzulesen, damit keine Schwierigkeiten bei der Benutzung von Bettercap entstehen[1]. Das Tool ist relativ intuitiv und mit genügender Vorbereitung können alle Bettercap-Aufgaben schnell gelöst werden. Außerdem lohnt es, sich mit der offiziellen Dokumentation auf der Webseite von Bettercap vetraut zu machen. [2]
Jede Gruppe kann selbst wählen, ob Bettercap GUI oder Bettercap mittels Shell während des Versuchs benutzt wird. Falls Sie die die Caplets der GUI anwenden wollen, dann sollten Sie auch ihre Inhalte umfangreich im Bericht erläutern. Der Versuch ist aber so gedacht, dass alle benötigte Befehle einzeln eingegeben werden können. (Also die Anwendung von Caplets ist NICHT notwendig, aber ist erlaubt)

3.2 About Proxying

Ein Proxy ist eine Kommunikationsschnittstelle in einem Netzwerk, die als Vermittler arbeitet. Anstatt Daten zwischen zwei Parteien A und B direkt auszutauschen, werden sie stattdessen über den Proxy geleitet. A adressiert ein Datenpaket an B, und schickt es an den Proxy. Der Proxy leitet es an B weiter. Die Antwort von B, die an A adressiert ist, erreicht dann zunächst den Proxy, welcher sie an A weiterleitet. Proxies existieren auf vielen ISO Ebenen, in diesem Versuch geht es um TCP und HTTP(S). BetterCap implementiert für HTTP/HTTPs sowie TCP sogenannte transparente Proxies. Transparente Proxies sind solche, die ohne Konfiguration der Clients A oder B funktionieren.

Sie werden in Bettercap benutzt, um HTTP/HTTPS oder lowlevel TCP Datenfluss zu sniffen und zu manipulieren. Unter anderem kann JavaScript in vom Opfer besuchte Webseiten injected werden um z.B. alle Bilder zu ersetzen oder Cookies zu stehlen. In diesem Versuch werden alle Gruppen einmal die Rolle des Angreifers einnehmen, um mit einem transparenten Proxy die Verbindung zu manipulieren. In Abbildung 7 ist die Position des Angreifer-Proxy zwischen einem Host und dem Internet zu sehen.

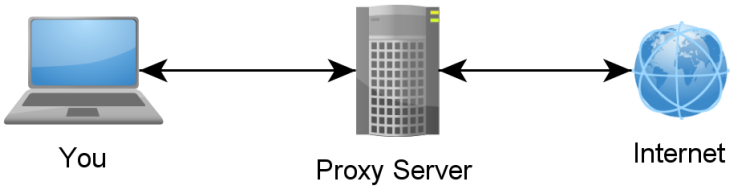

Abbildung 7: Proxy

Wenn ein Proxy für gewünschte Verbindung mittels Bettercap gesetzt wird, wird Bettercap sich auch über Spoofing und benötigten Iptables-Regeln kümmern, um den Datenfluss des Opfers zum Proxy weiterzuleiten. Die Bettercap Module dafür können mittels des Befehls **help http.proxy** oder **help https.proxy** ausgelesen werden.

[1] Bettercap Usage Examples: `https://www.cyberpunk.rs/bettercap-usage-examples-overview-custom-setup-caplets`, Bettercap JS-Injection Tutorial:`https://www.youtube.com/watch?v=m-H9W9ZOzBI`
[2] Offizielle Webseite: `https://www.bettercap.org/intro/`

3.3 Arp-sk – Swiss-Knife tool for ARP

Ist ein Tool zur einfachen Erzeugung von ARP Nachrichten. Das Tool ist in zwei Versionen verfügbar. Für Linux basierend auf der libnet und Windows als winarp-sk (WinPcap).

3.4 ARP0c

Ist ein ARP Spoofing Tool, das im Gegensatz zu arp-sk selbstständig die Kontrolle über die Erstellung und den Versand der ARP Nachrichten übernimmt. Außerdem verfügt ARP0c über eine integrierte bridging Funktion, um abgefangene Pakete weiterzuleiten. Von ARP0c existiert, neben der von uns hier verwendeten Linux Version, eine Windows Version, die sich in der Anwendung leicht von der Linux Version unterscheidet.

3.5 Dsniff

ist eine Sammlung von Netzwerkanalysetools. Neben dem Passwortsniffer dsniff und dem ARP poisoning Modul arpspoof sind noch diverse andere Programme enthalten. Dabei handelt es sich um spezialisierte Sniffer wie mailsnarf und urlsnarf, Tools für SSH1 mitm (sshmitm) Angriffe, connectionkiller (tcpkill) und lustige Anwendungen wie webspy, welches ermöglicht, automatisch alle Webseitenbesuche eines Opfers selbst am eigenen Browser nachzuvollziehen. Für unsere Anwendung reichen arpspoof und dsniff aus. Anders als ARP0c und Ettercap ist dsniff nicht selbst in der Lage abgefangene Pakete weiterzuleiten. Hier muss wieder auf IP-Forwarding oder andere Software (z.B. fragrouter) zurückgegriffen werden.

4 Gegenmaßnahmen

Maßnahmen gegen ARP-Spoofing sind theoretisch an vielen Stellen möglich, wegen praktischen Aspekten wie Arbeitsaufwand oder der weiten Verbreitung des ARP Protokolls aber nur sehr eingeschränkt praktisch umsetzbar. Lösungen, die keine Änderung des Protokolls beinhalten, sind meist nur von eingeschränktem Nutzen, da es zwar hilfreich ist einen bereits laufenden Angriff zu erkennen, aber eventuell bereits wichtige Daten abhanden gekommen sind.

4.1 Schutz durch Manuelle Eingabe

Der einzig sichere Schutz gegen ARP-Spoofing besteht derzeit wohl in der manuellen Wartung aller an der Kommunikation beteiligten ARP-Cache Tabellen. Trägt man vorher verifizierte IP—MAC Adresszuordnungen von Hand permanent ein und verbietet weitere Änderungen, so kann man zumindest für kleine Netzbereiche oder an Stellen, die größtmögliche Sicherheit erfordern, diese herstellen. Da der Aufwand aber mit jedem weiteren Eintrag stark zunimmt, ist diese Methode in der Praxis nur sehr eingeschränkt anwendbar. Microsoft Windows (bis XP) war bis ins Jahr 2004 weiterhin verwundbar, da statische Einträge jederzeit durch dynamische überschrieben werden konnten.

4.2 ARP Spoofing entdecken – Netzwerk- und Hostmonitoring

Erfolgreiches ARP-Spoofing erfordert oft, dass der Angreifer ständig gefälschte ARP Nachrichten an viele Empfänger versendet. ARP Nachrichten gehören zwar zum normalen Datenaufkommen eines Netzwerks, sind aber abhängig von der Anzahl der im Netz vorhandenen Geräte, und bedingt durch ARP-caching eher selten.

Durch Mitschneiden des gesamten Netzwerkverkehrs (z.B. am Switch), oder auch durch einfaches Abhören an einem Interface (Promiscuous mode), kann ein gehäuftes Auftreten von ARP Nachrichten meist recht gut erkannt werden. Durch Auswertung der ARP Nachrichten kann dann auf einen eventuellen Angreifer geschlossen werden.

Arpwatch überwacht die Kommunikation zwischen dem lokalen und anderen Hosts. Zu jedem kontaktierten Host wird die IP—MAC Zuordnung mit einer Zeitmarke gespeichert. Ändert sich diese Zuordnung, so wird ein zusätzlicher Eintrag erstellt, und der Anwender über die Änderung der Zuordnung informiert. Dieser Ansatz kann natürlich nur dann Erfolg haben, wenn zu Beginn der Aufzeichnungen kein laufender Angriff falsche IP—MAC Zuordnungen erzeugt.

5 Hinweise

5.1 Voraussetzungen für die Teilnahme

- Grundlagenkenntnisse bei der Bedienung von grafischen und Text-basierten Linux-Systemen.
- Die Versuchsanleitung muss gelesen und verstanden werden. Ferner müssen Sie die Kontrollfragen beantworten können.
- Zum besseren Verständnis ist es hilfreich, die angegebenen Links anzuschauen. Eventuelle Nachfragen können gleich zu Beginn des Versuchs geklärt werden.
- Machen Sie sich mit den benötigten Tools vertraut, damit Sie die erforderlichen Kommandos beherrschen.

5.2 Versuchsaufbau

Auf den Rechnern ist Ubuntu Linux und alle vorher erwähnten Tools installiert. Sie bearbeiten den Versuch mit Kommandozeilen-Befehlen. Die GUIs können auch benutzt werden, ist aber nicht zu empfehlen. Der Befehl für Bettercap GUI lautet:

```
sudo␣bettercap␣-caplet␣http-ui
```

Nach der Eingabe in Terminal bekommen Sie ein Links, der im Webbrowser die Bettercap GUI öffnet. Falls Bettercap GUI nicht startet, es soll die Hinweise auf `https://www.bettercap.org/usage/` beachtet werden. Für diesen Versuch wird explizit die Arbeit im Shell empfohlen, denn die leichter zu bedienen ist. Für diesen Versuch sind Sie unter dem Account „student“ angemeldet. Das Passwort für root-Rechte lautet 12345.

Die Beschreibungen der einzelnen Schritte enthalten die Vorgabe in **fetter Schrift** und die Erläuterungen dazu jeweils in normaler Schrift. Falls Sie etwas nicht verstehen, fragen Sie bitte zu Beginn den Betreuer. Ziel ist es, dass Sie die Vorgabe jedes Schrittes erfüllen und das Ergebnis für sich dokumentieren. **Schreiben Sie bei jedem Schritt das Zwischenergebnis, die Durchführung und exakte Befehlsangaben auf, damit der Bericht vollständig den Versuch abdeckt.** Unvollständige Berichte werden vom Versuchsleiter nicht akzeptiert, was dazu führen kann Sie die Schritte dann noch mal durchführen müssen!

5.3 Schriftliche Versuchsauswertung

Jedes Team fertigt eine schriftliche Auswertung an. Diese sollte insbesondere die bei jedem Schritt verwendeten Befehle und eine Erläuterung zu deren Ausgabe enthalten. Also unbedingt dokumentieren,

was Sie bei der Versuchsdurchführung getan haben und **warum** Sie dies getan haben. Alle Fragen der Aufgabenstellung sollten durch ihre Dokumentation beantwortet werden.

6 Versuch

6.1 Schritt 1

Sniffen Sie mit Wireshark im Netzwerk. Was für Pakete erhalten Sie? Was ist anders als beim vergangenen „nmap und Wireshark"-Versuch? Dokumentieren Sie das Ergebnis.

Welche Funktionen von Wireshark (z.B. Filter, Statistiken) haben Sie benutzt? Dokumentieren Sie diese mit Screenshots.

6.2 Schritt 2

Schalten Sie Wireshark ein. Verbinden Sie sich mittels eines Webbrowsers mit dem Server im Netz (oder führen Sie „ping -c 1 hostname" aus). Schauen Sie dann mittels „arp -n" in den ARP-Cache des Clients. Was steht da?

Finden Sie den ARP-Request und die dazugehörige Response in Wireshark. Notieren Sie den benutzten Filter und machen Sie Screenshots der Pakete. Falls wireshark nicht ausgeführt wurde, kann ein Eintrag im ARP Cache mit dem Befehl „sudo arp -d [ip.des.Ziels]" entfernt werden, um danach einen neuen erzeugen und aufzeichnen zu können.

6.3 Schritt 3

Bevor eine ARP-Attacke gestartet werden kann, muss der Angreifer-Rechner das Opfer pingen, sodass seine IP-Adresse und MAC-Adresse bei dem Angreifer in der ARP-Tabelle erscheinen. Mit dem Befehl 'arp -n' muss wieder festgestellt werden, dass alle benötigten Adresse in der Tabelle enthalten sind. (Opfer, Server, Gateway). Um deren IPs herauszufinden, sollte man den Netzplan des Labors benutzen.

6.4 Schritt 4

ACHTUNG! Hier gibt es ein paar Fehler, wodurch Sie versehentlich alle Gruppen von ihrer Arbeit abhalten können! Es reicht teilweise ein Tippfehler eines einzelnen Zeichens. Wird dies dennoch (mutwillig) getan, kann die betroffene Gruppe vom Versuch ausgeschlossen werden. Lesen Sie folgendes aufmerksam:

- Der Parameter 'arp.spoof.internal' SOLL immer auf 'false' gesetzt sein und wird während des Versuchs nicht geändert. Per default ist der Wert schon 'false'. Versehentliches Einschalten dieses Parameters führt zum Poisoning des gesamten Netzwerkes und blockiert die Angriffe aller anderen Gruppen.
- Sowohl Target 1 als auch Target 2 müssen jeweils **genau** einem Opfer (entweder **Partnergruppe** oder **Server**) zugewiesen werden!
  - Es dürfen NICHT beide Opfer nur EINEM Target zugewiesen werden.
  - Es darf KEIN Target LEER bleiben.
  - Jedes Target enthält EXAKT EINE IP.
  - IPs werden ohne Schrägstriche, Sonderzeichen o.Ä. geschrieben! Beispiel: **192.168.1.6**

Prüfen Sie die korrekte Setzung der Parameter in Bettercap mittels des Befehls '**active**' oder '**get arp.spoof.***'.
Welche Targets sind im 'arp.spoof.targets' Feld? Falls die leer sind, dann ist nach default als Target ganze Netzwerk eingerichtet! Alle defaults-Einstellungen eines Moduls können mittels **help *Name des Moduls*** ausgelesen werden. In der Bettercap-Shell sieht es so aus:

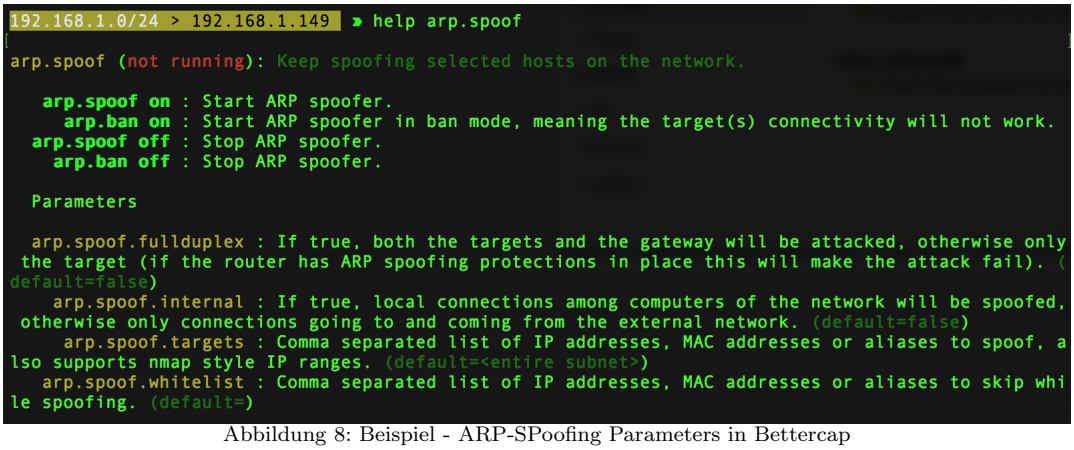

Abbildung 8: Beispiel - ARP-SPoofing Parameters in Bettercap

- Achten Sie darauf, dass die Parameternamen **richtig geschrieben werden**. Sollten Sie sich vertippen, so wird ein neuer Parameter mit falsch geschriebenem Name erstellt, welcher für das Programm keine Relevanz hat. Angriffe nutzen dann die (falschen) default-Werte anstelle der echten Parameter. Für 'arp.spoof.targets' ist das besonders kritsch, weil der default-Wert dieses Parameters 'ganzes Netzwerk' ist.

Aufgabenstellung. Ziel: **Führen Sie einen ARP-Spoofing Angriff aus.**
Dieser Schritt muss in **Absprache** mit Ihrer Nachbargruppe gemacht werden und wird von beiden Gruppen abwechselnd ausgeführt.
Ein einfacher Weg, die Netzwerk-Konfiguration inklusive der ARP-Tabellen zurückzusetzen, ist über die Oberfläche von Ubuntu. Klicken Sie oben rechts auf das Symbol für das Netzwerk und klicken Sie auf "Kabelnetzwerkverbindung 1".

- Gruppe 1 (Angreifer): Starten Sie Bettercap mit root-Rechten (sudo) und geben sie mit '-iface [Interface]' als weiteren Parameter das Interface an, auf welchem gesnifft werden soll. Dies könne Sie mit dem Konsolenbefehl 'ip r' ermitteln. Ermitteln Sie die in Bettercap verfügbaren Parameter des Moduls 'arp.spoof.' mit dem Befehl 'help arp.spoof'. Welche davon können für Sie von Nutzen sein? Dokumentieren Sie Ihre Ideen. Erklären Sie die benutzten Parameter und erläutern Sie die Ziele aller Befehle einzeln. Hinweis: Lediglich zwei Parameter sollen gesetzt werden.
  Es ist verboten, den Angriff ohne Erlaubnis des Betreuers zu starten.
  Nachdem die Versuchsleitung die Konfiguration des Angriffs überprüft hat, können Sie zuerst das eingebaute Sniffer-Modul mit 'net.sniff on' starten oder Wireshark als Sniffer benutzen. Auf solche Weise werden die Zugangsdaten des Opfers in Bettercap oder in Wireshark abgefangen und dargestellt. Danach starten Sie den Angriff mit 'arp.spoof on'. Sniffen Sie die Verbindungen Ihrer Nachbargruppe und lesen Sie Benutzername und Passwort des Opfers aus. Dokumentieren Sie die ARP-Poisoning-Attacke in Wireshark, indem Sie zwei ARP-Response-Pakete finden, die zur ARP-Attacke gehören (ein Paket pro Target), und Sie anhand der darin enthaltenen

relevanten Felder des Pakets beschreiben, wie der Angriff funktioniert.

Beschreiben Sie in Ihrem Bericht, welche Rolle die in Wireshark/Bettercap Sniffer mitgeschnittenen Pakete spielen. **Stellen Sie den Angriff als Grafik dar**, in welcher die drei Parteien (Server, Angreifer, Opfer) sowie die Nachrichten enthalten sind. Finden Sie mittels Wireshark-Filter die Login-Daten der angegriffenen Partei im Mitschnitt des Datenverkehrs und fügen Sie diese in Ihren Bericht ein.

- Gruppe 2 (Opfer): Warten Sie auf den Beginn des Angriffs durch die Angreifergruppe. Surfen Sie danach auf einen verwundbaren Server und loggen Sie sich ein. Sie können den lokalen Server `http://server.netzlabor` verwenden (akzeptiert alle User/Passwörter). Alternativ können Sie die Website `http://vulnweb.com` verwenden, welche eine unsichere Login-Form für solche Zwecke bereitstellt. Überprüfen Sie wieder den ARP-Cache. Was stellen Sie fest? Wie lässt sich der Angriff aus Opfersicht bemerken?

Um den Angriff zu beenden, benutzen Sie den Befehl:

```
arp.spoof␣off
```

Oder einfach '**quit**' bzw. '**q**' im Terminal. Beenden Sie den Angriff und wechseln Sie die Rollen.

6.5 Schritt 5

Wie im Schritt 4, aber diesmal sniffen Sie eine verschlüsselte Verbindung! Bettercap ist in der Lage on-the-fly gefälschte Zertifikate zu generieren. Lesen Sie aufmerksam den Link: `https://www.bettercap.org/modules/ethernet/proxies/https.proxy/`.
Welche Adresse sollen sie in **https.proxy.address** eintragen? Welcher Befehl ist dafür geeignet?
Bettercap weißt darauf hin, dass es automatisch neue Zertifikate erzeugt, wie in Abbildung 9 zu sehen.

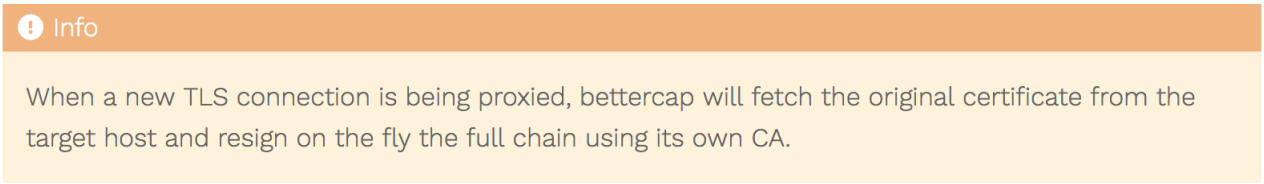

Abbildung 9: Bettercap https-proxy

- Gruppe 1: Sniffen Sie die verschlüsselten Verbindungen ihrer Nachbargruppe. Setzen Sie **net.sniff.local** auf 'true' und lassen Sie die **net.sniff on** laufen. Bettercap entschlüsselt die TLS-Verbindung für Sie und präsentiert die HTTP-Daten in der Konsole des Fensters. Starten Sie das Sniffing mittels Klick auf Start → Sniffing im Wireshark. Zeigen Sie wieder die aufgezeichneten Login-Daten Ihrer Partnergruppe. Finden Sie die Daten sowohl in Wireshark als auch in Bettercap? In welchen Programmen sind die Login-Daten lesbar, und warum (nicht)?
- Gruppe 2: Benutzen Sie nun HTTPS für den Login auf der WebApp-Attrappe. Überprüfen Sie die Sicherheitsinformationen des Browsers (das farblich markierte Feld im linken Bereich der Adressleiste anklicken) zunächst VOR dem Angriff der Partnergruppe und anschließend NACH dem Angriff. Was hat sich im Zertifikat vor und nach dem Angriff geändert? Welche Fehlermeldungen werden gezeigt? Wie weißt der Browser den Nutzer auf den Angriff hin, und was muss der Benutzer tun, damit der Angriff erfolgreich passiert?
  Notieren Sie sich den Hash-Fingerprint der beiden Zertifikate (Original und Angreifer). Stellen

Sie fest, dass der Issuer vom gefälschten Zertifikat mit dem Bettercap Issuer übereinstimmt. Den Bettercap Issuer kann man in der Doku auf der Bettercap Webseite finden.

Wechseln Sie Rollen und wiederholen Sie den Vorgang.

Hinweis: Um die etwaige Fehlermeldung "SEC_ERROR_REUSED_ISSUER_AND_SERIAL" zu unterbinden, können sie Firefox neustarten oder einen anderen Webbrowser verwenden. Dies tritt auf, wenn sich die Seriennummer eines Zertifikats ändert, *während Firefox es gerade für eine Verbindung nutzt.* Können Sie sich vorstellen, aus welchem Grund der Browser diese Meldung anzeigt? Wie könnte ein Angreifer sie umgehen?

6.6 Schritt 6

Sichern Sie ihren Rechner! Benutzen Sie „arp -s“ und legen statische Routen für den Router und die Server. Wiederholen Sie Schritt 4. Dokumentieren Sie die Aufzeichnungen der Wiederholung. Wie weit ist der Angriff jetzt möglich? Überlegen Sie sich auf welcher Seite der Angriff noch funktioniert oder wie er eingeschränkt wurde. Diskutieren Sie: Handelt es sich hier um eine für (Firmen-) Netze geeignete Sicherheitsmaßnahme?

6.7 Schritt 7

Löschen Sie zuerst alle statischen Einträge aus Schritt 6.

Vergewissern sie sich, dass *arpwatch* auf dem richtigen Interface ethX (X = 0...7) lauscht[3]:

```
sudo␣service␣arpwatch␣stop
sudo␣arpwatch␣-i␣ethX␣-d
```

Probieren Sie Schritt 4 wieder aus. Welche Aufgabe erfüllt arpwatch und welche nicht? Wo können Meldungen von arpwatch angesehen werden? Was passiert?[4]

Beschreiben Sie die Bedeutung der Meldungen von arpwatch. Informationen dazu können Sie mittels **man arpwatch** in der Man-Page nachlesen.

Ende des Versuchs

6.8 Bonusaufgabe für Begeisterte: JavaScript-Injection

In dieser Aufgabe werden fortgeschrittenere Features von Bettercap untersucht. Nun sollen Sie injizierten JavaScript benutzen, um z.B. Cookies zu stehlen, eigenen Code in die HTML Seiten einzufügen, auf einen gewünschte Seiten zu redirecten oder ein Bild auf der Webseiten einschleusen. Suchen Sie sich zwei bis vier der genannten mögliche Tricks aus, führen Sie den Angriff durch, und dokumentieren Sie das Ergebnisse. Recherchieren Sie mögliche JS-Snippets, die von Nutzen sein könnten.
Lesen Sie die Ausgabe vom Befehl **help https.proxy** aufmerksam. Welche Parameters sollten Sie verwenden um eine Injection durchzuführen? Initialisieren Sie das Feld *http.proxy.address* mit IP-Adresse ihres Rechners. Warum ist dies notwendig?

- Gruppe 1: Im Ordner */home/student/Praktikum/spoof/* gibt es ein JS Datei 'havefun.js', die lediglich eine Zeile enthält: *alert(1);*. Setzen Sie den Pfad zu dieser Datei im Parameter

[3] Informationen über angeschlossene Interfaces erhält man zum Beispiel mit **ifconfig**
[4] Der Befehl **tail -f** könnte hilfreich sein, um Änderungen zu beobachten.

https.proxy.injectjs. Starten Sie den Angriff mittels **https.proxy on**. Bei einem erfolgreichen Angriff sollte im Web Browser des Opfers ein Popup mit dem Inhalt "1" erscheinen.
Bemerkung: ARP-Spoofing muss immer 'on' bleiben. Warum?
Wenn Sie zusätzlich andere Parameters geändert haben, dokumentieren Sie das. Falls der Angriff funktioniert, probieren Sie die Datei 'havefun.js' zu ändern. Schalten Sie ARP-Spoofing, sowie Proxying aus, während der Bearbeitung der Datei. Nach dem Speichern, schalten Sie die Module erneut ein. Dokumentieren Sie, welche Auswirkung bei Nachbargruppe erreicht werden soll. (z.B Bilder, JS-Code in HTML Seite, redirecting, Cookie usw.)

- Gruppe 2: Besuchen Sie die HTTPS-Webseiten und akzeptieren Sie Fehlermeldung erneut.
  Bemerkung: Manchmal sollten Sie den Browser neustarten, damit der Angriff der Nachbargruppe erfolgreich ist. Falls die Nachbargruppe etwas ins HTML-Page eingefügt hat, können Sie mit der Tastenkombination:

  `Strg+U`

  HTML-Page anschauen und nach gewünschte Code suchen.

Nachdem genügende Anzahl von Injections durchgeführt war, wechseln Sie die Rolle.

Hinweis: Angriffe auf Web-Applikationen könne aus verschiedenen Gründen scheitern. Beispielsweise könnte es sein, dass Sie ihren injizierten Code korrekt verfasst haben, jedoch die Website aufgrund der Content Security Policy die Ausführung von JavaScript kategorisch unterbindet. Dann schlägt der Angriff fehl, da der eigentlich korrekte Payload nicht vom Browser ausgeführt wird.

Probieren Sie verschiedene Seiten aus, und achten Sie auf die Ausgaben der Developer Tools im Browser, um Ihren Angriff zu debuggen.

7 Kontrollfragen

Sie sollen mindestens folgende Fragen beantworten können (bevor der Versuch beginnt):

1. Was ist der Unterschied zwischen einem Hub und einem Switch?
2. Was ist der Unterschied zwischen einer MAC- und einer IP-Adresse?
3. Wozu dienen die Protokolle ARP, RARP, IP, TCP, UDP, ICMP, DNS, SNI, HTML?
4. Wie funktioniert ein Proxy?
5. Wozu dienen die Protokolle HTTPS, SSH, TLS/SSL?
6. Wer ist ein CA?
7. Was ist eine Halbe/Duplex-Verbindung?
8. Was sind geswitchte Netzwerke? Welche Rolle spielen sie beim Sniffen?
9. Wie snifft man trotzdem in geswitchten Netzwerken?
10. Welcher Befehl vom Bettercap setzt die IP-Adressen, um zwischen zwei Parteien zu spoofen?
11. Unter der Verwendung welches Befehls wird Bettercap ein on-the-fly Zertifikat erstellen?
12. Welche bekannten Protokolle übertragen Passwörter im Klartext?
13. Kann man nur im Ethernet sniffen?
14. Würden Sie einem Besucher bei Ihnen einfach so einen Netzwerkanschluss anbieten um z.B. zu drucken oder schnell was im Netz nachzuschauen?
15. Würden Sie das auch bei einem sicherheitskritischen Netz machen?

Literatur