
Diplomarbeit

Klassifikation und Modellierung von Angriffen auf Wireless LAN

erstellt von

Robert Lehmann

Technische Universität Dresden

Fakultät Informatik
Institut für Systemarchitektur
Professur Rechnernetze

Betreuer: Dipl. -Inform. Stephan Groß
Verantwortlicher Hochschullehrer: Prof. Dr. rer. nat. habil. Dr. h. c. Alexander Schill

Eingereicht am: 31. Dezember 2006

Platzhalter Aufgabestellung

Inhalt

	Einleitung	9
1	Wireless LAN	11
1.1	Veränderte Bedrohungslage für Netzwerke durch WLAN	12
1.2.1.1	WEP-Authentifizierung	17
1.2.1.2	WEP Vertraulichkeit	18
1.2.1.3	WEP Integrität	20
1.2.1.4	WEP-Verfügbarkeit	22
1.2.1.5	Konsequenz aus den bekannten Schwächen in WEP	23
1.2.1	WEP – Wired Equivalent Privacy – Sicherheitslücken und Implementierungsfehler	16
1.2.2.1	Schwachstelle im 4-Way-Handshake bei WPA/WPA2	24
1.2.2	WPA – WiFi Protected Access und 802.11i (WPA2)	24
1.2.3	Fazit	28
1.2	Sicherheitsmechanismen für WLAN	14
2	Attack Trees	29
2.1	Struktur von Attack Trees	30
2.2	Enhanced Attack Trees	34
2.3.1	Umsetzung der Erweiterungen im Support Tool for Enhanced Attack Trees	37
2.3	Support Tool for Enhanced Attack Trees	35
3	Angriffe auf Wireless LAN	39
3.1	Angreifermodelle	39
3.2.1	Aktive Suche nach Wireless LANs	42
3.2.2	Passive Suche nach Wireless LANs	43
3.2	Finden und Aufdecken von Wireless Networks	42
3.3.1	Belauschen von Übertragungen	45
3.3.2	Angriffe auf den geheimen Schlüssel	45
3.3.3	Angriffe auf den Schlüsselbitstrom	51
3.3.4	Man-in-the-Middle-Angriffe	54
3.3	Angriffe gegen die Vertraulichkeit	44
3.4	Angriffe gegen die Integrität	57

3.5	Angriffe auf die Verfügbarkeit	58
3.6	Fazit	59
4	Angriffe in der Realität	61
4.1	Versuchsanordnung	61
4.2	Angriffe ohne Verschlüsselung – Mac Spoofing	63
4.3.1	mit Aircrack-ng	65
4.3.2	mit aircrack-ng und airodump-ng	66
4.3.3	Abwehrstrategien gegen Passive Schlüsselangriffe	66
4.3	Passiver Angriff auf den geheimen Schlüssel	64
4.4.1	ARP Replay über aireplay-ng	67
4.4.2	Paket Erstellung packetforge-ng und ChopChop	72
4.4.3	Aktiver Angriff mit Wesside	75
4.4	Aktiver Angriff auf den Schlüssel	67
4.5.1	Airsnarf – Hotspot Phishing	75
4.5.2	AirPWN – Antwort - Modifikation	77
4.5	Man in the Middle	75
4.6	Zusammenfassung	80
4.7	Fazit	81
5	Generierung von Policies für Bro wireless aus Attack Trees	83
5.1	Einführung wireless IDS	83
5.2	Grundlagen Bro IDS	84
5.3	Analyse BRO wireless	84
5.4	Beispiel Policy zur Erkennung von Angriffen mit ChopChop	85
5.5	Möglichkeiten und Grenzen der automatischen Generierung von Policies aus Attack Trees	88
6	Zusammenfassung	91
	Anhang A	93
	Anhang B	93
	Anhang C	94
	Anhang D	94
	Literaturliste	99

Tabellen und Abbildungen

Abbildung 1-1: Beziehung WLAN, 802.11, Wi-Fi	11
Abbildung 1-2: vereinfachter 4-Way-Handshake	25
Abbildung 1-3: PTK Generierung	26
Abbildung 2 -1 Einfacher Attack Tree.	30
Abbildung 2-2 Baum mit Attributen	32
Abbildung 2-3: Enhanced Attack Tree	35
Abbildung 2-4: Oberfläche des Support Tool for AT	36
Abbildung 3-1: Darstellung zu Angreifermodellen	42
Abbildung 3-2: GUI AirSnort	46
Abbildung 3-3: Optionen von aircrack-ng	49
Abbildung 3-4: Aircrack: aircrack test.ivs	50
Abbildung 4-1: Attack Tree für MAC Spoofing	63
Abbildung 4-2: Airsnort nach erfolgreich gebrochenem Schlüssel	65
Abbildung 4-3: airodump-ng: nach 8 Minuten Laufzeit	68
Abbildung 4-4: aireplay Aufruf	68
Abbildung 4-5: DeAuthentication mit aireplay	68
Abbildung 4-6: Reaktion des ARP Replay Modus	69
Abbildung 4-7: Anstieg der in der Data-Spalte deutet auf Erfolg.	69
Abbildung 4-8: Attack Tree für Angriff auf den geheimen Schlüssel	71
Abbildung 4-9: Paketauswahl mit ChopChop	72
Abbildung 4-10: Ausgabe tcpdump	73
Abbildung 4-11: Aufruf Packetforge-ng	73
Abbildung 4-12: Attack Tree Angriff mit ChopChop und Packetforge	74
Abbildung 4-13: Captive Portal von Aircsnarf	76
Abbildung 4.-13: Attack Tree zu Aircsnarf	77
Abbildung 4-14: AirPWN mit Google Startseite	78
Abbildung 4-15: Attack Tree zu airPWN	79
Abbildung 5-1: ChopChop einzeln betrachtet	86
Tabelle 1-1: Der Niedergang von WEP	17
Tabelle 1-2: WLAN Sicherung	16
Tabelle 4-1: Konfiguration des WLAN	62

Einleitung

Seit Jahren steigt die Anzahl an privat oder kommerziell betriebenen Wireless LANs stetig an. Leider, so scheint es, lassen Vorzüge wie Bequemlichkeit oder mobiler Zugang den Benutzer sehr schnell die Sicherheitsrisiken vergessen. Tatsächlich wurden in den vergangenen Jahren immer wieder eklatante Sicherheitslücken in den alten Sicherheitseinrichtungen von Wireless LAN bekannt. Obwohl sich an dieser Front technisch einiges getan hat, ist nach wie vor der überwiegende Teil der betriebenen Netze verwundbar gegen eine Vielzahl von Angriffen.

Ziel dieser Arbeit wird es sein, Aussagen über die tatsächlich von Angriffen bzw. Angriffsimpementierungen ausgehende Bedrohung für den Endverbraucher zu treffen. Sofern möglich, werden im Rahmen der einzelnen Betrachtungen auch Strategien zur Abwehr entwickelt und vorgestellt.

Kapitel eins wird sich mit den grundlegenden Techniken und ihren Unzulänglichkeiten befassen. Dabei wird zum Teil sehr detailliert auf einzelne Punkte eingegangen. Andere, für diese Arbeit weniger wichtige Punkte werden dagegen nicht einbezogen.

Im Kapitel zwei wird mit Attack Trees eine einfache und übersichtliche Methode für die Modellierung von Angriffen zur Verfügung gestellt. Diese Methode wird im Verlauf um das Konzept der Enhanced Attack Trees erweitert.

Die Kapitel drei und vier beschäftigen sich im Anschluss mit einzelnen Angriffen im Detail. Es werden erst die Ideen, die hinter den einzelnen Angriffen stehen, betrachtet und im Anschluss ausgewählte Werkzeuge im Laborversuch getestet. Dies soll dazu führen, im Anschluss eine Bewertung dieser Werkzeuge vornehmen zu können.

Das fünfte und letzte Kapitel soll dann Aspekte der automatischen Generierung einer Policy für Bro IDS betrachten.

1 Wireless LAN

Wireless LAN bezeichnet im Allgemeinen ein kabelloses lokales Funknetz. In den Medien wird das Buzzword „WLAN“ aber gerne auch synonym für Wi-Fi (Wireless Fidelity) und 802.11 verwendet, streng genommen ist das aber falsch [Abbildung 1-1]. Die allgemeine Bezeichnung Wireless LAN oder WLAN schließt auch solche Vertreter wie zum Beispiel HomeRF mit ein. Wobei man hier sagen kann, dass aus der Gruppe der WLANs nur 802.11 eine wirkliche Bedeutung hat, andere führen ein Nischendasein oder sind überhaupt nie am Markt angekommen.

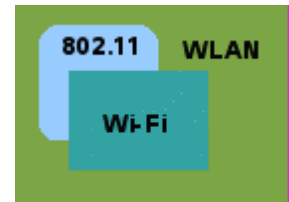


Abbildung 1-1:
Beziehung WLAN,
802.11, Wi-Fi

An zweiter Stelle der Missverständnisse im Bereich Wireless LAN steht: 802.11 gleichbedeutend Wi-Fi. Obwohl diese beiden Begriffe dicht zusammenliegen, haben sie doch unterschiedliche Wurzeln. 802.11 ist der 1999 von der IEEE eingeführte Standard für WLAN. Wi-Fi ist dagegen eher ein Zertifikat, das von der Wi-Fi Alliance – einem Zusammenschluss von Firmen, die im Bereich WLAN tätig sind – für Produkte vergeben wird. Die Voraussetzungen, die ein Wi-Fi konformes Produkt erfüllen muss, unterscheiden sich stückweise vom 802.11-Standard der IEEE und zielen hauptsächlich auf Kompatibilität zwischen Produkten unterschiedlicher Hersteller ab. Grundlegend kann man sagen, dass Wi-Fi zum Teil Weiterentwicklungen im 802.11 vorweg nimmt, aber auch einzelne Passagen – vorerst – nicht umsetzt. So geschehen bei WPA (Wi-Fi Protected Access), welcher schon 2002 von der Wi-Fi Alliance beschlossen wurde und bereits Teile des erst Mitte 2004 von der IEEE ratifizierten 802.11i-Standards enthält. Genauere Ausführungen dazu aber in den zugehörigen Abschnitten.

Zum Aufbau dieses Kapitels: Zunächst wird die allgemeine Bedrohungslage durch die Verwendung von Wireless LAN beschrieben. Einzeltechnologien zum Abwenden dieser Bedrohungen werden dann in Auszügen dargelegt. So wird eine Grundlage für die anschließende Diskussion der Verwundbarkeiten geschaffen.

1.1 Veränderte Bedrohungslage für Netzwerke durch WLAN

In erster Linie ist ein WLAN ein LAN mit allen Möglichkeiten, Problemen und Schwächen. Für die meisten Probleme und Schwächen, und vor allem die daraus erwachsenden Angriffspunkte gibt es für Local Area Networks sehr gut erprobte Sicherheitseinrichtungen, die eben diese Schwierigkeiten beheben. Eine Firewall zum Beispiel sorgt dafür, dass der gesamte Traffic in ein oder aus einem Netzwerk nach Sicherheitskriterien gefiltert wird. Die Nichterfüllung dieser Kriterien führt dazu, dass die Daten abgewiesen werden und der Zugang somit verwehrt bleibt. Vereinfacht gesagt, ist das möglich, weil in einem an Kabel gebundenen Netz der Punkt, an dem kontrolliert werden muss, feststeht, nämlich genau dieser eine Zugang durch die Firewall. Es wird dann zweierlei angenommen, erstens: Neben dieser Verbindung gibt es keine zweite Möglichkeit, um Daten in das Netz zu bringen. Zweitens: Kein Individuum hinter der Firewall unternimmt jemals einen Angriff.

Neben dieser „aktiven“ Zugangskontrolle, die sich mehr auf den Übertragungsweg bezieht, ist in jedem LAN auch immer eine „passive“ Zugangskontrolle in Form einer festgelegten Anzahl kontrollierbarer Ports an Switches o.ä. vorhanden. Man kann davon ausgehen, dass jeder Netzwerkverantwortliche weiß, wo sich physische Zugangspunkte zum eigenen LAN befinden. Er kann vielmehr sogar sicher davon ausgehen, dass an Stellen, an denen sich keine Netzwerkinfrastruktur – Verkabelung, Switches, Router, etc – befindet, prinzipiell auch kein Zugang möglich ist. Dieses sehr stark vereinfachte Beispiel verdeutlicht, wie gut sich ein LAN in der Regel von der Außenwelt abgrenzen lässt.

Genau diese Annahmen über Zugänge und Abgrenzbarkeit gelten aber nicht mehr, sobald man an Stelle eines LANs ein WLAN betreibt. In einem WLAN gibt es eben keine überschaubare Anzahl von Zugangspunkten zum Übertragungsmedium und damit letztendlich zum Netzwerk, da sich ein Funksignal in alle Richtungen ausbreitet und es somit im gesamten Abdeckungsbereich möglich ist, Zugang zum Übertragungsmedium zu erhalten.

Jetzt kann man argumentieren, dass die Signalausbreitung über Richtfunkantennen und Sendeleistung gesteuert werden kann. Leider ändern diese beiden Maßnahmen aber kaum etwas an dem Problem. Damit

hält man eventuell zufällige Teilnehmer aus dem Netz, nicht aber ernsthafte Angreifer. Denn mit großer Sicherheit übersteigen Empfangsempfindlichkeit und Sendeleistung eines Angreifers die eines normalen Teilnehmers bei weitem. Was mit dem richtig zusammengestellten Equipment möglich ist, zeigt der aktuelle Wireless LAN Langstreckenweltrekord von 124.9 Meilen [WiFiWR].

Damit wird schon einmal deutlich, dass man nahezu nichts gegen das passive Teilnehmen in Form einfachen Mithörens an einem Funknetz unternehmen kann. Hier geht es erst einmal um das technische Teilnehmen. Es ist dabei zunächst unerheblich, was mittels Kryptographie erreicht werden kann: dass nämlich ein Mithörer zwar betriebene Kommunikation bemerkt, aber kein Wort versteht, was dann das einfache Mithören wenig Erfolg versprechend gestaltet.

Sogargegenaktives Teilnehmen ist mangrundsätzlicher steinmal machtlos. Denn im Gegensatz zum wired LAN, bei dem es durchaus wahrscheinlich ist entdeckt zu werden, wenn man sich damit ohne Berechtigung verbindet, sind die Chancen, aktiv an einem WLAN teilzunehmen, ohne entdeckt zu werden, gut. Es ist sogar sehr wahrscheinlich, dass man in einem ungesicherten WLAN nicht entdeckt wird.

Die Möglichkeit des unbeobachteten passiven und aktiven Teilnehmens an Wireless LANs führt unweigerlich zu Problemen mit Vertraulichkeit und Integrität. Es werden also Mechanismen benötigt, die diese Probleme beheben oder zumindest so stark abschwächen, dass man für den Übertragungsweg eine dem *LAN ähnliche* Basissicherheit erhält.

Würde an diesem Punkt mit der Beschreibung der allgemeinen Bedrohung durch Wireless LAN abgeschlossen, so wäre das nicht nur unvollständig, sondern auch fahrlässig. Bislang wurde stillschweigend Folgendes angenommen: Dem Betreiber eines Netzes ist klar, dass er (auch) ein WLAN betreibt. In genau dieser Aussage liegt aber das Problem. Viele Access Points lassen sich heute auch von Laien installieren. So ist es nicht selten, dass Mitarbeiter, ohne grundlegend böse Absicht, einen WLAN-Zugang zu einem Firmen-LAN schaffen. Es wird einfach ein handelsüblicher AP mit dem Arbeitsplatzzugang verbunden, um etwa auch außerhalb des eigenen Büros Zugang zum Internet/Intranet zu haben. Damit wird eine Situation geschaffen, in der sich ein Administrator nicht mehr bewusst ist, dass er nun auch mit Angreifern rechnen muss, die nicht über den normalen Angriffsweg Internetzugang kommen. Es wird also die physische Begrenzung des

LANs stark erweitert, nämlich um den gesamten Bereich, der nun vom AP abgedeckt wird.

Aber solche, auch als Rogue Access Points bezeichnete Zugänge werden nicht nur unbedacht von Mitarbeitern installiert, sondern auch von Angreifern. Häufig ist es im Rahmen des normalen Kundenverkehrs möglich, in Firmengebäude zu gelangen und so einfach einen Rogue AP anbringen zu können. Beide Fälle machen deutlich, dass es in jedem Fall Sinn macht, WLAN zu überwachen, auch wenn man *eigentlich* keines betreibt. In dieser Arbeit soll es aber bei der Verdeutlichung der Möglichkeit eines *Angriffs* auf diesem Weg belassen werden. Abwehrstrategien sind unter anderem Thema in [NEU, ADGKK, AD].

1.2 Sicherheitsmechanismen für WLAN

In den folgenden Abschnitten werden WEP, WPA und 802.11i (WPA2) als Möglichkeiten zur WLAN-Sicherung auf ihre Schwachstellen hin untersucht. Ziel ist es dabei in keinem Fall, die vollständige Funktionsweise und Beschreibung zu jedem einzelnen Mechanismus darzulegen, sondern gezielt die Punkte herauszuheben, in denen es bekannte Schwächen gibt. Für vollständige Funktionsbeschreibungen sei an dieser Stelle auf Spezifikationen der IEEE und Wi-Fi Alliance oder zusammengefasst auf [EdAr 04] verwiesen.

Zunächst soll aber kurz auf vermeintliche Sicherungen im WLAN eingegangen werden und auch darauf, warum sie eigentlich keinerlei Schutz bieten. Die Rede ist von *Access Control List (ACL)* und *hidden Service Set Identifier (SSID)*, beides wird man in nahezu jedem Gerät, das als Access Point (AP) betrieben werden kann, finden.

Access Control List

Eine ACL im Bereich WLAN ist nichts anderes als eine Liste von MAC-Adressen. Empfängt ein AP Daten, wird zunächst überprüft, ob die Absender-MAC sich in dieser Liste befindet. Ist das nicht der Fall, wird der 802.11-Frame abgewiesen. Problem an dieser Stelle ist, dass die MAC-Adresse mit jedem Frame im Klartext übermittelt wird. Ein mithörender Angreifer hat also ständig die Chance, an eine für diesen AP gültige MAC zu kommen. Da es mit nahezu jeder Netzwerkkarte möglich ist, seine eigene MAC frei zu wählen, macht das die ACL als wirkliche Sicherheitseinrichtung nutzlos.

Der Prozess des Aneignens einer fremden MAC wird als MAC-Spoofing bezeichnet. Heute hat die ACL nur noch eine anerkannte Funktion: zufällige Teilnehmer fernzuhalten.

Abseits der Sicherheitsproblematik sind WLAN ACLs auch in Sachen Performance kritisch. Für Netze mit mehreren hundert Teilnehmern zeigt sich schnell, dass ACLs an ihre Grenzen gelangen. ACLs skalieren ungenügend.

Hidden Service Set Identifier

In der Regel bietet jeder AP eine Option, die SSID, welche mit jedem ausgesendeten Beacon regelmäßig übermittelt wird, zu verbergen. Da diese aber benötigt wird, um sich mit einem AP zu verbinden, schließt man damit – vermeintlich – unerwünschte Teilnehmer aus. Für legitime Teilnehmer stellt eine verborgene SSID kein Hindernis dar, man kann ja voraussetzen, dass ihnen diese bekannt ist.

Wie schon bei ACLs scheint das auf den ersten Blick eine einfache und effektive Möglichkeit zu sein, den Zugang zum WLAN zu beschränken. Wie zu erwarten, ist das leider nicht der Fall.

Zum einen gibt es die Möglichkeit, die SSID eines WLANs per Brute Force bzw. Dictionary Attack (Tool: MDK2 u.a.) zu gewinnen. Dieses Vorgehen ist aber überdurchschnittlich zeitaufwändig. Die zweite, deutlich mehr Erfolg versprechende Methode liefert die SSID des angegriffenen WLANs über passives Beobachten. Sobald sich ein regulärer Nutzer mit dem AP verbindet, übermittelt er, im Klartext, die SSID. Kann der Angreifer genau diese Nachricht mithören, ist ihm danach die SSID bekannt. Das bekannteste Tool auf diesem Sektor ist ohne Zweifel Kismet, natürlich bieten aber auch andere Implementierungen diese Möglichkeit.

Offensichtlich bieten also weder ACL noch hidden SSID eine ernstzunehmende Sicherung für Wireless LAN. Daraus folgt, dass weiterführende Maßnahmen ergriffen werden müssen, um LAN und WLAN auf das gleiche Sicherheitsniveau zu bringen.

1.2.1 WEP – Wired Equivalent Privacy – Sicherheitslücken und Implementierungsfehler

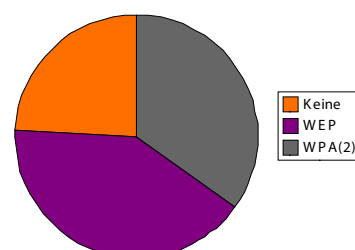
Wired Equivalent Privacy – diese Bezeichnung impliziert eine dem wired LAN gleichgestellte Sicherheit. Tatsächlich war WEP jedoch nie darauf ausgelegt, allumfassende Sicherheit zu bieten. Das Ziel der IEEE war lediglich, mit WEP die gleiche Basissicherheit für LAN und WLAN zu erreichen.

In der Realität bietet WEP aber schon seit 2001 nahezu keine Sicherheit mehr. Denn seit den ersten Veröffentlichungen über vermutete Sicherheitslücken im Jahr 2000 werden bis heute immer wieder neue oder verbesserte Tools implementiert [BHL, BI], um WEP auszuhebeln. Mit den effektivsten ist es unter Umständen möglich, den geheimen WEP-Schlüssel in unter zehn Minuten zu brechen, zu den Details mehr in Kapitel 3.3.2 und 4.3. Ein kleiner Zeitverlauf des Niedergangs von WEP ist in Tabelle 1-1 [Quelle: LE 06] dargestellt.

Obwohl WEP realistisch betrachtet gar keine Sicherheit mehr bietet, hat es dennoch einen berechtigten Platz in dieser Arbeit. Sechs Jahre nach den ersten Veröffentlichungen zu Sicherheitsproblemen bei WEP ist es immer noch im großen Rahmen im Einsatz. Tabelle 1-2 stellt die Ergebnisse einer nur knapp einstündigen Fahrt durch das Stadtzentrum Dresdens dar. Diese Ergebnisse werden durch eine aktuelle Untersuchung in [DÖR] gestützt, in der umfassend die Verbreitung der verschiedenen Verschlüsselungen untersucht wird.

Alle	1396	
Keine Sicherung	339	24.3%
WEP	569	40.7%
WPA(2)	488	35%

Tabelle 1-2: WLAN Sicherung



Aus Sicht der Verbreitung und des Einsatzes ist es also durchaus sinnvoll, WEP zu betrachten. Es ist hier zu betonen, dass in diesem Feldversuch die Sicherungen auf höheren ISO/OSI Schichten vollkommen außer Acht gelassen wurden.

1.2.1.1 WEP-Authentifizierung

Authentifizierung ist der Prozess, in dem alle Kommunikationspartner (bei WLAN wünschenswerterweise sogar wechselseitig) beweisen, dass sie der sind, der sie behaupten zu sein. Obgleich Authentifizierung die Basis ist, um überhaupt von sicherer Kommunikation sprechen zu können, existiert sie in WEP nicht. Tatsächlich wurde im ursprünglichen 802.11 eine Authentifizierung sogar vorgesehen, war aber vom ersten Tag an so wertlos, dass sie von der Wi-Fi Alliance nie ernsthaft umgesetzt wurde [EdAr 04].

<i>Datum</i>	<i>Beschreibung</i>
September 1995	Potentielle Schwachstellen in RC4 [WA]
Oktober 2000	Erste Publikation über WEP-Schwachstellen: „Unsafe at any key size; An analysis of the WEP encapsulation“ [WAL]
Mai 2001	„An inductive chosen plaintext Attack against WEP/ WEP2 [AR]
Juli 2001	CRC Bit flipping-Attack – Intercepting Mobile Communications: The Insecurity of 802.11 [BGW]
August 2001	FMS-Angriffe – „Weaknesses in the Key Scheduling Algorithm of RC4“ [FMS]
August 2001	Veröffentlichung von AirSnort [SG]
Februar 2002	Optimierte FMS-Angriffe von [HU]
August 2004	KoreK-Angriffe (einmalige IV) – Veröffentlichung von chopchop und chopper [KO]
Juli/August 2004	Veröffentlichung von Aircrack [DE] und WepLab [SA], beide setzen den KoreK Angriff um
September 2005	Fragmentation Attack Tool wesside: „The Final Nail in WEP’s Coffin“ [BHL]

Tabelle 1-1: Der Niedergang von WEP

Grund dafür waren zwei gravierende Designfehler. Zum einen basiert der ganze Prozess auf einem einfachen Challenge & Response-Verfahren. Soll heißen, will sich eine Station mit dem AP verbinden, so stellt sie einen Authentication Request. Von Seiten des AP wird mit einem Authentication Challenge (128 Bit Zufallszahl) geantwortet. Die Station verschlüsselt die Zufallszahl und sendet sie als Authentication Response zurück. War der letzte Schritt erfolgreich, antwortet der AP nun ein letztes Mal mit Authentication Success, daraufhin gilt die Station als authentifiziert. Damit ist nicht nur die

Station authentifiziert, sondern auch ein mithörender Angreifer mit 128 Bit Klartext/Schlüsseltext-Paar versorgt. Da aber für die Authentifizierung und die nachfolgende Verschlüsselung der Übertragung der gleiche Schlüssel benutzt wird, ist dies überaus gefährlich. Damit nicht genug, der Angreifer ist gleichzeitig auch quasi selbst authentifiziert. Denn nach dem Challenge & Response-Verfahren gibt es keine Möglichkeit, um festzustellen, ob kommunizierte Daten auch von der tatsächlich authentifizierten Station oder von einer mit vorgetäuschter Identität stammen. Im Schatten dieser beiden Fehler verblasst das Fehlen einer wechselseitigen Authentifizierung fast vollkommen, aber auch diese existiert bei WEP Authentication nicht.

Durch das Fehlen einer sicheren Authentifizierung werden einige Angriffe überhaupt erst möglich, zum Beispiel Rogue AP. Der Vollständigkeit halber ist zu ergänzen, dass das viel später ratifizierte 802.1x [IEEE 1] theoretisch auch in Verbindung mit WEP angewendet werden kann. In der Praxis wird bei WEP immer Open Authentication verwendet. Das hat zur Folge, dass jeder, der sich bei einem AP zu authentifizieren versucht, dies auch schafft. Die Zugangskontrolle, die in der Regel zumindest vom Erfolg oder Misserfolg der Authentifizierung beeinflusst ist, wird von dieser Phase der Kommunikation vollständig in die nachfolgende, die eigentliche Datenkommunikation verlegt. Darin wird dann stillschweigend angenommen, dass jeder, der den richtigen Schlüssel besitzt, um die Daten zu entschlüsseln, automatisch ein legitimer Teilnehmer ist. Es stellte sich beim Einsatz von WEP doch schnell heraus, dass dies ein schwerer Fehler war.

1.2.1.2 WEP Vertraulichkeit

In Abschnitt 1.1 wurde klar, dass für Wireless LAN ein Mechanismus benötigt wird, um Daten vor den Augen unerwünschter Dritter unlesbar zu machen, also zu verschlüsseln. Für die Realisierung der Verschlüsselung bei WEP ist die Stromchiffre RC4 zuständig. Bereits kurz nach dem Markteintritt wurden aber erste gravierende Schwächen in der Umsetzung entdeckt:

Zu kurze geheime Schlüssel. Die erste Version von WEP erlaubte nur 40Bit-Schlüssel, was selbst mit Brute Force-Angriffen keine Herausforderung darstellt. Spätere WEP-Versionen unterstützen 104Bit-Schlüssel, was das Problem abschwächt, aber grundlegend nichts daran ändert.

Kein Schlüsselaustausch-Protokoll. WEP sieht keine Möglichkeit vor, Schlüssel automatisch zu aktualisieren. Das führt dazu, dass es für Wireless LANs mit vielen Teilnehmern, etwa Firmen, überaus aufwändig ist, den geheimen Schlüssel regelmäßig zu wechseln.

Zu kurze IVs (Initialization Vector). Für WEP sind nur 24Bit-IV vorgesehen. Selbst wenn man davon ausgeht, dass eine IV erst so spät wie möglich wieder verwendet wird, ist das schon nach bereits 16Mio Paketen der Fall. Erschwerend kommt hinzu, dass im Standard keine Methode für die Generierung der IVs festgelegt wurde. Bei den meisten Umsetzungen wird die IV zufällig generiert, was die Wiederholrate sogar noch anhebt.

Hinweis: Dies kommt zu Stande, weil die zufällig aus der Menge aller IVs gezogene IV nicht aus der Menge subtrahiert wird und es somit die Möglichkeit gibt, dass sie im Anschluss direkt noch einmal gezogen wird. Von daher bietet das Iterieren der IV den geringsten Wiederholungsgrad.

Das führt zu einer Unsicherheit und zwar völlig unabhängig von der Länge des geheimen Schlüssels [WAL]. Kann ein Angreifer den Schlüsselbitstrom zu einer IV gewinnen, so kann er automatisch alle weiteren Frames mit gleicher IV entschlüsseln, ohne überhaupt Kenntnis vom geheimen Schlüssel zu haben. Weiterhin kann er für diese IV auch beliebige Daten verschlüsseln.

Problem dabei bleibt erst einmal die Gewinnung des Schlüsselbitstroms. Aber auch das ist, zumindest teilweise, direkt möglich, da von bestimmten Abschnitten eines Frames der Klartext bekannt ist. So sind zum Beispiel einige Bytes im Header jedes Frames bekannt, da sie für jeden gleich sind. Effektivere Methoden, um große Mengen des Schlüsselbitstromes zu erhalten, werden in Kapitel 3.3.3 beschrieben.

WEP erlaubt die Verwendung von „weak“ IVs. Um zu erklären, woraus diese Schwäche erwächst, ist es nötig, zuerst kurz die grundlegende Funktionsweise von RC4, wie es bei WEP verwendet wird, zu erläutern. RC4 besteht aus lediglich zwei recht einfachen Algorithmen. Der erste Algorithmus, KSA (Key Scheduling Algorithm),

benutzt den geheimen Schlüssel und die IV des aktuellen Frames und liefert ein 256-Byte großes Array, dessen Inhalt eine Permutation der Zahlen 0-255 ist. Aus dem so initialisierten Array erzeugt der zweite Algorithmus PRGA (Pseudo Random Generation Algorithm) pro Iterationsschritt ein Byte Schlüsselbitstrom. Um den eigentlichen Ciphertext zu erhalten, wird der Schlüsselbitstrom XOR mit dem Klartext verknüpft.

Fluhrer, Mantin und Shamir zeigen in [FMS], dass eine Klasse von Schlüsseln existiert, die sie „Weak Keys“ nennen, für die eine überdurchschnittliche große Anzahl von Bits in den ersten paar Byte Schlüsselbitstrom von nur wenigen Bit im eigentlichen Schlüssel abhängen. In Verbindung mit Klartext/Schlüsseltext-Paaren, der ersten paar Bytes, ergibt sich daraus eine Möglichkeit, Aussagen zu bestimmte Bytes des geheimen Schlüssels zu machen.

Um diesen Angriff auf den Schlüssel sinnvoll beginnen zu können, ist zuvor aber eine Information wichtig: Handelt es sich bei dem angegriffenen Schlüssel um einen „Weak Key“? Für den eigentlichen Schlüssel kann ein Angreifer das nicht sicher entscheiden, das ist aber auch vollkommen unnötig. Die Aussage, ob ein vollständiger WEP-Schlüssel – also IV+SK – „weak“ ist, kann alleine anhand der IV getroffen werden.

Sowohl für das Problem der „weak“ IVs wie auch für das daraus resultierende der schlechten ersten Bytes Schlüsselbitstrom gibt es einfache Lösungen, die aber in WEP¹ nie umgesetzt wurden. IVs können verworfen werden, wenn sie den Schlüssel zu einem „Weak Key“ machen würden. Für das Problem der schlechten ersten Schlüsselbits wird in [RSA] vorgeschlagen, die ersten 256 Byte einfach zu verwerfen, da PRGA danach so viele Iterationen durchlaufen hat, dass die Beziehung Schlüsselbitstrom – „geheimer Schlüssel“ für einen Angreifer nicht mehr herzustellen ist.

1.2.1.3 WEP Integrität

Unter Integrität fasst man die Forderung nach Aktualität, Vollständigkeit und Richtigkeit zusammen. Leider wird keiner der drei Punkte von WEP erfüllt.

¹ es existieren jedoch Produkte einzelner Hersteller die dies außerhalb einer Standardisierung tun

Aktualität (Replay Prevention)

Warum Aktualität eine so wichtige Forderung ist, wird sich in Kapitel 4.4 genau zeigen. Das Fehlen dieser Überprüfung wird dazu benutzt, um etwa Angriffe auf WEP-Schlüssel dramatisch zu beschleunigen. Replay Prevention wurde in WEP nicht gebrochen, sondern nie implementiert.

Vollständigkeit und Richtigkeit (Modifikation Detection)

Für diese Aufgabe ist in WEP das so genannte ICV (Integrity Check Value) zuständig. Der ICV wird über ein CRC32-Verfahren aus dem Klartext vor dem Versenden eines jeden Frames erstellt, anschließend an den Klartext angehängt und beide werden mit RC4 verschlüsselt.

CRC hat ein grundlegendes Problem: Es ist zwar geeignet, um zufällige Fehler, etwa bei der Übertragung, zu erkennen; es bietet aber absolut keinen Schutz vor absichtlichen Manipulationen durch einen Angreifer, der nach dem Verändern der ursprünglichen Daten den CRC-Wert einfach neu berechnen und ihn somit wieder zur Gültigkeit führen kann. Hinzu kommt, dass ein linearer Zusammenhang zwischen Daten und dem zugehörigen CRC-Wert besteht. Dies bedeutet, dass man für veränderte Bits im Klartext genau sagen kann, welche Bits im CRC-Wert sich ändern werden.

Für WEP könnte man nun erst einmal annehmen, dass der CRC-Wert, zumindest solange der jeweilige WEP Schlüssel nicht gebrochen ist, sicher bleibt, da:

$$\bullet \quad (N, \text{CRC}(N)) + \text{RC4}(\text{IV}, \text{SK})$$

gilt. Der erzeugte CRC-Wert wird also auch RC4 verschlüsselt, bevor eine Übertragung erfolgt. In [BGW] wird aber gezeigt, dass sich durch den Charakter von XOR diese Linearität erhält. Daraus folgt, dass ein Angreifer beliebige Bits im Schlüsseltext ändern kann und weiß, welche Bits im CRC-Wert korrespondierend geändert werden müssen, damit dieser gültig bleibt. Als Formel dargestellt, bedeutet das:

$$\bullet \quad \begin{matrix} (N_{\text{mod}}, \text{CRC}(N_{\text{mod}})) \\ (N_{\text{delta}}, \text{CRC}(N_{\text{delta}})) \end{matrix} + \text{RC4}() = (N_{\text{ori}}, \text{CRC}(N_{\text{ori}})) + \text{RC4}() +$$

wobei N die Nachricht repräsentiert und $+$ die XOR Funktion. N_{delta} stellt die Bitmaske der Veränderung an N_{ori} dar. Die Bitmaske der Veränderung und ihr CRC-Wert XOR , verknüpft mit der Originalnachricht und dem dazugehörigen CRC Wert, liefert also N_{mod} , eine beliebig modifizierte Nachricht, die unter dem ebenfalls modifizierten $CRC(N_{\text{delta}})$ gültig ist. Ein Angreifer hat nun also die Nachricht und den CRC-Wert verändert, ohne dass er auch nur ein Bit des Schlüsselstroms kennen muss.

Das scheint in der Praxis erst einmal kein ernstes Problem darzustellen, denn der Angreifer kann ja nur „etwas“ ändern, weiß aber im Großen und Ganzen nicht, was er da überhaupt ändert. Weit gefehlt! Ebenfalls in Kapitel 4.4.2 werden Angriffe vorgestellt, die ICV und Netzinfrastruktur nutzen, um einzelne Frames zu entschlüsseln bzw. beliebig viele Bit Schlüsselstrom zu einer gegebenen IV zu gewinnen, ohne Kenntnis vom geheimen Schlüssel zu haben.

1.2.1.4 WEP-Verfügbarkeit

Dieses Schutzziel wurde bewusst an das Ende der Auflistung gestellt. Denn obwohl für jede Art von Netz, und sogar Computersystem im Allgemeinen, von großer Wichtigkeit, muss der Wunsch nach allumfassender Verfügbarkeit immer enttäuscht werden, wenn man von einem genügend mächtigen Angreifer ausgeht [PF]. Was „genügend mächtig“ nun im Detail bedeutet, hängt vom jeweils betrachteten System ab. Für Wireless LAN kann man aber sagen, dass selbst ein vergleichsweise schwacher Angreifer effektiv die Verfügbarkeit beeinflussen kann. Eine Sicherstellung von Verfügbarkeit im Sinne von Unterbinden von DoS-Angriffen ist in Wireless LAN-Umgebungen nicht möglich. Der Grund dafür liegt einmal mehr in der freien Zugänglichkeit des Übertragungsmediums. Das führt hier sogar soweit, dass unbeabsichtigt DoS-Angriffe gestartet werden, etwa durch einen nicht funktionierenden Mikrowellenherd. Hier liegt die Begründung in den ähnlichen benutzten Frequenzbändern: Mikrowellenherd 2.45GHz, 802.11 2.4GHz (5GHz ist bei 802.11 möglich). Der überwiegende Teil der zurzeit bekannten DoS-Angriffe auf 802.11 benötigt aber weder „spezielle“ Hardware noch Kenntnisse im RF-Bereich.

Dass hier wieder von 802.11 gesprochen wird und nicht mehr speziell von WEP, hat einen Grund: Die meist in der Praxis auftretenden DoS-

Angriffe erfolgen abseits von WEP und Co. auf der Managementebene von 802.11. Natürlich gibt es auch DoS-Angriffe, die speziell auf WEP oder WPA ausgerichtet sind. Tools für diese Art von Angriffen existieren, wenn überhaupt, nur als Proof of Concept.

Im Rahmen dieser Arbeit werden keine DoS-Angriffe bzw. Angriffstools betrachtet, die ausschließlich das Ziel haben 802.11 lahmzulegen. Einige, auch als DoS-Angriff zu gebrauchende Exploits werden aber zum Teil benötigt, um spezielle Ereignisse zu einem vom Angreifer festgelegten Zeitpunkt eintreten zu lassen. So kann etwa das massenhafte Aussenden von Authentication/DeAuthentication Frames als DoS gebraucht werden, aber in geringer Zahl vom Angreifer auch dazu benutzt werden, um einen legitimen Nutzer zum erneuten Verbindungsaufbau mit dem AP zu bewegen.

1.2.1.5 Konsequenz aus den bekannten Schwächen in WEP

Als Reaktion auf die immer neuen Schwachstellen in WEP wurden diverse proprietäre WEP-Erweiterungen auf den Markt gebracht, die das Ziel hatten, zumindest einige der Problemstellen zu überdecken. Vertreter dieser WEP-Erweiterungen sind:

- WEP2 mit längeren IVs und mindestens 128Bit-Schlüsseln
- WEPplus Vermeidung von weak IVs

Leider hatten beide Erweiterungen wenig bis gar keinen Effekt. Beide konzentrierten sich ausschließlich darauf, die in [FMS] beschriebenen Schwachstellen zu mindern, was ihnen jedoch nicht gelang. Längere Schlüssel führen lediglich zu einem linearen Ansteigen der zum Brechen des Schlüssels benötigten Zeit [EdAb 04]. Das Filtern von weak IVs setzt voraus, dass alle WLAN Interfaces in einem Netz dazu in der Lage sind. Hinzu kommt, dass später entwickelte Angriffe, die nicht mehr weak IVs, sondern nur unique (einzigartige) IVs benötigen, von diesem Mechanismus gar nicht betroffen sind.

Die einzige Möglichkeit zum ernsthaften Sichern einer 802.11 WEP-Verbindung bietet die Nutzung von Mechanismen wie VPN (Virtual Private Network). Es wird also die Sicherheitsproblematik auf eine höhere Schicht im ISO/OSI-Modell verlagert. Dabei verbleiben die unteren Schichten aber unsicher. Problematisch beim Einsatz von VPN ist der erhöhte

Administrations- und Infrastrukturaufwand, was dazu führt, dass in administrierten Umgebungen, etwa Firmen, dieser Weg zwar möglich ist, sich aber für SOHO und Private quasi ausschließt. Für Wireless LAN wurde folglich eine Möglichkeit gesucht, die sicher war, aber dennoch so einfach zu handhaben wie WEP.

1.2.2 WPA – WiFi Protected Access und 802.11i (WPA2)

WPA ist die Antwort der WiFi-Alliance auf das Bedürfnis der Konsumenten nach einer funktionierenden Verbindungssicherung für 802.11. Dabei ist WPA ein Zwischenstandard zu WEP und 802.11i (WPA2). Die Notwendigkeit für eine solche Zwischenlösung ergab sich daraus, dass ältere Hardware nicht die technischen Voraussetzungen mitbrachte, um direkt von WEP auf 802.11i zu wechseln. Hinzu kam die immer wieder verschobene Ratifizierung von 802.11i.

WPA/WPA2 beheben *alle* Schwächen von WEP, mit Ausnahme der Verfügbarkeit. Wie es im jeweiligen Fall genau bewerkstelligt wurde, einzelne Schritte der Kommunikation zu sichern, ist nicht Gegenstand dieser Arbeit. Für detaillierte Informationen zu den Funktionsweisen von WPA und WPA2 sei auf die jeweiligen Spezifikationen [IEEE 2] bzw. auf [EdAr 04] verwiesen. Hier soll lediglich, wie auch schon im Kapitel zu WEP [1.2.1], auf die Komponenten eingegangen werden, aus denen Schwachstellen erwachsen.

Dass WPA und WPA2, obwohl sie sich in einigen Punkten unterschieden, hier gemeinsam betrachtet werden, hat einen einfachen Grund: Zurzeit ist nur eine wirkliche Schwachstelle in WPA/WPA2 bekannt, welche die Sicherheit direkt gefährden kann. Diese Schwachstelle entsteht aber sowohl bei WPA wie auch bei WPA2 am gleichen Punkt der Kommunikation, aus demselben Grund.

1.2.2.1 Schwachstelle im 4-Way-Handshake bei WPA/WPA2

WPA(2) sieht zwei Betriebsmodi vor: WPA-Enterprise und WPA-Personal. WPA-Enterprise ist dabei primär für die Anwendung im Firmenbereich ausgelegt. Da die angesprochene Schwachstelle in diesem Fall nicht existiert, muss WPA(2)-Enterprise hier nicht weiter betrachtet werden. Im

Rest dieses Abschnitts wird immer dann, wenn von WPA gesprochen wird, implizit WPA(2)-Personal gemeint.

Um auf den 4-Way-Handshake eingehen zu können, ist zuvor eine kurze Einführung in die Schlüsselhierarchie von WPA notwendig, sonst blieben Schritte unklar.

Der im Zusammenhang mit WPA bekannteste Schlüssel ist der so genannte PSK (Pre Shared Key). Entgegen WEP, bei dem der Secret Key direkt verwendet wurde, wird der PSK niemals direkt benutzt, um die Übertragung zu verschlüsseln. Aus dem PSK wird zunächst der PMK (Pairwise Master Key) abgeleitet. Dafür stehen je nach vorliegender Form des PSK, 256 zufällige Bits oder Passphrase (8-63Byte), zwei Methoden zur Verfügung:

- Methode für 256Bit Zufall: $PMK = PSK$
- Methode für Passphrase: $PMK = PBKDF2(Passphrase, ssid, ssid_length, 4096, 256)$

Während die erste Methode klar ist, bedarf es bei der zweiten einer kurzen Erklärung. Die detaillierte Arbeitsweise von PBKDF2 ist in [RSA 2] beschrieben. An diesem Punkt reicht die kurze Erläuterung aus, dass es sich

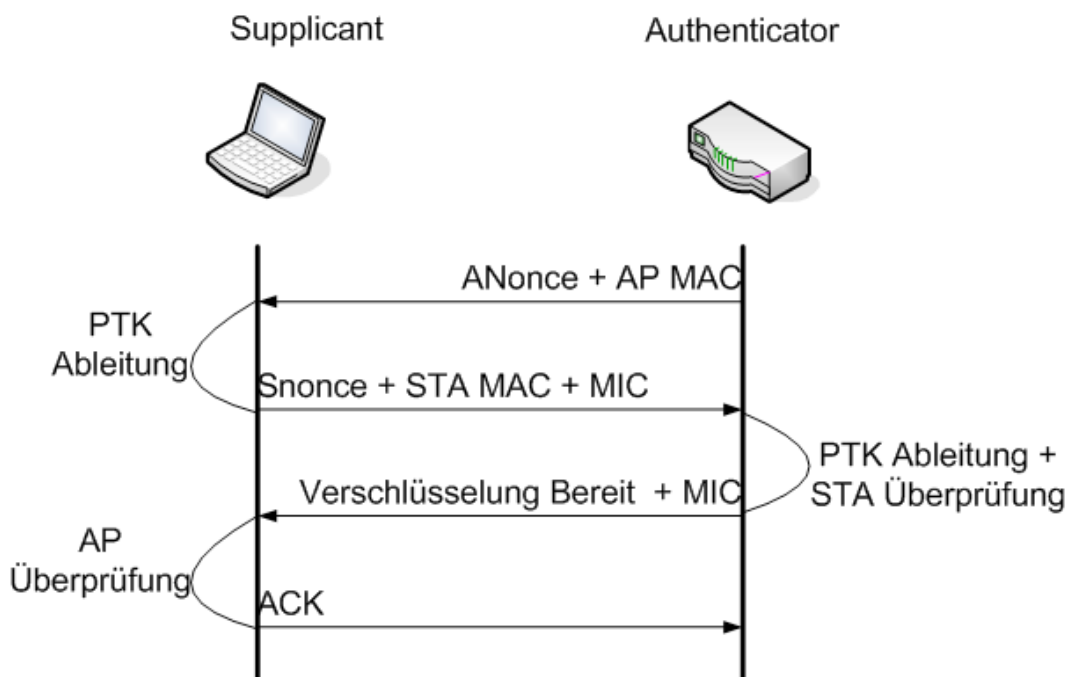


Abbildung 1-2: vereinfachter 4-Way-Handshake

dabei um eine Schlüsselableitungsmethode handelt. Für die Konkatenation „Passphrase+ssid+ssid_length“ liefert sie nach 4096-maligem Hashen 256 Bit PMK.

Mit dem so erhaltenen PMK kann nun in den 4-Way-Handshake eingestiegen werden. Abbildung 1-2 stellt den gesamten Ablauf vereinfacht dar.

Vor dem Übertragen der ersten Nachricht erzeugen Client (hier auch **Supplicant**) und Access Point (hier auch **Authenticator**) jeweils eine *Zufallszahl* genannt **SNonce** und **ANonce**.

Nachrichte Authenticator->Supplicant: Der Authenticator übermittelt dem Supplicant in der ersten Nachricht ANonce und AP-MAC. Diese Nachricht ist vollkommen ungeschützt. Ein Verändern oder Verfälschen hätte aber lediglich zur Folge, dass der 4-Way-Handshake nicht zu Stande kommt.

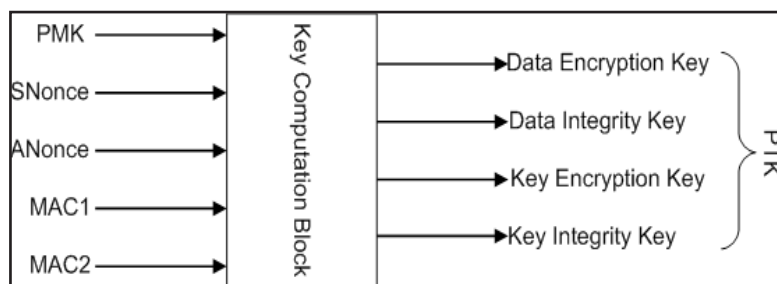


Abbildung 1-3: PTK Generierung

Mit SNonce, ANonce, AP-MAC und eigener MAC kann der Supplicant nun unter Benutzung des PMK den so genannten PTK (Pairwise Transient Key) ableiten [Abbildung 1-3]. Der PTK seinerseits repräsentiert eine Menge von Schlüsseln:

- Data Encryption Key (128 Bit)
- Data Integrity Key (128 Bit)
- Key Encryption Key (128 Bit)
- Key Integrity Key (128 Bit)

Key Encryption Key und Key Integrity Key dienen der Sicherung des Schlüsselaustausches, Data Encryption Key und Data Integrity Key sichern die nach dem 4-Way-Handshake folgende Datenkommunikation.

Hinweis: im PTK unterscheiden sich WPA(2) TKIP und WPA2 AES-CCMP. Für AES-CCMP fallen Data Encryption Key und Data Integrity Key zusammen.

Nachricht B Supplicant->Authenticator: Ähnlich wie bei Nachricht A sendet nun der Supplicant seine SNonce an den Authenticator. Der Unterschied ist aber, dass der Supplicant in diesem Schritt eine mit Hilfe des Key Integrity Key berechnete MIC (Message Integrity Code) an die Nachricht anhängt. Ab diesem Punkt ist die Kommunikation gegen Manipulation geschützt, es wird zu diesem Zeitpunkt aber noch nicht verschlüsselt.

Der Authenticator ist bei Erhalt von Nachricht B nun in der Lage, SNonce zu extrahieren und seinerseits den PTK abzuleiten. Als Konsequenz kann der Authenticator nun selbst den MIC der soeben erhaltenen Nachricht B errechnen, diesen mit der empfangenen vergleichen und so feststellen, ob der Supplicant wirklich Kenntnis vom PSK hat.

Nachricht C Authenticator -> Supplicant: Diese Nachricht enthält die Mitteilung, dass der Authenticator nun bereit ist, mit der verschlüsselten Kommunikation zu beginnen. Die wiederum angehängte MIC versetzt den Supplicant nun in die Position, die Identität des Authenticators zu bestätigen.

Nachricht D Supplicant -> Authenticator: Nachricht D ist der letzte Frame, der unverschlüsselt versendet wird. Er enthält lediglich eine Bestätigung für den Authenticator und beendet den 4-Way-Handshake.

Da alle Nachrichten unverschlüsselt versendet werden, kennt ein Angreifer nun ANonce, SNonce sowie die beide MACs. Für eine beliebig gewählte bzw. geratene PSK kann er jetzt PMK->PTK ableiten und anhand der an die Nachrichten B-C angehängten MIC überprüfen, ob seine gewählte PSK tatsächlich die von Authenticator und Supplicant benutzte ist. Wie sich die Angreifbarkeit des 4-Way-Handshakes faktisch auf die Sicherheit von WPA in der Praxis auswirkt, wird in Kapitel 3.3.2 dargelegt.

An dieser Stelle muss herausgestellt werden, dass dies die zurzeit einzige bekannte Schwachstelle in WPA(2) ist, die eine praktische Bedeutung hat.

Aber keineswegs ist sie als annähernd so problematisch zu betrachten wie eine beliebige der vielen Schwachstellen in WEP.

Eine starke PSK macht diesen Angriff nahezu wirkungslos. „Stark“ bedeutet, dass der Passphrase möglichst nicht in einem Standardwörterbuch vertreten sein sollte und – was bei solchen Betrachtungen meist außer Acht gelassen wird – sich mit Tools wie John The Ripper auch nicht leicht aus Wörterbucheinträgen ableiten lässt. Für die Wahl des Passphrases gelten die allgemeinen Regeln für starke Passwörter. Werden sie beachtet, ist auch WPA-Personal im angedachten Einsatzgebiet gegenwärtig sicher.

1.2.3 Fazit

Obwohl WEP nahezu keine Sicherheit mehr bietet, wird es dennoch benutzt. Die Gründe sind hier vielfältig und reichen von immer noch nicht vollständig umgestellter Hardware, was unweigerlich zu Kompatibilitätsproblemen führt, bis hin zu Service Providern, die nach wie vor Hardware vertreiben, die nur WEP unterstützt [C't 17/2006] bzw. von Hause aus mit WEP als Standard konfiguriert ist. Gerade der letzte Punkt erscheint angesichts der seit fünf Jahren bekannten Sicherheitslücken [FMS] in WEP als unverantwortlich. Das führt dazu, dass WEP der (*Hacker*-)Welt wohl noch eine längere Zeit erhalten bleiben wird. Genug Motivation also, um neue Tools zu schaffen oder bestehende zu verbessern. Beispiele für eine Anzahl solcher Werkzeuge werden im Kapitel 3 angeführt und im Kapitel 4 einem Praxistest unterzogen.

Im Zusammenhang mit WEP noch von Sicherheit zu reden, ist in der Praxis falsch und nicht mehr tragbar. Gerade vor dem Hintergrund des am 26.07.2006 vom Landgericht Hamburg gefällten Urteils zur Haftung von WLAN-Betreibern im Falle einer Rechtsverletzung, die über ihr Netz begangen wird, erscheint das Betreiben eines ungesicherten – oder mit WEP quasi ungesicherten – Wireless LAN noch einmal stärker problematisch.

2 Attack Trees

Das Hauptproblem bei der Darstellung von Systemen ist im Allgemeinen, eine Repräsentation zu finden, die übersichtlich sowie möglichst einfach und verständlich ist, aber dennoch Möglichkeiten bietet, sehr komplexe Systeme mit ihren Zusammenhängen zu illustrieren. Wünschenswert ist es zudem, eine Darstellung so zu wählen, dass Personen aus anderen Fachbereichen zügig Zugang zur jeweiligen Thematik finden, ohne sich intensiv einzuarbeiten.

In Bereichen außerhalb der Informatik ist es seit geraumer Zeit üblich, solche Modelle nicht nur zur einfachen Darstellung von Systemen zu benutzen, sondern vor allem Daten aus Fehler- und Schwachstellenanalysen in sie eingehen zu lassen. Die im Modell kombinierten Erkenntnisse fließen dann in den Entwicklungsprozess des realen Systems zurück, um es weiter zu verbessern bzw. Fehler zu beheben. In der Literatur wird als Beispiel für diese Art von Analyse häufig [NuReg] angeführt. Das Konzept der Fault Trees [NuReg] stellt in gewisser Hinsicht einen Vorläufer der Attack Trees dar. Ziel von Fault Trees ist es, im Beispiel [NuReg], mögliche Schwachpunkte und daraus resultierendes Fehlverhalten, Unfallherde, Verwundbarkeiten und Angriffspunkte an Nuklearanlagen zu modellieren.

Für den Bereich der Informatik – besonders den Bereich Security – haben sich Attack Trees [SCH] als geeigneter Formalismus zur Modellierung von Sicherheitsschwachstellen in Systemen erwiesen. Ziel ist es, für ein spezifisches Szenario alle möglichen Angriffe zu erfassen. Bruce Schneier beschreibt Attack Trees wie folgt: „*Attack trees provide a formal, methodical way of describing the security of systems, based on varying attacks.*“

Neben Attack Trees stehen Petri-Netze zur Angriffsmodellierung zur Verfügung. Ein Vergleich beider Ansätze ist in [StSch] zu finden. Attack Trees stellen sich hier als gutes und übersichtliches Konzept zur Modellierung heraus, das besonders gut geeignet ist, um Personen mit unterschiedlichem Fachwissen schnell in spezielle Thematiken einzuführen. Als unzureichend werden jedoch unter anderem die Fähigkeit zur Beschreibung von nötigen Vorbedingungen und die nicht vorhandenen Möglichkeiten zur Zyklenkonstruktion genannt. Petri-Netze als umfangreicheres Konzept bieten diese Möglichkeiten, verlieren für größere Systeme jedoch schnell an Übersichtlichkeit.

Erweiterungen zu den ursprünglichen Attack Trees, z.B. Vorbedingungen und Nachbedingungen, sind unter anderem in [MEL] beschrieben. Zusätzliche Erweiterungen, wie sie unter anderem in [Opel] vorgenommen werden, sind Bestandteil des letzten Abschnitts dieses Kapitels.

In Abschnitt 2.2 und 4 werden Enhanced Attack Trees (EAT) [CY] als Erweiterung genutzt. Besonders für die in der aktuellen Arbeit behandelten Wireless LANs werden Teile von EAT benötigt, um Unterangriffe ausreichend strukturieren zu können.

2.1 Struktur von Attack Trees

Aufgabe eines Attack Trees ist es, ein vollständiges Angriffsszenario gegen eine gewählte Umgebung darzustellen. Dabei sind Attack Trees aber nicht auf Sicherheitsprobleme aus der Welt der Informatik begrenzt, sondern können auch auf allgemeine Sachverhalte angewendet werden. Anhand des Beispiels in Bild 2-1 wird dies deutlich. Hier dient ein Attack Tree zur Modellierung eines möglichen Angriffs auf einen Safe. Anhand dieses Beispiels aus [SCH] sollen nun schrittweise Struktur und Bedeutung einzelner Elemente detailliert beschrieben werden.

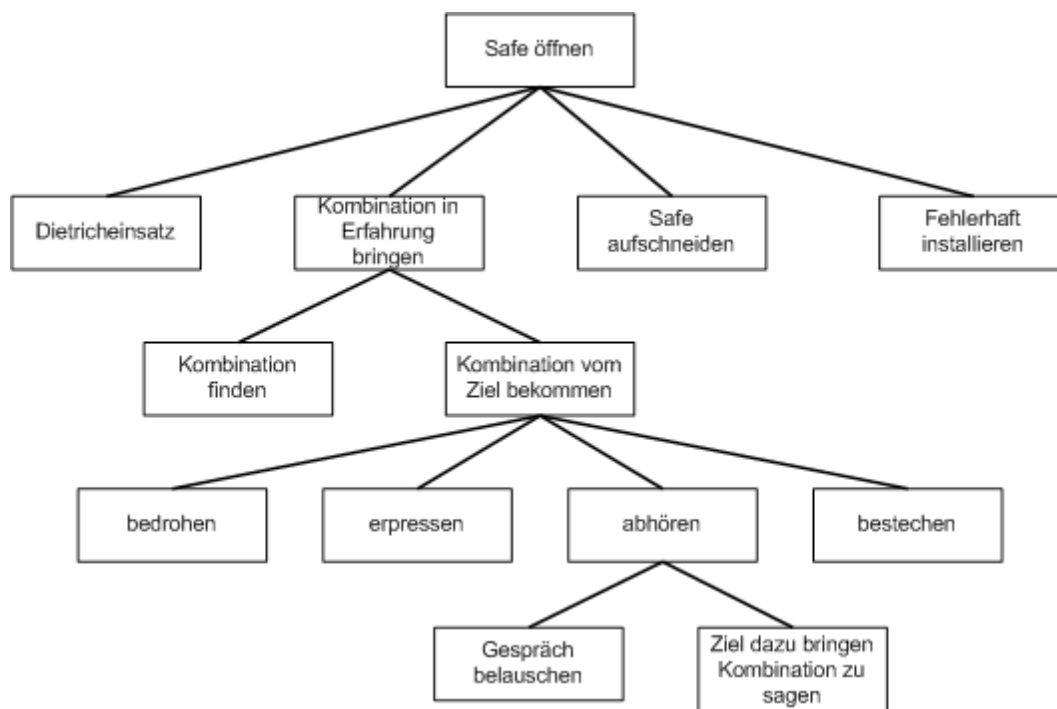


Abbildung 2 -1 Einfacher Attack Tree.

Schneiers Attack Trees finden weite Akzeptanz im Bereich Security, Gründe dafür sind hauptsächlich die intuitive Verständlichkeit und geringe Komplexität. Denn Schneiers ursprüngliche Attack Trees bestehen aus lediglich drei Komponenten: Kanten, Knoten und Knotenattributen.

Kanten

Die Verwendung erfolgt wie in Bäumen üblich zum Verbinden von Knoten. Einzelne Kanten können hervorgehoben werden, um beispielsweise einen bestimmten Weg von Blättern zur Wurzel zu kennzeichnen. In Bild 2-2 bezeichnen die gestrichelten Kanten alle Wege durch den Baum, die unter gegebener Forderung – Aufwand kleiner 15k € – möglich sind. Außerdem wird an Bild 2-2 sehr gut deutlich, warum es unbedingt notwendig sein kann, Wege auf diese Art zu bewerten.

Im gewählten Beispiel: Nimmt man an, dass sich im Safe zurzeit 15k € befinden und der Angreifer dies weiß. Ist offensichtlich, dass alle Angriffe, die insgesamt mehr als 15k € kosten, wohl nicht erfolgen werden, also unter den gewählten Umständen „unmögliche“ Angriffe darstellen. Erhöht man den Geldbetrag im Safe auf 100k €, ergeben sich auf einmal andere sinnvolle Möglichkeiten für einen Angriff. Da aber auch die zurzeit nicht sinnvollen Angriffe durchaus denkbar erscheinen, gehören sie in ein vollständiges Angriffsszenario.

Knoten

Schneier sieht zwei Varianten von Knoten vor: AND- und OR-Knoten. Die Semantik der Knoten ist „straightforward“. Damit ein AND-Knoten als erreicht gilt, müssen alle Kindereignisse eintreten. Für OR-Knoten gilt, dass mindestens ein Kindereignis eingetreten sein muss.

Im Beispiel [Abbildung 2-2] ist es ausreichend, dass einer der Angriffe bedrohen, erpressen, abhören *oder* bestechen erfolgreich ist, um an die Kombination zu kommen. Der Angriff abhören seinerseits ist aber nur dann erfolgreich, wenn es gelingt, ein Gespräch überhaupt zu belauschen *und* in diesem Gespräch die Kombination auch erwähnt wird. Vereinfachend wird nur eine Art von Knoten gekennzeichnet, in diesem Fall die AND-Knoten. Alle übrigen Knoten sind als OR-Knoten zu betrachten.

Auf dem Weg von den elementaren Bestandteilen eines Angriffs (Blättern) zum Erreichen des Angriffsziels (Wurzel) werden eine Reihe von Knoten passiert. Aus der Sicht des Angreifers ist jedes Erreichen eines dieser

Knoten gleichbedeutend mit einem Teilerfolg. Klar wird an dieser Stelle, dass auch ein Teilerfolg schon das eigentliche Ziel des Angreifers gewesen sein kann. Jeder Teilbaum kann also auch für sich einen vollständigen Attack Tree repräsentieren. Warum eben auch ein Teilerfolg schon eine Gefahr darstellt, wird in Kapitel 4.4 klar, in dem ein eigentlich eigenständiger Angriff benutzt wird, um den „Hauptangriff“ massiv zu beschleunigen.

Knoten-Attribute

Mit Kanten und Knoten lassen sich Attack Trees konstruieren, die Strukturen und Zusammenhänge von Angriffen beschreiben. Es ist aber auch wünschenswert, über einzelne Pfade spezielle Aussagen treffen zu können. Damit das überhaupt möglich ist, können Knoten mit Attributen versehen werden.

Der Einsatz von Attributen lässt sich erneut am besten mit dem bekannten Beispiel [Abbildung 2-2] erläutern. Das Wissen über die Art, Anzahl und Größenordnung der zu wählenden Attribute stammt bei einem realen Attack Tree aus empirischen Daten und Einschätzungen.

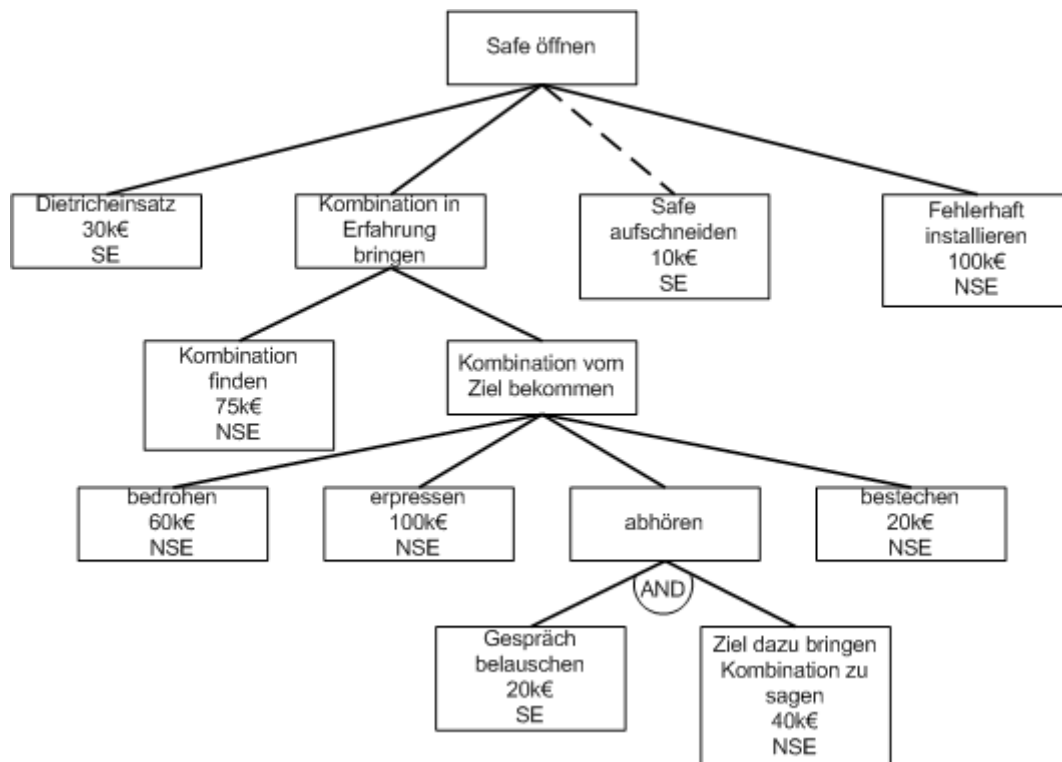


Abbildung 2-2 Baum mit Attributen: Kostenaufwand (Ganzzahl) und spezieller Werkzeugbedarf (Boolesch) SE (falsch) = Special Equipment; NSE (wahr) = No Special Equipment

Initial werden nur die Blattknoten mit Attributen versehen. Von dort aus werden die Werte im Baum aufwärts propagiert. Bei Wahrheitswerten ist dieses Konzept noch eindeutig.

Im Beispiel 2-2 würde der *AND*-Knoten „abhören“ mit dem Wert SE belegt, da ja alle Kindereignisse eintreten müssen und für mindestens eines spezielles Werkzeug benötigt wird. Für den *OR*-Knoten „Kombination vom Ziel bekommen“, der den Wert NSE erhält, ist die Entscheidung entsprechend.

Bei Ganzzahlattributen ist der Prozess des Wertepropagierens vielschichtiger. Im Beispiel sind semantische Belegung für AND und OR noch recht eindeutig zu ermitteln. So erscheint die Wertung von AND als Summenbildung und OR als Minimum der Kindattribute als einzig plausible Variante. Für andere Attribute beziehungsweise Szenarien sind andere Belegungen nahe liegend. Geht man davon aus, dass der Angreifer gar nichts erbeuten möchte, sondern nur das maximale Maß an Schaden versucht anzurichten, erscheint die Maximumbildung als sinnvoller. In dieser Weise lässt sich noch eine Reihe von sinnvollen Semantiken konstruieren. In [Opel] wird so zum Beispiel vorgeschlagen, Minimum, Maximum, Summe, Produkt und Mittelwert als mögliche Semantiken anzubieten. Welche Berechnungsform für ein bestimmtes Attribut gewählt wird, um Werte von Blättern zur Wurzel zu propagieren, obliegt am Ende der Kreativität des Entwerfenden.

An dieser Stelle sei noch angeführt, dass die einzelnen Attribute nicht losgelöst voneinander betrachtet werden müssen. Kombinationen wie zum Beispiel „alle Angriffe, die unter 40k€ kosten und keine speziellen Werkzeuge benötigen“ sind jederzeit möglich.

Der letzte Punkt in diesem Abschnitt soll die Attribuierung von Kanten ansprechen. Obwohl es in Baumstrukturen nicht unüblich ist, neben den Knoten auch die Kanten mit Werten zu versehen, wird dies bei AT nicht getan. Es ließen sich zwar Umstände konstruieren, in denen solche zusätzlichen Bewertungen Sinn machen. So könnte eine Ganzzahl bewertete Kante zum Beispiel ausdrücken, dass ein Ereignis n -Mal auftreten muss, damit es als Angriff und nicht als normales Verhalten gewertet wird. Aber diese Bewertung ließe sich in gleichem Umfang mit der gleichen Aussage auch in ein Attribut des betreffenden Ereignisses einbetten. In den klassischen Attack Trees sind solche Attribute zwar nicht vorgesehen – hier können generell nur Blätter mit Attributen versehen werden –, im folgenden Abschnitt werden jedoch

Enhanced Attack Trees beschrieben, in denen diese Attribute prinzipiell existieren.

2.2 Enhanced Attack Trees

Im vorangegangenen Kapitel wurden AT beschrieben, mit denen es möglich ist, logische und hierarchische Beziehungen zwischen Events (Leafes) und Subgoals (Nodes) zu modellieren. Für viele Anwendungen sind die gebotenen Beschreibungsmöglichkeiten in Schneiers AT ausreichend.

In vielen anderen Fällen wird jedoch eine weitere Beziehung benötigt, um Bedrohungen ausreichend realistisch darstellen zu können, die zeitliche Abfolge. Enhanced AT [CY] stellen dafür zwei Möglichkeiten zur Verfügung: O-AND (Ordered-AND) als zusätzliche Knotenvariante und TTL (Time To Live) als obligates Attribut für jeden Knoten (Events und Subgoals).

An dieser Stelle erscheint es sinnvoll, das Safe-Beispiel aus Kapitel 2.1 zu erweitern, um die Funktion von O-AND und TTL zu erklären.

Der Knoten „Dietricheinsatz“ wird dabei zuerst vom Blatt zum O-AND-Knoten befördert und hat nun seinerseits die Kinder: „Alarmsystem umgehen“ (Blatt), „Raum bei Nacht betreten“ (Blatt), „Dietricheinsatz“ (Blatt). Dass es bei diesen Ereignissen nicht nur darum gehen kann, dass sie alle auftreten, sondern dass sie alle in der richtigen Abfolge auftreten, scheint plausibel. Es ist ganz klar nicht realistisch, ein Schloss zu knacken, bevor man den Raum betreten hat, in dem es sich befindet. Was wiederum auch nur möglich ist, wenn man die Sicherheitseinrichtungen zuvor überlistet hat [Abbildung 2-3]. Logisch sind O-AND und AND gleich belegt, mit der Ergänzung, dass Kommutativität für O-AND ausgeschlossen ist. Am Blatt-„Alarmsystem umgehen“ zeigt sich im Beispiel am einfachsten die Funktion des Attributs TTL.

Angenommen, nach jeglicher Art von Manipulation am Sicherheitssystem startet dieses nach 60s neu, so hat ein Angreifer genau 60s Zeit, um alle anderen für das Subgoal „Dietricheinsatz“ benötigten Events eintreten zu lassen. Denn nach der Lebenszeit von 60s wird das Event „Alarmsystem umgehen“ automatisch wieder als nicht eingetreten bewertet, und der Elterknoten kann somit nicht mehr erreicht werden. Die TTL für jeden Knoten kann unabhängig gewählt¹ werden und beginnt abzulaufen, sobald er als erreicht gilt.

1 „Gewählt“ bedeutet hier: anhand von Mess- oder Erfahrungswerten gewählt.

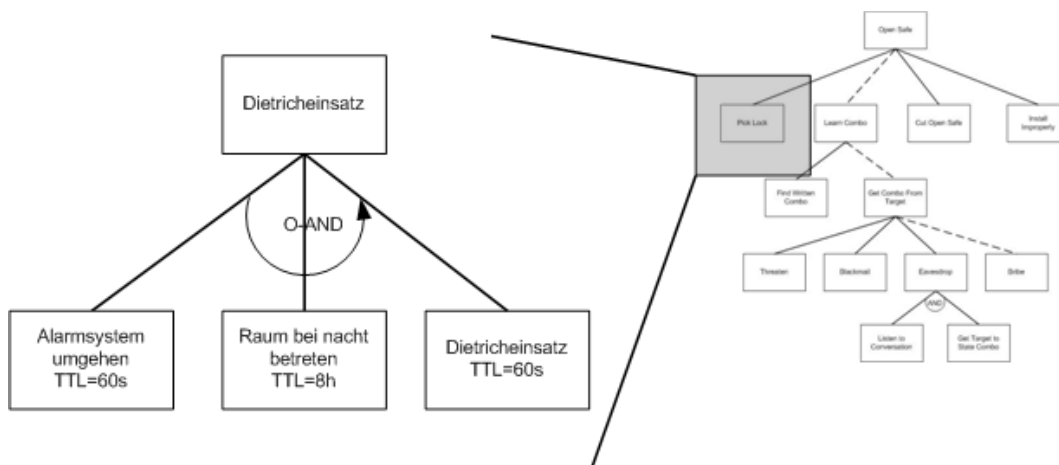


Abbildung 2-3: Enhanced Attack Tree

Die dritte Erweiterung, die durch Enhanced AT beschrieben wird, ist das Attribut PC (Percent Confidence). Jedes Subgoal besitzt genau einen PC-Wert. Er sagt aus, mit welcher Sicherheit es sich bei den bis jetzt aufgetretenen Events (genau genommen die Events, die zu genau diesem Subgoal führten) um den Angriff handelt, der durch den Rootnode verkörpert wird. Die Berechnung wird im Detail im folgenden Abschnitt beschrieben. PC dient vor allem dazu, ein Frühwarnsystem zu realisieren.

2.3 Support Tool for *Enhanced* Attack Trees

Als Grundlage für alle in dieser Arbeit umgesetzten Enhanced Attack Tree (EAT)-Konzepte dient das Support Tool for Attack Trees [Opel/2] von Alexander Opel. Sämtliche von Schneier geforderten Merkmale werden von [Opel/2] implementiert. Darüber hinaus werden einige Erweiterungen vorgenommen, die aber für diese Arbeit nicht relevant sind und außer Acht gelassen werden können. Eine vollständige Liste der Features ist [Opel] zu entnehmen. An dieser Stelle soll nur kurz auf die Punkte eingegangen werden, die für die aktuelle Arbeit unmittelbar benötigt werden.

Präsentation des AT

Bisher wurden in dieser Arbeit alle AT als Baumdiagramm, also graphisch, dargestellt und stillschweigend angenommen, dass dies auch die günstigste Präsentationsform ist. Leider zeigt sich jedoch, dass bei allen AT, die über triviale Beispiele hinausgehen, eine übersichtlichere Darstellungsform gebraucht wird. Schneier schlägt an dieser Stelle eine listenähnliche

Struktur vor. Im Support Tool for Attack Trees wird genau dieser Vorschlag umgesetzt [Abbildung 2-4].

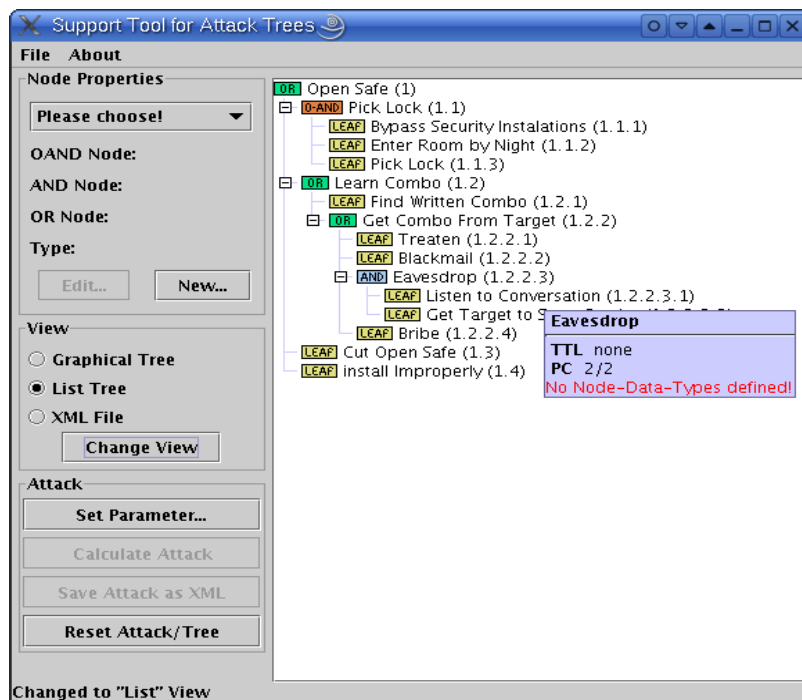


Abbildung 2-4: Oberfläche des Support Tool for AT

Datenhaltung (Datiformat)

Für jeden Attack Tree entsteht eine gesonderte XML-Datei (speziell GraphML), die sämtliche zugehörigen Informationen enthält. [Opel/2] wurde hier so modifiziert, dass alle Erweiterungen im Rahmen von EAT in die „.graphml“-Datei einfließen.

Berechnung der Knotenattribute

Attribute für Subgoals werden automatisch berechnet, sobald jeder Zweig im Baum mit einem Leaf endet. Jeder neu hinzugefügte Leaf wird dabei automatisch mit allen Attributen initialisiert. Startwert ist ein zuvor festgelegter Default-Wert. Die Vorgehensweise beim Erstellen oder Bearbeiten von Attributen wird in [Opel] Kapitel 4.1.1 u. 2 beschrieben.

Hier sei angemerkt, dass berechnete Subgoal-Attribute als Tool Tip zum jeweiligen Knoten betrachtet werden können, eine permanente Speicherung ist aber nicht vorgesehen.

2.3.1 Umsetzung der Erweiterungen im Support Tool for *Enhanced Attack Trees*

Die grundlegenden Erweiterungen, die getätigt werden müssen, um das Basisprogramm EAT-fähig zu machen, wurden bereits in Abschnitt 2.2 beschrieben. Hier soll nun auf Feinheiten der Umsetzung und auf die Auswirkungen eingegangen werden.

Neuer Knotentype: O-AND

Das Programm wurde um den Knotentype O-AND erweitert. Knoten vom Type O-AND können vom Anwender genau so verwendet werden wie die bereits bekannten OR/AND-Knoten. Alle Berechnungsformeln für Knotenattribute, wie AND-Formula, XOR-Formula etc. [Opel], sind nun ebenfalls für O-AND verfügbar.

Neue Attribute TTL und PC

Für jeden Node und Leaf besteht die Möglichkeit über, ein Pop-Up Menü den TTL-Wert zu verändern. Wird der Wert nicht geändert, ist er als Default auf „none“ gesetzt. Das TTL-Attribut wird in die automatische Neuberechnung der Attribute – dies geschieht normalerweise bei jeder Veränderung am Baum – nicht mit einbezogen, da es ja für jeden beliebigen Knoten möglich ist, TTL individuell zu setzen und der Wert pro Knoten nicht von anderen abhängt.

Anders als bei TTL gibt es bei PC gar keine direkte Möglichkeit, den Wert über Benutzereingaben zu manipulieren. Die Berechnung erfolgt voll automatisch, sobald am Ende jedes vollständigen Weges durch den Baum ein Event (Leaf) steht, und wird bei jeder Veränderung, die die Baumstruktur betrifft, wiederholt. Wie die Berechnung im Detail erfolgt, ist im Folgenden als Pseudocode dargestellt. Diese Erweiterungen sind in die Klasse Utilities.java eingeflossen.

```

PcCalc (Node currentNode, double minleafcount)
{
if ( currentNode == RootNode())
    {currentNode.PC = MinLeafCount(currentNode) / minleafcount} // das wird immer 1 weil Root
else
    {
        if (currentNode .Parent().Type() == „or“)
        {
            minleafcount = minleafcount * MinLeafCount(currentNode)
            currentNode.PC = MinLeafCount(currentNode) / minleafcount // der ungekürzte Bruch ist aus
                                                                    //Anschauungszwecken hier Absicht
        }
        else
            {currentNode.PC = MinLeafCount(currentNode) / minleafcount}
        }
    forall (currentNode.Child())
        { PcCalc(currentNode.Child(), minleafcount) }
}

MinLeafCount (Node currentNode)
{
if (currentNode.Type() == „or“ || currentNode.Type() == „leaf“)
    {return 1}
else
    {
        count = 0
        forall (currentNode.Child())
            { count += MinLeafCount(currentNode.Child()) }
        return count
    }
}

```

Alle eingearbeiteten Erweiterungen werden auch in der „graphml“-Datei gespeichert. Insbesondere die beiden feststehenden Attribute TTL und PC werden nun für Subgoals gespeichert und müssen nicht bei jedem Laden der Datei neu eingegeben bzw. berechnet werden.

3 Angriffe auf Wireless LAN

Nachdem im ersten Kapitel ausführlich beschrieben wurde, wo generelle Schwächen von Wireless LAN und seinen Sicherheitssystemen liegen, werden in diesem Kapitel einzelne Angriffe, genau genommen Implementierungen einzelner Angriffe, gegen diese Schwachstellen im Detail betrachtet. Hinzu kommen Kombinationen einzelner Implementierungen, womit zum Teil eine deutliche Verbesserung in der Performance des Angriffs erzielt werden kann.

Zunächst müssen aber mögliche Angreifertypen betrachtet werden, denn das Ziel dieser Arbeit soll nicht unbedingt sein, theoretische Machbarkeit zu zeigen – das wurde in anderen Arbeiten schon zur Genüge getan –, sondern am Ende Aussagen treffen zu können, welche Gefährdung vom entsprechenden Angriff für den Endverbraucher direkt ausgeht. Dafür wird es notwendig sein, Annahmen über einen erwarteten Angreifer zu machen, etwa Bedarf an Mitteln, Bedarf an Zeit und nicht zuletzt Bedarf an Know-How.

Dieser Abschnitt soll auch die Grundlage bilden, um im vierten Kapitel eine Klassifikation nach den in der Themenstellung geforderten Kriterien durchführen zu können.

Alle in dieser Arbeit betrachteten Tools stehen zum freien Download bereit. Gerade für das Betriebssystem Windows steht auch eine Anzahl von kommerziellen Tools bereit (zum Beispiel: [LF] oder [TM]), diese werden an dieser Stelle jedoch nicht einbezogen.

3.1 Angreifermodelle

Die Klassifikation von Angreifern ist der erste Schritt, um Aussagen über die tatsächliche Bedrohung, welche von Angriffen und Angriffstools ausgeht, überhaupt treffen zu können und so diese selbst zu klassifizieren. Am Ende entscheiden freilich Kriterien wie Schwierigkeit bei der Installation sowie der Bedienung und nicht zuletzt auch der sichtbare Erfolg bei der Benutzung auch mit darüber, wie schnell sich solche Werkzeuge verbreiten und welcher Art von Angreifer sie zugänglich werden. Im Allgemeinen ist es sinnvoll, Angreifer nach ihrer Motivation (spielen, Schaden anrichten, Wissensgewinn u.a.), ihren Mitteln (Zeit, Geld, Infrastruktur, Hardware

etc.) und ihrem Wissensstand (Grundkenntnisse bis Expertenwissen) zu kategorisieren.

In [EdAr 04] werden Angreifer nach ihrer Motivation und dem benötigten Know-How eingeteilt. Dabei wird eine Spanne vom spielerischen Angreifer bis zum Ego-Angreifer abgedeckt. Während der spielerische Angreifer lediglich Tools aus dem Internet lädt, sie kurz testet und meist schnell aufgibt, besitzt ein Ego-Angreifer detaillierte Kenntnisse im jeweiligen Bereich und entwickelt selbst neue Angriffe.

Ein ähnlicher Ansatz in [VGM] klassifiziert schon spezieller für den Wireless LAN-Bereich. Hier werden Angreifer in drei Gruppen eingeteilt: (1) Neugierige Personen, sie greifen nicht böswillig an, sondern suchen technisches Verständnis; (2) Bandbreitendiebe, sie beabsichtigen lediglich einen Breitbandinternetzugang mitzubেনutzen, verfolgen aber nicht das Ziel, darüber eine Straftat im Sinne von illegalem Informationsgewinn oder stellvertretenden Handlungen zu begehen. Hier wird das Opfer unbewusst geschädigt. Die letzte Gruppe stellt die wirklichen Black Hats dar (3), hier ist das Motiv klar: illegale Informationsbeschaffung bzw. Begehung von Straftaten über das Netz des Angegriffenen. Das Opfer wird dabei absichtlich geschädigt, bzw. eine Schädigung wird billigend in Kauf genommen.

Es zeigt sich, dass die beiden angeführten Klassifikationen für diese Arbeit nicht vollständig geeignet sind. Da allein die Gefahr, die von existierenden Angriffen und Tools ausgeht, bewertet werden soll, würden bei der [EdAr 04] Klassifikation quasi alle Angreifer in die gleiche Gruppe der spielerischen Angreifer fallen. Auch die zweite Einordnung scheint nicht geeignet, hier wird fast ausschließlich die Motivation herangezogen, um eine Entscheidung über die Kategorie des Angreifers zu treffen. Bedarf an Mitteln und nötiges Know-How werden hier weitestgehend außer Acht gelassen.

Für die hier benutzte Einordnung von Angreifern ist die Motivation weitgehend uninteressant, vielmehr stehen der Bedarf an Mitteln, besonders Zeit und Hardware, und das nötige Know-How im Fokus. Ob ein Angreifer also testen möchte, inwieweit ein Angriff auf dieses oder jenes WLAN möglich ist, um dann Sicherheitsempfehlungen zu geben, oder ob er angreift, um sich illegal Zugang zu Daten zu verschaffen, ist für diese Arbeit nicht von Belang. Erfüllen guter und böser Angreifer sonst die gleichen Voraussetzungen, fallen sie auch in dieselbe Gruppe.

Die **erste Klasse** von Angreifern besitzt weder Know-How noch spezielle Hardware, das einzige, was hier im Überfluss vorhanden ist, ist Zeit. Es

wird davon ausgegangen, dass sowohl Wissen wie auch der Wunsch, sich Wissen anzueignen, bei dieser Gruppe nicht ausgeprägt ist. Angriffe werden nicht über das Ausführen von einfachen Windowsprogrammen hinausgehen. Bei einem Misserfolg kann man auch annehmen, dass diese Art von Angreifer schnell aufgibt. Der Grund für den Misserfolg liegt zum großen Teil am Nichtvorhandensein der richtigen Hardware und des richtigen Betriebssystems. Was richtige Hardware und richtige Betriebssysteme sind, wird im folgenden Kapitel klar, in dem Angriffe in einer Testumgebung nachgestellt werden.

Eine **zweite Klasse** von Angreifern hat noch nicht deutlich mehr Wissen über die Thematik, ist aber mit wesentlich leistungsfähigerer Hardware ausgerüstet. Das sind in der Regel: Laptop, leistungsstarke WLAN Interfaces mit zusätzlichen Antennen usw. In diese Gruppe fallen Bandbreitendiebe und WarDriver (siehe nächster Abschnitt). Sie nehmen sich nicht die Zeit, lang andauernde Angriffe gegen ein spezielles WLAN auszuführen, sondern gehen nach einem missglückten Versuch zum nächsten weiter. In Kapitel eins wurde gezeigt, dass es noch genug ungesicherte oder unzureichend gesicherte Netze gibt. Wichtig ist hier, dass diese Gruppe auch kein spezielles Interesse an einem bestimmten personen- oder firmengebundenen Netz hat. Sie versuchen nicht gezielt anzugreifen und haben weder Bewusstsein, noch interessiert es sie, dass sie mit ihrem Angriff eventuell Schaden zufügen.

Die **dritte** und zugleich letzte **Klasse** von Angreifern stellt die größte Bedrohung für den Endverbraucher dar. Diese Gruppe ist mit passender Hardware ausgerüstet, kennt sich in der Thematik WLAN hinreichend gut aus und unterliegt nicht mehr irgendwelchen Beschränkungen, wenn es um Kenntnisse von Tools und Betriebssystemen geht. In diese Gruppe fallen ernsthafte Angreifer mit sowohl guten wie auch schlechten Absichten, also Black Hats und White Hats. Sie greifen nicht mehr wahllos, sondern zielgerichtet an. Bei anfänglichen Misserfolgen ist auch nicht damit zu rechnen, dass Angreifer schnell aufgeben.

Die Grenzen zwischen den einzelnen Angreiferklassen sind aber durchaus als fließend zu betrachten, und die Zuordnung zu einer Klasse ist im besten Fall eine Momentaufnahme des Angreifers. Es steht ihm ja jederzeit frei, sich Wissen anzueignen und passende Hardware zu erwerben. Die Abbildung 3-1 verdeutlicht den Zusammenhang noch einmal.

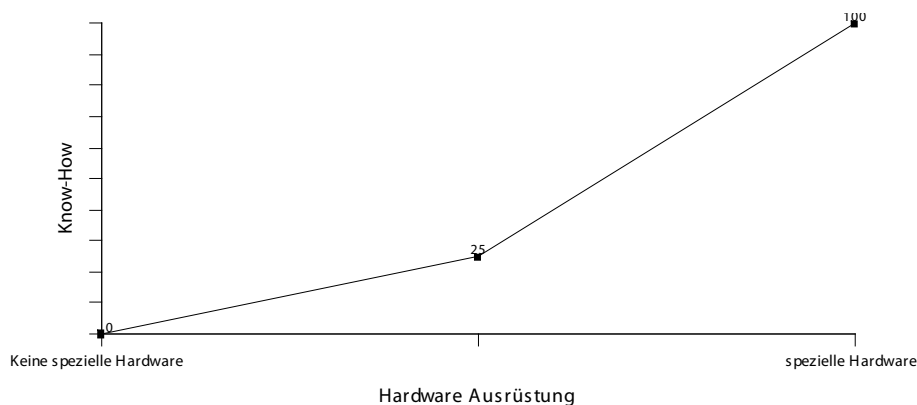


Abbildung 3-1: Zusammenfassende Darstellung zu Angreifermodellen

3.2 Finden und Aufdecken von Wireless Networks

Den ersten Schritt vor dem Angriff stellt immer das Auffinden von vorhandenen Wireless Networks dar. Für diese Aufgabe steht eine Reihe von Tools zur Verfügung. Hier wird eine Auswahl der am weitesten verbreiteten betrachtet. Alle diese Tools haben zwar die gleiche Aufgabe, nämlich Wireless Networks zu finden und zu identifizieren, lösen sie aber auf unterschiedliche Weise.

3.2.1 Aktive Suche nach Wireless LANs

Aktive Suche bedeutet, dass von der Scanning Software permanent so genannte „Probe Requests“ ausgesandt werden. Access Points in Reichweite antworten daraufhin mit einem „Probe Response“. Diese Antwort enthält neben Fähigkeiten des AP auch die gesuchte SSID und BSSID (Mac des AP).

Die aktive Suche nach APs hat aber entscheidende Probleme. Im ersten Kapitel wurde der so genannte „hidden SSID“-Modus beschrieben, der von APs unterstützt wird. Ist dieser Modus aktiviert, antwortet der AP nur mit einem Probe Response, falls der Request bereits mit der korrekten SSID versehen war. Anderenfalls erhält der Angreifer keine Informationen.

Ein zweites Problem bei dieser Art des Scannens ist, dass es aktiv geschieht. Da hier andauernd Anfragen verschickt werden – und das

in deutlich größerer Zahl als beim normalen Betrieb –, kann ein solcher Angriff schnell entdeckt werden. Ernsthafte Angreifer bevorzugen es aber in der Regel, so lange wie möglich keinen Hinweis auf ihre Anwesenheit zu geben.

Die Methode des aktiven Scannens ist aber trotz ihrer Nachteile die am weitesten verbreitete. Zu verdanken ist dieser Fakt dem Programm Netstumbler [NS]. Grund für die hohe Verbreitung ist, dass es auf dem am weitesten verbreiteten Betriebssystem – Microsoft Windows – arbeitet. Netstumbler bzw. die Funktionalität wurde in Tools für andere Plattformen reimplementiert, so zum Beispiel dstumbler [DB] für BSD oder Wellenreiter als Perl Script. Mit Pocket Stumbler existiert auch eine Version, die auf Pocket PC lauffähig ist.

Netstumbler lässt sich windowstypisch extrem einfach installieren und bedienen und unterstützt jede gängige Hardware. Es sind keinerlei Kenntnisse über Linux-Betriebssysteme oder wireless LAN an sich nötig. Diese Punkte machen es zur ersten Wahl für gelegentliche WarDriver. Dieses Tool steht, als eines der wenigen hier noch zu betrachtenden, auch der ersten Klassen von Angreifern zur Verfügung.

Am Rande sei noch das bereits erwähnte Tool Wellenreiter genannt. Es bietet zusätzlich zum einfachen aktiven Scan noch die Möglichkeit, Hidden SSIDs über einen Wörterbuchangriff zu ermitteln. Solche Methoden sind aber sehr auffällig und werden von den meisten Angreifern gemieden.

Hinweis: WarDriving ist vom Begriff WarDailing abgeleitet und bedeutet, dass mobil, mit Laptop o.ä., nach WLANs gesucht wird.

3.2.2 Passive Suche nach Wireless LANs

Diese Suchmethode unterscheidet sich vollkommen von der eben beschriebenen. Passive WLAN-Scanner senden selbst keinerlei Daten aus. Sie vertrauen auf die von APs in regelmäßigen Abständen ausgesandten (Standard 0.1sec Intervall) Beacon Frames. Diese zu den schon mehrmals erwähnten Management Frames gehörigen Frames enthalten neben anderen Informationen auch die SSID und BSSID des AP.

Der Betrieb von solchen passiven Tools hat aber einige Anforderungen an die Hardware und an den benutzten Treiber. Beide müssen es zulassen, dass sich das WLAN-Gerät in den so genannten Mode Monitor schalten

lässt. Im Mode Monitor werden sämtliche empfangenen WLAN Frames an das Betriebssystem weitergeleitet, in anderen Betriebsmodi werden alle nicht für dieses WLAN Interface bestimmten Frames herausgefiltert und verworfen. Für aktive Scanner gilt diese Forderung nicht, in [VGM] wird der dort beschrittene Weg erläutert.

Bezeichnend für passive Tools sind aber vor allem zwei Eigenschaften: Zum einen ist das Beweisen ihrer Präsenz unmöglich, da sie zu keinem Zeitpunkt selbst Daten senden, zum anderen ist es mit ihnen möglich, die SSID in jedem Fall in Erfahrung zu bringen, unabhängig davon, ob der AP im Hidden SSID-Modus arbeitet. Verantwortlich dafür ist der Mode Monitor, da hier sämtlicher Datenverkehr zur Auswertung zur Verfügung steht. Insbesondere schließt das auch die Daten ein, die ein legitimer Client bei seinem Verbindungsaufbau mit dem AP austauscht. In diesen Daten ist die SSID und die zugehörige BSSID immer unverschlüsselt enthalten.

Der bekannteste Vertreter aus dieser Gruppe ist Kismet [KI]. Neben den grundlegenden Funktionen eines Wireless LAN-Scanners – das Auffinden von WLANs – bietet Kismet noch eine Vielzahl von erweiterten Möglichkeiten, z.B. das Mithören und Abspeichern des kompletten Datenverkehrs. Kismet bietet eine komplexe WLAN-Datenanalyse mit der Fähigkeit, eine Vielzahl von Angriffen auf WLAN zu erkennen (Beispiel: Netstumbler). Für eine komplette Liste der Möglichkeiten sei an dieser Stelle auf die Dokumentation auf [KI] verwiesen.

Die Möglichkeit zum Auffinden und Mithören von WLAN-Verkehr bietet ebenfalls das Tool airodump-ng. Es wird hier gesondert erwähnt, da es Teil der aircrack-ng Suite ist, welche im Verlauf dieses Kapitels noch häufiger eine Rolle spielen wird.

Passive Wireless LAN-Scanner sind in der Regel die erste Wahl für erfahrene Wardriver, Sicherheitsprofis und Hacker. Auf Grund der Hardwareanforderungen schränkt sich der Kreis der potentiellen Nutzer aber schon ein. Hinzu kommt, dass Kismet auf Windowsplattformen nur stark eingeschränkt funktionstüchtig ist.

3.3 Angriffe gegen die Vertraulichkeit

Hat der Angreifer über eines der im vergangenen Abschnitt vorgestellten Tools ein WLAN gefunden, kann er sich daran machen, seinen Angriff zu planen und durchzuführen. Angriffe gegen Wireless Networks können

dabei aus einem oder mehreren der im Folgenden beschriebenen Schritte bestehen.

3.3.1 Belauschen von Übertragungen

Am Anfang nahezu jedes Angriffs steht das Belauschen von mindestens einem Teil der Übertragung, um Informationen über das WLAN zu erhalten. Dies können Informationen über den AP oder über Clients sein, etwa MAC-Adressen, Aktivität von Clients usw. Während die eben erwähnten Informationen grundlegende Ziele in Abschnitt 3.2 waren, geht es dem Angreifer nun darum, möglichst viele (verschlüsselte) Frames zu sammeln. Für unverschlüsselte Netze erhält er so direkt Einblick in die übertragenen Daten und erlangt so Informationen, die er für weiterführende Angriffe, die über das WLAN hinausgehen können, benötigt. Ist die Übertragung jedoch verschlüsselt, werden die gesammelten WLAN-Frames benutzt, um etwa den geheimen Schlüssel zu attackieren oder Teile des Schlüsselbitstromes zu rekonstruieren. Mehr hierzu in den folgenden Abschnitten dieses Kapitels.

Das Mitschneiden von Datenkommunikation, auch Packet Capture oder Sniffen genannt, ist im WLAN prinzipiell mit jeder gängigen Capture Software, sofern WLAN Extensions unterstützt werden, möglich. Klassische Beispiele hierfür sind Ethereal und tcpdump, die Liste lässt sich jedoch beliebig fortführen. Von besonderem Interesse sind hier erneut Kismet und airodump-ng, beide sind ausschließlich auf das Sniffen von WLAN-Frames ausgelegt. Airodump-ng bietet zudem eine direkte Möglichkeit, nur die für einen Angriff auf den geheimen WEP-Schlüssel benötigten Daten, IVs und geringe Teile der Daten, nicht aber den gesamten Frame zu speichern.

3.3.2 Angriffe auf den geheimen Schlüssel

Ein angreifbarer geheimer Schlüssel ist eine der größten Gefahren, der sich ein Sicherheitssystem gegenüber sehen kann. Wie in Kapitel eins schon beschrieben, ist bei WEP genau das der Fall: Der Schlüssel lässt sich direkt über Verfahren, die deutlich effektiver als Brute Force und Wörterbuch-Attacken sind, angreifen [FMS]. Die Grundlagen dieser Verwundbarkeit wurden bereits kurz erläutert und sollen hier auch nicht weiter vertieft werden. Nur soviel, mit dem Release des ersten Tools zum Cracken von

WEP im Jahr 2002 (Wepcrack und kurze Zeit später AirSnort) wurden ca. 6 Mio. Frames benötigt, um mit einem Erfolg beim Angriff auf den Schlüssel rechnen zu können. Der Grund für die große Anzahl an benötigten Frames war, dass nur ein sehr geringer Teil der gesammelten Daten wirklich benutzt wurde, um auf Bytes im Schlüssel zu schließen. Aber schon dieses erste Tool machte es vergleichsweise einfach, einen WEP-Schlüssel zu cracken, selbst für Nichtprofis im WLAN- und Linux/Unix-Bereich.

Zunächst aber noch einmal zurück zu den ersten Angriffen auf den WEP-Schlüssel. Wep_tools [LAVA] und WepAttack [WeA] waren die ersten Implementierungen, die sich mit dem Zurückgewinnen des geheimen Schlüssels befassten. Beide verließen sich aber noch vollständig auf Brute Force bzw. Wörterbuch-Angriffen. Verbesserungen im Gegensatz zu einer klassischen Brute Force-Methode, die in beiden Tools umgesetzt wurden, beziehen sich auf Implementierungsschwächen beider Schlüsselgenerierung im AP, welche den möglichen Schlüsselraum von 40 bzw. 104 Bit verkleinern. Die nachfolgende Generation von WEP-Angriffstools machte beide aber schnell uninteressant für tatsächliche Angriffe auf WEP, weshalb sie hier auch nur der Vollständigkeit halber genannt wurden.

Zu diesen Tools der zweiten Generation gehören unter anderem WepCrack und AirSnort (pre 0.2.7). Sie benutzen nicht mehr Brute Force, sondern gehen algorithmisch nach [FMS] vor. Während WepCrack eine Sammlung von Perl-Scripten ist, die auf der Konsole ausgeführt werden, ist AirSnort voll graphisch und sowohl für Windows wie für Linux verfügbar. Abbildung 3-2 zeigt das Userinterface des Programms AirSnort [SG].

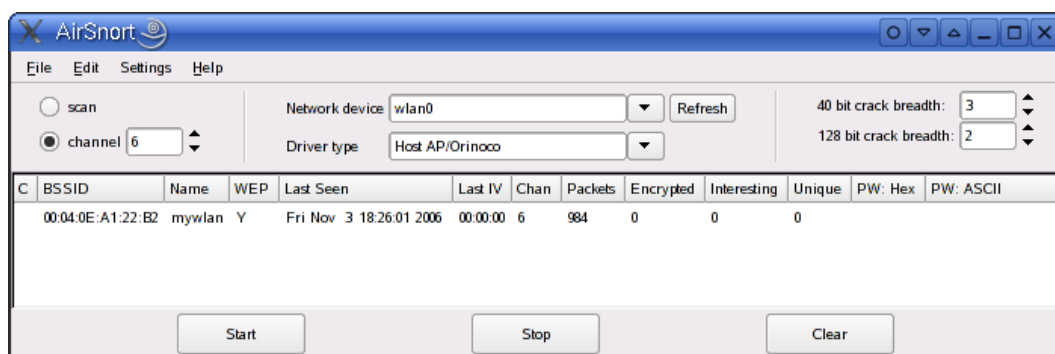


Abbildung 3-2: GUI AirSnort

Das AirSnort GUI ist überaus intuitiv und selbst Neulingen im WLAN-Bereich leicht verständlich. Es werden die nötigsten Optionen zur Auswahl des WLAN Interfaces, Treibers, Channels oder Scanmodus zur Auswahl gestellt (oberer Teil des GUI). Der untere Teil des GUI listet die gefundenen

Wireless LANs mit dazugehörigen Informationen und, sobald gefunden, mit dem geheimen Schlüssel auf. AirSnort (pre 0.2.7) versucht automatisch, den Schlüssel zu berechnen, sobald zehn weak IVs für einen AP gefunden wurden (siehe AirSnort Manual Pages).

Diese Tools der zweiten Generation, obwohl deutlich effektiver als die Brute Force Methode, benötigen immer noch große Datenmengen (Rund 4 Mio Pakete für 104 Bit Schlüssel [FMS]), um den Schlüssel tatsächlich angreifen zu können. Möglichkeiten, die Datenmenge deutlich zu reduzieren, wurden von David Hulton in [HU] beschrieben, er bezieht dabei vor allem Optimierungsvorschläge aus [SIR] ein. Diese Optimierungen schließen die Erweiterung der Klasse der weak IVs und das Betrachten von mehr als nur dem ersten Worte der RC4 Ausgabe ein (FMS zieht ursprünglich nur das erste Ciphertext-Wort zur Schlüsselberechnung heran, da man dort mit größter Wahrscheinlichkeit den Klartext zum Schlüsseltext kennt, für das zweite Wort ist das aber ebenfalls der Fall). Damit wird aus weniger Datenmenge mehr Information gewinnbar (ca. 2 Mio Pakete benötigt laut [HU]). Die einzige bekannte Implementierung dieser Verbesserungen ist dweputils, welches Bestandteil der schon erwähnten BSD-Airtools ist.

Die bis jetzt genannten Methoden bzw. Tools haben, neben der relativ großen Menge an benötigten Frames, ein weiteres Problem, welches zum Totalversagen dieses Ansatzes führen kann. Sie beruhen ausschließlich auf Weak IVs. Im ersten Kapitel wurde bereits erwähnt, dass einige WLAN Hardware über Firmenware-Updates, die nach der Veröffentlichung von [FMS] zur Verfügung gestellt wurden, Weak IVs filtert, was dazu führt, dass FMS nahezu vollständig versagt. Gefilterte IVs schaffen erst einmal eine subjektive Verbesserung der Sicherheit von WEP, wenn auch durch die weitere Verringerung der Anzahl möglicher IVs andere Angriffe gefördert werden, vor allem IV-Reuse. Diese letzte Hoffnung, man könne WEP mit nachgeschobenen Maßnahmen doch noch etwas sicherer gestalten, erlosch im zweiten Quartal 2004 endgültig.

Am 08.06.2004 wurde im NetStumbler-Forum unter dem Pseudonym KoreK ein Quelltext [NS 2] veröffentlicht, der eine Reihe von Angriffen implementierte, mit denen es möglich sein sollte, den WEP-Schlüssel nach bereits 100.000 Frames zu errechnen. Aber nicht nur, dass die Anzahl an benötigten Daten um eine Größenordnung sank, die neuen Angriffe benötigten auch nur noch einzigartige IVs und nicht mehr Weak IVs.

Dieser unter dem Namen KoreK-Attacke bekannt gewordene Angriff basiert auf statistischer Kryptoanalyse. Zusammengenommen bilden die 17

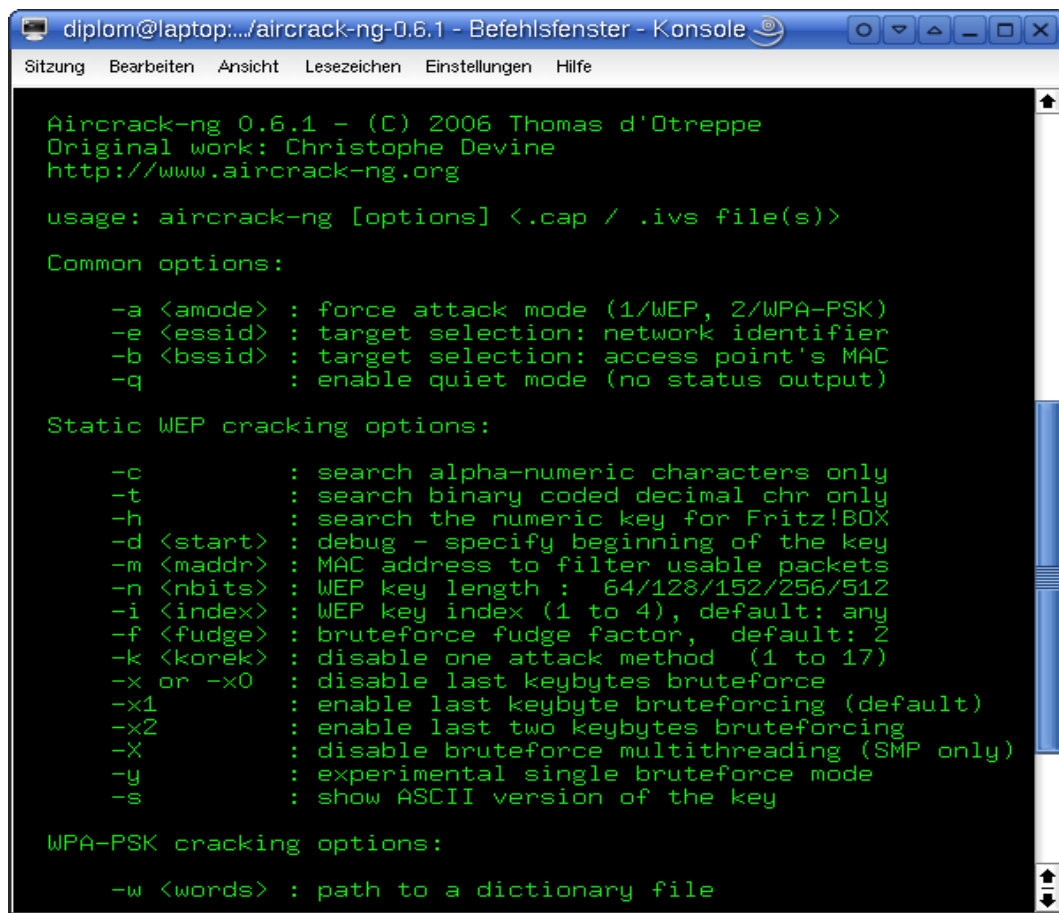
Einzelangriffe, aus denen die KoreK-Attacke besteht, eine Art Voting-System: Wird von einem Angriff ein bestimmtes Byte an einer bestimmten Stelle des Schlüssels als wahrscheinlich ermittelt, so erhält dieses Byte eine Stimme. Wie die einzelnen Angriffe im Detail funktionieren und wie die Stimmen gewichtet werden, würde den Rahmen dieser Arbeit bei weitem sprengen, es muss daher auf den Quelltext der Proof of Concept-Implementierung Chopper [NS 2] verwiesen werden.

Kurz nach der Veröffentlichung der KoreK-Attacke wurde sie in den Werkzeugen aircrack (später aircrack-ng) und WepLab umgesetzt, seit Version 0.2.7 auch in AirSnort. Für den Benutzer von AirSnort ergeben sich keine Veränderungen, da das GUI und mögliche Optionen weitgehend gleich geblieben sind. Die beiden Werkzeuge WepLab und aircrack bieten im Gegensatz zu AirSnort kein graphisches User Interface, sondern laufen von der Kommandozeile. Beide Werkzeuge bieten ähnlich umfangreiche Optionen und Tuningmöglichkeiten. Hier wird auf Grund der ähnlichen Leistungsfähigkeit [OS] nur eines von beiden betrachtet. Die Wahl fällt dabei auf aircrack, da es neben dem eigentlichen Programm aircrack zum Brechen des WEP-Schlüssels noch eine Anzahl anderer nützlicher Werkzeuge, wie dem schon erwähnten airodump, im Paket mitbringt.

Die Abbildung 3-3 Zeigt die möglichen Optionen des aktuell in der Version 0.6.1 befindlichen Werkzeugs aircrack-ng.

Aircrack erfordert als Eingabe lediglich eine Datei mit aufgezeichneten IVs, alle weiteren Optionen sind wahlfrei. Weitestgehend sind die Optionen selbsterklärend und bedürfen keiner weiteren Erläuterung. Eingegangen wird hier daher lediglich auf die Optionen -f (Fudge Factor: default=2) und -k (disable KoreK Attack #).

Wie bereits dargestellt, handelt es sich bei den KoreK-Angriffen um ein Verfahren, das auf statistischer Kryptoanalyse basiert. Liefert die Capture-Datei nun zu wenige statistische Informationen, um den geheimen Schlüssel direkt zu ermitteln, wird die Möglichkeit genutzt, anhand des bereits vorhandenen Trends für einzelne Bytes die wahrscheinlichsten Schlüssel zu probieren. Der Fudge Factor legt dabei fest, wie tief aircrack probiert, also wie tief er im Baum absteigt, wenn ein neuer Schlüssel getestet wird. Ein höherer Fudge Factor benötigt mehr Rechenzeit, liefert aber bei gleicher Anzahl an IVs mit größerer Wahrscheinlichkeit den korrekten Schlüssel.



```
diplom@laptop:~/aircrack-ng-0.6.1 - Befehlsfenster - Konsole
Sitzung Bearbeiten Ansicht Lesezeichen Einstellungen Hilfe

Aircrack-ng 0.6.1 - (C) 2006 Thomas d'Otreppe
Original work: Christophe Devine
http://www.aircrack-ng.org

usage: aircrack-ng [options] <.cap / .ivs file(s)>

Common options:

  -a <amode> : force attack mode (1/WEP, 2/WPA-PSK)
  -e <essid> : target selection: network identifier
  -b <bssid> : target selection: access point's MAC
  -q          : enable quiet mode (no status output)

Static WEP cracking options:

  -c          : search alpha-numeric characters only
  -t          : search binary coded decimal chr only
  -h          : search the numeric key for Fritz!BOX
  -d <start>  : debug - specify beginning of the key
  -m <maddr>  : MAC address to filter usable packets
  -n <nbits>  : WEP key length : 64/128/152/256/512
  -i <index>  : WEP key index (1 to 4), default: any
  -f <fudge>  : bruteforce fudge factor, default: 2
  -k <korek>  : disable one attack method (1 to 17)
  -x or -x0  : disable last keybytes bruteforce
  -x1        : enable last keybyte bruteforcing (default)
  -x2        : enable last two keybytes bruteforcing
  -X         : disable bruteforce multithreading (SMP only)
  -y         : experimental single bruteforce mode
  -s         : show ASCII version of the key

WPA-PSK cracking options:

  -w <words> : path to a dictionary file
```

Abbildung 3-3: Optionen von aircrack-ng

Die Notwendigkeit, einzelne Korek-Angriffe zu deaktivieren, ergibt sich für bestimmte geheime Schlüssel. Bei diesen Schlüsseln führen bestimmte Angriffe zu falschen Ergebnissen (false Positives) und sorgen so dafür, dass die statistische Auswertung verfälscht wird und somit der Schlüssel verspätet oder nie gefunden wird.

Im Vorfeld kann ein Angreifer nicht sagen, wie die Optionen -f und -k zu wählen sind. In der Regel wird mit den Standardeinstellungen begonnen und sobald die Schlüsselsuche für eine als genügend groß betrachtete Capture-Datei nach einer gewissen Zeit nicht erfolgreich ist, wird begonnen, mit einer oder beiden Optionen zu experimentieren. Als grobe Faustregel kann man sagen: Wenn ein 104 Bit-Schlüssel bei 800k IVs nach fünf bis zehn Minuten nicht gefunden ist, ist eine Veränderung an diesen Optionen sinnvoll. Letztendlich obliegt die Wahl der Einstellungen aber der Erfahrung und *Kreativität* des Angreifers.

Abbildung 3-4 zeigt aircrack bei der Arbeit bzw. das gelieferte Ergebnis. Im oberen Teil werden die benötigte Rechenzeit, der getestete Schlüssel

```

mc - /home/diplom/aircrack-ng/aircrack-ng-0.6.1 - Befehlsfenster - Konsole
Aircrack-ng 0.6.1

[00:00:06] Tested 2 keys (got 1048576 IVs)

KB    depth  byte(vote)
0     0/ 1    11( 195) E4( 20) 5A( 17) B7( 15) 1A( 13) 68( 13)
1     0/ 1    22( 277) 71( 34) E6( 34) 08( 28) 0B( 26) 26( 25)
2     0/ 1    33( 145) 2B( 108) 7F( 33) 07( 30) D2( 30) 06( 25)
3     0/ 1    44( 205) 07( 51) C6( 39) EF( 39) 06( 35) 99( 27)
4     0/ 1    55( 114) 5B( 53) A3( 50) 79( 28) 5A( 25) 5E( 25)
5     0/ 1    66( 157) 4E( 28) 76( 28) 17( 27) 56( 27) 9F( 26)
6     0/ 1    77( 974) 91( 33) DF( 33) AC( 29) 4E( 26) AD( 25)
7     0/ 1    88( 415) 11( 45) 5B( 44) 24( 39) 12( 37) 2B( 36)
8     0/ 1    99( 155) 80( 138) 8F( 54) A3( 44) 44( 41) 7C( 39)
9     0/ 1    00( 298) 36( 173) D9( 117) 9F( 67) A5( 49) F3( 37)
10    0/ 1    11( 313) 29( 95) 98( 66) 92( 59) BD( 54) CE( 44)
11    0/ 1    12( 183) 0A( 101) 9E( 76) 79( 68) 73( 58) 78( 50)
12    0/ 1    13( 162) 7D( 64) 58( 59) 57( 45) A2( 45) 8C( 38)

KEY FOUND! [ 11:22:33:44:55:66:77:88:99:00:11:12:13 ]

```

Abbildung 3-4: Aircrack: aircrack test.ivs

und die verfügbaren IVs angezeigt. Die 13 Zeilen im mittleren Teil zeigen jeweils an, um welche Schlüsselbytes es sich handelt, in welcher Tiefe probiert wurde und dann, von links nach rechts absteigend mit jeweiliger Voting-Anzahl in Klammern, die jeweils wahrscheinlichsten Bytes für dieses Schlüsselbyte.

Die Bindung an die Kommandozeile und die umfangreichen Optionen machen ein gewisses Basiswissen über die Funktionsweise und Vorgehensweise dieser WEP-Cracker notwendig, um sie effektiv nutzen zu können. Wem der Sinn aber nach Einfachheit und nicht nach maximalem Erfolg steht, der kann nach wie vor auf das einfache und übersichtliche AirSnort zurückgreifen. Während AirSnort durchaus noch der zweiten Klasse von Angreifern voll zur Verfügung steht, tendiert aircrack schon sehr stark in die Richtung der dritten Klasse.

Hinweis: Die angeführten Größenordnungen der benötigten Pakete basiert auf den Angaben in den jeweiligen Veröffentlichungen. In der Praxis schwankt die Anzahl der benötigten Pakete signifikant! Es ist so zum Beispiel nicht unüblich, dass für die klassische FMS-Methode ca. 10 Mio Pakete benötigt werden. Abhängig ist das bei den von Weak IVs abhängigen Angriffen von der Auftrittshäufigkeit derselben. Ein Teil der Schwankungen in der Anzahl der benötigten Pakete kommt aber auch durch den Schlüssel selbst zu Stande, in diesem Fall sind dann sowohl FMS- wie auch KoreK-Angriffe betroffen.

In [OS] wird ein Vergleich der benötigten Pakete für die verschiedenen Tools vorgenommen. Demnach erreicht KoreK bei ca. 5 Mio Paketen eine

nahezu hundertprozentige Chance, den Schlüssel zu berechnen, während das klassische AirSnort beinahe 15 Mio Pakete benötigt, um sich der 100% anzunähern.

Der letzte verbliebende Angriff ist zugleich der einzige, der auf WPA bzw. WPA2 existiert. Im ersten Kapitel wurde bereits sehr ausführlich beschrieben, wie ein aufgefangener Four-Way-Handshake benutzt werden kann, um den PSK zu ermitteln. Hier soll nun auf die Implementierung cowWPAtty eingegangen werden. cowWPAtty wurde mit leichten Veränderungen auch in aircrack eingebunden. Erste cowWPA-Versionen setzen genau den im ersten Kapitel beschriebenen Brute Force-Angriff um. Es wird eine Capture-Datei mit Four-Way-Handshake, ein Wörterbuch und die SSID als Eingabe gefordert und der PSK geliefert. Diese Methode ist extrem zeitaufwändig, da für jeden Wörterbucheintrag PMK neu berechnet werden muss (siehe Kapitel eins). Neuere Versionen von coWPAtty (ab Version 3.0) bieten die Möglichkeit, an Stelle eines Wörterbuches eine Hashdatei mit vorher berechneten PMKs zu übergeben. Unter [COWi] steht eine ca. 7 GB große Sammlung von Hashdateien für die gängigsten 1.000 SSIDs zur Verfügung. Auch mit dieser Variante bleibt dieser Angriff nach wie vor lediglich ein sehr zeitaufwändiger Brute Force/Wörterbuch-Angriff. Jedoch ist es mit einer vorausberechneten Hashdatei möglich, die Anzahl von pro Sekunde probierten Möglichkeiten circa um drei Größenordnungen anzuheben. Aber auch mit dieser massiven Steigerung der Leistung bleibt WPA sicher, sofern grundlegende Punkte zur Wahl eines starken PSK beachtet werden.

3.3.3 Angriffe auf den Schlüsselbitstrom

Anders als im vorangegangenen Abschnitt haben die hier vorgestellten Werkzeuge überhaupt nicht die Absicht, den geheimen WEP-Schlüssel zu erlangen. Vielmehr geht es darum, den von RC4 erzeugten Schlüsselbitstrom (auch PRGA – Pseudorandom Generation Algorithm genannt) zu einer gegebenen IV zu finden (siehe IV Verwendung Kapitel eins). So wird die Gelegenheit geschaffen, alle zukünftigen und vergangenen Frames mit gleicher IV zu entschlüsseln und beliebige Daten mit dem zu dieser IV gehörenden PRGA zu verschlüsseln.

Das Problem ist aber, erst einmal genug PRGA-Ausgabe zu erhalten. Zur Lösung gibt es mehrere Ansätze, die sich jeweils in ihrer Vorgehensweise unterscheiden. Die erste Veröffentlichung zu diesem Thema [AR] wurde nie in einem öffentlich zugänglichen Werkzeug umgesetzt, es erscheint jedoch

notwendig, ihre Funktionsweise darzustellen, da sie die Grundlage für ein Werkzeug Namens ChopChop ist, welches im Anschluss diskutiert werden soll.

Als Ausgangspunkt benötigt der Angreifer einige Bytes PRGA, in [AR] wird vorgeschlagen zu warten, bis ein DHCP-Discover auftaucht, und aus ihm die nötigen Anfangsbytes zu gewinnen. Da bei diesem Paket auf alle 38 Byte Klartext geschlossen werden kann, liefert Klartext XOR DHCP-Discover nun $38(=n)$ Byte PRGA. Ziel ist es, mit den Vorhandenen 38 Byte weitere PRGA Bytes zu ermitteln. Dies geschieht, indem ein Paket von $35(=n-3)$ Byte gewählt wird. Zu diesem Paket wird ICV nach dem bekannten Verfahren berechnet und an die 35 Byte angehängt. Die resultierenden $39(=n+1)$ Bytes werden nun bis zum Byte n mit dem bekannten PRGA verschlüsselt. Für das verbleibende, noch unverschlüsselte Byte $n+1$ wird nun die PRGA-Ausgabe geraten, und das so verschlüsselte Paket wird an den AP gesandt. Wenn das richtige Byte geraten wurde, erhält man vom AP nach kurzer Zeit eine Antwort; wenn nicht, wurde falsch geraten, und es wird mit einer der verbleibenden 255 Möglichkeiten ein neuer Versuch gestartet. So erhält man immer nach maximal 256 Versuchen ein weiteres Byte PRGA Ausgabe.

ChopChop [KO] basiert auf dem eben beschriebenen Angriff, jedoch in umgekehrter Vorgehensweise (inverser Arbaugh-Angriff). Es wird zu einem beliebigen Paket PRGA und Klartext ermittelt. ChopChop entfernt die letzten vier Byte (ICV), verwirft das letzte Datenbyte unter der Annahme, dass sein Klartext 0 war. Ähnlich der in Kapitel eins beschriebenen Möglichkeit zum Erstellen einer Bitmaske, um die Integritätsprüfung zu umgehen, lässt sich nun der Einfluss des angenommenen Wertes 0 auf ICV errechnen und entfernen. Daten und modifizierte ICV werden wieder verbunden und wie beim Arbaugh-Angriff an den AP geschickt. War 0 das richtige Byte, wird ICV wieder gültig und somit vom AP akzeptiert; wenn nicht, verbleiben noch 255 weitere Möglichkeiten für dieses Byte. So wird weiter verfahren, bis alle Bytes des gewünschten Paketes entschlüsselt sind. Als Ergebnis erhält man neben dem Klartext auch den Schlüsselbitstrom. Zum Dekodieren benötigt ChopChop je nach Länge des Paketes und Auslastung des WLAN einige Sekunden bis wenige Minuten.

Der Nutzen solcher Werkzeuge erschließt sich nur noch versierten Anwendern direkt. Es ist nicht realistisch anzunehmen, man könne mit ChopChop in Echtzeit den gesamten Verkehr in einem auch nur halbwegs belebten WLAN entschlüsseln. Ein Ziel ist es hingegen, punktuell einzelne

Pakete zu entschlüsseln, um dem Klartext Informationen zu entnehmen, etwa IP-Adressen von Clients oder die Art der kommunizierten Daten. Der hauptsächliche Einsatzbereich liegt aber darin, auf diese Weise schnell genug PRGA-Ausgaben zu gewinnen, um eigene Pakete in das WLAN einschleusen zu können. Warum es Ziel eines Angreifers sein kann, beliebige Daten in ein WLAN zu injizieren, wird im Abschnitt zu Angriffen gegen die Integrität genau betrachtet. ChopChop ist – anders als die FMS bzw. KoreK, die für sich selbst genommen rein passiv sind – ein aktiver Angriff. Hier kann einem WLAN-Betreiber sehr wohl auffallen, dass er zur Zeit angegriffen wird, da sich permanent Pakete im WLAN befinden, die aus dem regulären Datenstrom herausstechen, sei es in Anzahl oder hinterlassenen Fingerprints. Abwehrstrategien zu erkennbaren Angriffen werden im Detail in Kapitel fünf im Rahmen von Abwehrstrategien gegen Angriffe auf WLAN diskutiert.

Eine weitere sehr interessante Möglichkeit, den PRGA anzugreifen, wird in [BI, BHL] beschrieben. Hier wird die im 802.11 [IEEE 3] vorhandene Möglichkeit zur Fragmentierung von Dateneinheiten (MSDU – Mac Service Data Unit) in MPDU (Mac Protocol Data Unit) genutzt. 802.11 unterstützt dabei bis zu 16 Fragmente einer MSDU, die jeweils unabhängig verschlüsselt werden. Geht man nun davon aus, dass ein Angreifer zu jedem nicht fragmentierten Paket, das per WLAN übertragen wird, acht Byte Klartext/Schlüsseltext-Paarung kennt (LLC + SNAP Header bilden acht Byte und sind in der überwiegenden Zeit bekannt! [BI, BHL]), kann er mit Hilfe der Fragmentierung nun $(4_{\text{Daten}} + 4_{\text{ICV}}) \times 16$ Byte beliebiger Daten in das WLAN einspeisen. Abzüglich der 8 Byte LLC/SNAP, 20 Byte IP Header und insgesamt 64 Byte ICV verbleiben 36 Byte beliebiger IP Payload.

Der weitaus interessantere Anwendungsfall des Fragmentierungsangriffs ist jedoch die Erzeugung von beliebig viel PRGA in sehr kurzer Zeit (1500 Byte PRGA in zwei Sekunden). Dazu wird ein Umsetzungsproblem zwischen 802.11-Standard und der Realität ausgenutzt. In [IEEE 3] ist ausdrücklich die Rede von: „Broadcast/multicast frames shall not be fragmented [...]“. In der Realität lässt die WLAN-Infrastruktur die Fragmentierung von Broadcast/Multicast jedoch sehr wohl zu [Bi, BHL].

Nach der oben beschriebenen Methode erzeugt der Angreifer ein Paket und schleust es mit Hilfe der Fragmentierung ein. Allerdings hat dieses Paket eine Besonderheit, es ist ein Broadcast-Paket. Der AP empfängt die einzelnen MPDU, und da eigentlich kein Grund besteht zu fragmentieren, wird wieder ein zusammenhängendes Paket erzeugt. Dieses Paket wird nun

mit Hilfe einer neuen IV verschlüsselt und versandt. Das auf diese Weise weitergeleitete Paket muss vom Angreifer nur noch abgefangen werden, und er erhält, da er ja das Klartext/Schlüsseltext-Paar kennt, nun zur neuen IV 64 Byte PRGA. Bei jedem weiteren Schritt wird dementsprechend mehr PRGA zu einer IV gewonnen.

Ähnlich ChopChop ist das Entschlüsseln einzelner Pakete bzw. die Erzeugung von PRGA-Ausgabe – allein gesehen – in der Praxis nicht besonders relevant. Mit der zum Fragmentierungsangriff veröffentlichten Proof-of-Concept-Implementierung wesside ist es jedoch möglich, ein WLAN vollständig automatisch knacken zu lassen. Die Fragmentierung wird dabei zum Erzeugen von Verkehr benutzt, die eigentliche Entschlüsselung übernimmt aircrack. Wesside ist für BSD-Betriebssysteme geschrieben worden.

Für wesside und ChopChop gilt: Beide stellen sehr spezielle Hardwareanforderungen. Dass für beide der Mode Monitor obligatorisch ist, da ja jeweils Daten injiziert werden und die Reaktion des AP beobachtet werden muss, ist an dieser Stelle offensichtlich. Zusätzlich muss aber Packet Injection unterstützt werden, um direkt Pakete über das WLAN Interface versenden zu können, ohne dass die Firmware der Karte diesen Prozess beeinflusst. In der Regel empfehlen sich dafür Karten mit Prism- (2,2.5,3) oder Atheros-Chipsätzen, hier sind modifizierte Treiber zum Beispiel in der aircrack Suite enthalten, welche Packet Injection erlauben.

Ein letzter Punkt, der an dieser Stelle wichtig erscheint, ist das Erstellen von IV Dictionaries. Sowohl mit ChopChop wie auch mit wesside ist es möglich, so genannte IV Dictionaries zu erstellen, d.h. eine Liste aller IVs mit zugehörigem PRGA. Da es nur 2^{24} ca. 17 Mio möglicher IVs gibt, ist die Speicherung von ca. 1.500 Byte PRGA durchaus realistisch, eine vollständige Liste benötigt ca. 23 GB Speicherplatz. Die im Abschnitt zu Angriffen auf den geheimen Schlüssel beschriebene Filterung von weak IVs vereinfacht diese Aufgabe noch einmal, da die Zahl der IVs weiter gesenkt wird.

3.3.4 Man-in-the-Middle-Angriffe

Nach Angriffen auf den geheimen Schlüssel bzw. auf den PRGA verbleibt noch eine weitere Möglichkeit, die Vertraulichkeit zu kompromittieren. Die so genannten Man-In-the-Middle (MitM)-Angriffe haben ein Ziel: unbemerktes

Zwischenschalten in eine Kommunikation. Dabei sind MitM-Angriffe kein ausschließliches Problem von Wireless Networks. In wired Netzen macht das so genannte ARP Spoofing bzw. ARP Poisoning seit Jahren Probleme. Diese Angriffe basieren auf gefälschten ARP-Antworten, womit das Opfer dazu gebracht wird, Daten an den falschen Empfänger zu senden.

Auch wenn ARP Spoofing im WLAN-Bereich durchaus auch vorstellbar ist – unter der Voraussetzung, dass die Verschlüsselung geknackt wurde bzw. das Netz unverschlüsselt überträgt –, gibt es dort doch einen deutlich einfacheren Weg. Wieder einmal spielen dabei die im 802.11 vollkommen ungesicherten Management Frames eine Rolle.

MitM-Angriffe setzen sich aus mehreren Schritten zusammen, für 802.11 sind das: Identitätsdiebstahl (Identity Theft, Impersonation, Masquerading genannt) und DoS (Denial of Service). Zunächst nimmt der Angreifer die Identität (SSID und MAC) des AP (Connection Hijacking) oder die Identitäten von Client und AP (MitM im eigentlichen Sinne) an. Dann wird durch das Senden von Deauthentication Frames die ursprüngliche Verbindung zwischen AP und Client aufgetrennt, um so den Client zu zwingen, die Verbindung neu zu etablieren. Da während der gesamten Zeit der eigentliche AP mit Deauthentication Frames überhäuft wird, verbindet sich der Client nun aber mit dem AP des Angreifers (eine Alternative stellt es dar, den gefälschten AP mit größerer Sendeleistung zu versehen, da bei gleichen SSIDs in der Regel die stärkere Signalquelle zum Verbinden bevorzugt wird). Ab diesem Zeitpunkt bestehen alle Möglichkeiten. Der Angreifer kann sich nun ebenfalls mit dem legitimen AP verbinden und jeden Verkehr zwischen Client und AP einfach weiterleiten oder nach Belieben modifizieren. Vorausgesetzt ist hier natürlich, dass entweder gar keine Verschlüsselung benutzt wird oder bei WEP eine der bereits vorgestellten Methode verwendet wird. Die Modifikation kann unter anderem auch das äußerst problematische Umleiten auf gefälschte Internetseiten beinhalten, um so etwa an Passwörter und Logins zu kommen, die in der Regel im WLAN nicht sichtbar wären (SSL) .

Im Übrigen ist über einen solchen dazwischengeschalteten AP durchaus auch eine verschlüsselte WPA/WPA2-Verbindung möglich. Die Sicherheit bleibt dabei erhalten, der Angreifer gewinnt lediglich die Möglichkeit, einzelne Pakete gar nicht oder verspätet zu versenden.

Eine vereinfachte Methode dieses in der Praxis doch recht aufwändigen, klassischen MitM-Angriffs stellt das Connection Hijacking dar. Der Angreifer macht sich nicht die Mühe, mit dem AP eine Verbindung zu schaffen, sondern

täuscht direkt ein LAN vor, ebenfalls mit dem Ziel, vertrauliche Daten zu erlangen. Vor allem dieser vereinfachte Weg findet beim Angriff auf Public Hotspot-Nutzer Verwendung. Nicht selten werden hier Anmeldung und Rechnungsleistungen über leicht zu fälschende Captive Portals realisiert. Der Angreifer kann auf diese Weise einfach und schnell auf Kosten eines Dritten den Service nutzen.

Auch für die Realisierung von MitM-Angriffen gibt es eine Reihe von Implementierungen, die speziell auf 802.11 ausgelegt sind. Der bekannteste Vertreter ist hier AirSnarf [SG 2]. AirSnarf ist im Grunde nur ein Script, welches :Hostap Treiber, iptables, httpd, dhcpd, sendmail und Net::DNS Perl benutzt, um einen vollständigen Access Point mit Captive Portal zu erzeugen. Clients, die zu diesem Rogue AP eine Verbindung aufbauen, erhalten, wie bei einem normalen AP, per DHCP eine IP. Jede Anfrage, die sie jedoch im Folgenden an den DNS stellen, wird so aufgelöst, dass sie auf dem Captive Portal enden, auf welchem sie dann aufgefordert werden, sich einzuloggen. Auf dieser Weise gestohlene Logins und Passwörter werden per Sendmail an den Nutzer geschickt. Die Konfiguration erfolgt über die Datei airsnarf.cfg.

Erweiterte Grundkenntnisse über Linux und die benutzten Netzwerktechnologien sind vonnöten, wenn über die Basiskonfiguration von AirSnarf hinaus etwa Routing in das Internet für bestimmte Adressen erfolgen soll.

Rogue Accesspoints gehören zu den wenigen Angriffen, die man im Wireless LAN per IDS automatisch erkennen kann. Strategien hierzu werden im folgenden Kapitel im Verlauf der Bewertung vorgestellt.

Als zweiter Vertreter der Man in the Middle-Angriffe soll im Rahmen dieser Arbeit das Tool airpwn betrachtet werden. Ziel dieser Implementierung ist es, Anfragen von Stationen an beliebige Server mit eigenen Inhalten zu beantworten. Airpwn betrachtet dabei alle möglichen Pakete, die ein Client absetzt, und sucht nach bestimmten angefragten Inhalten. Das können z.B. Bilder, Webinhalte oder bestimmte Anfragen (FTP) sein. Wird ein Frame mit passendem Inhalt gefunden, antwortet airpwn an Stelle des eigentlichen Servers an den Client. Das funktioniert aus einem einfachen Grund: Die Antwortzeit der meisten Internetserver liegt deutlich über der von lokalen Servern (lokale Server hier gleich airpwn). In der Regel akzeptieren Webbrowser o.ä. die erste gültige Antwort und ignorieren alle weiteren. So wird der Angreifer in die Lage versetzt, dem Opfer beliebige Inhalte aufzwingen zu können.

3.4 Angriffe gegen die Integrität

In Kapitel eins zeigte sich deutlich, dass der Schutz der Integrität in WEP nicht nur problematisch, sondern in der Praxis nicht vorhanden ist. ChopChop und teilweise auch wesside, welche genau hier ansetzen und die mangelhafte Integritätsprüfung (über rICV) ausnutzen, wurden bereits betrachtet und sollen hier nicht noch einmal Gegenstand sein. Vielmehr soll es nun um die auch zum Schutzziel Integrität gehörige Replay Protection gehen. An dieser Stelle vorweg genommen sei, dass es sich bei dieser Art nicht um einen Angriff handelt, der für sich selbst betrachtet eine praktische Bedeutung hat. Replay wird in der Regel zur Beschleunigung von Angriffen auf den geheimen Schlüssel benutzt.

Die Vorgehensweise ist dabei umweglos: Der Angreifer wartet, bis er eine ARP Request im WLAN sieht. Das kann er selbst im verschlüsselten Netz aus einem einfachen Grund: ARP Requests haben eine feste Länge von 34 Byte. Der Angreifer nimmt diesen auf und spielt ihn wiederholt (je nach Hardware einige hundert Mal pro Sekunde) in das WLAN ein. Erfolg dieser Aktion ist ein massives Verkehrsaufkommen und damit einhergehend eine große Anzahl von IVs im Netz. So wird es möglich, selbst in wenig ausgelasteten Wireless LANs, in denen das Capturen von regulärem Verkehr sonst Tage dauern würde, in kurzer Zeit so viele einzigartige IVs zu sammeln, dass ein Angriff auf den geheimen Schlüssel Erfolg versprechend ist.

File2air, void11, wnet dinject/reinject und aireplay(-ng) sind hier die am weitesten verbreiteten Werkzeuge. Aireplay-ng ist wie auch schon airodump-ng Bestandteil der aircrack-ng suite. Aus Gründen der Konsistenz soll hier wieder das zu aircrack gehörige Werkzeug näher betrachtet werden.

Aireplay-ng bietet neben der Möglichkeit, abgefangene ARP Requests automatisch für einen Replay-Angriff zu nutzen, auch die Möglichkeit, manuell eine Auswahl zu treffen. Aber es können nicht nur ARP Requests eingespielt werden, sondern auch beliebige Pakete, die etwa mit packETH oder packetforge-ng erzeugt wurden. An dieser Stelle kommt auch der mit den im vergangenen Abschnitt beschriebenen Werkzeugen gewonnene PRGA zum Einsatz, um diese beliebigen Pakete zu verschlüsseln. ChopChop ist genau genommen sogar eine Option für aireplay.

Um legitime Teilnehmer temporär vom AP zu trennen, bietet aireplay die Option, DeAuthentication Frames Netzen zu injizieren. Neben den schon erwähnten Anwendungsfällen von DeAuthentication Frames gibt es noch

einen weiteren, eher als Nebeneffekt zu bezeichnenden Fall. Einige Clients säubern nach einem Deauthenticate komplett ihre ARP-Tabelle, was beim erneuten Verbinden mit dem AP zur Generierung von ARP Requests führt (Quelle: airplay Manual Pages).

Durch das automatische Erkennen und Wiedereinspielen von ARP Requests bietet aireplay eine vergleichsweise simple Möglichkeit, Verkehr zu erzeugen. Leider ist die Methode nicht immer optimal, da zeitweise keine oder nicht zu gebrauchende ARP Requests erzeugt werden. In diesem Fall hat es Sinn, ein beliebiges Paket mit ChopChop zu entschlüsseln, um so ein selbst erzeugten ARP Request einzuspeisen.

3.5 Angriffe auf die Verfügbarkeit

In diesem Abschnitt soll darauf verzichtet werden, im Detail auf die verschiedenen Methoden zur Durchführung von Angriffen auf die Verfügbarkeit einzugehen. Denn die Optionen, die ein Angreifer hat, um ein WLAN einfach nur lahmzulegen bzw. zu stören, sind in ihrer Anzahl schier endlos. Die Möglichkeiten reichen von physischen Angriffen, dem Diebstahl des AP, über technische Angriffe, Frequenz Jamming, bis hin zu Angriffen, die Probleme in den zum WLAN gehörigen Protokollen ausnutzen, etwa Probleme im Michael Schlüsselaustausch bei WPA [HE].

Wie im Verlauf bereits einige Male angesprochen, werden Angriffe auf die Verfügbarkeit teilweise benötigt, um bestimmte Aktionen zu erzwingen, etwa den Reconnect eines Clients. Diese Art von Angriffen soll jedoch hier keine Rolle mehr spielen.

Verfügbarkeitsangriffe müssen aber nicht immer beinhalten, dass ein bestimmter AP nicht mehr zu erreichen ist. Es reicht im Umfeld eines Public Hotspots bereits zu, wenn ein Angreifer eine große Anzahl von nicht existenten APs vortäuscht, um so das Auffinden des gesuchten APs zu erschweren. Zu erwähnen ist in diesem Zusammenhang fakeAP.pl [FAP], welches genau diese Aufgabe übernimmt. Mit Hilfe von ifconfig und iwconfig wird eine beliebig große Zahl von APs vorgetäuscht, es können dabei jeweils gültige SSID und MAC aus einer mitgelieferten Liste zufällig gewählt werden. Die APs erfüllen jedoch keinerlei Funktion. FakeAP.pl wird auch benutzt, um als Honeytrap unerfahrene WarDriver zu täuschen. Erfahrene Angreifer werden von FakeAP.pl jedoch kaum behindert, da ein einfacher Blick auf die BSS Timestamp des Beacon Frames ausreicht, um festzustellen, dass der AP

eine Uptime von lediglich einigen Millisekunden hat. Das Tool Raw FakeAP [RFA] hat das gleiche Ziel wie fakeAP.pl, benutzt jedoch raw packet injection, um APs vorzutäuschen. So wird erreicht, dass auf der Anwendungsebene der volle Zugriff auf alle Felder des 802.11-Frames besteht, so eben auch auf die BSS Timestamp. Raw Fake AP ist ebenfalls in der Lage, auf Probe Requests zu antworten.

3.6 Fazit

So zahlreich wie die Sicherheitslücken in WEP sind auch die Implementierungen von Tools, die genau diese ausnutzen. Hier konnte nur ein kleiner Einblick in die Vielzahl von vorhandenen Tools geboten werden. Umfangreiche Listen sind unter [WiFoo], [WVE] oder auch [LP] zu finden. Jedoch sind dort teilweise noch Tools enthalten, die ihre Bedeutung aus verschiedenen Gründen verloren haben. Einige sind einfach veraltet und wurden durch effektivere Methoden ersetzt, andere basieren auf Sicherheitslücken, die in der Realität kaum noch anzutreffen sind.

Deutlich wurde aber, dass für jede bekannte Lücke im WEP-Sicherheitssystem auch eine Implementierung existiert. Wie komplex bzw. wie Erfolg versprechend einzelne Werkzeuge sind, wurde bislang nur angedeutet und soll ausführlich im folgenden Kapitel besprochen werden.

Bis zu diesem Punkt ist es offensichtlich geworden, dass WEP absolut unsicher hinsichtlich der Schutzziele Vertraulichkeit und Integrität ist. Die verbleibende Frage ist nun: Wem steht die Ausnutzung dieser Lücken offen und wie groß kann im Einzelnen der angerichtete Schaden sein?

4 Angriffe in der Realität

In den Kapiteln eins und drei wurde umfangreich auf die Grundlagen der Angriffe und auf die vorhandene Vielfalt an Werkzeugen eingegangen. Nun ist es Zeit, auf den Arbeitsablauf bei einzelnen Angriffen aus Sicht des Anwenders einzugehen. Dabei soll nach drei Gesichtspunkten bewertet werden: Schwierigkeit, Erfolgsaussichten, potentielle Gefährdung. Es geht nicht mehr darum, wie noch in Kapitel drei, die Werkzeuge mehr oder weniger losgelöst voneinander zu betrachten, hier wird auch auf das Zusammenspiel eingegangen.

Einzelne Angriffsszenarien werden mit den aus Kapitel zwei bekannten Attack Trees dargestellt. Anhand dieser Darstellungen und dem Wissen über die Funktionsweise der Angriffe sollen – sofern möglich – Abwehrstrategien entwickelt werden. Zum Teil wird es dabei möglich sein, auf bereits vorhandene Mechanismen zurückzugreifen.

Zum Umsetzen wird aus jeder Gruppe von Angriffswerkzeugen ein repräsentativer Vertreter gewählt. Es erscheint nicht sinnvoll, auf veraltete oder Reimplementierungen einer Gruppe (Schlüsselangriff, MitM, Verfügbarkeit) von Werkzeugen einzugehen, denn immerhin ist es ja Ziel, Angriffe in der Realität zu bewerten. Dabei kann davon ausgegangen werden, dass je nach Angreifer entweder der einfachste oder aber der effektivste Weg gewählt wird, nicht jedoch ein überholter oder sehr komplexer.

4.1 Versuchsanordnung

Die Versuchsumgebung besteht aus zwei Desktop PCs (Monitor und Opfer), einem Laptop (Angreifer) und einem Access Point (NETGEAR FWAG114). Da der Monitor lediglich benutzt wird, um das WLAN passiv zu überwachen, genügt hier ein einfacher USB-WLAN-Adapter (NETGEAR MA111), der im Mode Monitor betrieben wird. Auf der Seite des Opfers ist ein PC-Card-Adapter installiert, in dem eine PCMCIA-Karte (Linksys WPC11) betrieben wird. Im Angreifer wird ebenfalls eine PCMCIA-Karte der Marke ZCOM (XI-325HP+) verwendet. Alle WLAN Interfaces sind mit einem Prism 2-Chipsatz ausgerüstet.

Zum Betrieb der Testrechner wird Linux (BackTrack Final 1.0 (Slackware) [RME]) in einer LiveCD-Version verwendet. Dies birgt in diesem Fall gleich

mehrere Vorteile. Zum einen ist BackTrack eine Distribution, die speziell auf (Wireless) Penetration Testing ausgelegt ist. Dadurch ist nahezu der gesamte hier betrachtete Pool an Werkzeugen vorhanden. Außerdem sind alle WLAN-Treiber bereits mit den notwendigen Patches versehen, um Packet Injection im Mode Monitor zu gewährleisten. Des Weiteren wird durch die Verwendung eines Live Betriebssystems erreicht, dass die Testumgebung für jeden neuen Versuch sehr einfach in den Urzustand versetzt werden kann. Nicht zuletzt spielt aber auch eine Rolle, dass alle Versuche möglichst einfach durchzuführen sind und so wenig Vorbereitung wie möglich benötigen. Schließlich müsste ja im Fall einer regulären Linux-Distribution der Aufwand und die Schwierigkeiten bei der Installation und der Einrichtung von Treibern und Tools mit einbezogen werden, durch die Verwendung eines Live Systems entfällt das aber fast vollständig. So müssen deutlich weniger Annahmen über das Wissen des Angreifers gemacht werden. Es genügt zu sagen, dass grundlegende Kenntnisse in der Bedienung von Linux notwendig sind, um zumindest Teile der im Folgenden betrachteten Angriffe durchführen zu können.

In Tabelle 4-1 sind die notwendigen Daten zu Konfiguration des WLAN aufgelistet:

Netzwerk Name SSID	NETGEAR
Channel	11 (2.462 GHz)
Netz	192.168.0 (Default
Netzmaske	NETGEAR) 255.255.255.0
Router MAC Intern	00:09:5B:66:52:F8
Router MAC Extern	00:09:5B:66:52:F9
BSSID	00:09:5B:66:2E:73
Opfer MAC	00:06:25:2A:F7:68
Angreifer MAC	00:60:B3:2D:9D:BF
Monitor	00:09:5B:74:D8:6B
WEP	Passphrase: Diplomarbeit Key1-4: 258439718550D7E3 A4C2D7E57B

Tabelle 4-1: Konfiguration des WLAN

Für alle Versuche gegen WEP bzw. WPA kann davon ausgegangen werden, dass diese auch aktiviert sind. Bei Versuche zu Man in the Middle-Angriffen wird auf Verschlüsselung verzichtet, da diese an sich transparent sind und somit unabhängig von Verschlüsselung.

4.2 Angriffe ohne Verschlüsselung – Mac Spoofing

Der erste Angriff ist gleichzeitig auch der denkbar einfachste. Ziel des Mac Spoofings ist, die Zugangskontrolle über ACLs zu umgehen. In dem der Angreifer sich die MAC-Adresse eines legitimen Clients aneignet, um so dessen Identität zu übernehmen.

Notwendig sind dafür nur zwei sehr einfache Schritte: einen legitimen Client zu finden und dessen MAC-Adresse in Erfahrung zu bringen. Darauf folgt das Übernehmen der MAC-Adresse.

Zum Auffinden eines Clients und dem Ermitteln seiner MAC-Adresse können beliebige Scanning-Werkzeuge benutzt werden, hier wird Kismet [KI] verwendet. Ist eine MAC-Adresse gefunden, hängt das weitere Vorgehen vom Betriebssystem ab. Für Windows gibt es die Möglichkeit, die MAC entweder über die zum WLAN Interface mitgelieferte Software oder über macmakeup [MM] zu ändern. Im Linux-Segment ist das Ändern der MAC einfach über `ifconfig <interface> hw ether <MAC>` möglich, es steht aber auch ein Script (macchange) zur Verfügung, mit dem das gleiche Ziel erreicht wird. Ein einfacher Attack Tree, der den gesamten Ablauf beschreibt ist, in Abbildung 4-1 dargestellt.

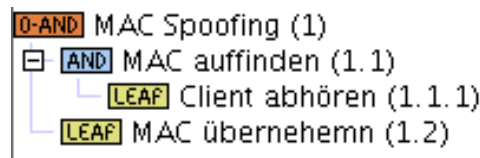


Abbildung 4-1: Attack Tree für MAC Spoofing

In [JW] wird eine Methode beschrieben, mit der sich MAC Spoofing aufdecken lässt. Dazu werden Sequenznummern aufeinander folgender Pakete verglichen. Springen diese hin und her, so kann man daraus auf MAC Spoofing schließen. In der Praxis ist diese Methode zur Aufdeckung von MAC Spoofing aber eher von beschränkter Einsatzfähigkeit.

Der Schwierigkeitsgrad ist hier sicherlich im unteren Bereich anzusiedeln. Anders sieht es mit den Erfolgsaussichten aus. Abgesehen von der Tatsache, dass es einen Client geben muss, dessen MAC man sich aneignet und dass vom WLAN Interface das Ändern der MAC unterstützt werden muss, ist der Erfolg nahezu sicher. Die Gefährdung, sowohl für den Angreifer entdeckt zu werden, wie auch für den Angegriffenen Schaden zu erleiden, ist bei

diesem Angriff zwischen nicht vorhanden und groß einzustufen. Ist in der Umgebung ein IDS in Betrieb, welches die in [JW] beschriebene Methode benutzt, so wird der Angriff sicherlich entdeckt. Dass der Angreifer selbst damit unmittelbar Gefahr läuft, auch entdeckt zu werden, bedeutet dies jedoch nicht.

Für den Client richtet sich die potentielle Gefährdung zum einen nach den Absichten des Angreifer (nur Bandbreitendiebstahl oder hoch kriminelle Aktivitäten) und nach vorhandenen Sicherheitseinrichtungen auf höherer Ebene (WEP bzw. WPA).

4.3 Passiver Angriff auf den geheimen Schlüssel

Die Überschrift „Passiver Angriff auf den geheimen Schlüssel“ ist genau genommen etwas irreführend. Tatsächlich ist es so, dass der eigentliche Angriff auf den Schlüssel sowohl bei aktivem wie auch bei passivem Angriff vollkommen identisch ist. Passiv bezieht sich ausschließlich auf die Methode zur Traffic-Sammlung. Überhaupt steht fest, dass von Seiten des Anwenders das Errechnen des geheimen Schlüssels die einfachere Aufgabe darstellt. Problematischer ist es, möglichst schnell an genügend Pakete mit einzigartigen IVs zu gelangen, um mit der Berechnung beginnen zu können.

Beim passiven Sammeln von Paketen wird nun ausschließlich darauf vertraut, dass im angegriffenen WLAN von Hause aus genug Verkehr erzeugt wird. Gerade für viele Wireless LANs im privaten Bereich führt das aber unweigerlich dazu, dass nicht mehr Stunden benötigt werden, um ausreichend Daten zu sammeln, sondern ganze Tage, womit der Angriff unrentabel wird.

Die Bewertung der Gefährdung für Angreifer und Opfer lässt sich an dieser Stelle vorwegnehmen. Da vom Angreifer zu keinem Zeitpunkt Spuren im Netz hinterlassen werden, kann er auf digitalem Weg auch kaum aufgespürt werden, er lebt also fast vollkommen sicher. Genau entgegengesetzt verhält es sich beim Opfer: Nicht nur, dass der benutzte WEP-Schlüssel sofort alle Sicherheitseigenschaften verliert. Der Angegriffene – dem ja der Angriff wahrscheinlich gar nicht bewusst wurde – wird auch keine wie auch immer gearteten Gegenmaßnahmen ergreifen (neuer WEP Schlüssel als Beispiel). Abgesehen vom Problem der Traffic-Generierung sind passive Angriffe die mit dem größten Gefahrenpotential.

Um die Testzeit für die passiven Tests in überschaubarem Rahmen zu halten, sendet das Opfer permanent ICMP-Pakete an die Broadcast-Adresse des Wireless LANs (ping -f -b 192.168.0.0). So wird auf *natürlichem* Weg eine große Anzahl von verschlüsselten Paketen erzeugt. Als Vergleichsgröße sei hier einmal angeführt, dass eine Anfrage an www.google.com circa 10-15 verschlüsselte Pakete in der Testumgebung erzeugte, eine Anfrage an www.spiegel.de hingegen circa 800.

4.3.1 mit Aircnort

Vorgestellt wurde Aircnort [SG] bereits in Kapitel drei. Die Handhabung ist, wie schon beschrieben, denkbar einfach. Aircnort übernimmt nach dem Starten automatisch das Aufzeichnen aller Kommunikation auf dem gewählten WLAN-Kanal und versucht, nach fester Anzahl von Paketen jeweils automatisch den Schlüssel zu errechnen. Mehr ist zur Bedienung nicht notwendig. Damit stellt Aircnort den einfachsten hier betrachteten Weg zum Errechnen des geheimen WEP-Schlüssels dar.

Aircnort führt jedoch nicht auf dem schnellsten Weg zum Ziel. Aus nicht erklärbaren Gründen war es mit dem in Aircnort implementierten Packet Capturer nur möglich, durchschnittlich 2200 Pakete pro Minute aufzuzeichnen. Im gleichen Zeitraum konnten mit airodump-ng knapp doppelt so viele Pakete erfasst werden. Dazu kommt, dass bei einigen Versuchen der Schlüssel mit Aircnort erst deutlich später (größere Anzahl an benötigten Paketen) errechnet werden konnte als mit dem Vergleichswerkzeug aircrack-ng. Zusammen betrachtet, schmälern diese beiden Punkte die Erfolgsrate über Zeit bei Aircnort doch empfindlich gegenüber anderen Werkzeugen. Dennoch bleibt Aircnort, auch ob des übersichtlichen Optionsumfangs, das einfachste Werkzeug.

Abbildung 4-2 zeigt Aircnort nach erfolgreichem Brechen des geheimen Schlüssels der Testumgebung.

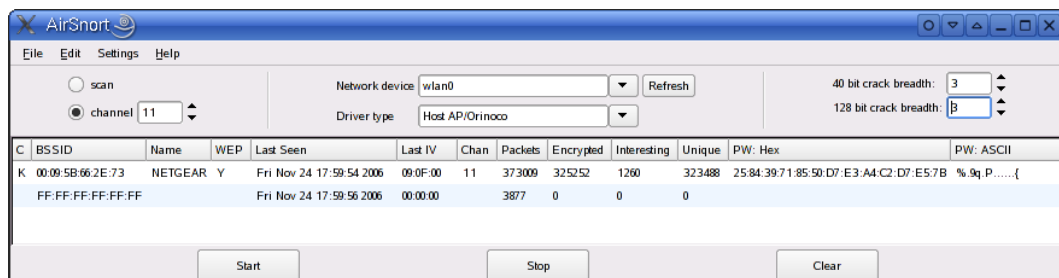


Abbildung 4-2: Aircnort nach erfolgreich gebrochenem Schlüssel

4.3.2 mit aircrack-ng und airodump-ng

Zur eigentlichen Aufzeichnung der Pakete wurde schon gesagt, dass airodump-ng die bessere Performance bietet. Technisch ließe sie sich auch mit Ethereal oder ähnlichen Tools bewerkstelligen. Airodump-ng wird von der Kommandozeile mit `airodump-ng -c 11 -w dump wlan0` ausgeführt (c:Kanal; w: Ausgabedatei). Abgesehen von der Bildschirmausgabe in Bild 4-3 liefert airodump-ng eine Ausgabe-Datei im allgemein üblichen pcap-Format.

Aircrack-ng wird sinnvollerweise nach 100.000 Paketen mit `aircrack-ng dump.cap` aufgerufen, weitere Optionen sind zum Tuning möglich. Mit verändertem Fudge Factor (-f 4) konnten Schlüssel beispielsweise aus Samples berechnet werden, bei denen Aircrack-ng kein Ergebnis lieferte. Die benötigte Rechenzeit steigerte sich jedoch auf teilweise über neun Stunden. In gut ausgelasteten Netzen ist es daher eher sinnvoll, einige tausend Pakete mehr abzuwarten.

Der Schwierigkeitsgrad ist nicht deutlich höher als beim Werkzeug Aircrack-ng. Obwohl die Optionsvielfalt und der Fakt, dass hier zwei Programme (airodump und aircrack) bedient werden müssen, dazu führen, dass die Komplexität, wenn auch nur geringfügig, über der von Aircrack-ng liegt.

Die Erfolgsaussichten hängen, wie schon bei Aircrack-ng, wesentlich von der Menge der generierten Pakete ab. Allerdings ist es durch die vielfältigen Optionen zum Tuning des Angriffs auf den Schlüssel möglich, mit deutlich weniger Paketen ans Ziel zu gelangen. Wie die einzelnen Optionen zu wählen sind, müssen letztendlich das Wissen und die Erfahrung des Angreifers entscheiden.

4.3.3 Abwehrstrategien gegen Passive Schlüsselangriffe

Gegen passive Angriffe gibt es keinen auf Dauer wirksamen Mechanismus zur Abwehr. Eine mögliche Strategie wäre, häufig den geheimen Schlüssel zu wechseln, so könnte man wenigstens wenig ausgelastete Netze schützen. Leider versagt dieser Ansatz vollkommen gegen die im Folgenden betrachteten aktiven Angriffe. Die einzig wirklich wirksame Lösung stellt die vollkommene Umstellung von WEP auf WPA/WPA2 dar.

4.4 Aktiver Angriff auf den Schlüssel

Gegen kleine Netze mit wenig Datenverkehr werden passive Angriffe schnell unrentabel. Als Alternative zum stundenlangen Warten auf ausreichend Pakete, kann der Angreifer selbst dafür sorgen, dass neue Pakete erzeugt werden. Dabei tauscht er dann die Anonymität der passiven Angriffe gegen Geschwindigkeit. Dieser Tausch sollte jedoch nicht überbewertet werden. Der Verlust an Anonymität geht nicht unbedingt mit einem dramatischen Anstieg der Gefahr für den Angreifer einher, entdeckt zu werden. Anders ausgedrückt es kann zwar festgestellt werden, dass angegriffen wird, aber es gibt keine Gegenmaßnahmen, die den Angreifer als Person in Gefahr bringen könnten. Zumindest besteht nun aber überhaupt die Möglichkeit, die Anwesenheit eines Angreifers auf technischem Weg nachzuweisen. Für das Opfer verschiebt sich die Gefährdung ebenfalls ein wenig. Zwar kann nun ein Angriff – wenn auch mit einem Aufwand, der für Privatpersonen und kleine Unternehmen unrealistisch erscheint – festgestellt werden, jedoch werden Netze zum potentiellen Ziel, die vorher keines waren. Ob tatsächlich aktiv angegriffen wird, hängt neben dem offensichtlichen Punkt des Verkehrsaufkommens natürlich auch von der Motivation bzw. den Zielen des Angreifers ab. Ist es einem Angreifer mehr oder minder egal, entdeckt zu werden, etwa einem Bandbreitendieb, wird er eher einen aktiven Angriff wählen. Geht es ihm hingegen darum, unerkannt und möglichst lange Informationen zu sammeln, so wird er, sofern möglich, die passive Methode bevorzugen.

Das Sammeln von Frames und der darauf folgende Angriff auf den Schlüssel sind bei passivem und aktivem Angriff identisch, daher wird hier auch nur noch auf die Methoden zur Generierung von Frames eingegangen.

4.4.1 ARP Replay über aireplay-ng

Die Ausgangssituation ist in Abbildung 4-3 dargestellt. Nach 8 min Laufzeit war es möglich, lediglich acht verschlüsselte Pakete zu beobachten – völlig unzureichend, um in absehbarer Zeit aircrack-ng starten zu können. Die beiden erkennbaren Stationen haben im Übrigen nichts mit der Versuchsumgebung zu tun, verdeutlichen aber noch einmal am realen Beispiel, wie problematisch die Beschränkung von WLAN ist.

CH 11][Elapsed: 8 mins][2006-11-24 15:07

BSSID	PWR	Beacons	# Data	CH	MB	ENC	ESSID
00:09:5B:66:2E:73	244	1768	8	11	54	WEP	NETGEAR

BSSID	STATION	PWR	Packets	Probes
(not associated)	00:05:3C:03:A4:87	185	112	guest
(not associated)	00:13:02:DE:1D:2B	188	20	linksys_0w_59345,B

Abbildung 4-3: airodump-ng: nach 8 Minuten Laufzeit

Es wird nun aireplay-ng benutzt, um automatisch einen ARP Request aufzufangen (das Erkennen eines ARP-Requests in dem ja eigentlich verschlüsselten Datenstrom ist anhand seiner festen Länge möglich) und ihn dann wiederholt in das Netz zu spielen. Abbildung 4-4 zeigt den aireplay-ng-Aufruf.

```
slax active_dump # aireplay-ng --arpplay -b 00:09:5B:66:2E:73 -h 00:06:25:2A:F7:68 wlan0
Saving ARP requests in replay_arp-1124-151338.cap
You should also start airodump-ng to capture replies.
Read 68 packets (got 0 ARP requests), sent 0 packets...
```

Abbildung 4-4: aireplay Aufruf: aireplay-ng <Modus> <BSSID> <Station><NIC>

Offensichtlich ist dieser Versuch vorerst nicht von Erfolg gekrönt (0 ARP Requests). Grund dafür: ARP Requests werden in Netzen mit wenigen Stationen nicht häufig versandt. Abhilfe schafft hier eine einfacher Trick: Ebenfalls mit aireplay werden DeAuthentication Frames an den AP geschickt (Abbildung 4-5), dies veranlasst die verbundenen Stationen, die Verbindung mit dem AP neu zu initialisieren. Der gewünschte Nebeneffekt dabei: Im Regelfall werden vor dem Neuverbinden bestehende ARP-Tabellen gelöscht. Sobald die Verbindung nun wieder hergestellt ist, wird die Station also gezwungen sein, einen neuen ARP Request zu verschicken, um zumindest die interne MAC des in den AP eingebauten Routers zu ermitteln.

```
slax active_dump # aireplay-ng --deauth 20 -a 00:09:5B:66:2E:73 -c 00:06:25:2A:F7:68 wlan0
15:21:59 Sending DeAuth to station -- STMAC: [00:06:25:2A:F7:68]
15:22:00 Sending DeAuth to station -- STMAC: [00:06:25:2A:F7:68]
15:22:01 Sending DeAuth to station -- STMAC: [00:06:25:2A:F7:68]
```

Abbildung 4-5: DeAuthentication mit aireplay:<Modus><Anzahl><BSSID><Station>

Hinweis: Wählt der Angreifer die Anzahl der versendeten DeAuthentication Frames zu hoch (wie abgebildet), so meldet Kismet einen Angriff.

Für das Opfer ist dieses Vorgehen fast nicht zu bemerken. Unter Umständen kommt es zum kurzen Aussetzen der Verbindung. In der Regel liegt die Zeitspanne im Bereich von einigen Sekunden, abhängig von der Anzahl der gesendeten DeAuthentication Frames.

```
slax active_dump # aireplay-ng --arpreplay -b 00:09:5B:66:2E:73 -h 00:06:25:2A:F7:68 wlan0
Saving ARP requests in replay_arp-1124-151338.cap
You should also start airodump-ng to capture replies.
Notice: got a deauth/disassoc packet. Is the source MAC associated ?
Notice: got a deauth/disassoc packet. Is the source MAC associated ?
Notice: got a deauth/disassoc packet. Is the source MAC associated ?
Notice: got a deauth/disassoc packet. Is the source MAC associated ?
Notice: got a deauth/disassoc packet. Is the source MAC associated ?
Notice: got a deauth/disassoc packet. Is the source MAC associated ?
Read 38085 packets (got 30177 ARP requests), sent 32840 packets...
```

Abbildung 4-6: Reaktion des ARP Replay Modus auf die eingestreuten Deauthentication Frames

In der obigen Abbildung [4-6] ist die Reaktion von aireplay-ng dargestellt, nachdem ein verbundener Client deauthentifiziert wurde. Jeder in das WLAN eingespielte ARP Request erzeugt nun zumindest eine Antwort. Es ist dabei nicht zwingend notwendig, dass dieser Angriff gegen eine wireless Station gerichtet ist. Dabei werden in diesem Fall mehr Pakete erzeugt, da der AP beim Weiterleiten jeden Frame erneut verschlüsselt.

CH 11][Elapsed: 26 mins][2006-11-24 15:24								
BSSID	PWR	Beacons	# Data	CH	MB	ENC	ESSID	
00:09:5B:66:2E:73	240	6886	16736	11	54.	WEP	NETGEAR	
BSSID	STATION		PWR	Packets	Probes			
CH 11][Elapsed: 27 mins][2006-11-24 15:25								
BSSID	PWR	Beacons	# Data	CH	MB	ENC	ESSID	
00:09:5B:66:2E:73	249	7130	20809	11	54.	WEP	NETGEAR	
BSSID	STATION		PWR	Packets	Probes			

Abbildung 4-7: Rapider Anstieg der in der Data-Spalte deutet auf Erfolg.

Auf diesem Wege konnte (Abbildung 4-7) offensichtlich ein Erfolg erzielt werden. Der schnelle Anstieg der gesammelten (verschlüsselten) Frames lässt es also nun selbst in diesem eigentlich fast vollkommen inaktiven WLAN zu, den Schlüssel zu attackieren. Dafür wurden in diesem speziellen Versuch sogar lediglich 275.758 Frames und 2:26 min benötigt, um mittels aircrack-ng den geheimen Schlüssel zu ermitteln. Insgesamt wurden ca. 70 Minuten benötigt, um genug verschlüsselte Frames zu sammeln und den Schlüssel zu berechnen. Das ist zwar schon ein recht gutes Ergebnis, unter Umständen lässt sich die benötigte Zeit aber noch optimieren, wenn für Capturing und Injection zwei Interfaces benutzt werden. Ziel dieser Arbeit ist es jedoch nicht, die Angriffe bis zum Optimum zu treiben. Deshalb wird hier darauf verzichtet, diesen noch einmal zu beschreiben.

Die derzeit einzige Möglichkeit, diesen aktiven Angriff abzuwehren, besteht darin, zu erkennen, dass es sich bei dem eingespielten ARP Request immer um exakt denselben handelt. Der AP könnte in diesem Fall das Weiterleiten unterbinden. Das Problem an diesem Ansatz ist jedoch, dass es von Seiten der Angriffswerkzeuge nur einer kleinen Veränderung an der Implementierung bedarf, um diese Bemühung nahezu wieder zunichte zu machen. Denn zurzeit ist es so, dass, egal wie viele verschiedene ARP Requests ein Angreifer(werkzeug) zur Verfügung hat, er/es immer ein und denselben benutzt. Das Anlegen einer ARP Request-Datenbasis und das anschließende zufällige Wählen eines Requests führt dazu, dass ein solches System umgangen werden kann. Nichtsdestotrotz kann man ARP Replay, wenn auch nicht unterbinden, zumindest anhand der unnatürlich großen Anzahl von ARP Requests erkennen. Die Festsetzung eines Schwellenwerts hängt direkt von der Anzahl der erwarteten Stationen über einen bestimmten Zeitraum ab. In einem privaten WLAN ist es so zum Beispiel unwahrscheinlich, dass auf normalem Weg über einen längeren Zeitraum mehrere tausend solcher Frames pro Minute generiert werden.

Abbildung 4-8 illustriert den gesamten hier beschriebenen aktiven und passiven Angriff noch einmal zusammenfassend als Attack Tree.

Um die Komplexität des vorliegenden Attack Trees nicht weiter zu steigern, wurde an einigen Punkten stark abstrahiert. Es bestünde jedoch ohne weiteres die Möglichkeit, einzelne Blätter weiter aufzuspalten. Dieser und folgende Attack Trees, die im Zusammenhang mit den in Kapitel vier beschriebenen Laborversuchen entstanden sind, bilden eigentlich eine Einheit. Sie wurden hier nur aus Gründen der Lesbarkeit aufgetrennt.

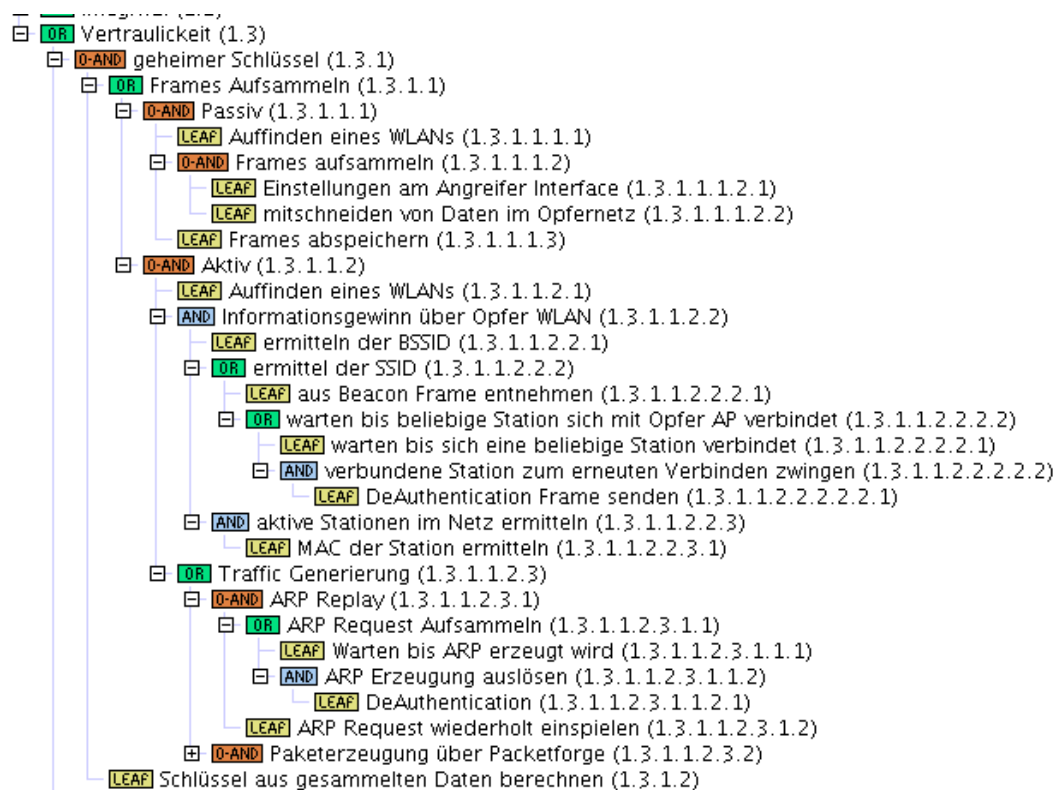


Abbildung 4-8: Attack Tree für Angriff auf den geheimen Schlüssel

Eine zusammenhängende Variante befindet sich auf der beiliegenden CD-ROM.

Abschließend soll auch hier wieder Schwierigkeit und Erfolgsaussicht bewertet werden, nachdem die Bedrohung für Opfer und Angreifer dem Abschnitt bereits vorangestellt wurde.

Obwohl dieser Angriff deutlich komplexer erscheint, als das bloße passive Angreifen des Schlüssels, ist er dennoch der zweiten Klasse von Angreifern (Kapitel 3) zuzuordnen, er tendiert aber schon in Richtung der dritten Gruppe. Um hier effektiv zu arbeiten, ist ein gewisses Vorwissen von Nöten (grundlegende Funktionsweise von 802.11, Funktion von DeAuthentication), von Expertenwissen kann jedoch noch nicht die Rede sein. Die größere Hürde stellt hier generell die Beschaffung der richtigen (Injection im Mode Monitor) Hardware dar.

In der hier benutzten Testumgebung waren die Versuche immer von Erfolg gekrönt. In der Praxis kann sich ein Angreifer aber einem AP gegenüber sehen, der die beschriebene Methode zur ARP Replay-Unterdrückung implementiert. Wie auch im Vorfeld schon erwähnt, kann unabhängig von ARP Replay-Prävention einem Angreifer immer wieder

eine Hardwarekombination auf Seiten des Opfers begegnen, die einzelne Angriffe unterbindet. Diese Konstellation ist aber in der Praxis eher die Ausnahme als die Regel.

4.4.2 Paket Erstellung packetforge-ng und ChopChop

Kann der Angreifer, aus welchen Gründen auch immer, keinen ARP Request abfangen, muss eine andere Möglichkeit gefunden werden, um Pakete zu erzeugen, welche Antworten erzwingen. Möglichkeiten sind dabei zum Beispiel, ARP- oder ICMP-Pakete selbst zu generieren und diese in das Netz zu schleusen, prinzipiell ist aber jedes Paket geeignet, auf das eine Antwort erfolgt. Dafür werden jedoch einige Informationen benötigt. Während BSSID und MAC-Adressen von Stationen und Router einfach abzulesen sind, ist die Netzadresse WEP-verschlüsselt. Es reicht aber aus, ein einziges Paket zu entschlüsseln, um an diese Information zu gelangen, dabei hilft ChopChop.

Die in aireplay-ng implementierte Version von ChopChop bietet entweder die Möglichkeit, einen beliebigen Frame live vom WLAN Interface zu entschlüsseln oder aus einer vorher aufgezeichneten pcap-Datei zu wählen (Abbildung 4-9).

```
BT chopchop # aireplay-ng --chopchop -r replay_src-1128-144518.cap wlan0

Size: 108, FromDS: 0, ToDS: 1 (WEP)

      BSSID = 00:09:5B:66:2E:73
      Dest. MAC = 00:09:5B:66:52:F8
      Source MAC = 00:06:25:2A:F7:68

0x0000: 0841 d500 0009 5b66 2e73 0006 252a f768 .A....[f.s..%*.h
0x0010: 0009 5b66 52f8 205d 910a 0000 468f 1bdd ..[fR. ]....F...
0x0020: c9d4 b23c ab27 1893 aae3 7a45 8157 4a75 ...<.'....zE.WJu
0x0030: 133b 499b 118b 53bf 37f6 6b41 08dc 88ce .;I...S.7.kA....
0x0040: 7d4b 5a38 c426 d446 115a b54b e344 6306 }KZ8.&.F.Z.K.Dc.
0x0050: 809e 0178 607f 2e30 1cbd c424 eddb 9988 ...x`.0...$....
0x0060: 5f31 4fe4 c96e 6201 34b9 c521 _10..nb.4..!

Use this packet ? █
```

Abbildung 4-9: Paketauswahl mit ChopChop

Um den hier aufgezeichneten Frame zu entschlüsseln, wurden weniger als zwei Minuten benötigt. Als Ausgabe liefert ChopChop zwei Dateien.

Zum einen den entschlüsselten Frame (Abbildung 4-10), aus welchem die noch benötigte IP-Adresse des WLANs ermittelt werden kann, und zum anderen eine .xor-Datei, die den zur IV dieses Frames gehörigen PRGA beherbergt.

```
BT chopchop # tcpdump -n -e -r replay_dec-1128-144722.cap
reading from file replay_dec-1128-144722.cap, link-type IEEE802_11 (802.11)
14:47:22.442734 BSSID:00:09:5b:66:2e:73 SA:00:06:25:2a:f7:68 DA:00:09:5b:66:52:f8 LLC, dsap SNAP (0xaa), ssap SNAP (0xaa), cmd 0x03: oui Ethernet (0x000000), ethertype IPv4 (0x0800): 192.168.0.2.32772 > 192.168.0.1.53: 8492+ AAAA? www.remote-exploit.org. (40)
BT chopchop #
```

Abbildung 4-10: Ausgabe tcpdump, mit der noch benötigten Netzadresse 192.168.0

Nun sieht sich der Angreifer in der Lage, mittels packetforge-ng selbst einen (in diesem Fall) ARP Request zu erzeugen (Abbildung 4-11). Dieser kann dann, wie im vergangenen Abschnitt beschrieben, mit aireplay-ng wiederholt in das WLAN eingeschleust werden.

```
BT chopchop # packetforge-ng --arp -w forgedpacket.out -a 00:09:5b:66:2e:73 -h 00:06:25:2a:f7:68 -k 192.168.0.1 -l 192.168.0.2 -y replay_dec-1128-144722.xor
```

Abbildung 4-11: Aufruf Packetforge-ng: -w <Ausgabe> -a <BSSID> -h <QuellMAC> -k <Quell IP> -l <Ziel IP> -y <PRGA Datei>

Der Effekt dieser Schritte ist letztendlich der gleiche wie beim Wiedereinspielen von ARP Requests im vergangenen Abschnitt. Er ist jedoch unabhängig von *echten* ARP Requests im WLAN. In der Testumgebung wäre diese Form des Angriffs prinzipiell nicht notwendig gewesen, da man ja hier durch DeAuthentication Frames jeweils die Erzeugung von ARP Requests auslösen konnte. Gegen APs, die immun gegen DeAuthentication sind (immun bezieht sich auf das Löschen der ARP Tabelle), ist ChopChop eine schnelle Möglichkeit, ARP Requests zu erzeugen.

In puncto Schwierigkeit stellt diese Form des Angriffs sicherlich die Spitze der in dieser Arbeit betrachteten Methoden bzw. Werkzeuge zum Angriff auf den geheimen Schlüssel dar. Es werden neben Kenntnissen zu Wireless

LAN auch mindestens Basiskenntnisse zu den Typen (ARP, ICMP etc.) der jeweils erzeugten Frames benötigt.

Problematisch stellt sich hier die Bewertung der Erfolgchancen dar. In der Testumgebung waren die Versuche zwar erfolgreich. Es wird aber in diversen Foren darauf hingewiesen, dass einige Access Points immun gegen ChopChop sind.

Vergleichsweise einfach stellt sich jedoch das Aufdecken eines ChopChop/Angriffs dar. Im Anhang A und auf der beiliegenden CD-ROM befinden sich die vom Monitor während des Versuchs aufgezeichneten Frames. Daraus wird ersichtlich, dass in einem Zeitraum von einer Sekunde ca. 170 neue Ziel-MAC-Adressen im Datenstrom auftauchen (hier sei noch einmal an die Funktionsweise von ChopChop aus Kapitel 3 erinnert). Das ist in einem WLAN dieser Größe absolut unrealistisch. Es bietet sich hier also an, dass ab dem Überschreiten eines Schwellenwerts von neun MAC-Adressen pro Zeitraum ein Alarm ausgelöst wird. Natürlich ließe sich nun argumentieren, dass bei reduzierter Framerate seitens ChopChop das Erkennen schwieriger bzw. unmöglich würde. Jedoch führen solche Maßnahmen auch dazu, dass ChopChop selbst an Einsetzbarkeit verliert. Die hier beschriebene Methode zur Erkennung lässt sich direkt auf eine Methode zur Abwehr übertragen. Eine Lösung ähnlich der für ihren eigentlichen Zweck nutzlosen ACLs (Kapitel 1) würde auf einfachem Weg dazu führen, ChopChop zumindest ineffektiv zu machen.

Auch hier soll der gesamte Angriff wieder zusammenfassend in einem Attack Tree (Abbildung 4-12) dargestellt werden.

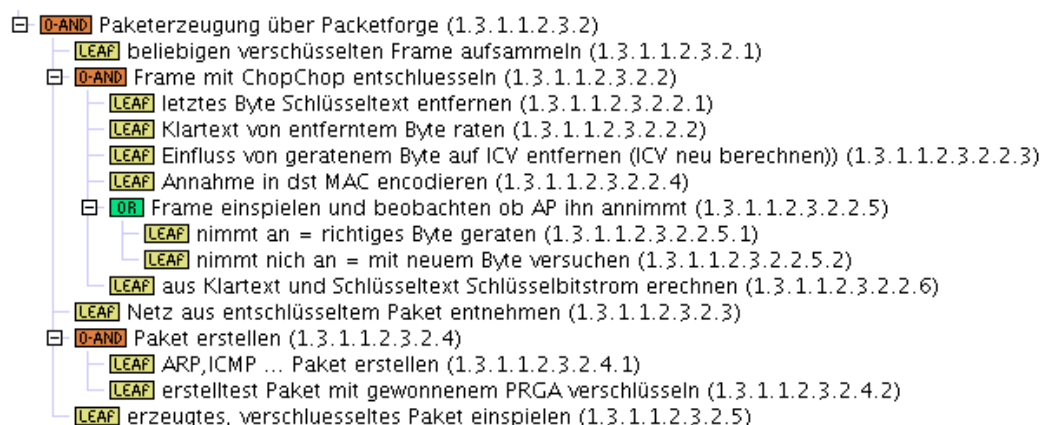


Abbildung 4-12: Attack Tree zum aktiven Angriff mit ChopChop und Packetforge

4.4.3 Aktiver Angriff mit Wesside

An dieser Stelle sollte nun eigentlich eine Versuchsreihe mit dem BSD-Werkzeug wesside durchgeführt werden. Leider war es im Rahmen der zur Verfügung stehenden Hardware nicht möglich, eine funktionstüchtige Umgebung zu konstruieren. Grund dafür war hauptsächlich, dass wesside nur mit auf Atheros basierenden WLAN Interfaces wirklich funktioniert. Leider wurden die zur Verfügung stehenden Atheros-Karten von vier verschiedenen BSD-Distributionen nicht oder nicht zufriedenstellend erkannt. Die geplanten Tests wurden daraufhin abgebrochen.

4.5 Man in the Middle

Als letzte Gruppe von Angriffen, die in dieser Arbeit betrachtet werden, sollen einige Laborversuche zu Man in the Middle-Angriffen durchgeführt werden. Dabei sei vorweg gestellt, dass MitM im eigentlichen Sinne, also das direkte Zwischenschalten in eine Kommunikationsverbindung, in einer Linux-Umgebung ohne weitere Werkzeuge möglich ist. Etwa über bridged Interfaces (brctl) oder den Betrieb eines Rogue AP als quasi Router, ähnlich der Funktionsweise von Aircrack-ng. Da es hier jedoch vordergründig um die Bewertung von Angriffswerkzeugen gehen soll, wird diese Methode außen vor gelassen.

4.5.1 Aircrack-ng – Hotspot Phishing

Aircrack-ng bedarf einiger Veränderungen, damit es auf der Linux-Distribution der Testumgebung fehlerfrei funktioniert, da ursprünglich eine Red Hat-Distribution vorausgesetzt wird. Die erste vorzunehmende Änderung war die Anpassung der Position von dhcpd und Webserver in der aircrack-ng.sh. Das normalerweise zum Zustellen der aufgezeichneten Login/Passwort-Paare benutzte sendmail wurde vollkommen deaktiviert und stattdessen die aircrack-ng.cgi dahingehend verändert, dass die aufgezeichneten Paare in eine Datei geschrieben werden. Anpassungen zu SSID, Netz, benutztem WLAN Interface usw. können in der aircrack-ng.cfg beliebig vorgenommen werden. Für die grundlegende Funktionalität ist aircrack-ng damit hinreichend konfiguriert. Weiterführende Änderungen, etwa Änderungen am Captive Portal oder

beliebige Routing-Vorschriften, können vorgenommen werden, sind aber nicht zwingend und obliegen Intention und Kreativität des Angreifers.

Mit den notwendigen Änderungen präsentiert Airnarft einem Opfer bei jeder Web-Anfrage die in Abbildung 4.13 dargestellte Webseite. Eine für Backtrack konfigurierte Version von Airsnarf befindet sich auf der beiliegenden CD-ROM. Eine Voraussetzung, die an die Hardware gestellt wird, ist das Unterstützen des Mode Master.



Please enter your username and password.

Username:

Password:

Abbildung 4-13: Captive Portal von Airsnarf

Diese Seite kann, wie schon erwähnt, beliebig gestaltet werden, um so Nutzer von legitimen Hotspots in die Irre zu führen.

Bislang konnten Schwierigkeit, Erfolgsaussicht und Gefährdung immer relativ losgelöst voneinander betrachtet werden. Hier ist es nun vollkommen anders. Alle drei Punkte, nach denen hier bewertet wird, sind eng miteinander verzahnt. Steigen Aufwand und Wissen, das ein Angreifer in die Vorbereitung steckt, so wird auch die Illusion, die dem Opfer geboten wird, perfekter, und die Erfolgsaussichten steigen. Damit geht auch direkt eine Verschiebung der Gefährdung für das Opfer einher. Ist eine reale Seite – anders als beim oben stehenden Bildschirmfoto – authentisch nachgebildet, wird wohl nur sehr versierten Nutzern direkt auffallen, dass es sich dabei um eine Fälschung handelt. Noch schwieriger wird es, wenn vollständige Web-Präsenzen gefälscht werden und zusätzlich andere Anfragen, die nicht mit gefälschten

Webseiten beantwortet werden, ins Internet weitergeleitet werden. Dafür ist jedoch Wissen über Perl-Net-DNS, mit welchem die DNS-Anfragen des Opfers behandelt werden, Linux Routing und iptables sowie ein zweites WLAN Interface von Nöten. Erfolgsaussichten und Schwierigkeitsgrad reichen also von niedrig bis hoch.

Einzig die Gefährdung für den Angreifer, entdeckt zu werden, bleibt relativ konstant. Grundlegend kann man, je nach Konfiguration dieser Rogue Access Points, einen Angriff auf drei verschiedenen Wegen erkennen. Täuscht ein Angreifer einfach nur die SSID vor, so wird mit dem in [NEU] beschriebenen Verfahren, trusted MAC-Listen, ein Angriff entdeckt. Werden vom Angreifer jedoch SSID und BSSID übernommen, aber eine anderer Kanal gewählt, so bleibt noch die Möglichkeit festzustellen, dass sich ein AP auf zwei Kanälen gleichzeitig befindet. Als letzte Möglichkeit kann der Angreifer zusätzlich noch den Kanal des legitimen APs übernehmen, dann verbleibt nur noch die Chance, das Vorhandensein eines Rogue AP über ständig schwankende Signalstärken zu erkennen. Diese letzte Möglichkeit kann jedoch schnell dazu führen, dass False Positives generiert werden, wenn Signalstärken aus natürlichen Gründen schwanken.

Abschließend auch hier wieder die Zusammenfassung dieses Angriffs, dargestellt als Attack Tree in Abbildung 4-13.

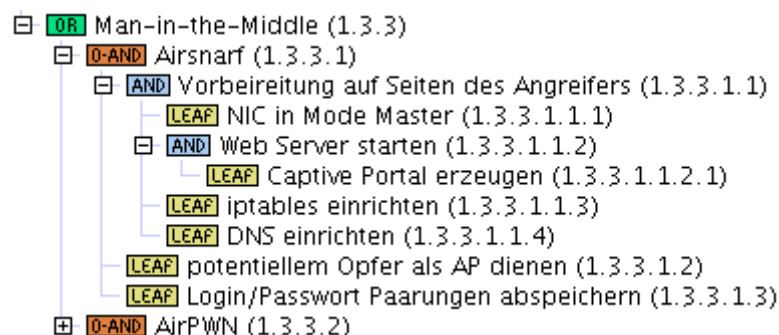


Abbildung 4.-13: Attack Tree zu Airsnarf

4.5.2 AirPWN – Antwort - Modifikation

AirPWN geht einen anderen Weg als Airsnarf, um Opfer zu beeinflussen. Hier wird nicht ein ganzer Access Point vorgetäuscht, sondern gezielt auf einzelne Anfragen mit einem eigenen, beliebigen Inhalt geantwortet. Auch bei diesem Werkzeug gilt wieder die Voraussetzung, dass das benutzte WLAN Interface Mode Monitor und Packet Injection unterstützt.

AirPWN kommt im Paket mit einigen vorgefertigten Konfigurationen, welche die Funktionsweise gut illustrieren. Im nachfolgenden Listing ist eine Beispielkonfiguration dargestellt:

```
begin airpwned_img
match ^GET [^ ]+\.?(i:jpg|jpeg|gif|png)
option reset
response content/airpwned_png
```

Damit wird bewirkt, dass jede Anfrage nach einem Bild zurückgesetzt und stattdessen mit `airpwned_png` geantwortet wird. So wird aus der bekannten Google-Startseite die in Abbildung 4-14 dargestellte:

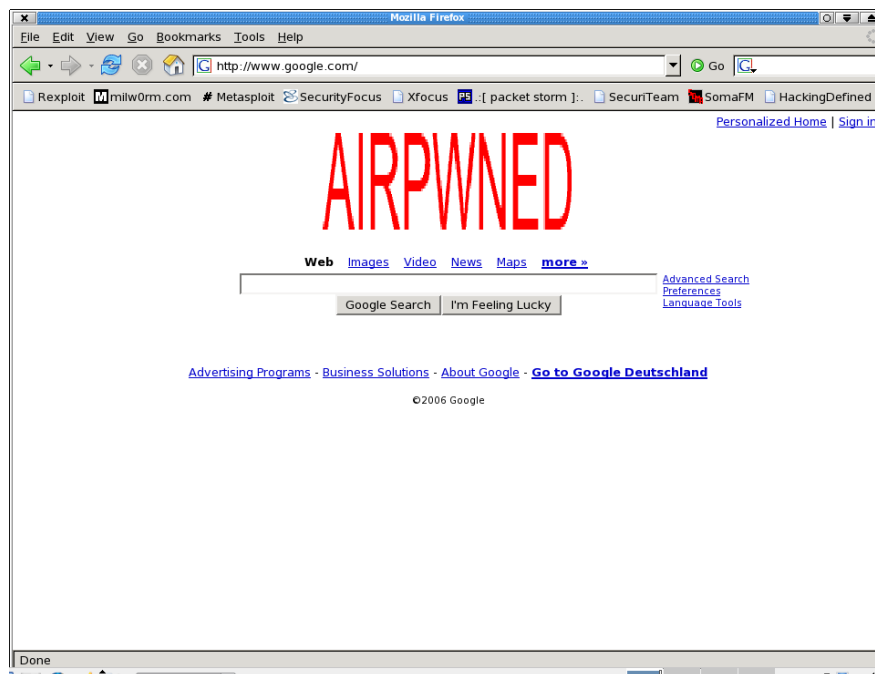


Abbildung 4-14: AirPWN mit Google Startseite

Diese Methode scheint nicht mehr als eine lustige Spielerei zu sein. Dieser erste Eindruck täuscht jedoch. Wie schon angeführt, lässt sich so beliebiger Inhalt einspielen, eben auch Scripte oder etwa virenverseuchte Daten. Vor diesem Hintergrund ist auch die Gefährdung für das Opfer zu bewerten. Im Testfall ist sie sicherlich nahe Null.

Diesen Angriff zu erkennen, ist möglich, aber nicht einfach. Im Anhang B sind einmal der Paketstrom einer normalen Anfrage an Google und einmal ein Paketstrom mit den injizierten Daten festgehalten. Es ist erkennbar, dass

beim verfälschten Datenstrom ein doppeltes Acknowledgement auftritt (erste ACK ist für den Angreifer bestimmt, das zweite wirklich für www.google.com). Das Problem dabei ist, dass es durchaus auch im normalen Betrieb vorkommen kann, dass einzelne Pakete doppelt empfangen werden. So wird das Erkennen dieses Angriffs wieder abhängig von einem Schwellenwert für doppelte Acknowledgements pro Zeit. Anderenfalls würden auch hier häufig False Positives generiert.

Die technische Erfolgswahrscheinlichkeit hängt maßgeblich von der Geschwindigkeit des Opfernnetzes ab, da bei diesem Angriff ausgenutzt wird, dass Anfragen an das Internet in der Regel deutlich mehr Antwortzeit benötigen als lokale Anfragen. Ist die Antwortzeit des Opfernnetzes sehr niedrig oder der PC des Angreifers aus anderen technischen Gründen nicht in der Lage, schnell zu antworten, schlägt dieser Angriff fehl. In der Testumgebung konnte dieser Punkt im Übrigen bestätigt werden. In einigen Fällen war es nicht möglich, das in Bild 4-14 dargestellte Ergebnis zu erzielen, da der Angreifer-Laptop zum einen zu langsam antwortete und zum anderen das Testnetz sehr geringe Antwortzeiten aufwies (kleiner 10ms).

Die Herausforderung bei der Durchführung dieses Angriffs liegt hauptsächlich in der notwendigen Protokollkenntnis (hier wurde ein HTTP-Beispiel benutzt, airPWN lässt sich prinzipiell aber auf alle TCP-Anfragen anwenden). Ein Angreifer kann nur Antworten fälschen, zu denen er die Formatierung von Anfrage und Antwort kennt. Die Bedienung des eigentlichen Werkzeugs airPWN mit seinen vorgefertigten Konfigurationen ist wohl auch noch der zweiten Klasse von Angreifern aus Kapitel drei möglich.

Abschließend auch hier noch einmal zusammenfassend der Attack Tree zu airPWN [Abbildung 4-15]:

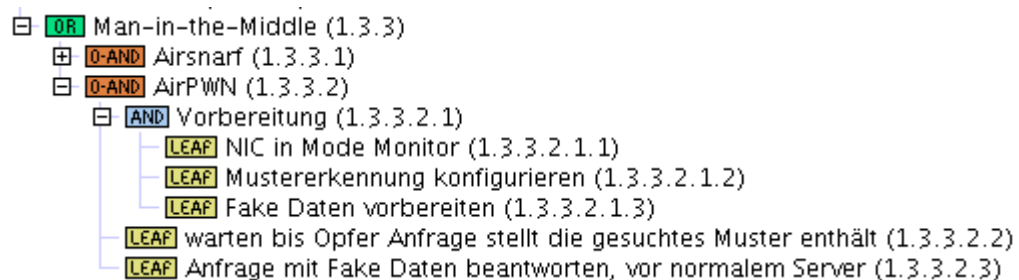


Abbildung 4-15: Attack Tree zu airPWN

4.6 Zusammenfassung

Als Zusammenfassung der Bewertungen der betrachteten Werkzeuge dient die nachfolgende Tabelle 4-2.

	Schwierigkeit	Erfolgsaussicht	Gefährdung
<i>passiver Angriff auf den Schlüssel</i>	abhängig vom Werkzeug, an sich eher gering	stark vom Verkehr abhängig	sehr niedrig für Angreifer, hoch für das Opfer
<i>aktiver Angriff mit ARP Replay</i>	etwas erhöht im Vgl. zu passivem	hoch, abgesehen von Ausnahmen	Es besteht die Möglichkeit, entdeckt zu werden.
<i>aktiver Angriff mit ChopChop und Packetforge-ng</i>	hoch, Vorwissen nötig	es gibt Hardware, die ChopChop blockiert, sonst hoch	Für Opfer extrem hoch. Für Angreifer wie ARP Replay.
<i>Airsnarf</i>	niedrig bis hoch	lokationsabhängig, mäßig bis hoch	stark abhängig vom Aufwand, den der Angreifer betreibt
<i>AirPWN</i>	mittel bis hoch	lokationsabhängig, mäßig	Stark abhängig vom Aufwand, den der Angreifer betreibt

4.7 Fazit

Alle hier aufgeführten Werkzeuge können, bei richtigem Einsatz, zu einer erheblichen Bedrohung für Nutzer werden. So schwierig oder unmöglich es teilweise erscheint, Gegenmaßnahmen einzuleiten, so einfach ist in der Praxis die Lösung. Sie lautet WPA.

Dass alle getesteten Werkzeuge nahezu durchgehend mit einem hohen Gefährdungspotential und großer Erfolgswahrscheinlichkeit bewertet wurden, liegt hauptsächlich daran, dass die Grundlagen der meisten Angriffe Jahre alt sind und damit sehr viel Zeit blieb, um sie immer weiter zu verfeinern. So wurde aus teilweise eingeschränkt nutzbaren Werkzeugen eine echte Bedrohung. An dieser Stelle sei noch einmal an das Beispiel Schlüsselangriff verwiesen, bei dem Tools früherer Generationen nahezu aussichtslos lange in Betrieb bleiben mussten, um ein Ergebnis zu erzielen, moderne Angriffe jedoch nur wenige Stunden benötigen.

Es erschien nicht sinnvoll, Werkzeuge, die zum Beispiel Brute Force-Angriffe auf Schlüssel implementieren, intensiv zu testen, da davon ausgegangen wurde, dass Angreifer sich den schnellsten, zumindest aber den einfachsten Weg auswählen. Ähnlich verhält es sich mit Angriffen, die auf sehr spezielle Infrastrukturen setzen, wie etwa Schwachstellen in proprietären Systemen.

Die im Verlauf ebenfalls angeführten Strategien zur Erkennung bzw. sogar zur Abwehr dieser Angriffe sind mit etwas Abstand zu betrachten. Es kann fast sicher davon ausgegangen werden, dass Betreiber – egal ob privat oder geschäftlich –, die immer noch auf WEP vertrauen, wohl wenig Sicherheitsbewusstsein haben und auch kein Interesse daran, ein Intrusion Detection System zu betreiben. Dennoch soll im folgenden, letzten Kapitel auf WirelessIDS eingegangen werden. Prinzipiell sind auch WPA-gesicherte Netze gegen einzelne Angriffe (etwa MitM) nicht gefeit.

5 Generierung von Policies für Bro wireless aus Attack Trees

Im letzten Kapitel soll untersucht werden, inwieweit es möglich und sinnvoll ist, automatisch Policies für ein IDS-System zu erstellen. Als Ausgangspunkt dienen dabei die bereits in den Kapiteln zwei und vier behandelten Attack Trees. Zunächst werden jedoch erst einmal die Grundanforderungen an ein IDS-System im Bereich wireless und die grundlegenden Funktionsweisen des Bro IDS-Systems [VP] vorgestellt.

5.1 Einführung wireless IDS

In kabelgebundenen Netzen sind IDS-Systeme schon seit Jahren üblich und ihre Aufgabengebiete klar abgesteckt. Für wireless Systeme stellen sich die Aufgaben, wenn auch nicht vollkommen neu, so doch stark verändert dar. Abgesehen von den aus kabelgebundenen Netzen bekannten Angriffen, die in der Regel ab Schicht 3 ISO/OSI-Modell aufwärts stattfinden, muss ein wireless IDS zusätzlich die Schichten 1 und 2 überwachen. Dass die Überwachung von höheren Schichten in einem Wireless LAN nicht trivial ist, wird offensichtlich, wenn man in Betracht zieht, dass auch ein Rogue AP durchaus mit Verschlüsselung arbeiten kann.

Hinzu kommt, dass, wie bereits in Kapitel eins beschrieben, die Abgrenzbarkeit bei Wireless LANs äußerst problematisch ist. Zum einen ist es nahezu unmöglich zu sagen, welche Fläche wirklich von einem AP abgedeckt wird, mit Rücksicht auf eventuell deutlich leistungstärkere Angreiferhardware. Aufgabe eines wireless IDS ist es freilich, den gesamten Abdeckungsbereich zu überprüfen. Zum anderen besteht in einem Wireless LAN nicht die Möglichkeit, an einem zentralen Punkt alle Kommunikationen direkt überwachen zu können. Der Grund dafür liegt in den 14 technisch möglichen Kanälen, auf den ein WLAN arbeiten kann. Reguläre WLAN-Hardware lässt es nur zu, sich auf einem Kanal zu einem Zeitpunkt zu befinden. Umso problematischer scheint es, zeitgleich 14 zu überwachen.

Da hier nur eine kurze Einführung in die Problematik gegeben werden soll, wird an dieser Stelle auf die Arbeiten [NEU] und [SR] verwiesen. Darin wird das Problem Wireless IDS sehr ausführlich behandelt. Dort sind ebenfalls detaillierte Anforderungsanalysen an ein wireless IDS aufgeführt.

5.2 Grundlagen Bro IDS

Es ist lediglich das Anliegen dieses Abschnitts, eine kurze Einführung in die Struktur und grundlegende Funktionsweise von Bro IDS zu geben. Detaillierte Analysen von Bro IDS sind [NEU] bzw. natürlich auch den offiziellen Bro-Referenzen [PT] zu entnehmen.

In Bro wird besonderer Wert auf eine klare Trennung zwischen Ereignisgenerierung und der standortspezifischen Sicherheitspolitik gelegt. Dies verdeutlicht die strukturelle Zweiteilung in Policy Layer und Bro Core. Letzterer ist seinerseits in drei Subsysteme (Policy Script Interpreter, Event Engine, Network Analysis) unterteilt. An dieser Stelle soll es genügen zu sagen, dass der Bro Core von der Netzwerkanalyse über die Erzeugung von Ereignissen bis hin zur Abbildung von Policies auf Bro Klassen alles gewährleistet.

Aufgabe des Policy Layers ist es, nachdem im Bro Core Ereignisse erzeugt wurden, standortspezifische Behandlungen dieser Ereignisse zu definieren. Dies geschieht mit so genannten Policy Scripten, die in der Bro eigenen Sprache geschrieben werden. Hauptbestandteil der Policy Scripte sind Event Handler, sie beschreiben, wie genau auf ein spezifisches Ereignis seitens des IDS-Systems zu reagieren ist. Mit Policy Scripten erfolgt die eigentliche Anpassung an das Sicherheitsbedürfnis eines Standortes.

Eine vollständige Referenz der Bro-Scriptsprache ist in [PT] gegeben. Im folgenden Abschnitt dieser Arbeit soll daher lediglich auf die in [NEU] vorgenommenen Erweiterungen eingegangen werden. Diese Erweiterungen schließen im Wesentlichen alle für die Erkennung von Angriffen auf Wireless LAN nötigen Mechanismen ein.

5.3 Analyse BRO wireless

In [NEU] wurde der Bro Core dahingehend erweitert, dass nun die Erkennung von Wireless Frames und deren Zerlegung in einzelnen Felder möglich ist. Wie dies im Detail geschehen ist, soll hier nicht Gegenstand sein. Es soll dabei belassen werden zu sagen, dass Bro Wireless prinzipiell in der Lage ist, auf der Policy-Ebene Ereignisse, die mit den drei in 802.11 definierten Frametypen (Management, Control und Data) in Zusammenhang stehen, zu behandeln. Dabei wird in Bro Wireless davon ausgegangen, dass sich die meisten Angriffe an Hand der Frame Header bzw. anhand des

Management Frame Body erkennen lassen. Deshalb sind in [NEU] folgende Ereignisse definiert, die sich ausnahmslos auf 802.11 Management Frames beziehen.

```
event wlan_assoc_req
event wlan_assoc_resp
event wlan_reassoc_req
event wlan_reassoc_resp
event wlan_probe_req
event wlan_probe_resp
event wlan_beacon
event wlan_disassoc
event wlan_auth
event wlan_deauth
event wlan_unknown_type
```

Damit wird deutlich, dass sich die in Kapitel vier vorgeschlagene Methode zur Erkennung von Angriffen mit ChopChop zurzeit nicht direkt realisieren lässt. Da für die Erkennung von ChopChop in jedem Fall Data Frames betrachtet werden müssen, was in der aktuellen Version von Bro Wireless nicht möglich ist..

Dennoch wird im folgenden Abschnitt ChopChop als Beispiel gewählt, da sich anhand dieses Angriffs die Schwierigkeiten bei der automatischen Erzeugung von Policies aus AT gut darstellen lassen. Es wird dafür einfach davon ausgegangen, dass von Bro ein Ereignis erzeugt wird, wann immer ein Data Frame eintrifft, welches die benötigten Informationen zugänglich macht. In einer Implementierung würde dies analog zur Erzeugung von Ereignissen bei Auftreten eines Management Frames geschehen. Bro selbst wird im Rahmen dieser Arbeit jedoch nicht erweitert.

5.4 Beispiel Policy zur Erkennung von Angriffen mit ChopChop

In den Kapiteln drei und vier wurde bereits auf die Funktionsweise und mögliche Einsatzgebiete von ChopChop eingegangen. Eine grundlegende Strategie wurde ebenfalls schon vorgestellt und soll nun vertieft werden. Dafür wird eine Policy für Bro IDS entwickelt, welche in der Lage ist, einen Angriff mit ChopChop zu erkennen. Zunächst aber noch einmal der AT zu ChopChop, losgelöst von dem in Kapitel vier betrachteten Szenario [Abbildung 5-1]. Die Repräsentation des ChopChop Attack Trees als graphml-Datei ist im Anhang D zu finden.

Wenn man die Abbildung und das Wissen über den Angriff aus den vergangenen Kapiteln in Betracht zieht, wird klar, dass nur die Punkte 1.5 und 1.6 aus Sicht eines IDS-Systems wahrnehmbar sind. Alle anderen Schritte erfolgen ausschließlich beim Angreifer und hinterlassen vorerst keine Spuren im Opfernetz.

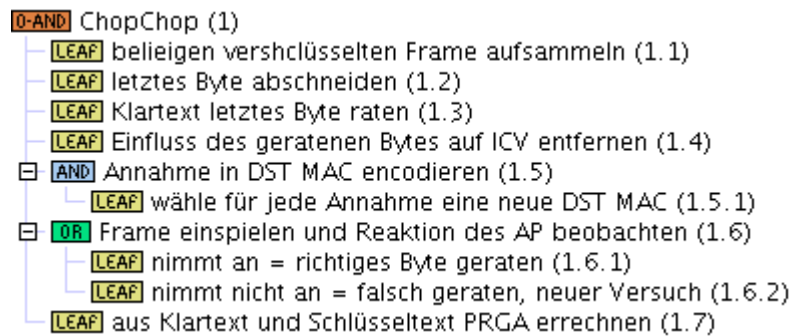


Abbildung 5-1: ChopChop einzeln betrachtet

Die beiden Punkte 1.5 und 1.6 liefern den Anhaltspunkt, wie ein ChopChop-Angriff erkannt werden kann, nämlich über die sehr hohe Anzahl pro Sekunde auftretender Ziel-MAC-Adressen. Es bietet sich also an, alle auftretenden Ziel-MAC-Adressen in einer Menge zu speichern. In der Bro Sprache ausgedrückt bedeutet das.

```
global mac_set: set[mac] & create_expire = 1sec;
```

So wird eine Menge von MAC-Adressen erzeugt und gleichzeitig gewährleistet, dass immer, wenn eine neue MAC hinzugefügt wird, diese automatisch nach einer Sekunde wieder aus der Menge entfernt wird.

An dieser Stelle setzt nun die im vergangenen Abschnitt angesprochene Annahme zur Generierung von Ereignissen beim Auftreten von Data Frames ein. Es wird davon ausgegangen, dass die im Folgenden definierte Datenstruktur und der zugehörige Event erzeugt werden.

```
type ieee80211_data_frame_hdr: record {
  fc: ieee80211_frame_control;
  duration: count;
  da: mac;
  sa: mac;
  bssid: mac;
  fragN: count;
  seqN: count;
};
```

```
global data_frame: event(c: ieee80211_data_frame_hdr);
```

Mit diesen Voraussetzungen kann nun der eigentliche Event Handler erstellt werden. Als Schwellenwert für die Anzahl pro Sekunde erlaubter neuer MACs wurde 50 gewählt, um selbst in Netzen mit vielen Teilnehmern keine False Positives zu erzeugen. ChopChop-Angriffe können mit diesem Schwellenwert recht zuverlässig erkannt werden. Im Laborversuch wurden rund 170 neue MACs pro Sekunde registriert. Die Wahl von Schwellenwerten zur Erkennung von Angriffen ist nicht unproblematisch. In der Regel ist es so, dass der Grad zwischen normalem und aggressivem Verhalten sehr schmal ist. Wird der Wert zu hoch gewählt, werden Angriffe übersehen, wird er zu niedrig gewählt, werden Fehlalarme erzeugt. Auf diesen Fakt wird auch schon in [NEU] aufmerksam gemacht.

```
global new_mac_count: count = 0;
global mac_count_limit: count = 50;
global check_interval: interval = 1sec;

event data_frame (c: ieee_data_frame_hdr) {

    local current_mac: mac;
    current_mac = c$sa;

    if ( current_mac !in mac_set ) {
        add mac_set [current_mac];
        new_mac_count = new_mac_count + 1;
    }
}

event lf_chopchop () {
    if (new_mac_count > mac_count_limit) {
        print "Anzahl an neuen MACs pro Sekunde überschritten. Möglicher ChopChop Einsatz "
    }
    new_mac_count = 0;
    schedule check_interval ( lf_chopchop () );
}

event bro_init () {
    schedule check_interval ( lf_chopchop () );
}
```

Es ist zu erkennen, dass bei jedem Auftreten einer neuen MAC innerhalb des gewählten Intervalls diese MAC der Menge der bekannten hinzugefügt und der Zähler um eins inkrementiert wird. Mit Hilfe von `schedule` wird erreicht,

dass jeweils einmal im Intervall geprüft wird, ob die Anzahl der neuen MACs kleiner als der Schwellwert ist. Sofern der Schwellwert überschritten wurde, wird eine Meldung erzeugt. Anschließend wird der Zähler zurückgesetzt.

Damit ist die Beispielpolicy zu ChopChop vollständig. Die Frage, die sich nun stellt, ist: Gibt es einen Weg, diese automatisch aus dem oben stehenden AT abzuleiten? Mit dieser Frage beschäftigt sich der folgende Abschnitt.

5.5 Möglichkeiten und Grenzen der automatischen Generierung von Policies aus Attack Trees

Wie bereits angedeutet, hat die Gewinnung von Policies aus AT ein generelles Problem. In einem AT sind in der Regel eine Reihe von Informationen und Schritten gegeben, die aus verschiedenen Gründen für ein IDS uninteressant sind. So sind im oben gewählten Beispiel [Abbildung 5-1] nur die Punkte 1.5 und 1.6 von Interesse für die eigentliche Policy. Alle anderen sind aus Sicht des AT jedoch nicht überflüssig, da sie für den menschlichen Betrachter zu einem vollständig modellierten Szenario dazu gehören. Es muss also zunächst ein zusätzliches Attribut für die Knoten des AT eingeführt werden, welches sie als relevant oder irrelevant für die Generierung einer Policy kennzeichnet. Diese Erweiterung würde – ähnlich der in Kapitel zwei – von AT zu EAT um den TTL-Wert eines Knotens beschrieben ablaufen.

Nachdem nun relevante Knoten erkannt werden können, ist die Gewinnung von Informationen zu einem speziellen Angriff aus diesen der nächste Schritt. Es zeigt sich jedoch schnell, dass keine allgemeine Herangehensweise beschrieben werden kann, wie dies zu geschehen hat. Zum einen ist, selbst nachdem alle irrelevanten Knoten herausgefiltert werden können, eine Interpretation des Zusammenspiels einzelner Teilbäume nicht ausreichend möglich. Im Beispiel wird zwar klar, dass die Knoten 1.5 und 1.6 in einer O-AND Beziehung stehen, es ist aber in keiner Weise ersichtlich, wie sich Veränderungen an 1.5 auf 1.6 auswirken. Dieses Problem entsteht nicht zuletzt, weil mit dem AT keinerlei verwertbare Informationen zur Verfügung gestellt werden, wie einzelne Attribute zu interpretieren sind.

Zum anderen, besteht in den in dieser Arbeit verwendeten EAT überhaupt keine Möglichkeit, Attribute nur für einzelne Knoten zu vergeben. Wie in Kapitel zwei besprochen, werden, mit Ausnahme von TTL und PC, Attribute

in AT/EAT immer nur für Blattknoten vergeben und propagieren sich dann selbständig in Richtung Wurzel. Sobald jedoch ein Attribut für ein Blatt eingeführt wird, muss dieses Attribut automatisch auch für alle anderen Knoten eine Bedeutung haben.

Für das gewählte Beispiel heißt das letztendlich, dass weder die Existenz noch die Belegung für `check_interval` und `mac_count_limit` automatisch abgeleitet werden können.

Diese kurze Betrachtung lässt die Aussage zu, dass AT/EAT zwar die Syntax eines Angriffs beschreiben, nicht aber seine Semantik. Es ist also möglich, die Struktur eines Angriffs zu erkennen. Das Entwickeln der eigentlichen Strategie – also festzustellen, wie anhand von gefundenen Indizien auf das Vorhandensein des eigentlichen Angriffs geschlossen werden kann – obliegt der menschlichen Beobachtungsgabe und Kreativität. So kann man zu dem Schluss kommen, dass zumindest für elementare Angriffe (in dieser Arbeit wären Elementare Angriffe etwa ChopChop, ARP-Replay etc.) keine automatische Policy-Erzeugung möglich ist. Leider ist genau dies aber das primäre Ziel. Denn ein großer Teil der für WLAN relevanten Angriffe besteht nur aus einem dieser elementaren Angriffe. So wird das einzige Indiz, was ein IDS für einen Angriff auf den geheimen Schlüssel finden kann, eine ARP Replay (oder andere Replay-Attacke) bleiben. Dies bedeutet, dass eine automatisch generierte Policy zur Erkennung von Angriffen auf den geheimen Schlüssel genau ein Statement, nämlich den Aufruf der elementaren Policy zur Erkennung von ARP Replay enthält.

Was sich also nicht automatisch generieren lässt, ist eine Policy für einen elementaren Angriff. Die Frage, die sich stellt: Ist dies lediglich ein Problem, welches aus der Verwendung von Attack Trees entsteht, und eignen sich andere Darstellungsformen eventuell dazu? Diese Frage lässt sich nicht eindeutig beantworten. Es gibt zwar Veröffentlichungen [HAG], die sich ebenfalls mit der automatischen Generierung von Policies beschäftigen und dazu eine andere Darstellungsform (Attack Graphs) wählen. Aber auch dort wird davon ausgegangen, dass elementare Angriffe prinzipiell erst einmal erkannt werden. Die automatische Generierung beschränkt sich dann lediglich auf das automatische Erzeugen von Angriffsszenarien, also der Verknüpfung von elementaren Angriffen.

Zum jetzigen Zeitpunkt ist nicht bekannt, ob eine Darstellungsform für Angriffe existiert, die es zulässt, eine zuverlässig funktionierende Policy abzuleiten und dabei nicht komplexer als die Policy selbst ist.

Für diese Arbeit kann man abschließend sagen, dass eine automatische Generierung einer vollständigen Policy für Bro IDS aus AT mit den derzeitigen Voraussetzungen unmöglich erscheint. Möglich ist einzig die Erkennung und Generierung von Angriffsszenarien. In der Tat wird dies über das PC-Attribut des EAT bereits automatisch gewährleistet. Eine Implementierung könnte dies als Grundlage nehmen

6 Zusammenfassung

Es konnte anhand einiger Beispiele gut illustriert werden, wie real die Bedrohung durch Angriffswerkzeuge im Wireless LAN wirklich ist. Dafür wurden zunächst die Grundlagen gelegt, um im Anschluss auf konkrete Angriffsstrategien eingehen zu können. Das wiederum schuf die Möglichkeit, verschiedene Werkzeuge einem Laborversuch zu unterziehen, was zu Aussagen über Schwierigkeitsgrad, Erfolgswahrscheinlichkeit und Gefährdungspotential führte. Die Struktur der einzelnen Angriffe wurde mit dem zuvor um das Konzept der Enhanced Attack Trees erweiterten Werkzeug [OPEL/2] dargestellt.

Im abschließenden Teil dieser Arbeit wurden Möglichkeiten zur automatischen Generierung von Policies für Bro IDS aus den in vorherigen Abschnitten entstandenen Attack Trees untersucht. Es konnten jedoch nur Anregungen und Ausblicke auf Implementierungen gegeben werden.

Letztendlich werden Betrachtungen wie diese auch weiterhin Gültigkeit behalten. Denn obwohl von Seiten der Industrie in den letzten Jahren beachtliche Fortschritte im Bereich WLAN-Sicherheit erzielt wurden, setzen sich diese Errungenschaften nur sehr langsam beim Verbraucher durch. Es ist zu erwarten, dass sich dieses Verhalten auch bei kommenden Security-Standards, wie etwa IEEE 802.11w Protected Management Frames, fortsetzen wird. Unsichere Implementierungen werden sicherlich noch geraume Zeit das Bild von 802.11 bestimmen.

Anhang A

Auf Grund, des Umpfangs nur auf CD-ROM

Anhang B

Normale Anfrage.

```
15:46:24.540369 IP 192.168.0.2.32772 > 192.168.0.1.53: 16126+ AAAA? www.google.com. (32)
15:46:24.546841 IP 192.168.0.1.53 > 192.168.0.2.32772: 16126 1/0/0 CNAME www.l.google.com. (52)
15:46:24.546922 IP 192.168.0.2.32772 > 192.168.0.1.53: 15596+ A? www.google.com. (32)
15:46:24.553395 IP 192.168.0.1.53 > 192.168.0.2.32772: 15596 4/0/0 CNAME www.l.google.com., A
72.14.221.99, A 72.14.221.104, A 72.14.221.147 (100)
15:46:24.556080 IP 192.168.0.2.37681 > 72.14.221.99.80: S 3583366865:3583366865(0) win 5840 <mss
1460,sackOK,timestamp 2226857 0,nop,wscale 2>
15:46:24.578076 IP 72.14.221.99.80 > 192.168.0.2.37681: S 3023865672:3023865672(0) ack 3583366866
win 8190 <mss 1460>
15:46:24.578110 IP 192.168.0.2.37681 > 72.14.221.99.80: . ack 1 win 5840
15:46:24.578463 IP 192.168.0.2.37681 > 72.14.221.99.80: P 1:491(490) ack 1 win 5840
15:46:24.601097 IP 72.14.221.99.80 > 192.168.0.2.37681: . ack 491 win 6432
15:46:24.611532 IP 72.14.221.99.80 > 192.168.0.2.37681: . 1:1431(1430) ack 491 win 6432
15:46:24.611549 IP 192.168.0.2.37681 > 72.14.221.99.80: . ack 1431 win 8580
15:46:24.612314 IP 72.14.221.99.80 > 192.168.0.2.37681: P 1431:2164(733) ack 491 win 6432
15:46:24.612323 IP 192.168.0.2.37681 > 72.14.221.99.80: . ack 2164 win 11440
15:46:24.635822 IP 192.168.0.2.37681 > 72.14.221.99.80: P 491:987(496) ack 2164 win 11440
15:46:24.642344 IP 192.168.0.2.37682 > 72.14.221.99.80: S 3586268992:3586268992(0) win 5840 <mss
1460,sackOK,timestamp 2226878 0,nop,wscale 2>
15:46:24.663856 IP 72.14.221.99.80 > 192.168.0.2.37681: . 2164:3594(1430) ack 987 win 7504
15:46:24.665504 IP 72.14.221.99.80 > 192.168.0.2.37681: . 3594:5024(1430) ack 987 win 7504
15:46:24.665528 IP 192.168.0.2.37681 > 72.14.221.99.80: . ack 5024 win 17160
15:46:24.667349 IP 72.14.221.99.80 > 192.168.0.2.37681: . 5024:6454(1430) ack 987 win 7504
15:46:24.667472 IP 72.14.221.99.80 > 192.168.0.2.37682: S 2154524617:2154524617(0) ack 3586268993
win 8190 <mss 1460>
15:46:24.667490 IP 192.168.0.2.37682 > 72.14.221.99.80: . ack 1 win 5840
15:46:24.669458 IP 192.168.0.2.37682 > 72.14.221.99.80: P 1:483(482) ack 1 win 5840
15:46:24.690696 IP 72.14.221.99.80 > 192.168.0.2.37681: . 6454:7884(1430) ack 987 win 7504
15:46:24.690744 IP 192.168.0.2.37681 > 72.14.221.99.80: . ack 7884 win 22880
15:46:24.692442 IP 72.14.221.99.80 > 192.168.0.2.37681: P 7884:9314(1430) ack 987 win 7504
15:46:24.693965 IP 72.14.221.99.80 > 192.168.0.2.37681: . 9314:10744(1430) ack 987 win 7504
15:46:24.693990 IP 192.168.0.2.37681 > 72.14.221.99.80: . ack 10744 win 28600
15:46:24.694195 IP 72.14.221.99.80 > 192.168.0.2.37682: . ack 483 win 6432
15:46:24.695114 IP 72.14.221.99.80 > 192.168.0.2.37682: P 1:288(287) ack 483 win 6432
15:46:24.695123 IP 192.168.0.2.37682 > 72.14.221.99.80: . ack 288 win 6432
15:46:24.716112 IP 72.14.221.99.80 > 192.168.0.2.37681: P 10744:10928(184) ack 987 win 7504
15:46:24.755575 IP 192.168.0.2.37681 > 72.14.221.99.80: . ack 10928 win 31460
```

Anfrage bei airPWN-Einsatz

```
15:38:46.232205 IP 192.168.0.2.32772 > 192.168.0.1.53: 34816+ AAAA? www.google.com. (32)
15:38:46.270944 IP 192.168.0.1.53 > 192.168.0.2.32772: 34816 1/0/0 CNAME www.l.google.com. (52)
15:38:46.275941 IP 192.168.0.2.32772 > 192.168.0.1.53: 1058+ A? www.google.com. (32)
15:38:46.281759 IP 192.168.0.1.53 > 192.168.0.2.32772: 1058 4/0/0 CNAME www.l.google.com., A
72.14.221.104, A 72.14.221.147, A 72.14.221.99 (100)
15:38:46.284420 IP 192.168.0.2.54162 > 72.14.221.104.80: S 3105349021:3105349021(0) win 5840 <mss
1460,sackOK,timestamp 2112296 0,nop,wscale 2>
15:38:46.306096 IP 72.14.221.104.80 > 192.168.0.2.54162: S 3758020498:3758020498(0) ack 3105349022
win 8190 <mss 1460>
15:38:46.306147 IP 192.168.0.2.54162 > 72.14.221.104.80: . ack 1 win 5840
15:38:46.320064 IP 192.168.0.2.54162 > 72.14.221.104.80: P 1:517(516) ack 1 win 5840
15:38:46.326364 IP 72.14.221.104.80 > 192.168.0.2.54162: P 1:222(221) ack 517 win 65535
15:38:46.326403 IP 192.168.0.2.54162 > 72.14.221.104.80: . ack 222 win 6432
15:38:46.342386 IP 72.14.221.104.80 > 192.168.0.2.54162: . ack 517 win 6528
15:38:46.342419 IP 192.168.0.2.54162 > 72.14.221.104.80: . ack 222 win 6432
15:38:46.372193 IP 72.14.221.104.80 > 192.168.0.2.54162: . 1:1431(1430) ack 517 win 6528
15:38:46.372234 IP 192.168.0.2.54162 > 72.14.221.104.80: . ack 1431 win 8580
15:38:46.372988 IP 72.14.221.104.80 > 192.168.0.2.54162: P 1431:2166(735) ack 517 win 6528
15:38:46.372998 IP 192.168.0.2.54162 > 72.14.221.104.80: . ack 2166 win 11440
```

Anhang C

Nur auf CD-Rom.

Anhang D

graphml-Datei zur ChopChop Abbildung.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<graphml>
  <key id="0" for="node" attr.name="type" attr.type="String" />
  <key id="1" for="node" attr.name="name" attr.type="String" />
  <key id="2" for="node" attr.name="attacktype" attr.type="String">
    <default>attack</default>
  </key>
  <key id="3" for="node" attr.name="ttl" attr.type="String">
    <default>none</default>
  </key>
  <key id="4" for="node" attr.name="pc" attr.type="long" />
  <graph id="0" edgedefault="directed">
    <node id="0">
      <data key="0">OAND</data>
```

```

    <data key="1">ChopChop</data>
    <data key="3">none</data>
    <data key="4">1</data>
</node>
<node id="1">
    <data key="0">LEAF</data>
    <data key="1">belieigen verschlüsselten Frame aufsammeln</data>
    <data key="3">none</data>
    <data key="4">0</data>
</node>
<edge source="0" target="1" />
<node id="2">
    <data key="0">LEAF</data>
    <data key="1">letztes Byte abschneiden</data>
    <data key="3">none</data>
    <data key="4">0</data>
</node>
<edge source="0" target="2" />
<node id="3">
    <data key="0">LEAF</data>
    <data key="1">Klartext letztes Byte raten</data>
    <data key="3">none</data>
    <data key="4">0</data>
</node>
<edge source="0" target="3" />
<node id="4">
    <data key="0">LEAF</data>
    <data key="1">Einfluss des geratenen Bytes auf ICV entfernen</data>
    <data key="3">none</data>
    <data key="4">0</data>
</node>
<edge source="0" target="4" />
<node id="5">
    <data key="0">AND</data>
    <data key="1">Annahme in DST MAC encodieren</data>
    <data key="3">none</data>
    <data key="4">1/7</data>
</node>
<edge source="0" target="5" />
<node id="6">
    <data key="0">OR</data>
    <data key="1">Frame einspielen und Reaktion des AP beobachten</data>
    <data key="3">none</data>
    <data key="4">1/7</data>
</node>
<edge source="0" target="6" />
<node id="7">
    <data key="0">LEAF</data>

```

```

    <data key="1">nimmt an = richtiges Byte geraten</data>
    <data key="3">none</data>
    <data key="4">0</data>
  </node>
  <edge source="6" target="7" />
  <node id="8">
    <data key="0">LEAF</data>
    <data key="1">nimmt nicht an = falsch geraten, neuer Versuch</data>
    <data key="3">none</data>
    <data key="4">0</data>
  </node>
  <edge source="6" target="8" />
  <node id="9">
    <data key="0">LEAF</data>
    <data key="1">aus Klartext und Schlüsseltext PRGA errechnen</data>
    <data key="3">none</data>
    <data key="4">0</data>
  </node>
  <edge source="0" target="9" />
  <node id="10">
    <data key="0">LEAF</data>
    <data key="1">wähle für jede Annahme eine neue DST MAC</data>
    <data key="3">none</data>
    <data key="4">0</data>
  </node>
  <edge source="5" target="10" />
</graph>
</graphml>

```


Literaturliste

- [LE 06] Guillaume Lehenbre: *Wi-Fi-Sicherheit – WEP, WPA und WPA2*, Nr. 1/2006 hakin9, 2006.
- [EdAr 04] Jon Edney, William A. Arbaugh: *Real 802.11 Security, Wi-Fi Protected Access and 802.11i*, Addison Wesley 2004.
- [WiFiWR] Justin Rigling, Ben Corrado, Brandon Schamer, Andy Meng: New World Record for Unamplified; <http://www.wifiworldrecord.com/>, 2005.
- [DÖR] Stefan Dörhöfer: *Empirische zur WLAN Sicherheit mittels Wardriving*; Diplomarbeit RWTH Aachen, 2006
- [NEU] Rene Neumerkel: *Entwicklung eines Angriffssensors für Wireless LAN*, Diplomarbeit Technische Universität Dresden; 2005.
- [ADGKK] Farooq Anjum, Subir Das, Praveen Gopalakrishnan, Latha Kant and Byungsuk Kim: *Security in an Insecure WLAN Network*, Telcordia Technologies.
- [AD] AirDefense Inc.: *Best Practices for Rogue Wireless LAN Detection*. http://www.airdefense.net/whitepapers/roguewatch_request2.php4, 2003.
- [BHL] Andrea Bittau, Mark Handley, Joshua Lackey: *The Final Nail in WEP's Coffin*, <http://tapir.cs.ucl.ac.uk/bittau-wep.pdf>, 2005
- [BI] Andrea Bittau: *The Fragmentation Attack in Practice*: www.toorcon.org/2005/slides/abittau/, 2005
- [WA] David A. Wagner: my-rc4-weak-keys; <http://www.cs.berkeley.edu/~daw/my-posts/my-rc4-weak-keys>, 1995
- [WAL] Jesse R. Walker: *IEEE P802.11 Wireless LANs Unsafe at any key size; An analysis of the WEP*; Intel Corporation, 2000
- [AR] William A. Arbaugh: *An Inductive Chosen Plaintext Attack Against WEP/WEP2.*, University of Maryland, 2001
- [BGW] Nikita Borisov, Ian Goldberg, David Wagner: *Intercepting Mobile Communications: The Insecurity of 802.11*, Seventh Annual International Conference on Mobile Computing And Networking, 2001

- [FMS] Scott Fluhrer, Itsik Mantin, and Adi Shamir: *Weaknesses in the Key Scheduling Algorithm of RC4*, Eighth Annual Workshop on Selected Areas in Cryptography, 2001
- [SG] The Shmoo Group: AirSnort; <http://airsnort.shmoo.com/>, 2001-2006
- [SG 2] The Shmoo Group: AirSnort; <http://airsnarf.shmoo.com/>,
- [HU] David Hulton: *Practical Exploitation of RC4 Weaknesses in WEP Environments*, www.dachb0den.com, 2002
- [DE] Christophe Devine später Thomas d'Otreppe: www.aircrack-ng.org
- [SA] José Ignacio Sánchez Martín: <http://weplab.sourceforge.net/>,
- [IEEE 1] Institute of Electrical and Electronics Engineers: *802.1X - Port Based Network Access Control*, 2001
- [IEEE 2] Institute of Electrical and Electronics Engineers: *802.11i IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 6: Medium Access Control (MAC) Security Enhancements*, 2004
- [IEEE 3] Institute of Electrical and Electronics Engineers: *802.11 Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, 1999
- [RSA] RSA Laboratories: *RSA Security Response to Weaknesses in Key Scheduling Algorithm of RC4*
- [RSA 2] RSA Laboratories: *PKCS #5: Password-Based Cryptography Standard*, 1999
- [PF] Andreas Pfitzmann: *Sicherheit in Rechnernetzen: Mehrseitige Sicherheit in verteilten und durch verteilte Systeme*, Technische Universität Dresden, 2000
- [C't 17/2006] C't: *T-One: C't Ausgabe 17 2006 S.140f*
- [NuReg] US Nuclear Regulatory Commission: *Fault Tree Handbok*, 1981

- [SCH] Bruce Schneier: *Attack Trees*, Dr. Dobb's Journal, 1999
- [StSch] Jan Steffan, Markus Schumacher: *Collaborative Attack Modeling*, 2002
- [MEL] Andrew P. Moore, Robert J. Ellison, Richard C. Linger: *Attack Modeling for Information Security and Survivability*, 2001
- [Opel] Alexander Opel: *Design and Implementation of a Support Tool for Attack Trees*, Otto-von-Guericke University, 2005
- [Opel/2] Alexander Opel: *Support Tool for Attack Trees*, <http://www.toengel.net/internship/data/AttackTree.jar>, 2005
- [CY] Seyit Ahmet Çamtepe, Bülent Yener: *A Formal Method for Attack Modeling and Detection*, 2001
- [LF] LinkFerret Network Monitor: www.baseband.com
- [TM] Tamosoft CommView for WiFi: www.tamos.com
- [VGM] Andrew A. Vladimirov, Konstantin V. Gavrilenko, Andrei A. Mikhailovsky: *WI-FOO The Secrets of Wireless Hacking*, Addison Wesley 2004
- [NS] Netstumbler: www.netstumbler.com
- [NS 2] Netstumber Forum: MacOS board , www.netstumbler.org/showpost.php?p=89036&postcount=11 , 2004
- [KO] Netstumbler Forum: Korek: chopchop (Experimental WEP attacks): www.netstumbler.org/showthread.php?t=12489, 2004
- [DB] Dachboden Labs: www.dachb0den.com
- [KI] Kismet: <http://www.kismetwireless.net/>
- [LAVA] Wep_tools: <http://www.lava.net/~newsham/wlan/>
- [WeA] WepAttack: wepattack.sourceforge.net/
- [SIR] Adam Stubblefield, John Ioannidis, Aviel D. Rubin: *Using the Fluhrer, Mantin, and Shamir Attack to Break WEP*, 2001
- [OS] Michael Ossmann: *WEP: Dead Again, Part 1 & 2*, <http://www.securityfocus.com/infocus/1814>, 2004
- [COWi] The Church of WiFi: www.churchofwifi.org

- [HE] www.heise.de: *Funknetz-Absicherung WPA offenbart Schwächen*, www.heise.de/newsticker/meldung/print/32542, 2002
- [FAP] Black Alchemy: fakeAP.pl, www.blackalchemy.to/project/fakeap/
- [RFA] Raw Wireless Tools: <http://rfakeap.tuxfamily.org/>
- [WiFoo] Recon and Attack tools: <http://www.wi-foo.com/ViewPageefe4.html?siteNodeId=55&languageId=1&contentId=-1>
- [WVE] Wireless Vulnerabilities & Exploits: www.wirelessve.org
- [LP] Lisa Phifer: *Wireless attacks, A to Z*, searchsecurity.techtarget.com/generic/0,295582,sid14_gci1167611,00.html, 2006
- [RME] Back Track Network Security Suite: www.remote-exploit.org
- [MM] Mac Makeup: www.gorlani.com/publicprj/macmakeup/macmakeup.asp
- [JW] Joshua Wright: *Detecting Wireless LAN MAC Address Spoofing*, 2003
- [HAG] L. Hamza, K. Adi, K. El Guemhioui: *Automatic generation of attack scenarios for intrusion detection systems*, IEEE, 2006
- [PAX] Vern Paxson: *Bro: A System for Detecting Network Intruders in Real-Time*, 1999
- [SR] Sandro Reichert: *Entwicklung eines verteilten Intrusion Detection Systems für Wireless LAN*, Belegarbeit Technische Universität Dresden, 2006
- [PT] Vern Paxson, Brian Tierney: *Bro Reference Manual*, 2004

Erklärung

Hiermit versichere ich an Eides Statt, dass die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Quellen von mir verfasst wurde.

Dresden, den 31. Dezember 2006