Technická univerzita v Liberci
Fakulta Pedagogická

Katedra informačních technologií
Studijní program: 2. stupeň
Kombinace: Informatika – Matematika

Zabezpečení dat v počítačových sítích
Security in computer networks

Diplomová práce 2001-FP-KIT

Autor: Jiří Vraný
Adresa: Žitná 828, Liberec 6, 460 06

Vednoucí DP: RNDr. Pavel Satrapa

Technická univerzita v Liberci
FAKULTA PEDAGOGICKÁ
461 17 LIBEREC 1, Hálkova 6 Tel./Fax: 048.5227332

Katedra: ..informačních technologií.....

ZADÁNÍ DIPLOMOVÉ PRÁCE

(závěrečného projektu)

diplomant: ...V r a n ý . Jiří.....
adresa: ..Energetiků 329, Stráž pod Ralskem.....
obor: .....Informatika - Matematika.....

Název DP: ....Zabezpečení dat v počítačových sítích.....
....Security in Computer Networks.....
.....

Vedoucí práce: ..RNDr. Pavel Satrapa.....

Termín odevzdání: .22.5.2001.....

Pozn. Podmínky pro zadání práce jsou k nahlédnutí na katedrách. Katedry rovněž specifikují zadání: východiska, cíle, předpoklady, metody zpracování, základní literaturu (zpravidla na rub tohoto formuláře). Zásady pro zpracování DP lze zakoupit v Edičním středisku TU v Liberci a jsou též k dispozici v UK TU, na katedrách a na Děkanátu Fakulty pedagogické.

V Liberci dne ..30.6.2000.....

...RNDr. Pavel Satrapa.....
vedoucí katedry

....doc. RNDr. Jaroslav Vild....
děkan

Převzal (diplomant):
Datum: 13.11.2000

Podpis:

KIT/IN
53 s.

Cíl:

Cílem diplomové práce je popsat problematiku bezpečnosti v počítačových sítích lokálního i globálního charakteru. Zejména problematiku zabezpečení administrátorského přístupu po síti. Dále je cílem práce seznámit se s dostupným zabezpečovacím softwarem, zejména se systémem Kerberos V a prakticky prověřit možnosti tohoto systému.

Literatura:

Garfinkel,S.,Spaford,G.: Bezpečnost v Unixu a Internetu v praxi, Computer Press, 1998

Frequently Asked Questions About Kerberos

HTTP:/WWW.NRL.NAVY.MIL/CCS/PEOPLE/KENH/KERBEROS

Systém KERBEROS IV

HTTP:/WWW.NATUR.CUNI.CZ/PRFDEC/KRB4.HTML

The KERBEROS Page

HTTP:/WWW.PDC.KTH.SE/KTH-KRB

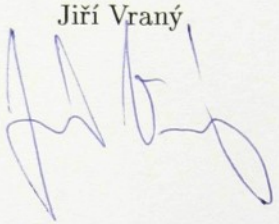
The KERBEROS Network Authentification Service

HTTP:/SUNSITE.AUC.DK/RFC/rfc/rfc1510.HTML

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostaně a že jsem uvedl všechnu použitou literaturu.

V Liberci dne 20.5.2001 Jiří Vraný



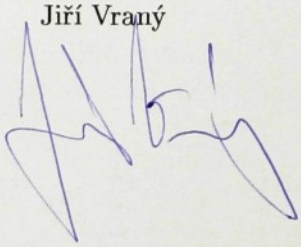
Byl jsem seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 o právu autorském zejména paragraf 60 (školní dílo)

Beru na vědomí, že Technická univerzita v Liberci (TUL) má právo na uzavření licenční smlouvy o užití mé diplomové práce a prohlašuji, že souhlasím s případným užitím mé diplomové práce (prodej, zapůjčení, kopírování apod.)

Jsem si vědom toho, že užít své diplomové práce či poskytnout licenci k jejímu využití mohu jen se souhlasem TUL, která má právo ode mne požadovat přiměřený příspěvek na úhradu nákladů vynaložených univerzitou na vyvtvoření díla (až do jejich skutečné výše). Diplomová práce je majetkem školy, s diplomovou prací nelze bez svolení školy disponovat.

Beru na vědomí že po pěti letech si mohu diplomovou práci vyzvednout v univerzitní knihovně TUL, kde bude uložena.

V Liberci dne 20.5.2001 Jiří Vraný



Chtěl bych poděkovat všem, kteří mi svou podporou umožnili dokončit tuto práci. Zejména vedoucímu diplomové práce RNDr. Pavlu Satrapovi za pomoc a poskytnuté rady a Zuzaně Lukešové nejen za kontrolu a opravu mých gramatických omylů.

Tato práce byla vytvořena pomocí sázecího programu LaTeX, na počítači s operačním systémem RedHat Linux 6.2.

Resumé

Diplomová práce pojednává o zabezpečení počítačových sítí. Cílem bylo popsat možnosti zabezpečení sítí a seznámit se s dostupným zabezpečovacím programovým vybavením, zejména s autentifikačním systémem Kerberos V5.

Teoretická část práce popisuje základní principy nejpoužívanější skupiny přenosových protokolů TCP/IP, nejznámější šifrovací algoritmy a také obvyklé formy útoků na počítačové sítě.

Práce je zakončena návrhem zabezpečení sítě základní nebo střední školy. Tento návrh je pokusem o spojení zabezpečovacích mechanizmů v dosud málo používanou kombinaci. Výsledkem této kombinace by mělo být zebezpečení sítě na dostatečné úrovni.

Summary

This Diploma thesis (DT) deals with security in computer networks. The aim of thesis was decribe network security and acquaint oneself with available protencting software, especially with authentication system Kerberos V5.

Theoretical part of thesis describes basic principles of the most used group of transfer protocols — TCP/IP. It also describes the best-known cipher algorithms and common types of attacks to the computer networks.

Last part of the DT is design of secured computer network at the basic or the high school. This design is an attempt at put together security principles to not so usual combination. The result of this combination will be equally secured network.

Obsah

1 Úvod

Tématem této diplomové práce je zabezpečení počítačových sítí. V současné době existuje pravděpodobně jen velmi málo oborů lidské činnosti, ve kterých by se nevyužívalo počítačů. Stále více se rovněž využívají počítačové sítě. A stále více lokálních počítačových sítí je připojeno k celosvětové síti - Internetu. Jen velmi naivní člověk by se domníval, že svět počítačů se výrazně odlišuje od jiných oborů lidské činnosti a že se v něm tedy nevyskytují lidé, kteří si se zákony a etickými normami příliš starostí nedělají.

Velmi stručná definice říká, že Internet je nezabezpečené místo. Tato práce popisuje některá rizika a metody jejich eliminace. Autor doufá, že v dohledné době se začnou naplňovat sliby politiků o školách připojených k Internetu. Právě v sítích škol je nutné dbát na zvýšenou bezpečnost. Špatně zabezpečená školní počítačová síť představuje riziko nejen pro danou školu, ale zejména pro ostatní uživatele Internetu - stává se totiž skvělým cvičištěm pro nové útočníky. Při návrhu bezpečnostních pravidel je ale dobré rozmyslet si, kde končí bezpečnost a začíná paranoia. Žádný systém totiž nelze zabezpečit na 100 %.

Vzhledem k významu skupiny protokolů TCP/IP v dnešních sítích popisuje úvodní kapitola hlavní protokoly patřící do této skupiny a důležité služby nad TCP/IP. V popisech je kladen důraz na bezpečnostní problematiku. Pochopení činnosti protokolů TCP/IP je důležitým předpokladem pro hlubší pohled do problematiky zabezpečení počítačových sítí.

V další kapitole jsou zopakovány základní principy kryptografie, základní kryptografické algoritmy a postupy. Protože v autentifikačním systému Kerberos je využíván algoritmus DES, je jeho princip popsán poněkud podrobněji.

Následující kapitola popisuje nejčastější formy útoků na počítačové sítě a navrhuje také vhodná bezpečnostní pravidla. Dále se tato kapitola zabývá problematikou firewallů, které se často používají jako ochrana sítě před vnějšími útoky. Závěr kapitoly je věnován některým autentifikačním systémům.

Předposlední kapitola popisuje autentifikační systém Kerberos. Vysvětluje základní principy autentifikace a důležité pojmy. Dále popisuje použitelnost systému Kerberos s různými operačními systémy a praktické možnosti nasazení tohoto systému.

Poslední kapitola obsahuje návrh zabezpečení imaginární školní sítě. V návrhu je využito teoretických poznatků předchozích kapitol.

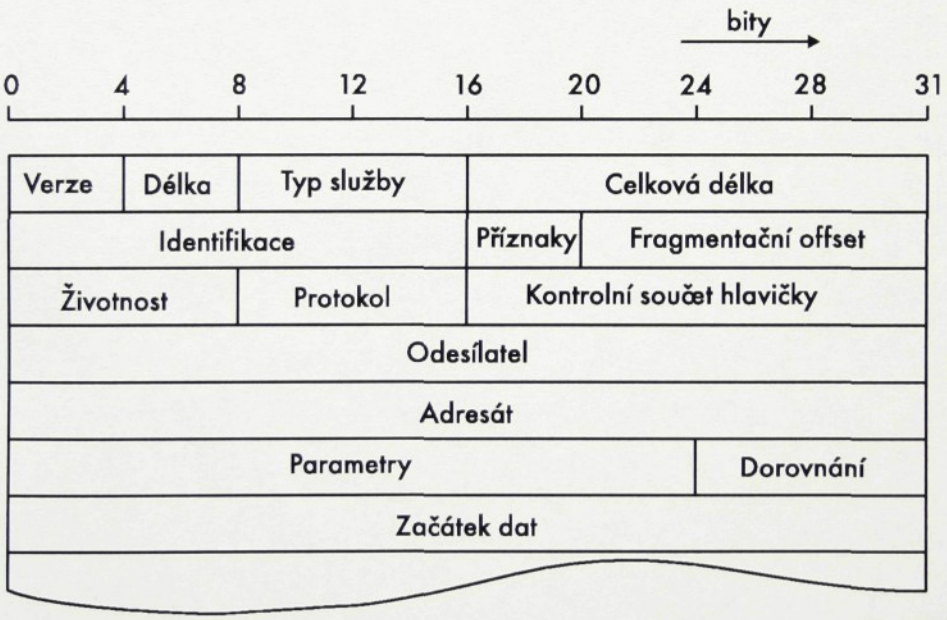
2 Sítě s protokolem TCP/IP

2.1 Popis protokolu

Nejvýznamnější skupinou síťových přenosových protokolů je v současnosti skupina označovaná jako TCP/IP. Používá se v sítích LAN i v síti Internet. Do této skupiny patří kromě základního IP také protokoly: TCP,UDP,RIP,ICMP a další. Pro pochopení dalšího textu je vhodné být seznámen s fungováním tohoto protokolu. Nejprve si popíšeme základní principy práce těchto protokolů.

2.1.1 IP - Internet Protokol

IP je hlavním protokolem skupiny TCP/IP. Definuje způsob, jak jsou zasílána data z jednoho počítače na druhý. Data jsou rozesílána ve formě bloků,kterým se říká „datagramy" nebo též „pakety". Strukturu IP paketu popisuje obrázek 2.1.



Obrázek 2.1: Struktura IP datagramu

- **Délka** udává velikost hlavičky IP datagramu v 32bitových slovech.
- **Celková délka** udává délku celého datagramu v bytech.
- **Identifikace, příznaky, fragmentační offset** kontrolují fragmentaci a znovu složení datagramu.

- **Protokol** udává, který protokol vyšší vrstvy byl k vytvoření datagramu použit.
- **Životnost** slouží jako ochrana proti zacyklení. Při průchodu směrovačem se snižuje o jedničku. Je-li rovná nule, datagram zanikne.
- **Parametry, dorovnání** jsou nepovinné položky.
- **Odesílatel, adresát** jsou IP adresy zdroje a cíle datagramu.

Každý počítač nebo síťové rozhraní v IP síti má jednoznačnou 32bitovou adresu. Obvykle se tato adresa zapisuje pomocí čtyř 8bitových čísel oddělených tečkou, například 147.230.16.1. IP Adresa udává dvě základní informace: adresu sítě a adresu konkrétního počítače. Jak velkou část adresy tvoří adresa sítě je určené *třídou adresy.*

IP protokol rozlišuje 5 základních tříd adres:

třída	horní bity	síť	počítač	adres v síti
A	0	7	24	16777214
B	10	14	16	65525
C	110	21	8	254
D	1110	skupinové adresy		268435456
E	1111	experimentální účely		

Třída adresy se určuje podle horních bitů. Kromě tohoto základního rozdělení mají některé adresy zvláštní význam.

- Adresa 0.0.0.0 se interpretuje jako stanice v této síti. Používá se např. při bootování bezdiskových stanic.
- Adresa 255.255.255.255 je hromadná adresa. Používá se pro zaslání paketu všem počítačům ve stejné síti.
- Adresa 127.0.0.0 je interní adresa počítače. Adresa 127.0.0.1 je přidělena internímu zařízení nazývanému *loopback.*
- Některé adresy nejsou určeny pro připojení do sítě Internet a není pro ně pro to určeno směrování. Tyto adresy se využívají v lokálních izolovaných sítích. Tyto adresy specifikuje [RFC1597].

Protože navržené dělení síť/počítač se ukázalo jako příliš hrubé,využívá se v současné době systému *podsítí.* Podsítí rozumíme jednu ze sítí třídy A–C rozdělenou na menší celky. Adresa potom udává tři základní informace: adresu sítě, adresu podsítě a adresu počítače. O tvaru konkrétního dělení na podsítě rozhoduje správce systému, obvykle podle topologie sítě. Jako doplněk k IP adrese potom slouží *maska podsítě*, která udává jaká část adresy je adresou počítače.

2.1.2 ARP a RARP

ARP znamená Adress Resolution Protocol. Slouží k zjišťování fyzických (MAC) adres,známe-li IP adresu. RARP znamená Reverse ARP a slouží tedy k opačnému účelu. IP pakety jsou totiž obvykle posílány pomocí ethernetu, který pro adresování počítačů používá vlastní (na protokolu IP nezávislé) adresy. Tyto adresy se nazývají MAC adresy a podobně jako IP adresy jsou celosvětově jednoznačné. MAC adresu přiděluje síťovému rozhraní již výrobce,a proto na rozdíl od IP adres nemají MAC adresy topologické uspořádání. Protokoly ARP a RARP popisuje RFC826 resp. RFC903.

2.1.3 TCP - transmision control protocol

TCP je spojovaný protokol se spolehlivým spojením. Spolehlivost spojení zaručuje, že je ověřeno doručení každého bajtu k adresátovi (nebo oznámení o nedoručení).Protokol má implementovánu detekci i korekci chyb. Data jsou přenášena formou datového toku (tedy nejsou chápána jako datagramy v IP) mezi dvěma počítači. Datový tok zaručuje, že data dojdou k příjemci ve stejném pořadím, v jakém byla odeslána. Tento datový tok je řízený: nemůže-li adresát přijímat data, přenos se pozdrží dokud není opět schopen přijímat.

Pro každé spojení je na obou stranách vyhrazen jeden *port*. Port identifikuje programy aplikační vrstvy, kterým náleží data ve vrstvě transportní. Přidělením portů pro přenos se tedy propojuje transportní a aplikační vrstva. Porty jsou určeny 16bitovým číslem, které je zakódováno v hlavičce TCP datagramu. Tato čísla se rozdělují na vyhrazená a volná. Vyhrazená čísla portů jsou přidělena jednotlivým službám podle předpisu, který je součástí RFC1700. Volné porty jsou určeny pro uživatelské programy (čísla > 1024)

Hlavní výhodou tohoto protokolu je spolehlivý přenos, který už aplikace nemusí dále kontrolovat. Jeho nevýhodou je vyšší režie zabezpečení přenosu, která vede ke zpomalení přenosu dat.

2.1.4 UDP - user datagram protocol

UDP je ve svých základních vlastnostech opakem TCP. Jedná se o nespojovaný a nespolehlivý protokol s volitelnou korekcí chyb. Nespolehlivost znamená, že ne každý paket bude doručen, případně že nebude zachováno pořadí paketů. Stejně jako v TCP se pro komunikaci používá dvojice portů. Hlavní výhodou UDP je vysoká rychlost díky nízké režii přenosu (podle [BezUaI] až desetkrát rychlejší než TCP). UDP se používá např. pro přenos směrovacích informací.

2.1.5 ICMP - internet control message protocol

Protokol ICMP slouží k výměně diagnostických a chybových informací mezi stanicemi komunikujícími pomocí IP. Podobně jako UDP je i tento protokol nespojovaný a nespolehlivý. Používá se pro výměnu směrovacích informací, oznámení o ukončení spojení pro síťové problémy apod.

2.1.6 Směrování

Protokoly skupiny TCP/IP jsou navrženy pro propojení velkého množství počítačů do globální sítě. Pro uchování informací o tom, kudy je nutné směrovat pakety tak, aby došly k cíli, se používají směrovací tabulky. Základní směrovací tabulku musí mít každé zařízení pracující v IP síti. K zajištění směrování se používají speciální stroje, kterým se říká *routery*. Pro výměnu směrovacích informací a doplňování směrovacích tabulek používají routery směrovací algoritmy. Mezi tyto algoritmy patří například RIP (routing information protocol) nebo OSPF (open shortest path first).

2.2 Služby nad protokolem TCP/IP

Počítače připojené do sítě s protokolem TCP/IP standardně poskytují celou řadu služeb. V prvopočátcích vývoje protokolů a služeb se patrně nepočítalo s tak masovým rozšířením. Jinak by totiž nemohlo dojít k tomu, že drtivá většina standardních služeb je velmi špatně zabezpečená proti nejrůznějším formám útoku. Následující kapitola ve stručnosti popisuje nejznámější služby a jejich potenciální bezpečnostní rizika.

2.2.1 DNS - domain name system

DNS je systém sdílených databází, který slouží k převodu IP adres na symbolická jména a naopak. Komunikace probíhá pomocí UDP paketů. V obvyklém pracovním režimu klient odešle UDP paket se žádostí o překlad jména na IP adresu. Například dotaz na jméno *bubo.vslib.cz* vrátí DNS server odpověď 147.230.16.1. DNS lze rovněž využít pro zjištění jména poštovního serveru dané domény, pro zjištění nadřazeného serveru nebo k výpisu všech adres v dané doméně.

Jména počítačů ani jejich IP adresy nebyly původně navrženy jako prostředek pro autentifikaci stroje. Bohužel celá řada programů (například r-příkazy) je právě k tomuto účelu využívá. Díky nezaručenému spojení může ale útočník za určitých podmínek podvrhnout DNS pakety. Technicky nemožné není ani zprovoznění nameserveru, který bude záměrně poskytovat falešné informace. Jména počítačů ani IP adresy by se proto neměly používat k autentifikaci.

2.2.2 SMTP - simple mail transfer protocol

Tento protokol slouží k rozesílání elektronické pošty. Kromě vlastního rozesílání rovněž provádí ověření existence adresáta, vyplní zpáteční adresu a další služby. Lze ho zneužít k získání informací o systému, který se útočník chystá napadnout, například lze zjistit informace o správci systému, o existujících uživatelských účtech apod.

Služby SMTP zajišťuje serverová aplikace - poštovní server. Nejznámějším poštovním serverem ve světě UNIXU (a potažmo tedy i TCP/IP) je program *sendmail*. Tento velmi komplexní a složitý program bohužel v minulosti obsahoval velké množství chyb, které pochopitelně mnohokrát vedly k narušení systému. Je proto nutné používat co nejaktuálnější verze a věnovat zabezpečení poštovního serveru značnou pozornost.

2.2.3 Telnet

Program Telnet slouží pro přihlášení ke vzdálenému počítači. Pro autentifikaci se používá jméno a heslo. Protože ale veškerá komunikace mezi serverem a klientem není nijak šifrována, je Telnet velmi náchylný na odposlech. Na dobře zabezpečeném serveru by se dnes standardní verze Telnetu vůbec neměla objevit, protože představuje příliš velké bezpečnostní riziko. V systému Kerberos je ale možno využít programu Telnet, který veškerou komunikaci v obou směrech šifruje. Pokud není používán Kerberos je vhodné použít místo Telnetu SSH (secure shell).

2.2.4 NTP - network time protocol

Tento protokol slouží k synchronizaci času mezi jednotlivými počítači. Základem pro synchronizaci je síť NTP serverů s různou přesností času. Nejpřesnější časový údaj poskytují primární NTP servery (servery první úrovně). Ty jsou určitým způsobem (např. rádiovým spojením) napojeny na velmi přesné hodiny. Protože ne vždy je potřeba naprosto přesný čas, je v systému celá řada serverů nižší úrovně, které poskytují méně přesné časové údaje[1]. Synchronizace času je velmi důležitá, chceme-li používat systém Kerberos, který časové údaje používá při identifikaci uživatelů a při správě identifikačních lístků.

2.2.5 FTP - file transfer protocol

FTP umožňuje přenášet binární a textové soubory. Využívá několika TCP spojení. První spojení je řídící a slouží pro ovládání relace. Každý přenos souborů se potom realizuje pomocí nového TCP spojení. Podobně jako u Telnetu není ani u FTP spojení nijak šifrované. Je zde tedy stejné riziko zachycení paketu s heslem. Řešením může být například použití protokolu SCP

[1] Odchylky se pohybují v milisekundách, nejvýše sekundách

(secure copy) a to zejména v případech, kdy se realizuje autentifikované připojení (zadává se jméno + heslo). V případě anonymního FTP připojení je heslem e-mailová adresa uživatele a bezpečnostní rizika při zachycení paketu tedy nejsou velká.

Velkou pozornost je třeba věnovat zabezpečení FTP serveru. Protože TCP spojení používané protokolem vyžaduje přístup k portu 20 (který smí používat pouze superuživatel), musí FTP server běžet pod superuživatelským oprávněním. To pochopitelně představuje jisté bezpečnostní riziko. Konfigurace zabezpečeného FTP serveru je podrobně popsána např. v [BezUal] na straně 487.

2.2.6 TFTP - trivial file transfer protocol

Protokol TFTP slouží k přenosu souborů pomocí UDP. Pro svou jednoduchost se používá zejména k bootování bezdiskových stanic nebo X-terminálů. Neobsahuje žádnou identifikaci ani ochranu. TFTP server prostě umožní přístup komukoliv, kdo o něj požádá. Proto je nutné velmi přesně specifikovat adresáře a soubory, do kterých je pomocí tohoto protokolu umožněn přístup. Přesto z důvodů nulové identifikace představuje velké riziko a neměl by se používat pokud to není nutné.

2.2.7 Finger a Whois - informace o uživatelích

Program Whois poskytuje pouze základní kontakt na zadanou osobu (obvykle e-mail). Finger je daleko komplexnější a zobrazuje: *uživatelské jméno, skutečné jméno, čas přihlášení, místo přihlášení* a některé další informace o uživatelích. Tyto údaje ovšem mohou být vodítkem pro útočníka, který hledá informace o možném průniku do systému. Při konfiguraci serveru, by se tedy mělo zabránit dotazům z vnější sítě, nebo Finger raději úplně vypnout. Případně použít program, který dokáže poskytovat jiné informace lokálním a cizím uživatelům.

2.2.8 HTTP - hypertext transfer protocol

Tento protokol je využíván pro získávání informací z WWW (world wide web). WWW je dnes nejrozšířenější oblast služeb nad TCP/IP. Pro mnoho laiků dokonce platí jednoduchá rovnost Internet = WWW.

Základní princip práce HTTP vypadá následovně: klient kontaktuje server a požádá ho o určitý soubor. Server mu odešle textový dokument, formátovaný jako ASCII nebo HTML (jazyk pro tvorbu WWW stránek). Požadovaný soubor si klient zobrazí pomocí speciálního programu - *prohlížeče*. Bezpečnostním rizikem pro klienta je důvěra v získané dokumenty. Moderní skriptovací jazyky totiž umožňují snadnou tvorbu „nepříjemných" programů. Server naopak ohrožují dotazy klientů. Je-li pro zpracování do-

tazů spuštěn nějaký lokální program s nevhodnými parametry, může útočník převzít jeho kontrolu a následně narušit chod systému.

Přestože samotný protokol HTTP je velmi jednoduchý, problematika zabezpečení všech služeb WWW jednoduchá není. Navíc je tato oblast velmi živá a neustále se dále rozvíjí. Popsat konfiguraci správně zabezpečeného WWW serveru je proto téma na celou knihu. Proto zde uvedeme pouze některé nejvýznamnější zásady:

- Počítač by měl sloužit pouze účelům WWW serveru. Neměl by poskytovat žádné další služby a mělo by na něm být co nejméně uživatelských účtů. Adresáře pro ukládání WWW stránek lze uživatelům zpřístupnit jiným způsobem (například pomocí NFS).

- Další velké riziko představují CGI (common gateway interface) scripty. Jedná se o různě složité programy, které zvyšují možnosti WWW. Například nejrůznější počítadla přístupů, návštěvní knihy apod. Tyto skripty jsou spouštěny přímo na WWW serveru, nikoliv na straně klienta jako například skripty v jazyce PHP. Právě v tom je jejich velké nebezpečí. Pokud script obsahuje chyby, může ho zkušený útočník využít ke spouštění dalších programů na serveru. Proto by se na serveru měly používat pouze bezpečné a ověřené scripty. Neznámé a nepoužívané scipty, které se mohou objevit například po instalaci serveru je bezpečnější smazat. V žádném případě nelze uživatelům dovolit spouštět libovolné scripty. CGI scripty ovšem výrazně zvyšují možnosti WWW, proto by byla škoda je nepoužívat. Je ale nutné psát sripty tak aby byly co nejbezpečnější a důkladně je otestovat. Pro psaní bezpečných scriptů platí podobná pravidla jako pro psaní ostatních programů spouštěných na interntovém serveru (viz str. 26.

- WWW server musí být nastartován v superuživatelském režimu. Nikdy by ale neměl běžet pod právy superuživatele. Ihned po startu je proto nutné změnit UID serveru. Nové UID lze nastavit v některém konfiguračním souboru serveru (obvykle je to soubor `httpd.conf`). Pokud by server běžel v režimu superuživatele, běžely by v tomto režimu i všechny serverem spuštěné procesy.

Jak už bylo naznačeno, je celá problematika daleko obsáhlejší. I na zabezpečení WWW serverů lze ale aplikovat některá pravidla, uvedená v kapitole 4.3 - Jak se bránit.

2.2.9 RPC - remote procedure call

Již z názvu tohoto protokolu je patrné, že umožňuje spouštět procedury na vzdálených počítačích. Programátorům tak částečně odpadá nutnost síťového programování, postačí, když využijí služeb RPC. Protokol využívá jak TCP tak UDP. Základem pro jeho práci je autentifikace. Pro volně

přístupné služby lze nastavit nulovou autentifikaci. Druhá možnost - standardní Unixová autentifikace je nebezpečná, útočník může snadno změnit identitu a předstírat, že je někdo jiný. Proto je nejvhodnější používat variantu nazvanou *Secure RPC*, která používá autentifikační schéma založené na kryptografii s veřejným klíčem a na algoritmu DES. Máme-li zprovozněn systém Kerberos, můžeme v RPC nastavit autentifikaci tímto systémem.

Používání RPC přináší řadu výhod, současně ale zvyšuje počet míst, na kterých je nutno systém dobře zabezpečit před napadením. Před použitím některého RPC systému je tedy velmi důležité promyslet, co a jak chceme používat. Mezi nejznámější RPC služby patří dva systémy od společnosti Sun: NIS a NFS.

2.2.10 NIS a NIS+

Network Information Service (NIS) byl vyvinut pro prostředí malých, přátelských sítí. Pro síť připojenou do Internetu představuje ale velmi nebezpečnou volbu. Jedná se o distribuovanou databázi účtů, hesel, IP adres, klíčů a případně dalších údajů. Umožňuje uživatelům práci se stejnou databází na různých stanicích, tak jako by její kopie existovala zvlášť na každém z nich. Všechny přenosy tohoto systému jsou nešifrované a tedy snadno odposlechnutelné. Zjevnou slabost tohoto systému si uvědomila i společnost Sun a následkem byla nová verze NIS+. NIS+ má opravenou většinu bezpečnostních chyb, dokáže rovněž využívat autentifikaci Secure RPC. Přesto jsou názory na jeho používání značně rozporuplné. Jako vždy se bezpečnostní rizika odvíjejí od konfigurace systému.

2.2.11 NFS - network filesystem

NFS je zejména v prostředí Unixu velmi rozšířená služba pro sdílení disků. Existují ovšem i verze pro další operační systémy (MAC-OS, Windows). Systém vypadá na první pohled velmi jednoduše. Klient si připojuje disky serveru, téměř stejně jako disky lokální. Celý systém je založen na protokolech UDP a RPC. Server pracuje jako bez stavový - případný reboot klienta nebo serveru nevyžaduje úpravy stavů na opačné straně spojení. Díky možnosti použití Secure RPC je systém relativně solidně zabezpečen, přesto je zde stále riziko odposlechu přenosů (ty na rozdíl od autentifikace šifrovány nejsou).

Základní jednotkou systému je tzv. *file handle*, unikátní řetězec, který jednoznačně identifikuje každý adresář a soubor na disku. Tento řetězec je generován serverem a klient nemůže nijak interpretovat jeho obsah. Znalost tohoto řetězce, dává potenciální možnost přistupovat k danému svazku. Není proto vhodné zpřístupňovat svazky s oprávněním pro superuživatele.

Pro klienta je zde opět nebezpeční podvržených informací. Touto cestou se mohou šířit chybné programy nebo některé trojské koně. Klient by měl

svazky připojovat bez možnosti používat SUID programy, speciální zařízení, případně dokonce bez práv na spouštění souborů.

2.2.12 X Windows

X Windows jsou grafický síťový systém pro Unixové systémy. Základem je architektura klient/server. Klientem je v tomto případě aplikace, která pomocí X protokolu komunikuje se serverem. Server má na starosti řízení klávesnice, myši, grafického displeje a případných dalších zařízení. Klient i server mohou běžet na jednom stroji stejně jako na několika různých. V tomto principu je síla systému ale zároveň jeho bezpečnostní slabina.

Zná-li aplikace IP adresu Xserveru (tedy naší stanice) a číslo portu, může se připojit na server, pokud není chráněný. To většinou není a tak se aplikace, připojí, aniž by uživatel cokoliv zjistil. Aplikace pak sdílí prostředky serveru, takže například může načítat všechny znaky napsané na klávesnici a tak odposlechnout heslo. Rovněž číslo portu je snadné získat, obvykle je to číslo o něco větší než 6000.

Takový systém by pochopitelně nebyl k ničemu, a proto bylo do systému X Windows implementováno několik mechanismů pro řízení přístupu. Od nejjednodušší ochrany pomocí seznamu povolených hostů, přes různá použití algoritmu DES (XDM authorization, Secure RPC) až po autentifikaci pomocí systému Kerberos.

2.2.13 R - příkazy

Skupina příkazů používaná pro přístup a ovládání vzdálených stanic. Patří sem například `rlogin`, `rsh`, `rcp` a některé další. Příkaz rlogin umožňuje přihlášení ke vzdálenému serveru podobně jako program Telnet. Na rozdíl od programu Telnet nemusí uživatel zadávat své heslo, pokud jsou splněny následující podmínky:

- Volání musí přicházet z privilegovaného portu (TCP 513). Některé OS (například DOS) ale používání těchto portů nikterak neomezují.
- Volající počítač a uživatel musí být uvedeni na cílovém počítači v tabulce prověřených hostů.
- Jméno uživatele musí souhlasit s příslušnou IP adresou.

Jako v předcházejících případech i zde nebylo v návrhu příliš počítáno s nepřátelským prostředím. V malé uzavřené síti jsou tyto příkazy výborným nástrojem, který umožní pracovat na více strojích bez neustálého zadávání hesel. V nepřátelském prostředí ovšem neposkytují téměř žádnou ochranu (implementované bezpečnostní schema lze snadno obejít).

Pokud použijeme pro autentifikaci systém Kerberos, získáme mimo jiné možnost používat zabezpečené r-příkazy. Důvěryhodnost hostů je pak garantována systémem, který provádí jejich identifikaci.

2.2.14 SSH - secure shell

Secure shell je moderní program, který je dobrou náhradou programu Telnet i r-příkazů. Pro spojení se vytváří kryptovaný kanál mezi klientem a serverem. Nad tímto kanálem je pak možno vytvořit TCP/IP spojení nebo spojení systému X11.

Pro šifrování používá Ssh algoritmus RSA s veřejnými a soukromými klíči. Ssh si vytváří databázi veřejných klíčů okolních počítačů, v případě že klíč neexistuje vygeneruje se automaticky při prvním spojení. Toto opatření eliminuje možnost útoků s podvrženou identifikací. Aby se spojení mohlo vytvořit, musí být splněny následující podmínky:

- Veřejný klíč klienta (na straně klienta klíč serveru) musí být shodný s klíčem uloženým v databázi. Ta je v systémech na bázi Unixu uložena obvykle v jednom z následujících souborů:
  - `/etc/ssh/ssh_known_hosts`
  - `$HOME/.ssh/known_hosts`.

  Pokud klíč není shodný, spojení se nepovolí. Pokud klíč neexistuje, spojení se povolí a přejde se k další identifikaci.
- Pokud se jméno stanice (a uživatele) vyskytuje v tabulkách:
  - `/etc/hosts.equiv, /etc/shosts.equiv`
  - `$HOME/.rhosts, $HOME/.shosts`

  je spojení navázáno bez dotazu na heslo. Pokud tomu tak není, je uživatel vyzván, aby zadal heslo. Zadané heslo je zašifrováno a odesláno tak, že je zabezpečeno proti odposlechu. Ještě bezpečnější je ale tuto identifikaci zakázat a použít následující způsob.
- Tato autentifikace využívá principu RSA. Pokud je na serveru v adresáři `$HOME/.ssh/authorized_keys` uveden veřejný klíč klienta, je automaticky nalogován bez dotazu na heslo. Logování je možno omezit na spouštění třeba jen jediného programu.

Pro identifikaci spojení využívá Ssh algoritmu RSA. Další komunikace už je ale šifrována symetrickým algoritmem. Podporovaných algoritmů je celá řada: IDEA, DES, Triple-DES, blowfish, TSS. Implementována je rovněž možnost nešifrovat přenos vůbec. Implicitně je nastaveno kódování algoritmem IDEA.

Jak už bylo uvedeno nad kryptovaným Ssh kanálem je možno vytvořit TCP/IP spojení. Ssh pak šifruje veškerý IP provoz. Tato možnost se dá velmi dobře využít například při návrhu virtuálních sítí nebo pro bezpečné připojení vzdáleného počítače.

Kromě kryptování spojení dokáže Ssh přenášená data také komprimovat. Při vzdálených připojeních tak přispívá k větší rychlosti spojení. Ssh je tak na výkon počítače o něco náročnější než standardní programy (Telnet), ovšem pro dnešní stroje nepředstavuje prakticky žádnou zátěž.

2.2.15 Současné trendy

V současnosti už je asi každému jasné, že Internet není rajská zahrada. Potenciálních útočníků jsou zde celé armády. Je pochopitelné, že server základní školy v Horní Lhotě pravděpodobně nebude cílem 50 útoků za den. Přesto by měl být dobře zabezpečen každý server, připojený do internetu, protože špatně zabezpečené servery mohou posloužit jako výcvikový prostor nových útočníků nebo jako mezistanice pro další útok (a stopa pak vede k falešnému cíli).

V současnosti se často využívá šifrování choulostivých přenosů (jména a hesla), nebo dokonce šifrování celého přenosu. Dále se ustupuje od používání nebezpečných aplikací a protokolů (Telnet), které se nahrazují bezpečnějšími variantami (Ssh). Běžné nezabezpečené protokoly (http) se pro svou jednoduchost stále používají tam, kde nejsou vysoké bezpečnostní nároky. Je-li nutná větší bezpečnost přenosu, používají se jejich složitější varianty (https), které obsahují i ochranné mechanizmy.

V nově vytvářených programech a protokolech (IPv6 aj.) je na zabezpečení pamatováno již při jejich tvorbě. Lze tedy doufat, že v budoucnu bude možné bezpečnost počítačových sítí ještě zvýšit.

3 Kryptografie

3.1 Úvod do kryptografie

Kryptografie je věda, která se zabývá zejména šifrováním a dešifrováním dat. Její počátky sahají až do starověku a prakticky celá její historie je nějak spojena s vojáky nebo alespoň s politikou. Utajení zpráv před nepřítelem a naopak snaha přečíst zprávy, které nepřítel zašifroval, byly hnacím motorem pokroku v této oblasti. Starověké šifry se dnes uplatní snad jen při hrách skautských či turistických oddílů. Moderní kryptografie využívá nejvýkonnější výpočetní techniku a složité matematické postupy. Kryptografie řeší několik základních problémů:

- Návrh a analýza kryptosystémů. Aplikací matematických postupů se kryptografové snaží vyvinout nové algoritmy, dokázat bezpečnost stávajících. Dále je snaha stanovit sílu algoritmů a tak je určitým způsobem uspořádat.
- Dešifrování zpráv. Snaha o prolomení stávajících algoritmů ať již pomocí nějakého matematického postupu, nebo prostě hrubou silou (vyzkoušení všech možností). Každé prolomení algoritmu je zpravidla velmi mediálně poutavou záležitostí (vzpomeňme jen nedávný rozruch kolem údajného prolomení PGP), protože se vždy jedná o určité překonání dosavadních možností.
- Integrita dat a zpráv. Cílem je vytvořit šifrovací postupy, které příjemci vydají jednoznačný signál, zda byla zpráva během přenosu narušena a modifikována.
- Autentifikace osob. Jde v podstatě o jednoduchý problém - jak zjistit, že osoba, která se mnou komunikuje, je skutečně tím, za koho se vydává. Tato oblast se zabývá například problematikou elektronického podpisu.

Kryptografie pochopitelně řeší i mnoho dalších problémů a její význam zejména v počítačovém světě v poslední době neustále vzrůstá. Následující text popisuje některé základní kryptografické metody.

3.1.1 Základní pojmy

Šifrovací algoritmus je obvykle matematická funkce, která provádí vlastní šifrování a dešifrování dat.

Šifrovací klíč určuje, jak konkrétně se budou data šifrovat. Pro volbu vhodného klíče platí stejná pravidla jako pro heslo.

Délka klíče udává přibližnou sílu algoritmu. Algoritmy s delším klíčem jsou totiž odolnější proti útokům hrubou silou. Podle délky klíče (a dalších kritérií) se algoritmy dělí na slabé a silné. Silné algoritmy byly vždy pokládány za strategický materiál.

Nešifrovaný text jsou informace, které chceme zašifrovat.

Šifrovaný text je výsledek činnosti algoritmu.

3.1.2 Jednosměrné šifrování

Jednosměrné šifrování je metoda, při níž text jednoznačně zakódujeme, ale už ho nedokážeme zpětně rozkódovat. Tímto způsobem se například uchovávají hesla v souboru `/etc/passwd`. Při identifikaci se heslo znovu zašifruje pomocí standardní funkce *crypt*[1] a porovná se s uloženým.

Značným teoretickým problémem je dokázat neexistenci inverzní funkce.Tedy dokázat, že šifrovací funkce je skutečně jednosměrná. Jedinou obecně použitelnou metodou je v současnosti tzv. *útok prohledáním prostoru klíče*, kdy jsou postupně vyzkoušeny všechny možnosti. Tento útok se někdy označuje také jako *útok hrubou silou.*

Obecně se za jednosměrné šifry považují například: Unixová funkce crypt()[2], nebo například algoritmus MD5, používaný např. pro generování kontrolních součtů.

3.1.3 Symetrické šifrování

Symetrické šifrovací algoritmy jsou někdy označovány jako *algoritmy s privátním klíčem.* Jedná se o skupinu algoritmů, která pro šifrování a dešifrování textu používá stejný klíč. Tento klíč musí znát jak odesílatel tak příjemce zprávy. Tyto algoritmy se používají například pro ukládání choulostivých dat na disku nebo pro zabezpečený přenos mezi dvěma systémy. Jejich citlivým místem je pochopitelně klíč, zejména jeho bezpečné předání mezi oběma účastníky komunikace.

Podívejme se nyní na několik nejznámějších symetrických algoritmů:

ROT13: primitivní algoritmus bez klíče. Principem je substituční šifra známá již ve starověku. Každé písmeno abecedy se nahradí písmenem, které je o 13 znaků v abecedě dále. Tato metoda se označuje jako Caesarova šifra.

Crypt: původní šifrovací algoritmus Unixu. Používá klíč proměnlivé délky. Není považován za bezpečný, protože existují programy, které ho dokáží dešifrovat bez znalosti klíče. Jedním z takovýchto programů je

[1] Nezaměňovat s velmi slabým unixovým programem crypt, který funguje úplně jinak.

[2] V dalším textu je zmiňován silný algoritmus Triple DES, funkce crypt používá modifikovaný DES dokonce 25!

CBW (crypt breaker's workbench), jehož autorem je Robert Baldwin z MIT.

DES: je v současné době nejrozšířenějším symetrickým algoritmem. Používá klíč o délce 56bitů. Vzhledem k jeho významu v systému Kerberos se mu budeme věnovat více na dalších stránkách.

IDEA: international data encryption standard je považován za poměrně silný algoritmus. Přestože nebyl vyvinut v USA a nebyl tedy chráněn vývozními restrikcemi, není příliš rozšířen, protože je pro změnu chráněn řadou patentů. Tento algoritmus je využíván programem PGP pro šifrování elektronických podpisů.

Mezi symetrické algoritmy patří dále například RC2, RC4, RC5, skipjack a další.

3.1.4 Asymetrické šifrování

Asymetrické šifrovací algoritmy jsou známy také jako *algoritmy s veřejným klíčem*. Pro zašifrování textu používají jiný klíč než pro jeho dešifrování. Každý uživatel vlastní dva klíče - klíč veřejný a klíč soukromý. Musí platit, že ze znalosti veřejného klíče nelze odvodit klíč soukromý. Veřejný klíč se používá k zašifrování textu, soukromý klíč k jeho dešifrování. Veřejné klíče jsou uveřejněny ve volně přístupných seznamech, na WWW stránkách apod. Průběh vlastního šifrování je poněkud složitější než v případě symetrických algoritmů. Podívejme se nyní na základní princip asymetrického šifrování.

Uživatel M chce zaslat zašifrovanou zprávu S uživateli N. Jak tedy bude postupovat.

1. M použije veřejný klíč K_{Nv} (který zná nebo ho může snadno zjistit) a zašifruje zprávu S. Získá kryptogram $K_{Nv}(S)$, který odešle uživateli N.

2. N může kryptogram dešifrovat, protože jako jediný zná svůj privátní klíč K_{Np}. Platí $K_{Np}(K_{Nv}(S))=S$.

3. Nikdo jiný než uživatel N nemůže kryptogram rozluštit, protože podle předpokladu nelze ze znalosti veřejného klíče odvodit klíč privátní.

Hlavní výhody asymetrických algoritmů jsou:

- Není nutná výměna klíčů. Díky tomu odpadá problém bezpečné výměny tajného klíče.
- Snadný přístup nových uživatelů. Nový uživatel prostě jen dá k dispozici svůj veřejný klíč.
- Dobré vlastnosti pro použití v oblasti elektronického podpisu.

Hlavní nevýhodou je dozor nad klíči. Je potřeba jistým způsobem zaručit, že zveřejněný klíč skutečně patří uvedené osobě. Například zákony o elektronickém podpisu řeší tento problém pomocí tzv. *certifikačních autorit.*

Asymetrické algoritmy jsou také výrazně pomalejší než aloritmy symetrické. Operace exponeciální a modulární aritmetiky používané v těchto algoritmech, jsou výpočetně mnohem náročnější než kombinatorické operace používané v algoritmech symetrických.

Nyní si opět uvedeme nejznámější algoritmy této skupiny:

RSA: patrně nejznámější asymetrický algoritmus. Díky zveřejnění algoritmu před jeho patentováním, se rychle rozšířil do celého světa. Používá se jak pro šifrování tak v systémech elektronického podpisu. Může používat klíče různé délky, čím delší klíč tím je samozřejmě šifra bezpečnější. Základní matematický model algoritmu spočívá na složitosti rozkladu velkých čísel na prvočinitele.

ElGamal: podobně jako RSA je založen na modulární a exponenciální aritmetice. Rovněž použití a bezpečnost je zhruba stejná jako u RSA.

Diffie-Hellman: používá se zejména pro výměnu tajných klíčů. Nejedná se v pravém slova smyslu o šifrovací algoritmus, jde spíše o metodu podle níž se nejprve sestrojí tajný klíč a potom dojde k jeho výměně.

3.1.5 Kombinované šifrovací systémy

Tyto systémy v sobě spojují oba předcházející způsoby šifrování. Největším problémem u symetrického šifrování je bezpečná výměna tajného klíče. Asymetrické algoritmy jsou zase velmi výpočetně náročné. Jejich vhodnou kombinací lze právě tyto nevýhody snadno eliminovat. Nyní si stručně popíšeme jak takový kombinovaný systém pracuje.

Program nejprve vygeneruje náhodný tajný klíč. Ten je zašifrován pomocí asymetrického algoritmu a odeslán příjemci. Po dešifrování pomocí soukromého asymetrického klíč, má příjemce k dispozici tajný klíč pro symetrické šifrování. Další přenosy jsou pak už šifrovány rychlým symetrickým algoritmem.

V praxi je většina systémů šifrování veřejným klíčem realizována právě pomocí kombinovaných systémů. Například velmi populární šifrovací program PGP (pretty good privacy) používá kombinaci algoritmů RSA a IDEA[3].

3.1.6 Digitální podpisy

O digitálních podpisech se v poslední době hodně mluví a píše. V naší republice byl, podobně jako ve většině vyspělých států, přijat zákon o elektronickém podpisu. Jeho uvedení do praxe snad přinese lepší možnosti využití moderních komunikačních prostředků.

[3]existují verze PGP které používají triple DES

Základní princip je v podstatě stejný jako u asymetrického šifrování. Používají se rovněž stejné algoritmy. Jenom sled operací je inverzní. Zprávu (nebo její část) zašifrujeme naším privátním klíčem. Kdokoliv ji může rozšifrovat veřejným klíčem, který jsme dali volně k dispozici. Protože existuje právě jeden privátní klíč, který zakóduje zprávu tak, aby šla dešifrovat naším veřejným klíčem, má příjemce potvrzenou identitu odesílatele.

Digitální podpis se často kombinuje s *algoritmy pro výtah zprávy*. Pomocí nich se vytvoří pouze kontrolní součet zprávy. Toto jednoznačné číslo je pak zašifrováno asymetrickým algoritmem a připojeno k nešifrované zprávě. Takovýto dodatek se pak ještě více podobá obvyklému podpisu. Tento princip se využívá pro svou rychlost. Asymetrické algoritmy nejsou příliš rychlé a šifrování dlouhé zprávy (MB) se tak může neúměrně protáhnout.

Velmi často se pro elektronický podpis používá kombinace algoritmů MD5 pro vytvoření kontrolního součtu a RSA pro zašifrování. Další kombinací je využití algoritmů SHA (secure hash function) a ElGamalova algoritmu. Tato kombinace je známa jako algoritmus DSA (digital signature algorithm)

3.2 Restrikce používání kryptografických algoritmů

Masivnímu používání většiny kryptografických algoritmů brání a nebo donedávna bránily nejrůznější restrikce. Drtivá většina algoritmů je chráněna patenty, některé dokonce světovými patenty (IDEA). Druhým a možná výraznějším omezujícím prvkem byly donedávna vývozní restrikce USA. Silné šifry byly totiž řazeny mezi materiály strategického významu, do stejné skupiny jako například jaderné zbraně. Tyto restrikce byly nejrůznějším způsobem obcházeny a také často kritizovány. Americkým softwarovým společnostem totiž pochopitelně unikaly značné zisky na světových trzích. Teprve v roce 2000 se snad začalo blýskat na lepší časy. Vypršela platnost patentů některých algoritmů a byly částečně omezeny některé vývozní restrikce. Situace je však právně velmi složitá a zaběhlé restrikce se uvolňují velmi pomalu.

Mezi silné šifrovací algoritmy je počítán i algoritmus DES, který je základem šifrování v systému Kerberos. Proto byly a bohužel dosud jsou uplatňovány na Kerberos vývozní restrikce. Z oficiálních stránek stále není možno stáhnout si zdrojové kódy, nebo upravené aplikace, jinam než na území USA nebo Kanady. Jako vysvětlení je připojen následující krátký text:

> Novela amerických exportních regulací v lednu a říjnu 2000 může restrikce omezit. Zatím z důvodů právní nejednoznačnosti tohoto problému neplánujeme změnu našeho rozhraní pro získávání souborů.
>
> V současné době konzultujeme s našimi právníky, jak nové vývozní regulace ovlivní naši distribuci zdrojových kódů a binárních

souborů systému Kerberos. (Je to dlouhý a pomalý proces, a to z mnoha důvodů. Prosíme, neptejte se....). Včas oznámíme jestli a kdy se tyto změny odrazí na našich distribučních metodách.

V současné době si můžete zdrojové kódy opatřit na WWW stránkách projektu crypto-publish.org, za což děkujeme Marku Horowitzovi.

Restrikce jsou tedy změněny a systém Kerberos je například součástí linuxové distribuce RedHat6.2. Oficiální stanoviska vývojového týmu jsou však poněkud zdrženlivá. Upřímně řečeno příliš se tomu divit nelze, protože v USA lze žalovat kohokoliv pro cokoliv (tedy téměř) a není vyloučeno, že by se někdo mohl pokusit proslavit právě takovouto kauzou.

3.3 Algoritmus DES

3.3.1 Úvod

Algoritmus DES je hlavním šifrovacím algoritmem používaným v systému Kerberos. Přestože jsou v současnosti podporovány i jiné algoritmy, zůstává DES standardem. Vzhledem k jeho významu je mu proto věnována samostatná část textu.

Šifrovací algoritmus DES (Data Encryption Standart) patří do skupiny symetrických algoritmů. Byl vyvinut počátkem 70tých let a v roce 1975 patentován firmou IBM. V roce 1977 vydala vláda Spojených států amerických jeho popis jako publikaci Federálního standardu pro zpracování informací (FSI PUB) č. 46. Algoritmus byl také schválen jako norma ANSI (X3.92-1981/R1987).

Algoritmus se na první pohled jeví jako velmi silný, protože k prolomení hrubou silou je nutné vyzkoušet $7 \cdot 10^{16}$ kombinací. Nikdy však nebyl schválen pro šifrování informací klasifikovaných jako tajné. Přesto je hojně využíván i v současnosti a to jak v americké státní správě, tak v soukromém sektoru, zejména v oblasti financí. V knize [BezUaI] se píše o tom, že původně navržený algoritmus s názvem Lucifer, byl na žádost NSA (Národní Bezpečností Agentury) úmyslně oslaben. Podle některých spekulací existovalo HW zařízení schopné prolomit algoritmus DES metodou prohledávání klíče již v době jeho standardizace. NSA si tak úmyslným oslabením původního algoritmu chtěla zajistit možnost číst šifrované informace.

Podobné spekulace přenechme autorům špionážní literatury. Faktem ale je, že zvýšení výkonu počítačů, odstartovalo ve druhé polovině 90. let závody v prolamování algoritmu DES. V roce 1997 se internetovému sdružení distributed.net zapojením velkého množství dobrovolníků podařilo prolomit šifru za 41 dní. Což je relativně dlouhá doba, ale již v červnu roku 1998 představila americká společnost Electronic Frontier Foundation speciální šifrovací počítač, který prolomil DES během 56 hodin! V roce 1999 vypsala společnost

RSA prémii pro toho, kdo první pokoří hranici 24 hodin. Hlavním důvodem těchto snah bylo, kromě získání prestiže, zařazení algoritmu DES mezi slabé šifry a tedy exportní možnosti.

Zjevné slabiny algoritmu DES logicky vedly k jeho vylepšení. Přesněji řečeno k vylepšení jeho používání. Namísto prostého šifrování se používá metoda nazvaná *superšifrování*. Pod tímto superlativním názvem se ukrývá pouze další zašifrování již zašifrovaného textu. Pochopitelně jiným klíčem. Jednoduchá a přitom velmi účinná metoda. Opakování se provádí buď 2x (Double DES) nebo častěji 3x (Triple DES).

3.3.2 Popis algoritmu

DES je symetrická bloková šifra. Data jsou kódována v 64 bitových blocích. Základem pro šifrování je 56bitový klíč, který se používá pro šifrování i dešifrování dat (symetrická šifra). Vlastní šifrování je realizováno jako série bitových permutací, substitucí a rekombinačních operací prováděných s 64bitovými bloky dat a 56bitovým klíčem (každý osmý bit slouží pro paritní zabezpečení a při zpracování se ignoruje).

Základní model algoritmu využívá dvě kryptografické metody konfuzi a difuzi. Konfuze se realizuje pomocí substituce a difuze pomocí permutace. Vlastní krok algoritmu tedy představuje substituci následovanou permutací. Anglický termín pro tento základní blok operací je *round*. Algoritmus DES má takovýchto kroků 16. Základní schéma algoritmu popisuje obrázek 3.1.

64bitový blok dat je nejprve permutován takzvanou počáteční permutací PP. Potom se blok rozdělí na dvě 32bitové části R(0) a L(0). Tyto části jsou pak v 16 krocích šifrovány pomocí podklíčů K1–K16. Tyto podklíče (subkeys) jsou definovány takzvanou KS funkcí (key schedule). Na závěr je blok dat znovu permutován permutací inverzní k permutaci PP. Permutace PP a PP^{-1} jsou definovány normou. Schématické znázornění šifrovacího kroku je znázorněno na obrázku 3.2.

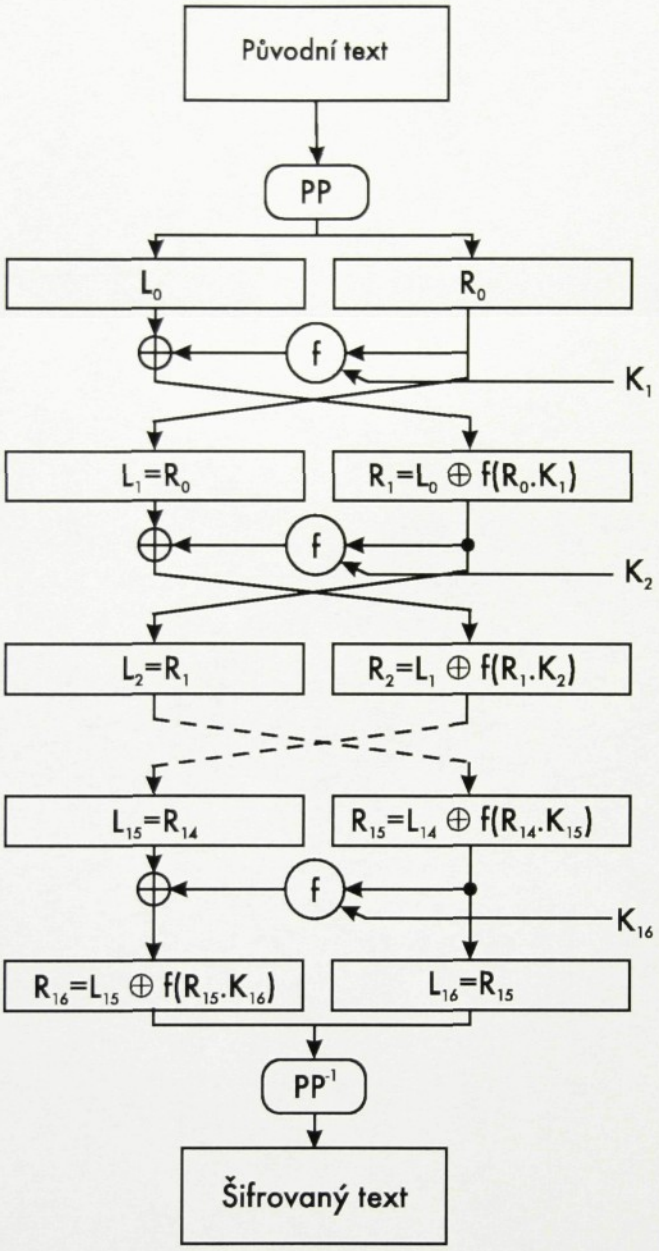
Vstupem jsou 32bitové bloky R_{i-1}, L_{i-1} a 64bitový klíč K_{i-1} vycházející z předcházejícího kroku (případně z počáteční permutace) Výstupem jsou upravená data R_i,L_i a nový klíč K_i. Výstupní funkce jsou definovány takto.

$$L_i = R_{i-1}$$

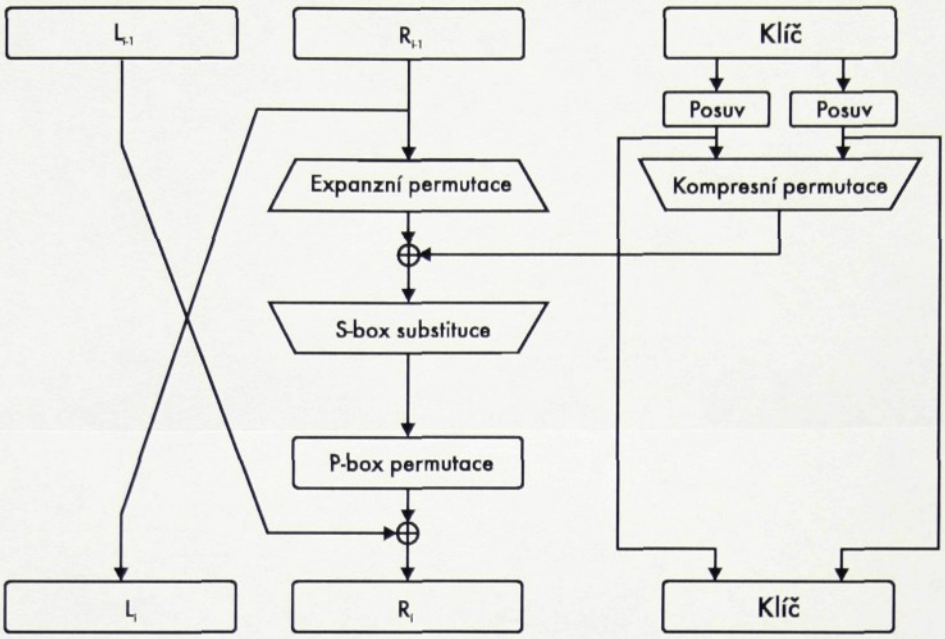
$$R_i = L_{i-1}\ XOR\ f(R_{i-1}, K_i)$$

Kde XOR je bitový součet modulo 2, f je šifrovací funkce a K_i je 48 bitový klíč vygenerovaný pomocí funkce KS z původního 64 (resp. 56) bitového klíče.

Šifrovací funkce f provede na bloku R expanzivní permutaci (tabulka permutace je opět definována normou) výsledek se bitově sečte modulo 2 s klíčem K. Na tento blok je aplikována S-Box substituce a poté ještě P-Box permutace. S-Box substituce je složena z osmi dílčích funkcí S_1 - S_8.



Obrázek 3.1: schéma algoritmu DES



Obrázek 3.2: schéma šifrovacího kroku

Také tyto funkce jsou definovány normou. Každá S funkce podle 8 bitového vstupu určí 4 bitový výstup tak, že prvním a posledním bitem adresuje řádku a prostředními 4mi bity adresuje sloupec v příslušné tabulce, kde je uvedena výstupní hodnota.

Dešifrování se provádí stejným postupem, pouze klíče K_i se použijí v obráceném pořadí. Celý algoritmus je navržen tak, že změna jediného bytu na vstupu ovlivní téměř všechny výstupní bity. Proto se algoritmus někdy používá pro generování náhodných čísel.

3.3.3 Bezpečnostní rizika

Jak už bylo uvedeno současné možnosti počítačů umožňují překonat algoritmus DES prohledáním klíče. Další metoda překonání DES algoritmu je opět popsána v [BezUaI], jedná se o takzvaný útok „se setkáním uprostřed". K provedení tohoto útoku je sice zapotřebí získat část zašifrovaného a nezašifrovaného textu, ale poté výrazně redukuje počet pokusů pro prohledání prostoru klíče, z původních 2^{212} na 2^{57} pokusů. Protože je potřeba ukládat si značné množství mezivýsledků (2^{56}), je tato metoda překonání spíše teoretická. Touto metodou je ovšem teoreticky možné překonat i dvojitý DES. Naprosto bezpečným je tedy pouze trojitý DES, který je odolný jak proti prohledání prostoru klíče, tak proti útoku se setkáním uprostřed. Data zašifrovaná tímto postupem jsou zabezpečena stejně, jako kdyby se při jed-

noduchém DES algoritmu použil klíč o délce 112 bitů. Což je velikost klíče, jaká byla navrhována v původním algoritmu Lucifer. Pokud bychom měli počítač, který za jednu sekundu dokáže provést 10^{12} pokusů, potřeboval by na překonání klíče $1.65\ 10^{14}$ let (odhadované stáří vesmíru je „pouhých" 10^{10} let). V systému Kerberos V je možno nastavit pro kódování lístků všechny tři metody použití algoritmu DES. Pro použití systému v místech, která patrně nebudou čelit soustředěnému útoku hi-tech protivníků, jako jsou například školy, je ale jednoduchý DES dostatečně bezpečný.

4 Útoky na počítačové sítě

4.1 Proč dochází k útokům

Obsahem následujícího odstavce nebude filozofická úvaha o důvodech, které ženou lidi ke zločinu. Pokud si ale předem promyslíme, co asi tak může útočníka zajímat, můžeme si lépe naplánovat bezpečnostní strategii.

V prvopočátcích sítí a počítačů vůbec byly útoky (mimo jiné) motivovány snahou o získání přístupu k výkonnějšímu stroji. Přestože dnes již tento důvod zdaleka není hlavní stále přetrvává.

V současnosti jsou asi nejčastějším cílem útoků servery uchovávající informace. Cílem útoků může být zničení, modifikace nebo odcizení těchto informací. Čím více by se za takové informace dalo získat peněz, tím častější jsou pochopitelně útoky. Oblíbeným cílem útoků jsou například databáze obsahující čísla kreditních karet.

Další možností je snaha o získání falešné autentifikace. Pod identitou cizí osoby pak může útočník provádět další pokusy o průniky, aniž by se musel obávat o své odhalení.

V následujících odstavcích si popíšeme, jak k některým útokům dochází.

4.2 Nejčastější způsoby napadení

4.2.1 Odposlech sítě

Drtivá většina současných sítí používá technologie Ethernet. Jednou ze základních vlastností této technologie je všesměrové vysílání. Datagram se dostane ke každému počítači v síti, stanice ale ignorují datagramy určené někomu jinému. Má-li uživatel možnost přistupovat (softwarově) k hardwaru (například počítač s DOS/Windows9x), může pomocí jednoduchých programových prostředků sledovat veškerou komunikaci v síti. Protože v současné době není většina provozu nad TCP/IP nijak šifrována, je velmi jednoduché odposlouchávat hesla (telnet, rlogin), zachytávat přenos souborů (FTP, NFS) atd. Útočníkem bývá většinou některý lokální uživatel. Z toho plyne nutnost zabezpečovat síť nejen proti útoku z venku, ale i proti útoku zevnitř.

Odposlechu sítě lze předcházet její vhodnou topologií. Počítače, které lze k odposlechu využít, můžeme umístit do samostatné podsítě. Komunikaci této „ nebezpečné“ podsítě s ostatními podsítěmi, lze pak filtrovat pomocí firewallu. Další ještě jednodušší možností je použití aktivních prvků - přepínačů (switch), které omezují všesměrové vysílání paketů po síti. Přepínač předáva pakety pouze do podsítě, ve které se nachází jejich příjemce.

4.2.2 Zjišťování hesla

Protože většina autentifikace je založena právě na schématu jméno/heslo, může útočník zjištěním hesla získat neomezený přístup do systému. Získá-li heslo privilegovaného uživatele nebo dokonce superuživatele, jde v podstatě o konec systému. Proto je mnoho útoků směřováno právě k získání hesla a jeho následnému využití.Způsobů, kterými se může pokusit získat nějaké heslo, má útočník k dispozici několik.

- **Nalezení účtů bez hesla**. Heslo sice útočník nezíská, přístup však ano. Špatně nastavený systém může umožnit zadání prázdného hesla. Někteří správci systémů dokonce vytváří nové účty, aniž by jim přidělili heslo. Ochrana proti těmto útokům je jasná: kontrolovat existenci účtů bez hesla[1], nastavit možnost volby hesla tak, aby bylo možné zvolit pouze heslo bezpečné.

- **Odpozorování hesla**. Velmi jednoduchý útok, stačí ve vhodnou chvíli koukat někomu přes rameno, probrat jeho osobní poznámky. Ochrana je jednoduchá - hesla si pouze pamatovat (nezapisovat) a sledovat kdo stojí za námi. Přesto, že se zdá tento útok triviální, není dobré ho podceňovat.

- **On-line útok hrubou silou** je další jednoduchý útok. Útočník se snaží postupně vyzkoušet všechny kombinace hesla (zadávaná většinou automaticky). Ochrana je opět poměrně jednoduchá. Po špatně zadaném heslu systém určitou chvíli čeká. Tak jak se zvyšuje počet špatně zadaných hesel, zvyšuje se i časová prodleva. Možnost vyzkoušení všech kombinací se tak stává nereálnou z časových důvodů. Jiná možnost ochrany: po n-tém špatném zadání hesla dojde k zablokování účtu, například na 24 hodin. Tato možnost není příliš vhodná, protože je možno ji využít pro úmyslné zablokování konkrétního účtu. Díky záznamům o špatných loginech je možné zjistit odkud on-line útok přišel a případně se proti němu bránit například filtrovacím pravidlem firewallu.

- **Off-line útok hrubou silou** . Je všeobecně známo, že hesla se na disku ukládají zašifrovaná. Podaří-li se někomu získat soubor s databází hesel, může se pokusit o jeho rozšifrování. V kapitole o šifrování je uvedeno, že hesla jsou šifrována pomocí jednosměrné funkce. Nelze je tedy dešifrovat jinak než hrubou silou. Oproti předchozímu případu jsou ovšem šance útočníka výrazně vyšší. Kromě prostého zkoušení všech kombinací, které je přece jen poměrně náročné, má možnost použít tzv. *slovníkový útok*. Při něm se místo náhodných kombinací zkouší

[1]Tyto účty se také označují jako JOE accounts. Pochopitelně mezi ně nepatří účet guest nebo jiné omezené účty.

slova obsažená ve slovníku určitého jazyka a některé variace (například číslice na začátku/konci, počáteční velké písmeno atd.). Programů pro slovníkový útok je celá řada[2] a měl by je čas od času použít správce systému pro kontrolu uživatelských hesel. Ochrana: důkladně střežit soubor obsahující hesla, „donutit" uživatele k používání bezpečných hesel. K souboru obsahujícímu hesla by normální uživatelé neměli mít přístup, dále je potřeba zajistit, aby programy pracující pod identifikací superuživatele (FTP, httpd aj.) nemohly být zneužity k získání tohoto souboru.

- **Trojský kůň** je modifikace běžného programu, která navíc vykonává nějakou skrytou činnost. Modifikovaný program `login` tak z pohledu uživatele pracuje úplně normálně, jen zadané heslo ještě zapíše nebo někam odešle. Tyto programy se mohou objevit zejména na nezabezpečených stanicích, kde má útočník možnost modifikovat programové vybavení. Například v systému DOS se dá takový kůň realizovat pomocí rezidentního programu. Ochrana proti těmto útokům je složitější. Pravidlem by mělo být, že pokud je stanice potenciálně nebezpečná, neměl by se na ní nikdy přihlašovat privilegovaný uživatel nebo superuživatel. Za potenciálně nebezpečné lze považovat jak stanice se systémem DOS tak Unixové stroje, které lze restartovat v jednouživatelském režimu.

Pokud je systém dobře nakonfigurován a spravován a pokud všichni uživatelé používají „bezpečná" hesla, lze se útoků zaměřeným na získání hesla dobře bránit. Pochopitelně čím větší počet uživatelů má účet na daném serveru, tím větší je možnost, že některý z nich bude mít špatné heslo. Proto by zejména na serverech, které jsou nejvíce „viditelné" z Internetu (například WWW server), mělo být co nejméně účtů. Další vhodnou ochranou je použití jednorázových hesel, které ale není vždy technicky možné.

4.2.3 Programové chyby a zadní vrátka

Nejčastěji se útočník dostane do zabezpečeného počítače, využije-li některou známou či neznámou bezpečnostní chybu v běžícím programu. Většina strojů běžících pod systémy typu UNIX (a tedy většina serverů na Internetu), používá pro základní programy stejný zdrojový kód. Ten si může prostudovat kdokoliv, tedy i schopný útočník. Nalezne-li v programu nějakou chybu, může ji zneužít ve svůj prospěch. Programové chyby se například vyskytovaly v programech `sendmail`, `fingerd`, `telnetd`. Většina těch nejznámějších chyb byla ale postupně nalezena a opravena. Pokud se objeví nová chyba, většinou na ní někdo upozorní a v brzké době jsou vydány opravné záplaty.

[2]Například program Crack od Aleca Muffleta

Při ladění nového programu jsou do něj velmi často implementovány kódy, které vlastní ladění usnadňují (například tím, že nevyžadují autentifikaci apod.). Těmto kódům se říká „zadní vrátka", jedná se většinou o vložení krátké sekvence, které umožní převzít kontrolu nad činností programu. Klasickým příkladem uvedeným v [BezUaI] jsou zadní vrátka programu `sendmail`, pomocí kterých se v roce 1988 šířil internetový červ. Zadní vrátka mohou v programu zůstat neúmyslně, ale mohou být pochopitelně implementována záměrně.

Nejlepší ochrana před tímto způsobem útoků je používání stabilních verzí prověřených programů. Vyhnout bychom ze měli programům z neznámých zdrojů, alespoň v případech, kdy nejsme schopni pochopit jejich zdrojový kód. V případě tvorby vlastních programů, které poběží na internetovém serveru, bychom měli dodržovat následující rady:

- Program by měl používat co nejnižší práva. Pokud je to možné neměl by používat práva superuživatele. Pokud je program potřebuje, je vhodné, aby se jich vzdal hned po ukončení akce.
- Zdrojový kód programu by měl být co nejstručnější a nejpřehlednější. V takovém kódu se pak lépe hledají chyby.
- Je potřeba dát pozor na chyby ve standardních knihovnách. Tyto chyby je nutné ošetřit v programu.
- Pokud program používá pevně alokovanou paměť, je potřeba ošetřit přetečení bufferu (takovou chybu například obsahoval `fingerd`). Vhodnější je použití dynamické alokace paměti, i zde je ale potřeba počítat s možným přeplněním paměti nebo disku (a to i při současných kapacitách).
- Velmi přesně je třeba ošetřit komunikaci programu s okolím, tak aby nemohlo dojít k jeho zhroucení při komunikaci.
- Program by měl být velmi důkladně otestován, než bude nasazen na serveru. Dvojnásob to platí, chceme-li program poskytnou třetí osobě.

4.2.4 Útok zablokováním služeb

Takto skupina útoků se výrazně odlišuje od předchozích. Útočník se nesnaží o průnik do systému, ale o vyřazení některé části systému z provozu. Tyto útoky mají v zásadě dvojí charakter: první skupina útoků směruje k přetížení určité služby, aby nemohla být k dispozici ostatním uživatelům. Druhá skupina útoků je směřována k destrukci nebo poškození systému tak, aby ho nebylo možné používat.

Útoky přetížením služby mohou přicházet jak ze sítě (přetížení síťové rozhraní) tak přímo ze serveru (má-li k němu přístup více uživatelů).

- Lokálně lze server přetížit opakovaným spouštěním procesů nebo přeplněním diskového prostoru. Proti těmto útokům se lze naštěstí efektivně bránit pomocí uživatelských omezení. V moderních operačních systémech lze nastavit jak diskové kvóty uživatelů, tak maximální počet procesů s jedním UID. Poměrně dobře lze také vypátrat útočníka. Tyto útoky nemusí být vždy způsobeny úmyslně uživatelem, může je způsobit i programová chyba.

- Útoky ze sítě spočívají v nadměrném zatížení síťového rozhraní. Pokud server vyřizuje velké množství žádostí, hromadí se přicházející žádosti ve frontě. Ani její kapacita není nekonečná, takže po čase dojde k tomu, že se žádosti začnou ztrácet. V případě TCP se budou žádosti opět vracet, což povede k dalšímu přetěžování. V konečném důsledku to vede k zablokování přístupu k určité službě. K těmto útokům lze také využít programové chyby. Velmi známá je například chyba zvaná *ping of death*, která byla odhalena až po vydání systému Windows 95. V tomto systému je totiž možno odeslání paketu *ICMP echo request* o nestandardní délce. Většina tehdejších ICMP systémů neověřovala délku paketu. Přijetí paketu o chybné délce vedlo pak ke zhroucení celého systému. Je zajímavé, že délku paktu neověřoval ani systém Windows 95.

Rovněž destruktivní útoky mohou být směřovány proti serveru i proti samotnému síťovému přenosu.

- Destruktivní útok proti serveru spočívá ve zformátování disku či smazání kritických systémových prostředků. Tomu lze zabránit vhodným nastavením systému a zejména ochranou superuživatelského hesla. Proti fyzickým útokům lze server chránit vhodným umístěním.

- Útočník se rovněž může pokusit o rušení síťového přenosu. Ať již fyzickým přerušením sítě (velmi snadné například u starých ethernetových rozvodů po koaxiálním kabelu,stačí jen odstranit terminátor). Umístění silného zdroje elektromagnetického vlnění do blízkosti nestíněného kabelu může výrazně ovlivnit kvalitu přenosu.

Útokům zablokování služeb lze předcházet vhodným nastavením systému, rozdělením sítě na několik podsítí, vhodným umístěním počítačů a používáním vhodného HW (záložní zdroje apod.). Pro ochranu před programovými chybami typu ping of death, je opět vhodné používat aktuální verze programů.

4.3 Jak se bránit

Již v úvodu této práce je uvedeno, že 100% bezpečný systém neexistuje. Protože možná obrana byla většinou popsána u konkrétního útoku, je ná-

sledující odstavec jakýmsi shrnutím toho nejdůležitějšího.

1. Na každém počítači provozovat (tedy nabízet okolním počítačům) pouze ty služby, které jsou nezbytně nutné. Čím více služeb počítač nabízí, tím více je možností pro útok.

2. Každý účet by měl mít bezpečné heslo. Pokud vlastní uživatel více účtů a není zprovozněn centrální autentifkační systém, měl by mít na každý stroj jiné heslo. Systémové účty, které nejsou používány k práci, ale jsou nutné pro chod systému (lp,bin,nobody), by měly mít zakódované heslo (*), aby se na ně nedalo přihlásit. Nesmí existovat běžné účty bez hesla.

3. Pro nabízené služby používat aktuální a stabilní verze programů. Aplikovat bezpečnostní záplaty. Před přechodem na novou verzi důkladně prostudovat, jaké změny přináší.

4. Nepoužívat nekryptované služby jako je `telnet`, `rlogin` a podobně. V současné době jsou k dispozici bezpečné náhrady jako secure shell (SSH). Zabrání se tak zbytečnému prozrazení hesel při odposlechu.

5. Nepoužívat zbytečně komplikované programy, pokud je lze nahradit jednoduššími. V jednoduchém programu budou spíše odstraněny všechny chyby. Například `sendmail` lze nahradit zjednodušenými variantami `zmailer`, `qmail`.

6. Minimalizovat počet služeb, které běží pod právy superuživatele. Řada běžných služeb např. (`finger`) může běžet pod právy běžného uživatele. Dále minimalizovat počet SUID a SGID programů.

7. Důsledně zaznamenávat všechny přístupy k počítači. Tyto záznamy - *logy* například mohou odhalit zdroj útoků. Pokud se podaří průnik do systému, máme záznam o tom, kdy se tak stalo (pokud po sobě útočník tzv. *nezamete*).

8. Nepoužívat v síti stanice, které lze využít k odposlechu (např. PC s DOSem) pokud je to možné.

9. Omezit přístup k portům serveru například použitím firewallu. Tím lze částečně zabránit v průzkumu vaší sítě.

10. Pravidelně zálohovat systém. V případě porušení systému lze obnovit původní stav.

11. Důležité systémové programy, které se nemění opatřit kontrolním součtem. Velmi snadno tak odhalíme jejich případnou modifikaci.

Aplikace výše uvedených pravidel může napomoci ke zvýšení bezpečnosti systému. Protože každý server má své specifické určení, je vždy jeho bezpečnostní konfigurace trochu jiná. Z prostorových důvodů se zde tedy nebudeme zabývat podrobnostmi.

Celou bezpečnostní problematiku lze zjednodušeně rozdělit na útoky lokální a útoky z vnější sítě. Proti útokům z vnější sítě se jako ochrany používá (kromě jiného) zvláštních serverů nazvaných firewally.

4.4 Firewally

Do počítačové terminologie se tento termín dostal ze stavebnictví. Označuje protipožární zeď, která má zabránit šíření ohně do dalších částí budovy. V knize [LIS] je použit celkem odpovídající překlad „hradba". Pokud zajdeme v alegorii trochu dále, můžeme si naší síť vybavenou firewallem představit jako středověký hrad, obehnaný vysokými hradbami. Ty sice spolehlivě chrání obyvatele před útokem, zároveň jim ale poněkud omezují pohyb. Podobně je tomu i ve světě počítačů. Firewall je speciální server, přes který přechází všechny datagramy směřující dovnitř i ven. Na jedné straně zabraňuje útoku, na druhé ovšem omezuje přístup k Internetu uživatelům lokální sítě. Snad v každé literatuře, která se o firewallech zmiňuje najdete výrazné varování ve smyslu.

> *Firewall nikdy nesmí nahrazovat ostatní bezpečnostní pravidla. Slouží pouze jako jejich doplňek.*

Toto varování má svůj význam. Žádná hradba přece není nepřekonatelná, záleží jen na útočníkovi zda si přinese dlouhý žebřík nebo se podkope. Pokud pak uvnitř hradu nenarazí na odpor (máme přece pevné hradby), následky jsou pro obránce fatální. Použití firewallu by tedy mělo být vždy kombinováno s použitím ostatních bezpečnostních pravidel. Případně s vhodným autentifikačním systémem jako je Keberos nebo DCE.

Firewally lze podle způsobu práce rozdělit na dvě základní skupiny. Jsou to datagramové a aplikační firewally. Podívejme se nyní, jaký je mezi nimi rozdíl.

4.4.1 Datagramové firewally

Činnost tohoto firewallu se dá přirovnat k filtru. Někdy se také používá termínu *propust*. Toto zařízení na základě předem definovaných pravidel propustí pouze některé datagramy. Pro volbu pravidel existují dvě protikladné strategie:

- *Vše je zakázáno.* V tomto případě projdou do sítě pouze pakety z povolených hostů. Ostatní pakety budou zahozeny. Obdobný princip platí i pro protokoly. Přenos je umožněn pouze pomocí povolených protokolů.

- *Vše je povoleno.* Opak předchozího. Projdou všechny pakety, zahozeny jsou pouze vybrané. Totéž platí pro protokoly.

Každá z těchto strategií má své výhody a své nevýhody. Lépe se konfiguruje druhý případ, a rovněž pro uživatele sítě za firewallem je příjemnější. Naopak zákaz všeho může být vhodný, například pokud je za firewallem nějaký důležitý server s omezeným přístupem. Ani jedna strategie ovšem není dokonalá.

Výhody datagramových firewallů jsou v pružnosti konfigurace a snadnější implementaci. Většinou lze využít stávajících směrovačů, na kterých se aktivují filtrovací pravidla. Ty lze aktivovat jak na směrovačích profesionálních (např. od firmy CISCO systems apod.), tak na běžném PC s Linuxem, které slouží jako směrovač. Pružnost konfigurace spočívá ve snadné modifikovatelnosti filtrovacích pravidel. Pokud zjistíme, že se do naší sítě někdo opakovaně pokouší probourat, lze filtrovacím pravidlem bez problémů zakázat přijímání jeho paketů. Obdobně můžeme zakázat přijímání paketů z některých hostů, vadí-li nám například obsah WWW stránek, které jsou na nich uloženy[3].

Nevýhodou jsou poněkud omezené možnosti, problémy s některými protokoly, které využívají více portů s vysokými čísly (např. FTP, Xwindows), nelze jednoduše a bezpečně otestovat korektnost nastavení.

4.4.2 Aplikační firewally

Princip činnosti aplikačního firewallu je úplně odlišný. Nepropouští se vůbec žádné pakety, pokud je dané spojení povolené, firewall ho sám naváže. Velmi často s tomuto typu firewallu říká *brána.*

Pro každou povolenou službu musí být na bráně spuštěna zástupná aplikace (odtud aplikační), které se říká *proxy služba.* Pokud chceme přistupovat ke službě za firewallem, dojde nejprve ke spojení s proxy službou. Ta si může v některých případech dokonce vyžádat autentifikaci. Proxy služba za nás provede připojení k dané službě a předává její odpovědi. V případě HTML proxy může server kromě zajištění bezpečnosti sloužit zároveň jako cache pro stránky. To může přispět k větší rychlosti a zprůchodnění linek.

Výhodou aplikačních bran je větší bezpečnost než u datagramových filtrů. Přístup ke službám se totiž povoluje na základně autentifikace a nikoliv pouze adresy. Díky existenci speciálních aplikací lze zabezpečit i služby FTP nebo Xwindows, které pomocí filtrů ošetřit nelze.

Nevýhodou je větší výpočetní náročnost a tedy zpomalování přenosu. Pro každou službu je rovněž potřeba speciální aplikace. Z toho vyplývá nutnost častých aktualizací a tedy vyšší náklady.

[3]Pokoušet se tímto způsobem zabránit studentům v prohlížení pornografických a jiných stránek, je úkol hodný Dona Quijota

4.4.3 Pravidla pro nasazení firewallů

Nejvhodnějším způsobem je kombinace datagramového filtru s aplikační bránou. Pro HTTP, FTP, Xwindows aj. použijeme proxy služby a pro zbytek provozu definujeme filtrovací pravidla. To nám umožní používat i nové služby, pro které ještě neexistují potřebné proxy aplikace. Tato varianta je nejčastěji používaná u profesionálních firewallů.

Při zabezpečení počítače, na kterém firewall poběží, je třeba dodržet již zmiňovaná pravidla.

- Žádné uživatelské účty.
- Nespouštět žádné služby, které nejsou nutné pro provoz firewallu. Zejména vypnout „nebezpečné" služby jako `telnetd, NFS, NIS+`
- Všechny konfigurační zásahy provádět z lokální konzole nebo pomocí šifrovaného připojení z bezpečného počítače.
- Počítač by měl být vyhrazen pouze k účelu ochrany sítě, případně směrování.

Nasazení firewallů přinese zvýšení bezpečnosti, zároveň ale omezí lokální uživatele. Je proto vhodné důkladně promyslet, zda nepřinesou více omezení než užitku. Stejně důkladně je třeba promyslet vlastní topologii sítě. Vždy je ale potřeba mít na paměti základní pravidlo pro nasazení firewallů a nespoléhat se při zabezpečení sítě pouze na ně.

4.5 Autentifikační systémy

Potřeba zajištění bezpečnosti v sítích vedla k vytvoření zcela nové kategorie programového vybavení - autentifikačních systémů. Tyto systémy zajišťují ověřenou autentifikaci uživatelů i služeb a také ochranu dat při přenosech v síti pomocí jejích šifrování.

Autentifikační servery výrazně zvyšují bezpečnost sítí před lokálními útoky. Tím spíše pak před útoky z vnější sítě. Jejich intenzivnímu nasazení brání složitá instalace[4], slabá podpora některých operačních systémů a rovněž exportní omezení silného šifrování.

Mezi autentifikační systémy lze počítat secure RPC zmiňované na str. 8. Dále sem patří několikrát zmiňovaný systém Kerberos, jehož praktické možnosti zkoumá tato diplomová práce. Existují i další podobné systémy, uveďme si ty nejznámější, které jsou uvedeny v [BezUaI].

[4]Složitostí instalace je myšlena její rozsáhlost a nutnost velkých změn stávajícího systému. Vlastní instalace programů na počítače složitá být nemusí.

4.5.1 DCE

DCE znamená distributed computing enviroment. Systém byl vyvinut ve společnosti Open Software Foundation. Základní principy práce odpovídají systému Kerberos. Pro vlastní komunikaci mezi servery a klienty je využíváno autentifikovaného DCE RPC.

4.5.2 SESAME

Také tento systém vychází z principů systému Kerberos. Systém byl vyvinut v Evropě. Zkratka SESAME znamená secure european system for applications in a multi-vendor enviroment. Hlavním rozdílem oproti systému Kerberos je použití asymetrického šifrování.

4.5.3 Heimdal

Heimdal je v podstatě Kerberos V4. Tento systém byl naprogramován ve Švédsku. Důvodem byla faktická nedostupnost systému Kerberos mimo USA. Systém nabízí v podstatě tytéž vlastnosti. Tento systém byl a pravděpodobně dosud je používán v MetaCentru Masarykovy Univerzity v Brně. Tamější programátoři se také podíleli na vylepšování systému.

5 Kerberos

5.1 Co je to Kerberos?

V řecké mytologii byl Kerberos trojhlavý pes střežící, aby žádný živý nevstoupil do podsvětí. Dnešní Kerberos je síťový autentifikační protokol střežící bezpečnost počítačové sítě. Nezabrání sice, aby do ní nevstoupil nikdo živý, ale dokáže znepříjemnit činnost mnoha potenciálním škůdcům.

Kerberos byl navržen a je dodnes používán na Massachussets Institute of Technology (MIT). Tam byl v roce 1983 odstartován projekt s názvem Athena, jehož cílem bylo zapojení počítačů do výuky. Cílem byla v dnešní době celkem běžná věc: možnost pracovat na libovolném terminálu a přesto mít možnost pracovat se svými soubory na síti. Součástí návrhu tohoto systému byl i autentifikační systém, který zamezuje zneužívání počítačů neoprávněnými osobami.

5.2 K čemu slouží?

Většina administrátorů se snaží chránit vlastní síť před útokem z vnějšku. K tomu se využívají zejména firewally, o kterých už byla zmínka dříve. Mnohdy ale dochází k tak paradoxním situacím, že v okamžiku kdy útočník překoná firewall, najde zbytek sítě prakticky nechráněn. Nezanedbatelné procento útoků může, a podle [BezUaI] také přichází, zevnitř sítě. Systém je vhodný právě jako ochrana proti útokům zevnitř, ale stejně dobře chrání před útokem z vnějšku. Na rozdíl od firewallů totiž poskytuje Kerberos komplexní systém ochrany.

Chce-li útočník získat z počítačového systému nějaké informace nebo dokonce kontrolu nad celým systémem, potřebuje jednu maličkost - heslo. V předchozí části textu bylo popsáno několik způsobů, které může použít. Systém Kerberos se těmto útokům brání velmi zajímavým způsobem. Databáze hesel je uložena na jednom jediném serveru - autentifikačním serveru. Ten rozesílá svým klientům: uživatelům a serverům poskytujícím služby, identifikační lístky - vstupenky. Veškeré choulostivé síťové přenosy, tedy zejména vstupenky jsou šifrovány algoritmem DES. Navíc se po síti nepřenáší žádná hesla a to ani v zašifrované podobě.

5.3 Přihlášení uživatele

Přihlášení do systému používajícího pro identifikaci protokol Kerberos vypadá na první pohled stejně jako v případě standardní identifikace. Stanice zobrazuje výzvu k zadání uživatelského jména a hesla. Pokud uživatel zadá správné jméno a heslo, je mu umožněn přístup do systému. Proces ověření hesla se ale od klasického způsobu značně odlišuje:

1. V okamžiku, kdy uživatel zadá heslo, odešle stanice autentifikačnímu serveru žádost o přidělení identifikační vstupenky (ticket-granting ticket neboli TGT) tato žádost obsahuje uživatelské jméno a aktuální systémový čas zašifrovaný zadaným heslem.

2. Server žádost přijme a pokusí se vyhledat zadané jméno v databázi. Nalezne-li odpovídající záznam zkouší dešifrovat zašifrovaný časový údaj heslem, které nalezne v databázi. Pokud je dešifrovaný časový údaj v toleranci, vystaví identifikační vstupenku, kterou zašifruje uživatelským heslem a odešle ji zpět.

3. Stanice přijme identifikační vstupenku, dešifruje ji a uloží na disk. Obvykle do souboru `./tmp/krb5cc_<uid>`. Tento soubor slouží jako úložiště lístků po celou dobu jejich životnosti. V okamžiku úspěšného dešifrování se ztrácí heslo uživatele, veškeré další přenosy se šifrují klíčem sezení, který je z hesla vygenerován a který je součástí identifikační vstupenky.

V původní verzi systému Kerberos se heslo zadávalo až po přijetí identifikační vstupenky a používalo se pouze k jejímu dešifrování. Tento způsob identifikace lze ale prolomit tím, že zaslanou vstupenku se útočník pokusí dešifrovat off-line slovníkovým útokem. V současné verzi musí nejprve uživatel serveru dokázat, že heslo zná. Uživatel je zároveň chráněn před falešným autentifikačním serverem, protože ten pochopitelně nezná správné heslo.

Co vlastně je identifikační vstupenka (TGT) a k čemu slouží? TGT je v podstatě „občanským průkazem" sloužícím k další identifikaci uživatele. Pokud by neexistovala, bylo by nutné se při každém použití síťových služeb znovu identifikovat, čímž by se celý systém stal nepoužitelným. Identifikační vstupenka je složena ze dvou částí. Klíče sezení K_{sez} (session key) a vlastního identifikačního lístku (TGT), který je zašifrován jednak klíčem sezení a dále klíčem služby přidělování vstupenek (TGS). Pokud se nyní uživatel rozhodne použít některou síťovou službu, například chce kopírovat data ze svého domovského adresáře, probíhá identifikace automaticky pomocí TGT.

TGT má v sobě obsaženy rovněž identifikační údaje o stanici, ze které je uživatel přihlášen. Tím je lístek chráněn před zachycením a použitím na jiné stanici. Ve chvíli, kdy uživatel ukončí sezení, systém smaže všechny jeho lístky včetně TGT. Další uživatel, který stanici použije, je tedy nemůže zneužít. Rovněž životnost TGT je omezená na 8 hodin. Tím se má zamezit pozdějšímu použití TGT, který mohl někdo zachytit a počkat až uživatel odejde.

Princip používání TGT je podobný způsobu identifikace v reálném životě, při návštěvě některých institucí nebo podniků. Ostraha u vchodu (autentifikační server) návštěvníka identifikuje, zajištěné údaje si uloží a vydá mu identifikační visačku (identifikační vstupenku). Návštěvník se potom

může pohybovat po objektu, aniž by se musel u každých dveří znovu a znovu představovat (tedy alespoň teoreticky).

5.4 Používání identifikační vstupenky

Představme si, že uživatel *Novák* si chce zkopírovat dokumentaci uloženou na serveru *Phobos*. Úspěšně se identifikoval a vlastní tedy identifikační vstupenku, tato vstupenka je uložena v ukládacím souboru /tmp/krb5cc_¡uid¿. Celý proces pak bude schematicky vypadat takto:

1. Uživatel *Novák* se podívá, zda vlastní lístek pro spojení se serverem Phobos. Pokud ho vlastní (lístek je uložen v odkládacím souboru) může se připojit přímo na server. Pokud ho nevlastní požádá o něj autentifikační server (AS).

2. Jakmile AS obdrží tuto žádost vygeneruje dvě kopie nového klíče sezení (session key). Tento klíč se bude používat pro přímou identifikaci mezi uživatelem a serverem.

3. Server vezme první kopii klíče, k ní přidá identifikační údaje serveru (IP adresa, Hostname apod. záleží na nastavení) a vzniklá data zašifruje pomocí klíče sezení, který je součástí TGT vlastněného uživatelem. Tím vzniká lístek A.

4. Druhou kopii klíče server doplní o identifikační údaje stanice, u které pracuje uživatel, vzniklá data pak zašifruje pomocí klíče serveru *Phobos*(service key). Tím vzniká lístek B.

5. Oba zašifrované lístky zašle AS uživateli.

6. Uživatel dešifruje lístek A pomocí svého klíče obsaženého v TGT. Tím získá identifikaci serveru *Phobos*. Tento lístek je uložen do odkládacího souboru.

7. Lístek B nedokáže uživatel dešifrovat, protože nezná klíč serveru (service key). Proto vytvoří třetí (nazvěme ho pro jednoduchost C) lístek, který obsahuje aktuální systémový čas a případné další údaje (například klíč pro šifrování další komunikace apod.). Tento třetí lístek zašifruje klíčem sezení, který získal z lístku A. Vzniklý lístek C a lístek B odešle serveru, jehož identifikační údaje získal také z lístku A.

8. Server *Phobos* dešifruje lístek B pomocí svého klíče (ten je uložen na serveru ve speciálním souboru). Po získání klíče sezení dešifruje lístek C a ověří si správnost času. Zároveň získá identifikační údaje stanice. Tím je identifikace hotova a může začít vlastní komunikace mezi serverem a uživatelem.

Časový údaj vložený do lístku C má zabránit zneužití lístku jeho zachycením a pozdějším použitím. Pro správnou funkčnost je nutná synchronizace časů. Aby se zabránilo komplikacím spojeným s možným zpožděním přenosu lístků, je možné nastavit určitou povolenou odchylku (například 3 minuty). Vstupenky i žádosti jsou stále šifrovány, takže jsou chráněny před odposlechem.

5.5 KDC - server pro přidělování klíčů

Systém Kerberos vykonává dvě hlavní činnosti. Identifikuje uživatele a přiděluje lístky. Pro každou činnost je vyčleněn zvláštní logický server. AS (authentication server) má na starosti identifikaci. TGS (ticket granting server) přiděluje vstupenky. Tyto dva logické servery jsou spuštěny na jednom fyzickém serveru. Tento server se označuje jako KDC (key distribution server). Tento server je bez nadsázky srdcem i mozkem celého systému.

Na tomto serveru jsou uloženy databáze uživatelů a hesel. Je tedy nutné, aby byl dokonale zabezpečen a to jak po stránce softwarové tak po stránce fyzické. Jeho důležitost je jednou z hlavních slabin systému. Pokud totiž dojde k vyřazení serveru z činnosti, je celý systém zastaven, neboť nemůže docházet k identifikaci. Proto je implementována možnost vytvoření sekundárních serverů. Tyto sekundární servery musí ovšem obsahovat kompletní kopie databází uživatelů a služeb a musí proto být zabezpečeny stejně jako primární server.

Při zabezpečování KDC platí stejná pravidla jako pří zabezpečování serveru. Tato pravidla jsou uvedena v kapitole o útocích. Server by měl být umístěn tak, aby k němu měly přístup pouze autorizované osoby, nejlépe střežen (ostrahou, zabezpečovacím zařízením) 24 hodin denně. Veškerá administrativa tohoto serveru by se měla provádět pouze z lokální konzole. Pokud je to jen trochu možné, nesmí server poskytovat žádné další služby.

5.6 Strukturování sítě

Je zřejmé, že autentifikační server si na nedostatek práce stěžovat nemůže. Pokud se jedná o malou síť nebo o velmi výkonný stroj patrně nezpozorujeme žádné problémy. Pokud se ale jedná o velkou a složitou síť, je vhodné rozdělit ji na menší oblasti (realms). Každá taková oblast má potom svůj vlastní autentifikační server.

Poslední verze systému se při strukturování vůbec nemusí omezovat na jednu instituci. Pokud například jako celou síť pojmeme všechny počítače v doméně .edu, budou jednotlivé oblasti představovat všechny subdomény .skola.edu, které budou vybaveny autentifikačním serverem. Tím se otevírá další oblast použití tohoto autentifikačního protokolu navrženého původně pro potřeby jedné univerzity.

V případě komunikace dvou počítačů v různých oblastech hovoříme o mezioblastní autentifikaci (cross realm authentication). Strukturu oblastí je vhodné promyslet dříve než začneme se samotnou instalací.

5.7 Kerberos a uživatelé sítě

Přestože princip autentifikačních mechanizmů se díky systému Kerberos radikálně změní, obyčejní uživatelé nemusí změny vůbec zpozorovat. Většinu operací totiž provede systém automaticky sám. Přesto musíme po nainstalování a spuštění systému předat uživatelům několik důležitých doporučení.

Především je nutné uživatele seznámit s principy přidělování a používání lístků. Pro práci s lístky slouží tři základní programy:

`kinit` - slouží pro získání TGT
`klist` - zobrazí obsah odkládacího souboru
`kdestroy` - smaže odkládací soubor

Na počátku sezení je automaticky spušten program `kinit`, který užvateli získá TGT. Obdobně je na konci sezení automaticky smazán odkládací soubor pomocí programu `kdestroy`. Také všechny ostatní upravené programy jako `telnet, ftp, rlogin apod.`, provádějí autentifikační procedury automaticky. Sami si ověří existenci lístku v odkládacím souboru a případně požádají o vydání nového lístku.

Vyjímkou jsou dlouhá sezení. Implicitní životnost lístků je osm hodin. Pokud je sezení delší musí uživatel po osmi hodinách použít programu `kinit` pro získání nového TGT. Tento mechanizmus má zabránit zneužití lístků třetí osobou po ukončení sezení.

Obyčejný uživatel není tedy přechodem na systém Kerberos nijak omezen a nemusí se ani učit ovládání nových programů. Přesto je určitě vhodné uživatele na přechod upozornit a umožnit jim vhodnou formou přístup k dokumentaci systému. Uživatelé by si také měli být vědomi zvýšeného rizika při vzdálování od přihlášeného počítače. Protože úroveň autentifikace je na počátku sezení vysoká, další procedury už probíhají automaticky, aby byla práce uživatele co nejpohodlnější. Pokud někdo umožní útočníkovi použít autentifikavaný počítač, je většinou sám označen za pachatele. Systémové záznamy a jednoznačná identifikace počítačů totiž umožňují velmi přesně odhalit původce nelegálních operací v síti. Argumentovat tím, že jsem si jen na chvilku odskočil, a mezitím můj počítač někdo používal, nemusí jako obhajoba stačit.

5.8 Podpora v různých operačních systémech

Kerberos bohužel vyžaduje velmi podstatný zásah do operačního systému. Tento protokol totiž vyžaduje, aby veškeré aplikace, které jsou využívány pro

práci se sítí (a budou tedy požadovat autorizaci), byly patřičným způsobem modifikovány. Této modifikaci se často říká „kerberizace" aplikace. Složitost tohoto procesu je závislá na složitosti vlastní aplikace. Úprava aplikací je popsána v dokumentaci dodávané se systémem. Je ovšem nezbytně nutné vlastnit zdrojové kódy aplikace, kterou chceme upravit. Získání původních zdrojových kódů je například u systému Windows nemožné (anebo velmi složité), v takovém případě je nutné napsat vlastní aplikace. Použitelnost systému Kerberos tím bohužel klesá.

Součástí instalačního balíku jsou zdrojové kódy nejdůležitějších upravených aplikací. Řada společností dodává systém Kerberos jako součást svých operačních systémů, protože zdrojové kódy byly dány volně k dispozici (pod volnou licencí MIT). Se systémem Kerberos se tak můžeme setkat ve většině modifikací systému UNIX. Masivnějšímu rozšíření bohužel dosud bránily exportní omezení USA. Doufejme, že se situace brzy vyjasní a dalšímu rozšíření systému nebude bránit v cestě žádná zákonná norma.

Značným problémem je v současnosti podpora systému Kerberos pod Microsoft Windows. Vzhledem k architektuře tohoto OS jsou veškeré zásahy do jeho struktury velmi komplikované. Jak už bylo uvedeno, je v tomto případě nutné napsat vlastní aplikace. To ale představuje značné programátorské schopnosti v oblasti Windows a hlubokou znalost činnosti systému Kerberos. I tento problém již byl vyřešen tvůrci systému. Na stránkách MIT jsou k dispozici instalační balíky i pro prostředí MS Windows, ale zatím jsou na ně stále uvaleny restrikce. MS podporuje systém Kerberos 5 v systému Windows 2000. Podle oficiální dokumentace by mělo vše fungovat bez problémů, příspěvky v konferencích ale většinou hovoří o opaku. Objevilo se ale několik programových záplat, které by měly problém vyřešit. Protože systém Windows 2000 nevlastním (jeho cena převyšuje hodnotu mého počítače), neměl jsem praktickou možnost vyzkoušet si jeho spolupráci se systémem Kerberos.

5.9 Nevýhody

Systém sebou bohužel přináší celou řadu nevýhod, které jeho použití velmi omezují. Nevýhody lze rozdělit do dvou velkých skupin:

- Každá síťová aplikace nebo služba musí být pro použití se systémem upravena. Pokud aplikaci neupravil někdo jiný, představuje to značné množství času a úsilí navíc. Služba musí být upravena na obou počítačích, které se zúčastní komunikace. Pokud chceme, aby se na zabezpečený server dalo přistupovat i z vnější sítě, z nezabezpečené stanice (například z domova), musíme na serveru provozovat jak upravenou tak neupravenou službu.

- Centrální autentifikační server je „Achilovou patou" systému. V pří-

padě, že server je z nějakého důvodu nedostupný, nikdo v celé síti nemůže pracovat. Na tomto serveru jsou rovněž uložena všechna hesla, která jsou navíc symetricky zašifrována klíčem serveru. Pokud by útočník prolomil klíč serveru je nutno změnit všechna hesla. Ochrana tohoto serveru musí tedy odpovídat jeho významu, vyžaduje množství času a prostředků.

- Odkládací soubor s lístky se nachází v adresáři `/tmp`, do kterého má přístup každý uživatel. Proto musí na každé stanici pracovat pouze jeden uživatel. Operační systém Unix ale umožňuje, aby na jedné stanici bylo přihlášeno více uživatelů. Pokud by k tomu došlo hrozí odcizení souboru s lístky a pozdější zneužití. Na toto nebezpečí je potřeba uživatele upozornit.

5.10 Zhodnocení systému

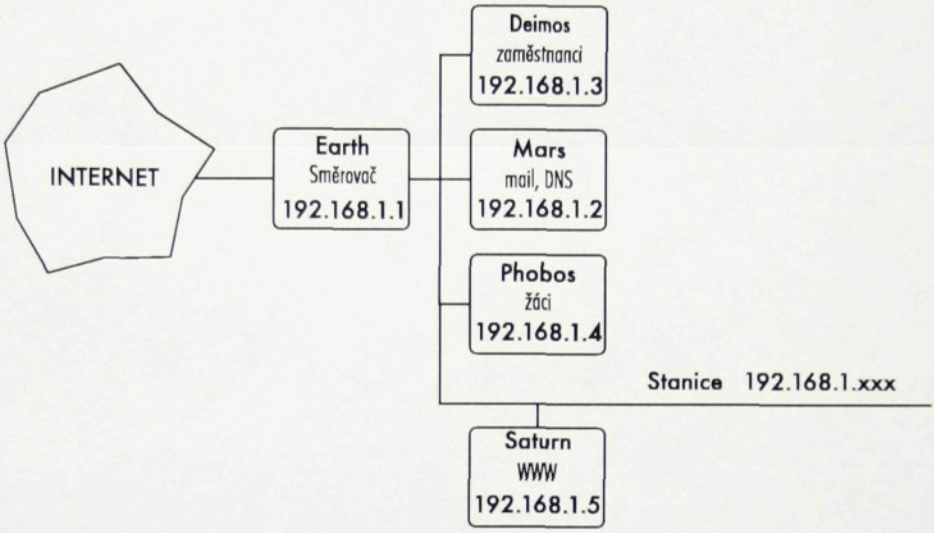
Centrální autentifikační systém výrazně zvyšuje bezpečnost sítě. Spolehlivě chrání proti odposlechům, protože veškerá citlivá komunikace je šifrována. Navíc nikdy nedochází k přenosu hesel a to ani v zašifrované podobě. Nasazení systému rovněž zabezpečí používání dosud nezabezpečených služeb jako `telnet` nebo FTP. Výše popsaná omezení ovšem značně zužují možnosti plnohodnotného nasazení systému.

Kerberos je vynikající náhradou za NIS. Základem obou systémů je stejná myšlenka, ale bezpečnost je v tomto případě mnohem lepší. Jak už bylo uvedeno díky známým zdrojovým kódům, je podpora klonů systému UNIX daleko širší. Na této platformě je za současných podmínek Kerberos nejlépe použitelný.

Přes zjevná omezení má ale systém Kerberos významné místo ve světě SW. Byl to první skutečně použitelný (a použitý) centrální autentifikační systém. Jeho principů využívá celá řada podobných systémů. Lze rovněž předpokládat že časem snad dojde k zrušení restrikcí ze strany MIT a systém bude možné nasadit na heterogenní sítě, ve kterých se využívá více operačních systémů najednou (což je pravděpodobně většina).

6 Návrh zabezpečené sítě

Tato kapitola je návrhem zabezpečení imaginární školní sítě, jejíž schéma vidíme na obrázku 6.1. Podobnou topologii by mohla mít například síť střední školy. Použité IP adresy patří do skupiny adres pro neveřejné sítě (viz. [RFC1597]). Při skutečné instalaci by pochopitelně tyto adresy nešly použít.



Obrázek 6.1: topologie sítě

6.1 Počáteční podmínky

Jak je vidět z počátečního schematu, jedná se o velmi jednoduchou síť, ve které jsou čtyři servery. Server Saturn slouží jako školní WWW server. Deimos obsahuje domácí adresáře zaměstnanců a citlivá školní data. Na serveru Phobos se nachází domácí adresáře žáků. Server Mars poskytuje ostatní důležité služby jako mailserver, DNS atd. Celá síť je k připojena k internetu, směrování zajišťuje server Earth. Škola používá doménu `skola.cz`.

Protože se jedná o imaginární síť, předpokládejme, že všechny stanice připojené do sítě jsou vybaveny operačním systémem Linux[1]. Dále předpokládejme, že síť je již částečně zabezpečena. Všechny stanice jsou nastaveny tak, že na nich neběží žádné serverové služby. WWW server je zabezpečen v souladu s běžnými pravidly. Rovněž na ostatních serverech nejsou provozovány nezabezpečené služby. Přesto lze bezpečnost sítě ještě zvýšit.

[1] Jsem si velmi dobře vědom, že tato možnost je čistě teoretická

6.2 Základní požadavky

Takovéto požadavky na zvýšení bezpečnosti nám předložilo imaginární vedení naší imaginární školy.

1. Zmenšit riziko poškození útokem z vnější sítě.
2. Zajistit, aby libovolný uživatel mohl používat libovolnou stanici.
3. Ochránit citlivá školní data také před útoky z vnitřní sítě.

6.3 Návrh řešení

Zabezpečení sítě si vyžádá poměrně velký zásah do stávajícího systému. Celé řešní si rozdělíme na několik kroků. Prvním krokem ke zvýšení bezpečnosti bude změna topologie sítě. Stávající síť rozdělíme na tři podsítě. Podsíť A vyhradíme pro servery, bude v ní zapojen server Saturn a server Mars. Podsíť B bude sloužit zaměstnancům školy a do podsítě C budou zapojeny počítače v učebnách, které používají studenti.

Pro zajištění druhého požadavku nainstalujeme v síti autentifikační systém Kerberos. To si ovšem vyžádá další server, který bude plnit úlohu KDC. Nasazením systému ale zvýšíme bezpečnost a zlepšíme pracovní podmínky všech uživatelů.

Nakonec nainstalujeme datagramový filtr na stávajícím směrovači Earth. Firewallu (filtru) využijeme jak pro zvýšení vnější bezpečnosti sítě, tak pro definici vnitřních pravidel. Definováním vnitřních omezení můžeme velmi snadno zabránit studentům v „průzkumech" zaměstnanecké části sítě.

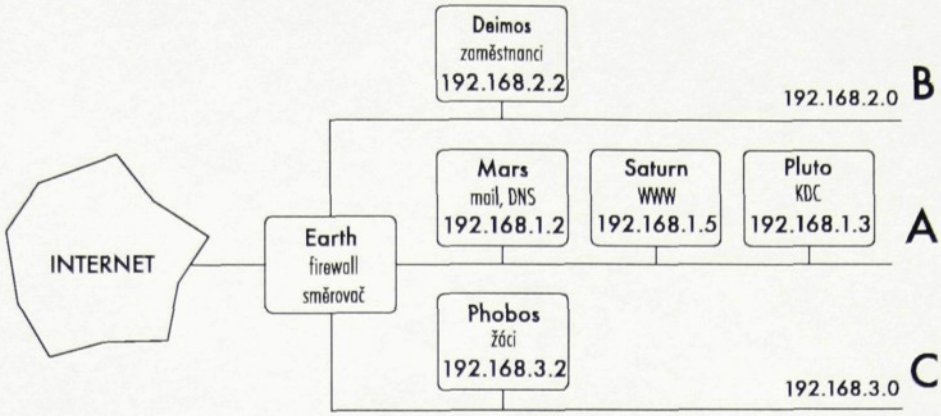
6.4 Bezpečnější topologie sítě

Rozdělením sítě zajistíme větší průchodnost a bezpečnost jednotlivých podsítí. Zároveň si připravíme dobrou výchozí pozici pro další úpravy. Novou topologii sítě popisuje obrázek 6.2.

Tento krok vyžaduje změnu stávajících IP adres. Jednotlivým podsítím přidělíme tyto adresy třídy C:

podsíť	adresa	maska podsítě
A	192.168.1.0	255.255.255.0
B	192.168.2.0	255.255.255.0
C	192.168.3.0	255.255.255.0

Ve skutečném případě bychom pravděpodobně neměli k dispozici dvě další adresy třídy C. Také vyčerpání celé třídy C pro podsíť, ve které bude nejvýše 5 počítačů (podsíť A) je nehospodárné. Méně náročné na počet adres by bylo



Obrázek 6.2: upravená topologie sítě

rozdělit původní adresu třídy C pomocí masek podsítí. Uvedené řešení je zde použito hlavně pro jednoduchost a transparentnost návrhu.

Vlastní změna topologie se realizuje úpravou stávajících směrovacích tabulek a překonfigurováním routeru Earth. Postup zde z úsporných důvodů uveden není. Konfigurace směrovače pracujícího pod OS Linux je popsána například v [LIS] na straně 31.

6.5 Instalace systému Kerberos

Celá instalace se dělí na několik částí. Instalace KDC serveru, instalace uživatelských stanic a instalace aplikačních serverů. Po vlastní instalaci je nutné zadat všechny uživatele do databáze KDC, což bude administrativně náročná záležitost. V případě školní sítě připadá takto náročný krok v úvahu pouze v období hlavních letních prázdnin.

Zdrojové kódy systému Kerberos lze získat v síti Internet například na WWW stránkách *Crypto-publishing project*. Adresa je uvedena v seznamu literatury jako [WWW 1]. Kerberos je také součástí některých Linuxových distribucí. Pro instalaci v naší síti použijeme distribuci Linux RedHat 6.2. Její součástí jsou jak předkompilované binární soubory, tak zdrojový kód. Pro vlastní instalaci budeme potřebovat tyto balíčky:

```
krb5-libs-1.1.1-9.i386.rpm
krb5-server-1.1.1-9.i386.rpm
krb5-workstation-1.1.1-9.i386.rpm
```

Instalace systému není složitá. Předpokládejme, že takovou instalaci bude provádět správce systému, tedy člověk, který má již nějaké zkušenosti s operačním systémem Linux. Naprostý začátečník by se raději do „ostré" instalace pouštět neměl. Hlavním zdrojem informací pro instalaci je [KrbIns] - instalační příručka systému Kerberos V5, která je součástí programového

balíčku.

6.5.1 Hlavní konfigurační soubor

Podobně jako u většiny Linuxových aplikací i u systému Kerberos se většina nastavení provádí pomocí konfiguračního souboru. Tento základní konfigurační soubor se jmenuje kdc5.conf a je obvykle uložen v adresáři `/etc`.

Konfigrační soubor se rozděluje na sedm základních sekcí:

libdefaults - definuje výchozí nastavení systémových knihoven

appdefaults - definuje výchozí nastavení systémových aplikací

realms - popisuje rozdělení systému na oblasti, je rozdělena na podsekce podle jednotlivých oblastí

domain_realm - definuje relace pro převod doménových jmen (systém DNS) na jména oblastí

logging - definuje pravidla pro vytváření záznamových souborů

capaths - určuje pravidla pro přímou mezioblastní autentifikaci

kdc - používá se pouze na KDC serveru, definuje například cestu k dalšímu konfiguračnímu souboru

Jednotlivé sekce nemusí být vyplněny všechny a nemusí ani následovat v uvedeném pořadí. Sekce začínají vždy identifikátorem uzavřeným v lomených závorkách například [realms]. Každá sekce obsahuje celou řadu definičních vztahů. Protože se jedná o poměrně velké množství voleb, popíšeme si pouze význam těch, které pužijeme při naší instalaci. Konfigurační soubor `/etc/kdc5.conf` bude v našem případě vypadat následovně:

```
[libdefaults]
ticket_lifetime = 24000
default_realm = SKOLA.CZ
clockskew = 180
[realms]
SKOLA.CZ = {
kdc = pluto.skola.cz:88
admin_server = pluto.skola.cz:749
default_domain = skola.cz
}
[domain_realm]
.skola.cz = SKOLA.CZ
skola.cz = SKOLA.CZ
```

V sekci [libdefaluts] jsme nejprve definovali maximální životnost lístků v sekundách. Dále jsme nastavili výchozí oblast. Protože máme malou síť, použijeme pouze jednu oblast, kterou nebudeme dále dělit. Proto bude na všech stanicích tento parametr stejný. Poslední parametr v této sekci nastavuje časovou toleranci. Pokud bude rozdíl porovnávaných časů větší než tři minuty (180 s), bude lístek prohlášen za neplatný.

Sekce [realms] definuje oblast SKOLA.CZ. Jak již bylo uvedeno použijeme pouze tuto jedinou oblast. Dále zde definujeme, který server slouží jako KDC a administrátorský server systému. V obou případech je to server Pluto, KDC používá port (TCP/UCP) 88, admin_server pak port 749.

Poslední sekce [domain_realms] definuje, jak jsou doménová jména přiřazena oblastem systému Kerberos. Díky použití jedné domény i oblasti je celá definice velmi jednoduchá.

Soubor /etc/kdc5.conf musí být na každé stanici připojené do sítě, máli tato stanice využívat systému Kerberos. Díky jednoduché struktuře naší sítě můžeme výše uvedený soubor použít na všech počítačích bez dalších úprav.

6.5.2 Instalace KDC serveru

KDC nainstalujeme na nový server, který se bude jmenovat Pluto. Na server nejprve nainstalujeme RedHat Linux 6.2. Vzhledem k budoucímu charakteru práce serveru nemusíme instalovat většinu balíčků. Pro bezpečnost celého systému autentifikace je nutné, aby tento server neposkytoval žádné další služby.

Pro instalaci KDC budeme potřebovat všechny výše uvedené balíčky. Ty se nainstalují do adresáře: /usr/kerberos. V tomto adresáři najdeme binární soubory modifikovaných aplikací pro stanice /usr/kerberos/bin a také pro server/usr/kerberos/sbin.

Nejprve upravíme soubor krb5.conf. K výchozímu tvaru souboru přidáme následující řádky:

```
[logging]
default = FILE:/var/log/krb5libs.log
kdc = FILE:/var/log/krb5kdc.log
admin_server = FILE:/var/log/kadmind.log
[kdc]
profile = /var/kerberos/krb5kdc/kdc.conf
```

Sekce [kdc] definuje cestu ke konfiguračnímu souboru serveru, který si zanedlouho popíšeme podrobně.

V sekci [logging] jsme nastavili umístění záznamových souborů. Všechny soubory budou umístěny v adresáři /var/log/. V těchto souborech jsou zaznamenány veškeré autentifikace, které server provedl. Dále nepodařené autentifikace a další operace serveru.

Konfigurace KDC se kromě souboru `krb5.conf` nastavuje také v souboru `kdc.conf`. Formát zápisu tohoto souboru je stejný jako u `krb5.conf`. Jsou zde ale pouze tři sekce. Podívejme se, co je v nich možné nastavit.

kdcdefaults - tato sekce je asi nejdůležitější, nastavují se zde důležité vlastnosti KDC serveru

realms - zde se opět nastavují různé vlastnosti jednotlivých oblastí

logging - v této sekci je možno nadefinovat soubory do kterých se zaznamenají přístupy k přímo k serveru, například při administraci

Soubor `kdc.conf` na serveru Pluto bude vypadat následovně:

```
[kdcdefaults]
acli_file = /var/kerberos/krb5kdc/kadm5.acl
admin_keytab = /var/kerberos/krb5kdc/kadm5.keytab
[realms]
SKOLA.CZ = {
master_key_type = des-cbc-crc
supported_enctypes = des-cbc-crc:normal des3-cbc-raw:normal
des3-cbc-sha1:normal des-cbc-crc:v4 des-cbc-crc:afs3
}
[logging]
kdc = FILE:/usr/local/var/krb5kdc/kdc.log
admin_server = FILE:/usr/local/var/krb5kdc/kadmin.log
```

V sekci [`kdcdefaults`] jsme nastavili umístění několika důležitých souborů. Soubor `kadm5.acl` obsahuje *access control list* - acl, tedy seznam uživatelů kteří jsou oprávněni modifikovat databázi. Dále je nastavena cesta k souboru `kadm5.keytab`, který obsahuje klíč pro identifikaci databáze. Dále jsme pro oblast SKOLA.CZ v sekci [`realms`] definovali hlavní typ šifrování a podporované šifrovací algoritmy.

V konfiguračních souborech lze samozřejmě nastavit ještě mnoho dalších parametrů. Většinou ale není nutné měnit hodnoty, které jsou nastaveny jako výchozí již od autorů programu.

Dalším krokem instalace KDC je vytvoření databáze serveru, ve které budou uložena hesla a jména uživatelů. Databázi vytvoříme příkazem:

```
/usr/kerberos/sbin/kdb5_util create -r SKOLA.CZ -s
```

Počítač požádá o zadání hlavního hesla databáze (Master password). Je asi zbytečné zdůrazňovat, že toto heslo by mělo být voleno jako bezpečné. Volba -s vede k vytvoření souboru, ve kterém je uloženo zašifrované heslo (stash file). Tento soubor je potřebný při automatickém startu KDC (během bootu), práva by měl mít nastavena tak, aby ho nemohl číst a modifikovat nikdo jiný než uživatel root.

Po vytvoření databáze musíme založit administrátorský účet databáze. Administrátoři databáze jsou zapsáni v souboru `kadm5.acl`. Cestu k tomuto souboru můžeme specifikovat při editaci souboru `kdc.conf`. Struktura souboru je velmi jednoduchá, každý řádek představuje jeden administrátorský účet, jeho omezení a pomocné volby. Kromě hlavního administrátora je vhodné vytvořit také další, kterým omezíme pravomoci (například nebudou moci měnit údaje uživatele root). Přesnou syntaxi souboru a jednotlivé volby popisuje dokumentace. Pro naše účely postačí jeden hlavní administrátor, řádek souboru pak bude vypadat následovně:

```
*/admin@SKOLA.CZ *
```

Teď už zbývá pouze účet přidat do databáze. To uděláme pomocí příkazů:

```
bash# /usr/kerberos/sbin/kadmin.local
kadmin.local: addprinc admin/admin@SKOLA.CZ
```

Opět budeme požádáni o zadání hesla pro administrátora. Jako poslední krok musíme k účtu vytvořit soubor `kadm5.keytab` který bude sloužit pro dešifrování lístků administrátora. Potřebný příkaz:

```
kadmin.local: ktadd -k /var/kerberos/krb5kdc/kadm5.keytab
=>[2]kadmin/admin kadmin/changepw
```

Tím je instalace KDC serveru v podstatě hotová. Můžeme spustit oba serverové daemony.

```
bash# /usr/kerberos/sbin/krb5kdc
bash# /usr/kerberos/sbin/kadmind
```

K dokončení vlastní instalace KDC je potřeba udělat ještě dvě věci.

1. Umožnit automatické zavedení serveru KDC (server ve smyslu program).
2. Omezit server tak, aby poskytoval pouze služby spojené s provozem systému Kerberos.

Pro zkušeného správce systému nepředstavují tyto operace nic složitého. Způsobů jak splnit oba požadavky je více. Uvedeme si zde ve stručnosti pouze jeden.

Pokud chceme oba daemony spouštět při bootu serveru (což je vhodné), můžeme zvolit z několika možností. Tradiční metoda je úprava souborů `/etc/inittab` a `/etc/rc.d/rc`. Velmi podobná je úprava souboru `/etc/rc.d/rc.local`, tento konfigurační skript je spouštěn v samotném závěru inicializace systému, po provedení všech ostatních scriptů. Do tohoto souboru přidáme následující řádky:

[2]Značka => znamená, že příkaz pokračuje na dalším řádku

```
# Zavedení KDC
/etc/rc.d/init.d/krb5server
```

Tyto řádky způsobí, že se na závěr inicializace systému zavolá skript `krb5server`, který zavede daemony systému Kerberos. Tento skript se do adresáře `/etc/rc.d/init.d/` uloží během instalace balíčku `krb5-server-1.1.1-9.i386.rpm`

Pro splnění druhého požadavku upravíme následujícím způsobem soubor `/etc/inetd.conf`. Server nebude poskytovat jiné služby než KDC a synchronizaci času.

```
# Tento počítač je autentifikační server systému Kerberos
(KDC).
#
# Služba pro synchronizaci času.
#
time stream tcp nowait root internal
time dgram udp wait root internal
# Služby systému Kerberos
#
krb5_prop stream tcp nowait root /usr/local/sbin/kpropd kpropd
eklogin stream tcp nowait root /usr/local/sbin/klogind
=> klogind -5 -c -e
```

Tím je instalace KDC serveru hotova. Protože naše imaginární školní síť je malá, nebudeme instalovat sekundární KDC servery. Ty totiž musí být zajištěny úplně stejně jako primární KDC. Další speciální server by také znamenal vyšší náklady.

6.5.3 Instalace stanic

Kerberizace stávajících stanic není nijak složitý proces. Pro instalaci nám postačí balíček workstation (`krb5-workstation-1.1.1-9.i386.rpm`). Po nainstalování balíčku přikročíme k úpravám konfiguračních souborů stanice.

Každá stanice v síti musí mít vlastní soubor `/etc/krb5.conf`. Obsah souboru na stanicích je stejný s obsahem souboru na KDC serveru (viz str. 43). Dále musíme upravit konfigurační soubor `/etc/services`, do kterého přidáme následující řádky:

```
kerberos        88/udp    kdc   # Kerberos V5 KDC
kerberos        88/tcp    kdc   # Kerberos V5 KDC
klogin          543/tcp         # Kerberos authenticated rlogin
kshell          544/tcp   cmd   # and remote shell
kerberos-adm    749/tcp         # Kerberos 5 admin/changepw
kerberos-adm    749/udp         # Kerberos 5 admin/changepw
eklogin         2105/tcp        # Kerberos auth. & encrypted rlogin
```

Tím je úprava konfiguračních souborů dokončena. Nyní musíme standardní

aplikace nahradit upravenými verzemi. Ty najdeme v adresáři `/usr/kerberos/bin`. Je potřeba ověřit, zda se tento adresář nachází v prohledávací cestě (měl by se tam automaticky přidat při instalaci balíčku). Pokud tam není, je nutné adresář přidat na začátek cesty, díky tomu se budou upravené aplikace používat místo neupravených. Součástí instalace jsou tyto upravené aplikace:

`FTP, gss-client,kdestroy, kinit, klist, kpasswd, krb524init, krlogin, krsh, ksu, rcp, rlogin, rsh, sclient, sim_client, telnet, uuclient, v4rcp, v5passwd`

Některé aplikace jsou přímou náhradou za původní, ostatní jsou nové. Záměnou aplikací je instalace stanic hotova. Při skutečné instalaci bychom pochopitelně museli provést školení uživatelů, tak jak je popsáno v části Kerberos a uživatelé (5.7).

6.5.4 Instalace aplikačních serverů

Před instalací aplikačního serveru je potřeba se rozhodnout, zda bude možné se na server přihlásit i z nezabezpečeného počítače (tedy počítače, který není autentifikován systémem). Pokud takováto přihlášení povolím, ztrácí server část svého zabezpečení. Někdy ale bohužel není jiná možnost. V naší imaginární instalaci můžeme povolit pouze zabezpečené přihlášení.

Je asi jasné že i na serverech musíme mít soubor `/etc/krb5.conf`. Podobně jako v předchozím případě potřebujeme balíček workstation. Součástí standardní instalace jsou čtyři modifikované daemony: `ftpd, klogind, kshd, telnetd`. Těmi nahradíme původní nezabezpečené služby.

Nejprve přidáme do prohledávací cesty adresář s těmito programy, obvykle je to `/usr/kerberos/sbin/`. Potom je nutné provést několik úprav v souboru `/etc/inetd.conf`. Nejdříve v souboru smažeme (zakomentujeme) všechny řádky týkající se služeb `ftp, telnet, shell, login, exec`. Tím jsme se zbavili nezabezpečených služeb. Přidáním následujících řádků zajistíme používání služeb autentifikovaných.

```
klogin    stream  tcp  nowait  root  /usr/local/sbin/klogind klogind -k -c
eklogin   stream  tcp  nowait  root  /usr/local/sbin/klogind klogind -k -c -e
kshell    stream  tcp  nowait  root  /usr/local/sbin/kshd kshd -k -c -A
ftp       stream  tcp  nowait  root  /usr/local/sbin/ftpd ftpd -a
telnet    stream  tcp  nowait  root  /usr/local/sbin/telnetd telnetd -a valid
```

Zbývá už jen vygenerovat soubor keytab pro identifikaci serveru. Pro tento soubor platí stejná zabezpečovací pravidla jako pro stash file KDC serveru. Pro generování souboru se používá program `kadmin`. Pro server Phobos, kde chceme používat služby host a ftp, bude syntaxe vypadat následovně:

```
bash# /usr/kerberos/sbin/kadmin
kadmin5: ktadd host/phobos.skola.cz ftp/phobos.skola.cz
```

Vytvořením klíčů serveru jsme dokončili instalaci systému Kerberos v naší pokusné síti.

6.6 Instalace datagramového firewallu

Datagramový firewall, společně se změnou topologie sítě, nám umožní lépe zabezpečit choulostivá školní data před útokem. K jejich zabezpečení pomáhá i autenitifikace systémem Kerberos, ale instalace filtru není nijak složitá a bezpečnostní sítě před útokem z vnějšku se ještě zvýší.

Vzhledem k již použitým zabezpečovacím prvkům nemusí firewall tvořit hlavní hráz. Dále by bylo vhodné, aby příliš neomezoval uživatele naší sítě. Proto pro návrh filtrovacích pravidel použijeme druhou strategii – vše je povoleno.

Nyní můžeme přikročit k návrhu filtrvacích pravidel. Navrhovaná pravidla a komentáře k nim jsou uvedeny v tabulce 6.1

Tato pravidla by zabránila přístupu ke každé potenciálně nebezpečné službě v naší síti. Tam kde je přístup nutný například u WWW serveru nebo mailserveru je pochopitelně povolen. Výhodou těchto pravidel je jejich snadná modifikovatelnost podle potřeby. Přidáním nebo odebráním pravidel lze zakázat/povolit nové služby.

Jak se pravidla zadávají záleží na použitém firewallu. Předpokládejme, že na směrovači Earth je nainstalován OS Linux. Abychom mohli použít Linux jako firewall, je nutné při konfiguraci jádra povolit volby `IP forwarding` a`IP firewalling`, případně zavést potřebný modul jádra. Pro zadávání pravidel se používá příkaz `ipfwadm` nebo `ipchains`, záleží na použité verzi jádra.

`Ipchains` je podporován od jádra verze 2.2. V základních principech je stejný jako starší programy, ale nabízí mnohem více možností. Pro uložení pravidel se používá zvláštních souborů označených *chains*. Při zadávání pravidla můžeme definovat, zda bude vstupní, výstupní nebo předávací. V programu `ipfwadm` to určuje přepínač. `Ipchains` používá pro každou skupinu pravidel zvláštní soubor (tyto soubory jsou implicitně nastaveny). Nabízí také možnost vytvoření vlastního souboru pravidel. Každý soubor může mít nastavenu vlastní politiku přijímání paketů (všechno povoleno/zakázáno). Více informací lze získat z manuálových stránek příkazu a z dokumentu [Firewall.HOWTO]. Většina pravidel, která jsou uvedena v tabulce, bude patřit do kategorie pravidel předávacích (forward). Vyjímku tvoří pravidla týkající se přístupu přímo k firewallu, ta budou patřit do kategorie vstupních (imput). První pravidlo z tabulky se zadá pomocí následujícího příkazu:

```
bash# ipchains -A forward -p TCP -s 0.0.0.0/0 -d 192.168.1.5
=>www -j ACCEPT
```

Stejným způsobem zadáme i všechna ostatní pravidla. Po zadání pravidel je firewall připraven k testovacímu provozu.

action	src	port	dst	port	flag	poznámka
allow	*	*	saturn	HTTP	TCP	přístup k WWW serveru
block	*	*	*	HTTP	TCP	jiný WWW server tu není
allow	síť B	*	saturn	ssh	TCP	admin. přístup k WWW
allow	síť A,B,C	*	pluto	kerberos	TCP	přístup ke KDC
allow	síť A,B,C	*	pluto	kerberos	UDP	přístup ke KDC
allow	síť A,B,C	*	pluto	kerb-adm	TCP	přístup ke KDC
allow	síť A,B,C	*	pluto	kerb-adm	UDP	přístup ke KDC
allow	mars	*	pluto	time	*	povolit synch. času KDC
block	*	*	pluto	*	*	ostatní přístup ke KDC zakázat
allow	mars	*	síť A,B,C	time	*	povolit synch. času
allow	*	*	mars	SMTP	TCP	přijímat poštu serverm mars
allow	mars	*	deimos	SMTP	TCP	deimos může převzít poštu
allow	mars	*	phobos	SMTP	TCP	phobos také...
block	*	*	*	SMTP	TCP	jinam se pošta nepřijímá
allow	*	*	mars	DNS	*	povolit nameserver mars
allow	síť A,B,C	*	*	DNS	*	a vnější nameservery
block	*	*	*	DNS	*	další nameservery zakázat
allow	síť B,C	*	síť B,C	NFS	UDP	povolnit NFS mezi B a C
allow	phobos	*	saturn	NFS	UDP	NFS ze saturnu na phobos
allow	deimos	*	saturn	NFS	UDP	a deimos
allow	phobos	*	mars	NFS	UDP	NFS z marsu na phobos
allow	deimos	*	mars	NFS	UDP	a deimos
block	*	*	*	NFS	UDP	ostatní NFS zakázat
allow	síť A,B,C	*	*	Xwin	TCP	povolit X v lok. sítích
block	*	*	*	Xwin	TCP	z vnější sítě zakázat
allow	síť A,B,C	*	síť A,B,C	ftp	TCP	povolit lokální ftp
allow	síť A,B,C	*	*	klogin	TCP	povolit Kerb. služby
allow	síť A,B,C	*	*	kshell	TCP	...
allow	síť A,B,C	*	*	eklogin	TCP	...
allow	síť A,B,C	*	*	telnet	TCP	...
allow	síť A,B,C	*	síť A,B,C	ftp-data	TCP	a přenos dat
allow	síť A,B,C	*	*	ftp	TCP	povolit ftp ven
allow	síť A,B,C	*	*	ftp-data	TCP	a přenos dat
block	*	*	*	ftp	TCP	ostatní ftp zakázat
allow	*	*	*	ssh	TCP	pro přístup povolit ssh
block	*	*	*	login	TCP	nezabezpeč. služby zakázat
block	*	*	*	shell	TCP	...
block	*	*	*	exec	TCP	...
allow	síť A,B,C	*	*	finger	TCP	povolit lokální finger
block	*	*	*	finger	TCP	z venku zakázat
allow	mars	*	earth	ssh	TCP	přístup na fw pouze z marsu
block	*	*	earth	*	TCP	...
block	*	*	*	1-1024	TCP	jiné priv. služby zakázat
block	*	*	*	1-1024	UDP	jiné priv. služby zakázat
allow	*	*	*	*	*	a vše ostatní povolit

Tabulka 6.1: Filtrovací pravidla

6.7 Zhodnocení návrhu

Takto zabezpečená síť by měla být odolná proti většině forem útoků. Všechny důležité služby a servery jsou zabezpečeny před zneužitím. Použití OS Linux dělá navržené řešení ekonomicky výhodným. Díky nízkým HW nárokům systému postačí jako server KDC i jako firewall běžné Pentium. Tedy počítač, který lze pořídit v částce pod 10000 Kč.

Návrh obsahuje v čechách nepříliš často používanou kombinaci zabezpečení pomocí autentifikačního systému a firewallu. Tato kombinace, využitím výhod obou prvků, ještě dále zvyšuje zabezpečení sítě. Podle dostupných informací je právě tato koncipováno zabezpečení počítačové sítě MIT, kde vznikl systém Kerberos.

Některé části návrhu (instalace systému Kerberos) byly ověřeny prakticky. Většinu návrhu, ale nebylo vzhledem k jeho charakteru možné otestovat v praxi. Použité postupy zabezepečení jsou ovšem platné a bylo by zcela určitě možné aplikovat je na konkrétní síť. Návrh by se v takovém případě musel pochopitelně upravit podle nových počátečních podmínek. Základní schéma návrhu zůstane ale stále stejné.

7 Závěr

Tato diplomová práce ve stručnosti shrnuje důležité bezpečnostní principy a pravidla. Většinu doporučení a rad lze použít nejen při zabezpečování sítě připojené do Internetu, ale i při zabezpečování malých nepřipojených sítí, dokonce i při zabezpečování osamělých počítačů.

Během tvorby práce se autor seznámil se systémem Kerberos a prohloubil si své znalosti o systému Linux. Systém Kerberos bohužel nesplnil všechna očekávání. Jeho hlavní nevýhodou je nedostatečná podpora některých operačních systémů. To značně omezuje možnosti jeho použití. Principy tohoto systému se však začínají uplatňovat i jinde. Lze také očekávat postupné odstranění nedostatků systému, zejména po uvolnění vývozních omezení v USA.

Práce nepostihuje všechny oblasti zabezpečení počítačových sítí. Vzhledem k rozsahu práce a neustálému rozvoji síťových technologií by to ani nebylo možné. Nové technologie sebou přinášejí nové bezpečnostní problémy, které bude nutné v budoucnu řešit.

Literatura

[BezUaI] Garfinkel S., Spafford G.: Bezpečnost v UNIXu a Internetu v praxi, Computer Press, Praha 1998, ISBN 80-7226-082-0

[LIS] Satrapa P.,Randus J.A.: Linux Internet server, Neokortex, Praha 1998, ISBN 80-902230-3-6

[KrbIns] Massachusets Institute of Technology: Kerberos V5 instalation guide, 1996

[Firewall.HOWTO] Grennan M.: Firewall and Proxy Server HOWTO v0.67, 1999

[WWW 1] Crypto-publishing project: http://www.crypto-publish.org/

[RFC1510] Kohl J., Neuman C.: The Kerberos Network Authentication Service (V5), RFC, 1993

[RFC1597] Rekhter Y., Moskowitz R., Karrenberg D., de Groot G.:Adress Allocation for Private Internets, RFC, 1994

[APZ] Borovanský J., Komárek P.,: DES kodér - semestrální úloha z APZ, FEL VUT Praha, 1999