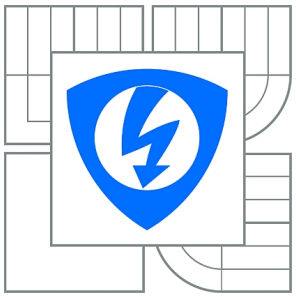
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

ZABEZPEČENÍ OPEN SOURCE PBX PROTI ÚTOKŮM

OPEN SOURCE PBX SECURITY AGAINST ATTACKS

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

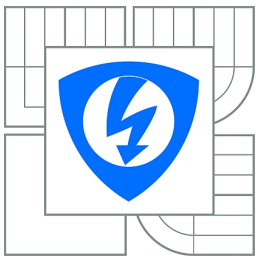
AUTOR PRÁCE — Bc. DAVID ORSÁK

AUTHOR

VEDOUCÍ PRÁCE — Ing. PAVEL ŠILHAVÝ, Ph.D.

SUPERVISOR

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bc. David Orsák	***ID:***	98609
Ročník:	2	***Akademický rok:***	2011/2012

NÁZEV TÉMATU:

Zabezpečení Open source PBX proti útokům

POKYNY PRO VYPRACOVÁNÍ:

Nastudujte problematiku útoků a možnosti zajištění bezpečnosti VoIP provozu a Open source PBX. Vytvořte nástroj pro experimentální generování různých typů útoků. Pomocí vytvořeného nástroje ověřte míru nebezpečnosti různých typů útoků. Proti jednotlivým útokům navrhněte opatření, s pomocí vytvořeného nástroje tato opatření ověřte a vyhodnoťte dle Vámi definovaných kritérií.

DOPORUČENÁ LITERATURA:

[1] Meggelen, J.V, Smith, J., Madsen, L. Asterisk™ The Future of Telephony. Sevastopol: O'Reilly Media, Inc., 2005. ISBN 0-596-00962-3.
[2] Jackson, Benjamin. Asterisk hacking :toolkit and liveCD / Burlington: Syngress, 2007. xii, 253 s. + ISBN 978-1-59749-151-8.
[3] Dwivedi, Himanshu. Hacking VoIP :protocols, attacks, and countermeasures / San Francisco : No Starch Press, 2009. xiii, 211 s. : il. ISBN 978-1-59327-163-3.
[4] Endler, David. Hacking exposed VoIP :voice over IP security secrets & solutions / New Yrok : McGraw-Hill, 2007. xxvii, 539 s. ISBN 978-0-07-226364-0.

Termín zadání:	6.2.2012	***Termín odevzdání:***	24.5.2012

Vedoucí práce: Ing. Pavel Šilhavý, Ph.D.
Konzultanti diplomové práce:

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT:

Tato diplomová práce pojednává o zabezpečení open source PBX ústředny proti bezpečnostním útokům. V rámci teoretické části je podrobně rozebrána problematika jednotlivých útoků použitelných na VoIP systémy se zvláštním důrazem na útoky typu odepření služby neboli Denial of Service. Dále jsou v teoretické části popsány metody zabezpečení inicializačního protokolu SIP. Samostatná kapitola je věnována detekci a prevenci průniku IDS a IPS se zaměřením na systémy Snort a OSSEC. V praktické části práce byl vytvořen generátor různých útoků proti PBX systémům, který byl následně použit pro podrobné testování. Podrobněji je pak PBX systém testován proti útokům typu DoS, proti kterým bylo vytvořeno zabezpečení ve formě aktivních prvků IDS Snort & OSSEC, které jsou schopny zabránit útokům v reálném čase. Toto zabezpečení bylo otestováno na jednotlivých ústřednách a v rámci porovnání byly změřeny možnosti systému před zabezpečením a po zabezpečení. Výsledkem této práce je generátor útoku VoIPtester a vytvoření konfiguračních pravidel pro IDS systémy

KLÍČOVÁ SLOVA:

PBX ústředna, inicializační protokol SIP, útoky na dostupnost služby, detekční systémy, Snort, OSSEC, VoIP, open source

ABSTRACT

This master's thesis deals with open source PBX security against security attacks. In the theoretical part is detailed description of problematic about attacks that could be used on VoIP systems with high focus on the Denial of Service attack. Furthermore are in theoretical part described methods of security of initialization protocol SIP. Individual chapter is devoted to intrusion detection and prevention of IDS and IPS systems, focusing on Snort and OSSEC. In the practical part of the work was created generator of attacks against various PBX systems, which was subsequently used for detailed testing. Special tests of PBX system are then used against DoS attacks, for which was created protection in form of active elements consisting of IDS Snort & OSSEC. These are capable to provide protection in real-time. The protection was tested on particular PBX systems and in matter of comparison were measured possibilities before and after of security implementation. The output of this work is attacks generator VoIPtester and creation of configuration rules for Snort and OSSEC.

KEYWORDS:

PBX system, SIP initialization protocol, Denial of Service, detection systems, Snort, OSSEC, VoIP

ORSÁK, D. *Zabezpečení Open source PBX proti útokům.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2012. 111 s. Vedoucí diplomové práce Ing. Pavel Šilhavý, Ph.D..

Prohlašuji, že svou diplomovou práci na téma zabezpečení Open source PBX proti útokům jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následku porušení ustanovení § 11 a následujícího autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestně právních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne . . . .

podpis autora

Děkuji vedoucímu diplomové práce Ing. Pavlovi Šilhavému, Ph.D., za velmi užitečnou metodickou pomoc a cenné rady při zpracování práce. Také bych chtěl poděkovat Hance Peterkové za její trpělivost a ochotu při korekci a úpravách práce po formální stránce.

V Brně dne . . . .

podpis autora

OBSAH

Seznam obrázků

Seznam tabulek

Úvod

Technologie VoIP (*Voice Over Internet Protocol*) patří mezi nejrychleji se rozvíjející komunikační technologie současnosti. Nepředstavuje pouze alternativu k veřejným telefonním sítím, jedná se o kompletní řešení, které může být využito jak ve firemní, tak v soukromé sféře. VoIP pracuje na otevřených systémech, využívá stávající IP sítě, standardní prvky a známé operační systémy. Díky tomu je VoIP technologie mnohem méně ekonomicky náročná v porovnání s klasickou telefonní sítí, není potřeba vytvářet novou infrastrukturu a je možné prakticky okamžitě vytvořit spojení mezi dvěma uživateli, kteří jsou od sebe vzdálení tisíce kilometrů. Toto je jedna z nesporných výhod, ale zároveň z použití stávajících datových sítí plyne mnoho nebezpečných aspektů, které se v běžné telefonní síti dříve nevyskytovaly nebo byly obtížně proveditelné. VoIP technologie s sebou přinesla bezpečností rizika, která jsou zaměřena přímo na telefonní hovory, ale také je nutno brát v úvahu všechny bezpečnostní problémy, které souvisejí s technologií IP. Jinými slovy řečeno, pokud dokáže někdo útočit na část sítě, může útočit také na VoIP. Jako nejznámější typ útoku, který lze provést v rámci VoIP sítě, je odposlech hovorů, jedná se o velmi jednoduchou záležitost, protože samotné hovory jsou často přenášeny v otevřené podobě a tím jsou dostupné všem, kdo mají k přenosovému kanálu přístup. Díky přechodu telefonní komunikace do počítačových sítí přibylo mnoho dalších útoků, které v rámci běžné telefonní sítě nebylo možné provést.

Srdcem každé sítě, která je založena na technologii VoIP, je PBX ústředna (*Private branch exchange*) sloužící jako centrální a zároveň jako výstupní prvek. Existuje mnoho variant PBX ústředen, jak komerčních řešení od velkých společností (CISCO, Siemens,…), tak zde také existují varianty, které jsou open source. Open source je otevřený počítačový software s veřejně dostupným zdrojovým kódem. Otevřenost zde znamená jak přístup k samotnému kódu, tak legální dostupnost (licence, která umožňuje využití zdarma). PBX ústředna bývá často terčem různých útoků, které mají za cíl negativně ovlivnit VoIP komunikaci. V tomto případě není podstatné zda-li se jedná o komerční řešení nebo o řešení open source. Útoky mohou být různého typu: jednak útoky typu DoS (*Denial of* Service) nebo útoky, které využívají nedokonalostí protokolu SIP. Cílem této práce je vytvořit zabezpečení open source PBX ústředny proti těmto útokům.

K samotné komunikaci je využívána řada protokolů, které zajišťují správu spojení od počátku až do konce hovoru. Jeden z nejvyužívanějších protokolů je inicializační protokol SIP (*Session Initiation Protocol*), jeho architekturu a princip komunikace popisuje kapitola 1. Existuje poměrně velká škála open source PBX systémů s podporou tohoto protokolu. Kapitola 2 popisuje tři vybrané zástupce této kategorie, jedná se o ústřednu Asterisk, YATE a FreeSWITCH.

Obecně lze útoky v IP telefonii rozdělit do několika kategorií: proniknutí do sítě, analýza provozu a shromažďování informací, útoky na dostupnost služby (*Denial of Service*), útok na integritu služeb a SPIT (*SPAM over IP Telephony*). Teoretický popis těchto kategorií je umístěn v kapitole 3. Tato práce podrobně rozebírá problematiku útoků typu DoS, které dokáží přetížit celý systém natolik, že je znemožněna komunikace mezi uživateli. Další, velmi problematickou částí v rámci VoIP, je nedokonalost SIP protokolu, která může být využita pro útoky na integritu služeb, jak je popsáno v kapitole 3.

Kapitola 4 popisuje možnosti zabezpečení pro inicializační protokol SIP. Protokol SIP je mocným nástrojem ve VoIP komunikaci, obsahuje ovšem řadu bezpečnostních slabin, které útočník může využít pro svůj prospěch. Z toho důvodu a obecně z důvodu bezpečnosti existují bezpečné varianty přenosu, které zajišťují důvěryhodnost, integritu a autenticitu v IP telefonii. Jedná se o protokol TLS (*Transport Layer Security*), který je možno implementovat v rámci SIP protokolu. Další možností, kterou lze využít společně se SIP protokolem pro zabezpečení je S/MIME (*Secure MIME*) protokol. Zabezpečení na nižších vrstvách síťového modelu řeší sada protokolů IPsec (*IP security*), jedná se o bezpečnostní rozšíření IP protokolu založené na autentizaci a šifrování každého IP datagramu. Problematika zabezpečení přenosu samotných dat je uvedena v kapitole 5, kde jsou popsány možnosti zabezpečení RTP protokolu jak pomocí šifrovaní SRTP (Secure Real-time Transport Protocol), tak zabezpečená komunikace výměny bezpečnostních klíčů ZRTP.

VoIP je komunikace v reálném čase proto je velmi citlivá na parametry QoS (*Quality of Service*). QoS je služba, jejichž cílem je doručit streamy, které jsou náročné na dobu přenosu, skrze síť tím nejefektivnějším způsobem. Ačkoliv neexistuje žádné pevné pravidlo, běžně se akceptuje, pokud signál putuje od volajícího k volanému pod 150ms. Pokud by tento čas překročil 300ms, bývá těžké se vyhnout rušení konverzace a pokud čas přesáhne 500ms, normální komunikace se stává téměř nemožnou. Podmínky za jak dlouho budou jednotlivé data doručeny, určuje několik faktorů, jedná se o šířku pásma dané sítě, ale také záleží na vytíženosti systému, který zpracovává požadavky. Jak již bylo uvedeno útoky, na které jsou PBX systémy obzvláště náchylné představují útoky typu DoS (*Denila of Service*) záplavového typu. Tyto útoky dokáží natolik ovlivnit parametry QoS, že spojení mezi uživateli bude prakticky nemožně. Kvůli tomuto faktu je potřeba vytvořit kvalitní zabezpečení nejen PBX ústředny, ale i pro systém obecně.

Zabezpečení proti těmto útokům je rozebráno v samostatné kapitole 6, která slouží jako popis systémů detekce a prevence útoků. Jedná se převážně o síťové analyzátory NIDS (*Network intrusion detection system*) a systémové analyzátory HIDS (*Host-based intrusion detection system*). Tyto dva nástroje představují velice efektivní řešení zabezpečení systému, aniž by byla ovlivněna funkce PBX ústředny.

Praktická část, zabývající se samotnými testy, je popsána v kapitole 7. Pro účely testování byl vytvořen generátor útoků VoIPtester, který umožňuje generovat různorodé útoky proti PBX systémům. Jedná se o útoky, které jsou zaměřeny na nedokonalost protokolu SIP, jako odchyt hesel v nezabezpečené komunikaci, nelegitimní ukončení hovoru nebo odstranění registrace hovoru. Další částí, kterou by se dalo považovat za stěžejní náplň této práce, je otestování všech ústředen jak se dokáží vyrovnat se záplavou paketů TCP SYN, UDP, ICMP a inicializačního protokolu SIP. VoIPtester nabízí dva nástroje k tomu účelu, díky nim byly provedeny detailní testy, které znázorňují, jak se dokáží ústředny s jednotlivými útoky vyrovnat. V druhé části kapitoly 7 je popsána implementace a zabezpečení pomocí NIDS a HIDS systémů proti těmto útokům a je zde otestována účinnost jednotlivých zabezpečení. Na závěr jsou shrnuty výsledky provedených útoků a uvedeny doporučené možnosti nastavení.

1. Inicializační protokol SIP

Inicializační protokol SIP (*Session Initiation Protocol*) je dle doporučení RFC 3261 --2 protokol, pracující na aplikační vrstvě. Umožňuje vytvořit, udržet a ukončit multimediální spojení. Jedná se o textově orientovaný protokol, který má do značné míry podobnou syntaxi jako HTTP protokol. SIP je inicializační protokol, který zajišťuje správu spojení. O přenos samotných dat se stará RTP protokol. Umožňuje jak UDP tak TCP spojení. SIP není limitován pouze na obsluhu telefonních spojení, ale lze ho také použít pro multimediální konference, instant messaging nebo online hry. Komunikace prostřednictvím protokolu SIP probíhá následujícím způsobem:

1. lokace uživatele – určení koncového systému komunikace;
2. dostupnost uživatele – určení dostupnosti uživatele ke komunikaci;
3. přenosové parametry – určení parametrů s ohledem na přenosová média;
4. nastavení relace – nastavení parametrů relace mezi volaným a volajícím;
5. management relace - včetně převodu a ukončení relace, možnost změny parametru či služeb během spojení.

SIP protokol je navržen jako otevřený protokol, ovšem při návrhu nebyl brán dostatečný důraz na možnosti zabezpečení, nepředpokládalo se využití NAT (*Network Address Translation*) a dalších služeb, které jsou v současné době samozřejmostí [20]. Díky postupnému vývoji protokolu dochází k neustálému rozšiřování a prohlubuje se složitost implementace. Samotný protokol je definován v RFC 3261, ovšem jedná se jen o základní definici, existuje mnoho dalších RFC, které rozšiřují možnosti protokolu.

1.1. Metody a odpovědi inicializačního protokolu SIP

Jak již bylo uvedeno v úvodní části, SIP protokol je inicializační protokol, který se stará o zprávu spojení. K tomuto účelu využívá několika typů metod. Každá tato metoda je přenášena jako samostatná zpráva mezi uživatelem a serverem. Jedná se o požadavky, u kterých je vyžadována odpověď, kormě zprávy ACK, ta je pouze informační.

Tab. 1.1: Metody signalizačního protokolu SIP [44]-[47] [45] [46] [43]

Metoda	Popis	Doporučení
INVITE	určena k navazování spojení, nebo ke změně parametrů již existujícího spojení	RFC 3261
ACK	slouží jako potvrzení od uživatele, že obdržel požadavek	RFC 3261
CANCEL	žádost o zrušení probíhající žádosti INVITE	RFC 3261
REGISTER	žádost o registraci klienta u registračního serveru	RFC 3261
OPTIONS	slouží k výměně informací podporovaných funkcí	RFC 3261
BYE	žádost o ukončení spojení	RFC 3261
REFER	požadavek jiného UA k relaci (např. inicializace spojení přes web)	RFC 3515
SUBSCRIBE	slouží jako požadavek pro zjištění stavu vzdáleného prvku	RFC 3265
NOTIFY	Slouží k informování koncové prvku	RFC 3265
UPDATE	aktualizace informace o stavu relace	RFC 3331
INFO	slouží k přenosu informací během relace	RFC 2976

Tab. 1.2: Odpovědi signalizačního protokolu SIP [44]

Kód	Popis	Příklad	Doporučení
1xx	Informační odpověď, požadavek byl obdržen a zpracovává se	100 Trying 180 Ringing 183 Session Progress	RFC 3261
2xx	Kladná odpověď, požadavek byl úspěšně obdržen a zpracován	200 OK 202 Accepted	
3xx	Přesměrování	300 Moved 302 Multiple Choices 305 Use Proxy	
4xx	Chyba na straně klienta.	401 Unauthorized 403 Forbidden 486 Busy Here	
5xx	Chyba na straně serveru	501 Not Implemented 503 Service Unavailable	
6xx	Globální chyba	600 Busy Everywhere 603 Decline	

1.2. Architektura inicializačního protokolu SIP

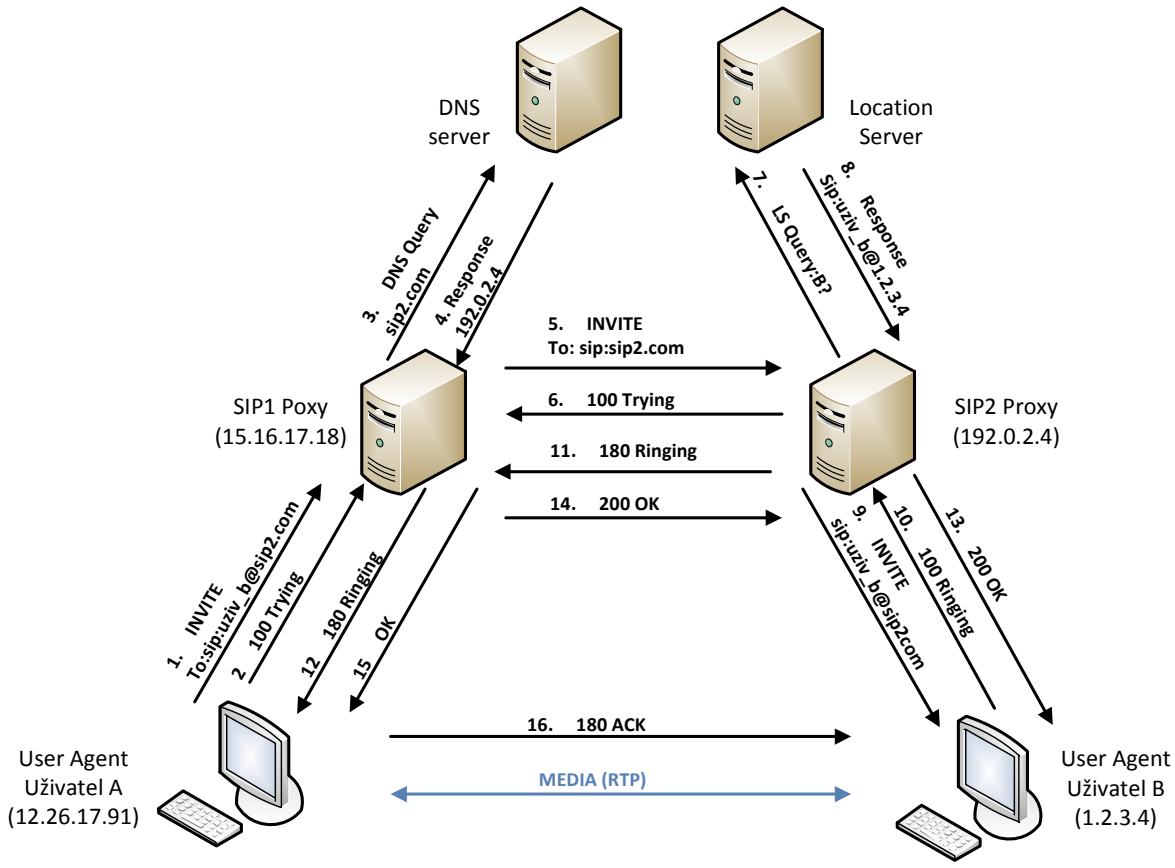
User Agent je zařízení na bázi protokolu SIP, může se skládat ze dvou logických částí: **User Agent Client** (UAC) a **User Agent Server** (UAS). User Agent Client (UAC) inicializuje spojení a posílá požadavky na UAS agenta. UAS agent odpovídá na zprávy vytvořené UAC agentem. Jelikož UA pošle jeden request a vzápětí na jiný request odpovídá, tak se role klienta UAC a serveru UAS mění, neboli tyto role platí jen po dobu transakce. Průběh komunikace je znázorněn na obrázku 1.1.

SIP server - jedná se o zařízení v síti, které se stará o správné vyřízení požadavku mezi účastníky spojení. Při přenosu jsou vysílány různé druhy požadavků, z toho důvodu musí SIP server zatupovat různé role.

- **Registrar server** - využívá se k registraci lokace uživatele, který se přihlásil do sítě. Pokud je požadavek na vytvoření nového spojení, tak Registrar server slouží jako adresář uživatelských jmen a IP adres. Zpracovává zprávy REGISTER a informace o poloze klienta předává lokalizační službě [59].
- **Redirect server** - přijímá žádosti o spojení od UAC nebo od proxy serverů, umožňuje směrování komunikace do jiné destinace. Další možností využití může být load-balancing, kde Redirect server může přesměrovat příchozí požadavky od klientů na základě momentálního vytížení [59].
- **Proxy server** - slouží k základnímu řízení komunikace protokolu SIP. Umožňuje navazování spojení mezi jednotlivými klienty [59].

1.2.1. Protokol SIP proxy architektura

Toto je teoretický princip komunikace při architektuře proxy serverů. Uživatel A vytváří spojení s uživatelem B. Uživatel A má URI uziv_a@sip1.com a Uživatel B má URI uziv_b@sip2.com. Uživatel A je UAC agent a je nakonfigurován, aby komunikoval při odchozích spojeních s proxy serverem. Komunikace začíná posláním zprávy INVITE proxy serveru SIP1. Tato zpráva signalizuje požadavek na UAS agenta Uživatele B (1) a server potvrdí požadavek (2). Uživatel B je identifikován svým URI, které značí, že uživatel B se nachází v jiné doméně. Na základě toho proxy server SIP1 předá původní požadavek INVITE k proxy serveru SIP2, který je zodpovědný za doménu sip2.com. IP adresu serveru SIP2 si proxy server zjistí od svého DNS serveru (3).



Obr. 1.1: Průběh komunikace signalizačního protokolu SIP [67]

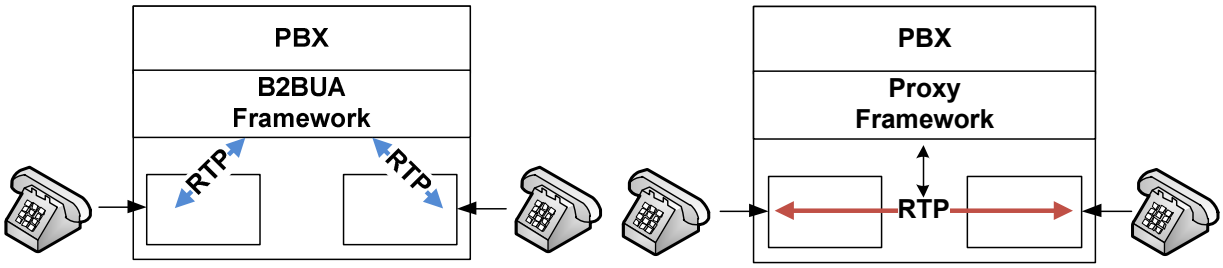
DNS server odpoví (4) s IP adresou serveru SIP2. Proxy server SIP1 může nyní odeslat zprávu INVITE serveru SIP2 (5). SIP2 potvrdí požadavek (6) a nyní hledá lokaci uživatele B (7), Location server odpovídá IP adresou uživatele B a zároveň indikuje, že uživatel B je dostupný (8).

Proxy server SIP2 může nyní poslat INVITE zprávu k uživateli B (9). Uživatel B odesílá potvrzující zprávu uživateli A (10, 11, 12) mezitím je upozorněna místní aplikace. Jakmile je hovor akceptován, UAS agent uživatele B odesílá zprávu OK uživateli A (13,14,15). Uživatel A odešle potvrzující zprávu uživateli B (16), tím je spojení vytvořeno a může být započato RTP spojení [67].

1.2.2. Protokol SIP B2BUA architektura

Další možností je architektura, která obsahuje pouze zařízení typu UAC, UAS a jejich kombinace B2BUA (B*ack-to-Back User Agent*). Díky tomu vzniká komunikace typu end-to-end. Tento způsob komunikace umožnil lepší implementaci zabezpečení, integrity a autentizace. Princip B2BUA komunikace je zobrazen na obrázku 1.2. [36].

Dle IETF standardu RFC 3261 je definován back-to-back user agent jako „logická entita, která přijímá požadavky a na základě definovaných pravidel zpracovává požadavky jako User Agent UAS. V případě, kdy je potřeba definovat jakým způsobem na požadavek odpovědět, tak se tato entita chová jako UAC a sama generuje požadavky. Na rozdíl od Proxy serveru udržuje stav dialogu a všechny žádosti musí procházet právě přes tuto entitu.“ [44] Server B2BUA umožňují větší flexibilitu než ostatní typy SIP serverů a vytváří rozhraní pro SIP aplikace, které nemohou být provedeny jako proxy aplikace.



Obr. 1.2: Rozdíl v komunikaci B2BUA a Proxy serveru

Příkladem může být spojení dvou uživatelů, které dokáže vytvořit jak proxy server, tak B2BUA server. Obě entity dokáží vytvořit spojení mezi dvěma uživateli, kdy ze strany uživatele není mezi spojeními rozdíl. Existuje zde ovšem zřejmý rozdíl v architektuře. B2BUA server vytvoří s každým uživatelem nezávislé spojení, oproti tomu proxy server je součástí relace pouze na začátku spojení, kdy inicializuje spojení. Samotná data jsou následně přenášena mezi jednotlivými uživateli[18].

Při architektuře proxy serveru má PBX zařízení informace o probíhajícím spojení, ale již nemůže jakýmkoliv způsobem ovlivnit spojení mezi uživateli. Jinými slovy proxy server pouze inicializuje spojení a následně se stává pasivním prvkem. Oproti tomu B2BUA server má úplnou kontrolu nad spojením, jelikož uživatelé komunikují přímo s ústřednou. B2BUA se chová jako koncový prvek a zároveň prvek, který ovládá spojení. Komunikace B2BUA je využívána většinou open source PBX zařízeními, jako je například Asterisk , FreeSWITCH a jiné. [36]

1.3. Adresace inicializačního protokolu SIP

Jelikož SIP byl založen na existujících standardech, které se již osvědčily na Internetu, používá zavedené metody pro identifikaci a připojení koncových bodů. To je zvláště vidět na systému adresace SIP účtů, které jsou podobné emailové adrese. Pro identifikace se používá URI (*Uniform Resource Identifier*)[40]. Skládá se ze dvou základních prvků: domény, kde je uživatelský účet umístěn a uživatelského jména (popřípadě telefonního čísla).

SIP: username@domain.com

Použitím této metody je jednoduché se připojit na konkrétní telefonní číslo nebo jméno. Samozřejmě platí, že uživatelské jméno na dané doméně musí být unikátní kvůli jednoznačné identifikaci uživatele. Uživatelské jméno a doména jsou jen dva základní prvky, které SIP URI obsahuje, mezi další patří číslo portu, heslo a další parametry [44]. Úplné znění hlavičky vypadá následovně:

sip:<user>[:<password>]@<host>[:<port>][;<uri-parameters>][?<headers>]

Kromě formátu SIP URI je definována i zabezpečená verze URI, SIPS URI (*SIP Secure*). Formát a struktura dat je v obou případech stejný, jen se místo „sip“ používá „sips“. To zaručuje, že je přenos šifrován pomocí TLS po celou dobu přenosu [44].

sips:<user>[:<password>]@<host>[:<port>][;<uri-parameters>][?<headers>]

2. Open source PBX ústředna

Existuje poměrně velké množství open source PBX, ovšem tato práce není zaměřena na jejich výčet ani detailní popis. Z toho důvodu jsou níže popsány heslovitě jen tři zástupci. Ústředna Asterisk je v dnešní době pravděpodobně nejrozšířenější Open source PBX ústřednou [71]. Ústředna FreeSWITCH dle dostupných informací nabízí vyšší výkon ve srovnání s ústřednou Asterisk na stejné hardwarové konfiguraci [16]. Open source ústředna YATE patří mezi velmi se rozvíjející projekty v poslední době a disponuje řadou pokročilých funkcí pro ISDN (*Integrated Services Digital Network*) a TDM (*Time Division Multiplex*) [81]. Každá z těchto ústředen bude podrobena testům, jak reagují na jednotlivé útoky a výsledky budou popsány v praktické části práce.

2.1. Open source ústředna Asterisk

Asterisk vytvořil Mark Spencer v roce 1999, důvodem vzniku byla potřeba nekomerčního PBX systému. V roce 2004 se jednalo o první plnohodnotný open source PBX systém [22]. V dnešní době existuje Asterisk ve verzi 10, předešlá verze systému je 1.8. Je ovšem nutno zmínit, že verze 1.8 je LTS (*Long Term Support*), tudíž se jedná o verzi kde je zaručena dlouhodobá podpora [56]. V tabulce 2.1 jsou znázorněny jednotlivé vývojové řady Asterisku.

Jedná se o nejrozšířenější open source systém pro VoIP technologii. Je dostupný na operačních systémech Linux, BSD, Windows (emulovaně) a MacOS X, poskytuje všechny funkce, které jsou očekávány od pobočkových ústředen. Asterisk je softwarová pobočková ústředna umožňující jak IP telefonii, tak digitální ISDN i analogovou telefonii. Mezi funkce, které podporuje, patří například IVR (*Interactive Voice Response*) a ACD (*Automatic Call Distribution*) [22].

Tab. 2.1: Vývojové řady ústředny Asterisk [56]

Verze	**Typ**	**Datum vydaní**	**Bezpečností záplaty do:**	**EOL**
1.2.X		2005-11-21	2007-08-07	2010-11-21
1.4.X	LTS	2006-12-23	2011-04-21	2012-04-21
1.6.0.X	Standard	2008-10-01	2010-05-01	2010-10-01
1.6.1.X	Standard	2009-04-27	2010-05-01	2011-04-27
1.6.2.X	Standard	2009-12-18	2011-04-21	2012-04-21
1.8.X	LTS	2010-10-21	2014-10-21	2015-10-21
10.X	Standard	2011-12-15	2012-12-15	2013-12-15

Podporované signalizační protokoly[75]:

- IAX - Inter-Asterisk Exchange – jedná se o protokol vytvořený tvůrci Asterisku [22]
- H.323
- SIP - Session Initiation Protocol
- MGCP - Media Gateway Control Protocol
- SCCP - Cisco Skinny

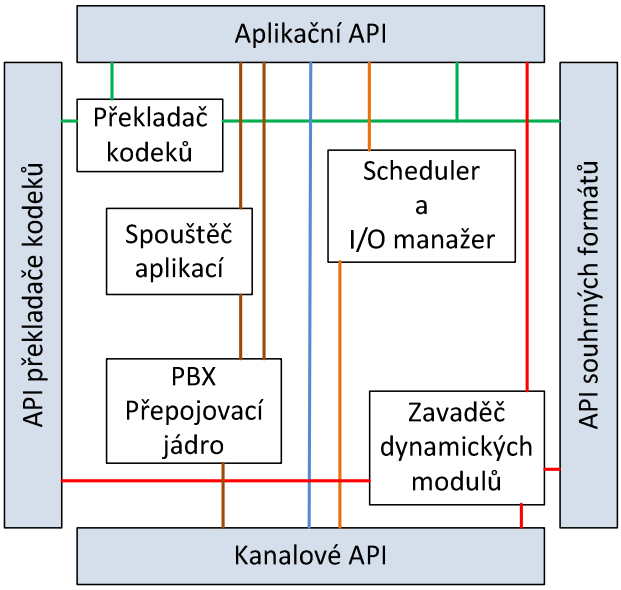
Podporované kodeky[75]:

- ITU-T G.711 (A-Law a µ-Law), G.726, G.723.1, G.729,
- GSM, iLBC, LPC-10 a Speex.

Architektura pobočkové ústředny Asterisk

Okolo centrálního jádra PBX jsou definovány specifické API (*Application Programming Interface*). Jádro ovládá vnitřní propojení PBX, specifické protokoly, kodeky a hardwarové rozhraní telefonních aplikací **[75]**.

PBX přepojovací jádro - základním prvkem systému Asterisk je přepojovací systém pobočkové ústředny, sloužící ke spojování různorodých uživatelů a automatizovaných úkonů.
Spouštěč aplikací - slouží ke spouštění aplikací jako je hlasová pošta, přehrávání souborů a procházení adresářů.
Překladač kodeků - používá moduly s kodeky pro kódování a dekódování různých audio formátů, které se používají v telefonním prostředí. Má k dispozici dostatečně velkou škálu kodeků, aby vyhovoval rozdílným potřebám a dosahoval co nejlepších výsledků v poměru mezi audio kvalitou a využitou šířkou pásma.
Scheduler a I/O Manažer - zajišťuje plánování úkolů na nejnižší úrovni a zajišťuje optimální využití výkonu systému za jakýchkoliv podmínek.



Obr. 2.1: Architektura pobočkové ústředny Asterisk [75]

2.2. Open source ústředna YATE

YATE (*Yet Another Telephony Engine*) jak název uvádí, označuje samotný engine, který je zaměřen na VoIP a PSTN sítě. Mezi hlavní přednosti patří snadná rozšířitelnost. Hlavní podporované součásti jsou audio video a datové služby nebo IM (*instant messaging*). Samotná ústředna je dostupná na nejpoužívanějších platformách Windows, Linux a MacOS X [80]. Tvůrcem je Paul Chitescu ze společnosti NullTeam, ta byla vytvořena v roce 2004. Jedná se o open source zveřejněný pod licencí GNU. Software je napsán v C++ a podporuje různé programovací jazyky jako například PHP, Python nebo Perl [77].

Podporované kodeky [9]:

- Speex
- G.729, G.723, G.722

Podporované signalizační protokoly [9]:

- H.323
- SIP
- IAX2
- PRI (PRA) ISDN
- JINGLE

Architektura pobočkové ústředny YATE

Jak už bylo napsáno dříve YATE se skládá z modulů, které jsou mezi sebou provázány dle určitých pravidel, což umožňuje dostatečnou flexibilitu. Zprávy v YATE mohou mít libovolný počet parametrů, a mohou být zaslány na více než jeden modul změnou priority [79].

Hlavní komponenty:

- **Core** - zajišťuje generování tříd jako String, Thread, Socket, Mutex
- **Message Engine** - zajišťuje zprávu tříd Message, Engine, Plugin
- **Telephony Engine** - zajišťuje zprávu tříd určených pro telefonii jako Driver a Channel
- **Yate Moduly** – jedná se o nedílnou součást nastavení ústředny, díky nim získává ústředna požadované vlastnosti. Dělí se do několika kategorií, patří zde například moduly signalizační, routovací, registrační, PBX nebo testovací.

V současné době je dostupná verze ústředny 4, která již podporuje TCP a TLS pro SIP protokol, umožňuje nastavení monitorování průběhu komunikace. YATE také podporuje SRTP protokol k zabezpečení komunikace [78]. Nastavení jednotlivých protokolů se definuje v souborech *ysipchan.conf*, *yiaxchan.conf* a *h323chan.conf*, kde lze definovat i využití TCP, TLS nebo SRTP [3].

2.3. Open source ústředna FreeSWITCH

Jedná se o framework určený k propojování různých komunikačních protokolů s použitím audia, videa, textu nebo jakékoliv jiné formy médií. Dostupný pro platformy Windows, Mac OS X, Linux, *BSD a další UNIX systémy. Je navržen tak, aby mohl využít co nejvíce již existujících knihoven. Systém je založen na modulární architektuře, to znamená, že je velmi snadno rozšiřitelný a jádro systému obsahuje jen nezbytné funkce. V podstatě se jedná o samostatnou knihovnu, která na sebe váže ostatní moduly, které zajišťují veškerou funkcionalitu [4].

FreeSWITCH byl vytvořen bývalými vývojáři Asterisku s cílem zaměřit se na modularitu, multiplatformní podporu a stabilitu, rok vzniku je 2006. FreeSWITCH je open source komunikační platforma vytvořena v programovacím jazyce C, distribuovaná pod licenci MPL 1.1 (*Mozilla Public License 1.1*). Podporuje řadu programovacích jazyků jako C/C++, Java, .NET, Javascript/ECMAScript, Python, Perl, Ruby, PHP, Lua a další [14]. Aktuální verze 1.2, stejně jako předchozí dva zástupci podporuje SIP TLS, SRTP a TCP spojení na místo UDP [58].

Podporované kodeky [58]

- PCMU – G.711 µ-law
- PCMA – G.711 A-law
- G.722
- G.726
- G.729
- GSM
- CELT and OPUS
- iLBC
- BroadVoice
- SILK
- Speex
- CODEC2
- Siren
- LPC-10
- G.723.1
- AMR
- iSAC

Podporované protokoly [58]

- SIP
- SCCP
- H.323
- LDAP
- Zeroconf
- XMPP / Jingle.

3. Problematika útoků v IP telefonii

Většina bezpečnostních problému v IP telefonii vzniká z toho, že VoIP pracuje na otevřených systémech a využívá již existující datové sítě. Z toho plyne, že bezpečnostním rizikem je pro VoIP i většina útoků, které lze provést obecně proti IP technologii. Díky tomu platí pro útok proti VoIP systémům téměř analogická pravidla. Další problém představují útoky, které jsou specifické právě jen pro VoIP systémy. Jelikož je tato práce zaměřena na inicializační protokol SIP budou specifické útoky na VoIP věnovány právě jemu.

Nelze jednoznačně definovat obecný postup, jakým může útočník provádět nelegitimní činnosti v IP telefonii. Existují ovšem dvě základní podmínky, které musí být splněny u většiny útoků nebo nelegitimních činností, ne jen v IP telefonii, ale obecně v datových sítích. Pokud je cílem útoku interní zařízení, v tomto případě PBX ústředna, je nutné zajistit přístup k LAN síti, kde se prvek nachází. Pokud dané PBX zařízení je zpřístupněno i ze strany internetu, je tento problém vyřešen už v počátku. Druhou, velice důležitou podmínkou je nutnost získat informace o daném prvku, který je cílem útočníkova útoku.

Proniknutí do sítě - jedná se o základní krok, bez kterého útoky není možné realizovat, ale pouze za předpokladu, že cíl útoku není veřejně dostupným prvkem, tedy že nemá veřejnou IP adresu. Vždy záleží na dané topologii a druhu sítě. Existuje několik možností jak získat přístup k síti buď to invazivním útokem zaměřeným na poškození směrovacích pravidel přepínače, anebo útokem typu „Man in the Middle“.

Analýza provozu a shromažďování informací - pokud útočník proniknul do sítě, dostává se mu možnost odposlechnout provoz v síti a shromáždit užitečné informace o síti a prvcích, které jsou v ní obsaženy.

Útoky na dostupnost služby - do této kategorie spadají útoky typu *Denial of Service* (DoS) a *Distributed Denial of Service* (DDoS), popřípadě útoky, které mají za cíl přerušit dané spojení. Jedná se o nejnebezpečnější typ útoku, jelikož jsou poměrně snadno proveditelné a jejich následky mohou ovlivnit celý systém. Pokud je například DoS synchronně spuštěn ze stovek počítačů, dokáže i zabezpečenému serveru popřípadě PBX způsobit velké potíže. VoIP komunikace je velice citlivá na QoS parametry, protože probíhá v reálném čase. Pří útoku mohou být pakety skutečného hovoru zpožděny, nebo může dojít k velké ztrátě paketů a hovor je velmi nekvalitní, případně dojde k přerušení relace [73].

Útok na integritu služeb - útočník se snaží určitým způsobem pozměnit případně zfalšovat přenášené zprávy. Je možné modifikovat přenášená data, popřípadě vložit nežádoucí data přímo do vytvořeného spojení[73].

SPIT (SPAM over IP Telephony) - reklamní telefonáty a jiné nevyžádané hovory. Po přijetí hovoru je automaticky přehrán hlasový záznam. Jedná se o alternativu emailového spamu [24].

3.1. Zajištění přístupu k LAN síti

K tomu, aby mohl být proveden úspěšný útok na VoIP, je potřeba splnit základní předpoklad a to získat přístup k síti, kde se PBX nachází. Přístup k síti lze získat několika způsoby, záleží na typu sítě. Pokud se jedná o bezdrátovou síť, je možné díky sledování sítě získat potřebné informace o síti, pokud je síť nezabezpečená jedná se o přístup triviální. V případě, že se jedná o bezdrátovou síť zabezpečenou pomocí WEP (*Wired Equivalent Privacy*), existuje mnoho podpůrných programů například Aircrack [1], které dokáží zabezpečení prolomit. U standardů WPA a WPA2, která využívají složitější šifrovací algoritmy, je prolomení podstatně komplikovanější [12]. Pokud se nejedná o bezdrátovou síť, zaleží na konkrétním zařízení, ke kterému chce útočník získat přístup.

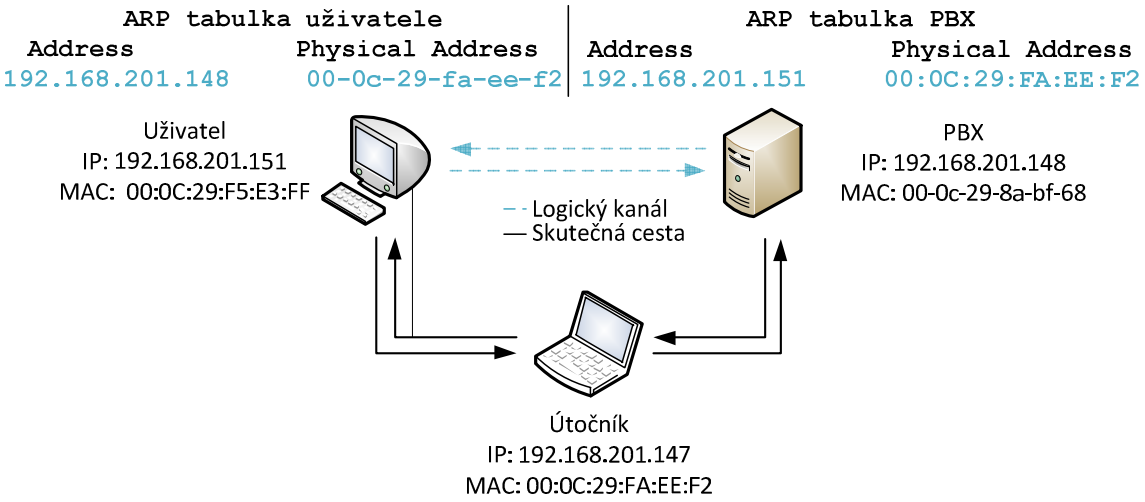
V případě přepínače je možné prolomit webové rozhraní a nastavit si odesílaní paketů na konkrétní port, bohužel v praxi je velmi často běžná kombinace admin/admin popřípadě admin/password. I zde existuje celá řada podpůrných aplikací, které umožní prolomit heslo například slovníkovým útokem, nebo metodou odposlechu. Webové rozhraní je většinou dostupné přes http, kde je uživatelské jméno a heslo přenášeno v nešifrované formě. Odposlech hesel je umožněn i pro protokol https, ovšem jen do určité míry v závislosti na použitém certifikátu. Další možností je obelstít ARP / MAC tabulku pomocí MAC Address Flooding. Útočník vysílá záplavu náhodných MAC adres na switch, pokud kapacita tabulky přestane stačit, tak se zařízení přepne do nouzového režimu fungující jako HUB.

Man in the Middle

Existují i jiné varianty, jak sledovat provoz na síti bez toho aniž by bylo potřeba drastických útoků, používajících techniky pro prolomení hesla, či poškození ARP tabulky. Jednou z možností je využití sniffovacích nástrojů, jako Ettercap. Ettercap disponuje řadou funkcí, které lze využít k odposlechu komunikace. Je vhodný pro útok typu Man in the Middle, dokáže zasahovat do probíhající komunikace bez přerušení spojení, nebo také umožňuje odchyt hesel řady protokolů. Na obrázku 3.1 je možno vidět odposlech síťové komunikace mezi ústřednou a jedním uživatelem. Existuje několik způsobů, jak provést přesměrování komunikace. Patří zde Port stealing, ARP Spoofing, DHCP Spoofing nebo ICMP Redirecting. Technika je vždy rozdílná, ale výsledek stejný, útočníkovi je umožněno sledování a ovlivnění komunikace, která mu není určena [57].

- **Port stealing** spočívá v oklamání switch a jeho CAM tabulky. Útočník pošle na switch rámec, kde upraví zdrojovou MAC adresu na MAC adresu oběti. Jako cílovou adresu uvede svoji MAC adresu. Switch následně upraví CAM tabulku a podle zdrojové MAC adresy si bude myslet, že oběť se nachází na portu, kde je ve skutečnosti útočník. Cílová adresa může být nastavena i jako broadcast, což je výhodné ze strany útočníka pokud je v síti více přepínačů [12].

- Principem **ARP Spoofingu** [33] je změna záznamu v ARP tabulce. Útočník pošle PBX ústředně (192.168.201.148) paket, kde je jako MAC adresa uživatele uvedena MAC adresa útočníka (192.168.201.147). Analogicky je uživateli (192.168.201.151) vyslán paket, kde je uvedeno, že MAC adresa PBX ústředny je adresa útočníka. Tímto je veškerá komunikace mezi PBX a uživatelem směřována přes útočníka.



Obr. 3.1: Odposlech LAN sítě pomocí ARP Spoofingu

- **DHCP Spoofing** využívá možnosti,kdy v lokální síti může existovat více DHCP serverů, pokud si oběť spustí počítač, útočník mu pošle falešné údaje. Tímto způsobem může být oběti podstrčena brána, kterou definuje útočník. Ten má pak kontrolu nad odchozími daty [12].
- **ICMP Redirecting** útok spočívá v posílání falešných zpráv ICMP o efektivnější cestě. Pokud uživatel obdrží zprávu ICMP Redirecting navrhující optimálnější cestu k cíli, aktualizuje si na základě ní svou routovací tabulku [12].

3.2. Analýza provozu a shromažďování informací v LAN síti

Skenování sítě a shromažďování informací o prvcích v síti, je klíčovou součástí každého útoku popřípadě testu zabezpečení. K tomu, aby mohl být proveden úspěšný útok na PBX, je potřeba mít dostatečné znalosti o nastavení sítě a prvcích, které jsou v síti obsaženy. Díky skenování sítě může útočník získat informace o operačních systémech a dostupných aplikacích. Tyto informace pak mohou být využity k provedení konkrétních útoků. Obecně skenery pracují tak, že odešlou dotaz na definovaný rozsah portů a zaznamenávají odpovědi. Tímto způsobem je možné získat mnoho užitečných informací:

- Jaké jsou dostupné IP adresy v síti.
- Jaké porty jsou otevřené ke komunikaci.
- Jaký typ a verze operačního systému je na daném zařízení.
- Jaké aplikace jsou dostupné na daných systémech.
- Kde je umožněno anonymní přihlášení.

Pro základní skenování LAN sítě je možno využít různých nástrojů, jako například Nmap. Nmap (Network Mapper) [28] je port scanner, který ke zjištění dostupnosti stanic používá různé metody zjišťování informací. Pro zjištění validních IP adres je použit protokol ARP. Podporuje také TCP i UDP a dokáže detekovat i operační systém. Pro detekci TCP portu Nmap pošle SYN paket a čeká, jak server odpoví. Server buď žádost o sestavení spojení potvrdí paketem ACK a nebo zamítne paketem s příznakem RST. K zjištění UDP portu odešle

skenovací nástroj UDP paket s prázdnou hlavičkou. Pokud skener obdrží odpověď ve formě UDP paketu, port je otevřen a aplikace naslouchá. Nmap dělí dostupnost portů do několika stavů:

- **Open** - port je otevřen a naslouchá;
- **Closed** - port není dostupný;
- **Filtered** - nmap nedokáže určit, zda je port otevřený, jelikož filtrování paketů zabraňuje dosáhnutí tohoto portu;
- **Unfiltered** – tento stav znamená, že port je dostupný, ale není možno určit, jestli je open nebo closed;
- **open|filtered** – nmap nedokáže určit, zda je port otevřen či nikoliv. Tento stav nastává v situaci, kdy nepřišla žádná odpověď na odeslaný požadavek.

```
Starting Nmap 5.51 ( http://nmap.org ) at 2012-04-30 16:04 CEST
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is
disabled. Try using --system-dns or specify valid servers with --dns-
servers
Nmap scan report for 192.168.201.151
Host is up (0.00029s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE
135/tcp open  msrpc
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:F5:E3:FF (VMware)
Device type: general purpose
Running: Microsoft Windows XP|2003
OS details: Microsoft Windows XP SP2 or SP3, or Windows Server 2003
Network Distance: 1 hop
```

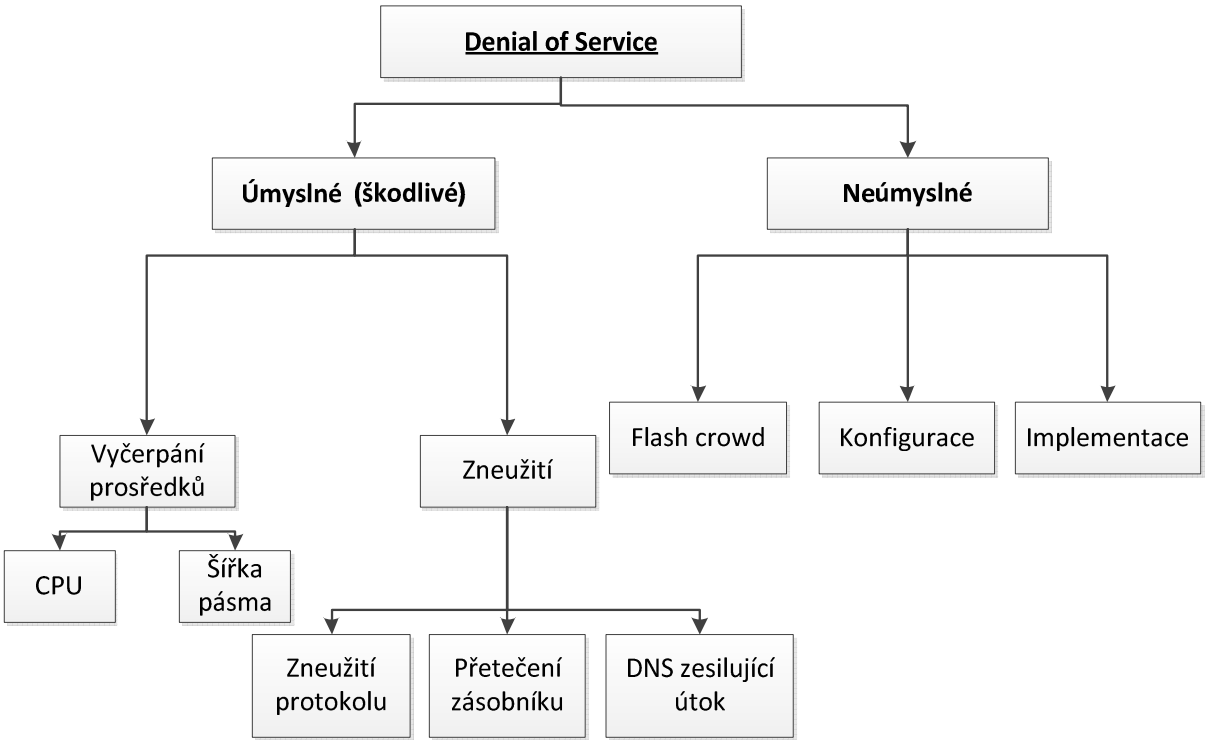
SIP je nejrozšířenějším protokolem pro VoIP komunikace a z toho důvodu existuje mnoho nástrojů sloužících k testování či k provádění útoků na danou VoIP síť. Jedním z nich je sada skriptů SipVicious [61], který slouží k základnímu testování protokolu SIP. Díky nástroji svmap je možné získat adresu všech entit v síti, které umožňují komunikaci SIP protokolem. Jako vstupní parametr postačuje IP rozsah sítě. Využívá také protokol ARP, tím se z rozsahu vyfiltrují nepoužívané IP adresy. Následně na to se odešle zpráva OPTIONS na předem definované IP adresy. Od PBX ústředny dostane odpověď 200 OK, spolu s informací o jaký typ ústředny se jedná.

```
svmap.py 192.168.201.1-255
| SIP Device           | User Agent                          | Fingerprint |
-------------------------------------------------------------------------
| 192.168.201.148:5060 | Asterisk PBX 1.6.0.26-FONCORE-r78 | disabled    |
```

3.3. Útok typu odmítnutí služby

Obecně lze útok typu Denial of Service klasifikovat jako snahu o zabránění legitimním uživatelům přístupu k službám nebo síťovým zdrojům. To může být způsobeno přetížením dané služby nebo celého serveru. Tento typ útoku může být směřován jak proti jednotlivým uživatelům, tak proti celé síti. Běžným cílem těchto útoků jsou servery, které zpřístupňují služby širší skupině uživatelů jako například web server, mail server nebo také PBX ústředna. Tento typ útoku je velmi rozšířený, jelikož jeho provedení, v poměru k tomu jaké dokáže způsobit následky, není příliš složité. Jeho rozšířenost v poslední době vzrostla hlavně díky webovým serverům různých vládních společností. Jednalo se o odpověď na projednávání zákonů SOPA [68] a PIPA [35]. Množství útoků vzrostlo v roce 2011 o 45% ve srovnání s rokem 2010. DoS útok není nijak striktně vázán na konkrétní protokol či služby. Může být proveden v různých vrstvách TCP/IP architektury a na různorodé služby. To znamená, že útok může být proveden jak na síťovou, aplikační, tak transportní vrstvu. Klasifikace DoS je znázorněna na obrázku 3.2. DoS útoky lze dělit do dvou základních kategorií, neúmyslné útoky a úmyslné neboli škodlivé útoky [62].

Neúmyslné útoky jsou obvykle výsledkem špatné konfigurace. Například hardwarová konfigurace systému neodpovídá jeho budoucímu vytížení, takže systém nebude zvládat příchozí požadavky od uživatelů. Může dojít k přetížení systému analogicky jako u DDoS útoku[12].



Obr: 3.2: Kategorie útoku typu odmítnutí služby

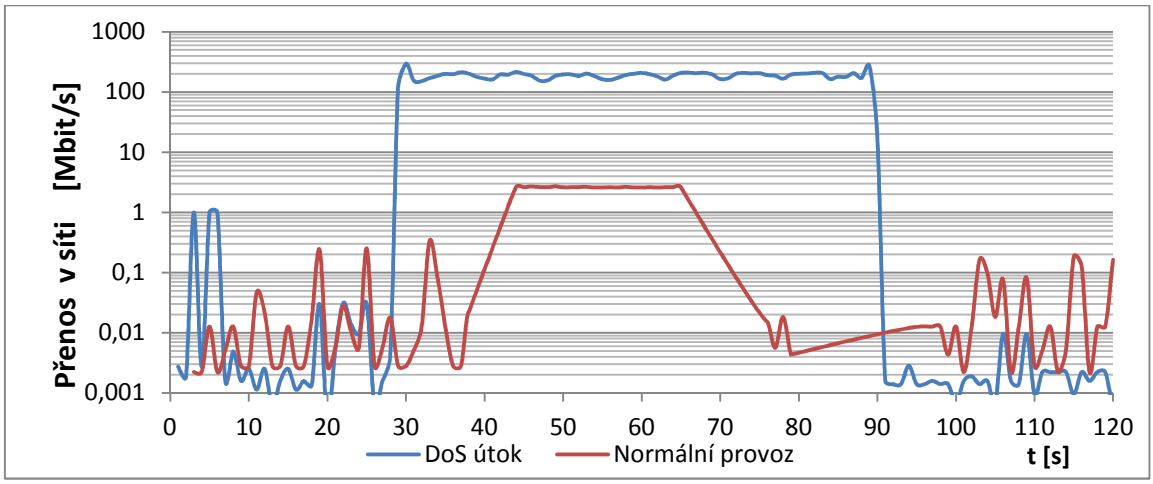
Úmyslné útoky mohou být dále rozděleny na útoky záplavového typu a útoky využívající nedokonalosti některých protokolů. Útoky záplavou paketů slouží k vyčerpání systémových prostředků oběti, z čehož vzniká neschopnost zpracovat příchozí požadavky. Jedná se převážně o vytížení procesoru, překročení šířky pásma nebo vytížení operační paměti systému. Teardrop attack může způsobit pád celého systému. Principem je zaslání nelegitimních IP fragmentů, například se mohou fragmenty překrývat. Díky špatné

implementaci sestavování IP datagramu mohlo dojít k zhroucení celého systému. Na útoky tohoto typu byly náchylné spíše starší systémy. Do této kategorie lze zařadit i útoky, kde díky nedokonalosti protokolu a nedostatečně zabezpečeného systému lze převzít částečnou kontrolu nad obětí útoku [62].

3.3.1. Útoky omezující šířku pásma

Útočník generuje velké množství paketů, které dokáží zahltit síť a tím odpojit oběť od sítě. Díky tomu se legitimní požadavky nemohou dostat k cíli, z pohledu sítě se oběť může jevit jako nedostupná. Příkladem může být Ping of Death, kde útočník zasílá velké množství ICMP paketů o velikost 65535 bytů k cílové stanici. Díky tomu je možné vygenerovat tok o velikosti přesahující 100 Mbit/s. Pokud je oběť připojena pouze touto rychlostí, je zaplněna celá šířka pásma oběti a nemohou být zpracovány žádné příchozí požadavky [7]. Obdobnou možností je UDP flood, kde opět dochází ke generování velkého množství paketů k oběti. Příjemce může obdržet UDP paket bez předchozího upozornění a je povinen jej zpracovat. Oběť musí zkontrolovat, zda příchozí paket náleží některé z naslouchajících aplikací a zda-li obsahuje smysluplný obsah [62]. Díky oběma těmto možnostem je jednak zahlceno síťové spojení, tak i značně vytěžován celý systém. Pro demonstraci záplavou paketů je níže uveden graf vytížení linky při záplavě ICMP paktů o velikosti 65535 bytů, viz obrázek 3.3.

Útočník byl schopen vygenerovat tok dat přesahující 200 Mbit/s. Jedná se o ukázku zaznamenaných dat ze semestrálního projektu, kdy každý útok probíhal po dobu 60 sekund. Útok byl proveden proti ústředně, která byla s uživatelem spojena linkou o kapacitě 100Mbit/s. Data byla zaznamenána z pohledu útočníka, ten disponoval linkou o kapacitě 1 Gbit/s.



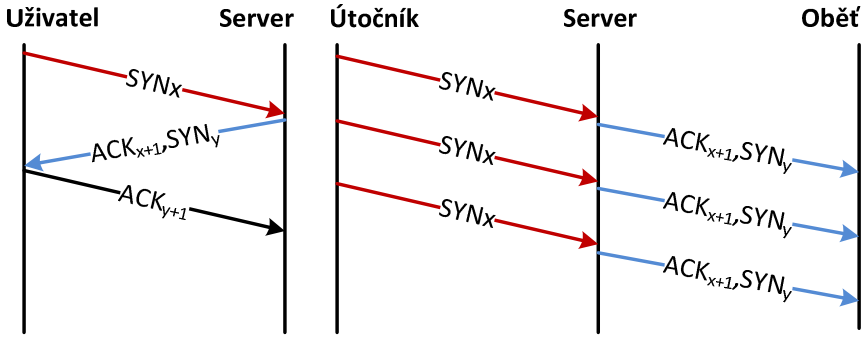
Obr. 3.3: Útok typu DoS záplavového typu

3.3.2. Útoky zaměřené na vyčerpání systémových prostředků

Do této kategorie lze zařadit několik útoků, ovšem princip je vždy stejný. Každý z těchto útoků se snaží vytížit systém natolik, aby byla znemožněna popřípadě omezena komunikace uživatelů s daným systémem. Jedná se o vytížení jak paměti systému tak hlavně dochází k vytěžování procesoru, to může mít za následek až odstavení celého systému z provozu. Jelikož procesor i paměť jsou sdíleny celým systémem, tak útoky v této kategorii nemusí být zaměřeny jen na samotný SIP protokol. Vytížení procesoru může jednoznačně ovlivnit

schopnost systému odpovídat na jednotlivé požadavky. To má za následek, že server je zaneprázdněn zpracováním požadavků od útočníka, ale legitimní požadavky jsou zahazovány. Do této kategorie mohou spadat částečně i útoky z předchozí části, jelikož i při nich může dojít ke značnému zatížení systému [25]. Tyto útoky již byly popsány v předchozí kapitole, z toho důvodu zde budou rozebrány jiné, také velmi efektivní útoky.

Jedním z těchto útoků je SYN flood. Tento typ útoku využívá nedokonalosti procesu navazování spojení mezi uživatelem a cílovou stanicí. Jedná se o takzvané třícestné navázání spojení (*three-way handshake*) [72], viz. obrázek 3.4. Při útoku je na server odesílána záplava paketů SYN, server odešle zpět příznaky SYN+ACK a rezervuje určitou část systémových prostředků, následně čeká na ACK od odesílatele (útočníka), ten ovšem požadovanou zprávu již nepošle [25] [7].



Obr. 3.4: Útok využívající třícestné navazování spojení

Pokud na sever přichází velké množství SYN paketů, tak postupně narůstá vytížení systému a může dojít až k úplnému vyčerpání systémových prostředků a oběť přestane odpovídat na jakékoliv požadavky. Určitým vylepšením tohoto útoku je zvolit IP adresu již existujícího systému v síti, díku tomu mohou být odchozí SYN+ACK pakety odeslány jinému uživateli a ten se pokusí příchozí požadavky zpracovat, i když pro něj nejsou určeny. Tímto způsobem dochází k vytížení systému obou obětí [62].

Jedná se o útok, který je zaměřen na TCP spojení, proto je často proveden proti webovým serverům. Často mívají ústředny i své webové rozhraní. Útok tohoto typu je schopen přetížit webový server, jako například Apache a z pohledu uživatele je rozhraní nedostupné. K pádu celého zařízení docházelo u starších systémů, v současných systémech je již obsažena ochrana SYN a RST Cookies [7]. Řešení pomocí Cookies odstraňuje problém s alokací systémových prostředků před úplným navázáním spojení, ale i přesto dokáže znepřístupnit některé služby.

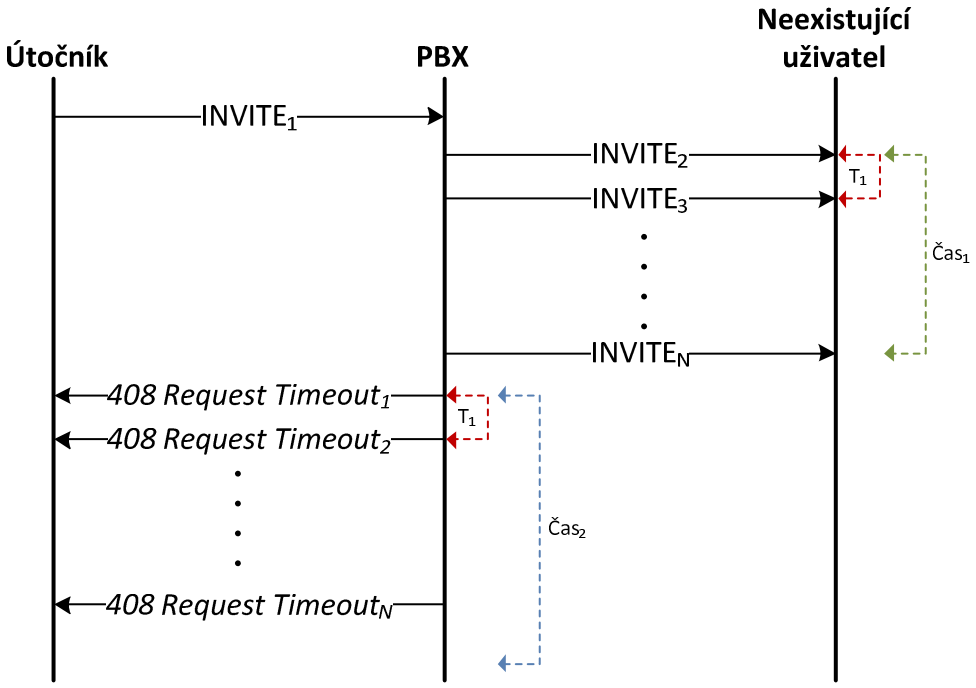
Útok záplavou paketů pomocí metody INVITE

Do kategorie útoku typu DoS patří také útoky využívající signalizačních zpráv INVITE. Pokud PBX obdrží požadavek, uloží si některé informace o transakci do paměti systému a to do té doby, než je transakce ukončena. PBX je schopno zpracovat určitý počet transakcí a to v závislosti na dostupné konfiguraci systému a možnostech samotné ústředny. Při teoretickém předpokladu, že ústředna zvládne zpracovat neomezené množství požadavků a je závislá pouze na možnostech operačního systému, je možno uvažovat následovně [62]. Při zvyšujícím se množství požadavků na PBX je postupně vytěžován systém a to až do doby, kdy je dosaženo maximální hodnoty. V tomto okamžiku již ústředna nemůže přijímat nové

požadavky a všechny další zprávy jsou zahozeny. Systém potřebuje určitý čas na zpracování všech již uložených požadavků, toho pak může využít útočník. Záplavu SIP zpráv je možné odesílat v určitých časových intervalech a ústředna zůstane nedostupná po celou dobu útoku. Příkladem může být záplava SIP metodou INVITE, která je nejefektivnější při útoku záplavou inicializačních zpráv [7].

PBX systém informace o INVITE požadavku udrží v paměti do té doby, dokud nezíská odpověď od uživatele, pro kterého je zpráva určena. Musí být zohledněn přenos zprávy po komunikačním médiu, zpracování požadavku uživatelem a následně doba, než ústředna obdrží požadavek od uživatele. Maximální možná doba pro udržení požadavku v systému je 32 sekund [44], za tento časový interval musí dorazit na PBX server odpověď od uživatele, v opačném případě je uživatel považován za nedostupného [44].

PBX systém musí zpracovat všechny požadavky, které obdrží. To znamená, že je nutné zpracovat i požadavky, které odkazují na neexistující uživatele. Jednotlivé požadavky by měly zůstat uchovány v paměti systému po definovanou dobu, tedy 32 sekund ($Čas_1$). Pokud daný uživatel není dostupný, je odeslána zpráva *408 Request Timeout response*[44]. Odpověď 408 bude přenášena po dobu 32 sekund ($Čas_2$). To znamená, že celková doba, jak dlouho je tento požadavek zpracováván je $Čas_1 + Čas_2$ tedy 64 sekund, obrázek 3.5.



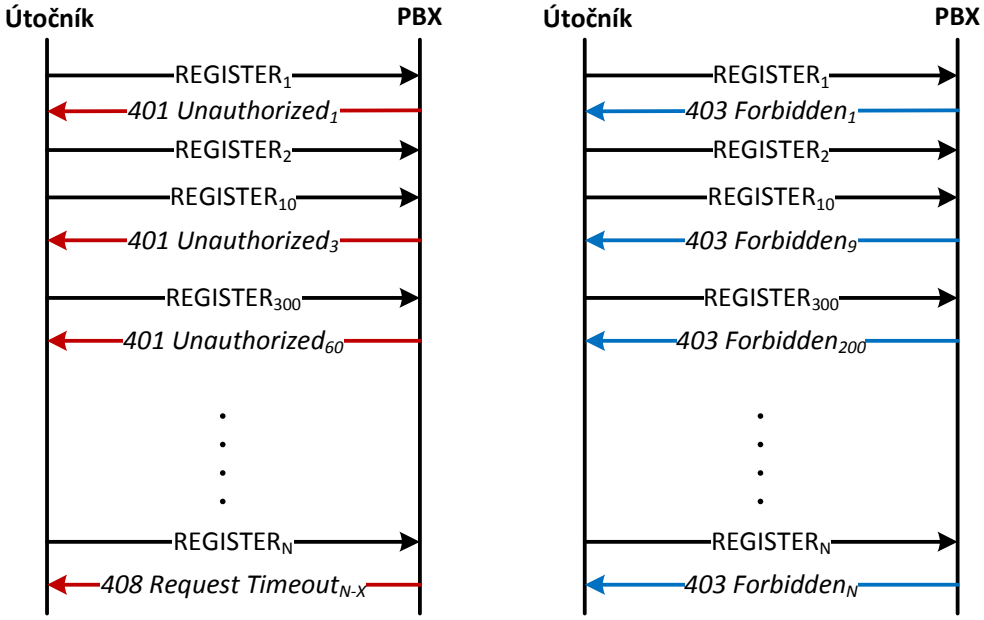
Obr. 3.5: Útok záplavou paketů pomocí metody INVITE

Analogicky platí, že PBX server pod útokem záplavou INVITE požadavků, tak doba odpovědí vzrůstá úměrně množství INVITE zpráv. K provedení efektivního útoku je nutné do SIP zprávy uvést již existující IP adresu systému, která je u PBX zaregistrována. Nebo v rámci URI je nutné uvést v následujícím tvaru **jmeno_uživatele@ip_adresa** namísto běžně užívaného formátu **jmeno_uživatele@jméno_domény.** Podstatné je zajistit, aby PBX systém považoval požadavek za skutečný a ne za špatně formulovanou zprávu, nebo neexistujícího uživatele [62].

V rámci SIP protokolu je typickým cílem DoS útoku SIP proxy server, ale útoky mají samozřejmě efekt i na B2BUA server. V případě B2BUA serveru dochází k razantnějšímu vytížení systému, jelikož server musí obstarávat kompletní zprávu spojení.

Útok záplavou paketů pomocí metody REGISTER

Další možností, jak vytížit systém PBX je pomocí SIP Autentizace. Pro autentizaci uživatelů se nejčastěji používá digest autentizace, která pracuje na systému výzva - odpověď. Princip autentizace je popsán v kapitole 4. Při digest autentizaci je generován *nonce*, který je vložen do odpovědi na zprávu REGISTER. Jedná se o *401 Unauthorized* [44] nebo *407 Proxy Authentication Required* [44]. Uživatel na základě této obdržené zprávy vygeneruje nový požadavek REGISTER, kde jsou již doplněny všechny potřebné údaje. Tím se ovšem naskýtá možnost útoku, která spadá do této kategorie, PBX ústředna si musí po celou dobu transakce uchovat informace o probíhající registraci [19]. Jedná se o podobný princip jako v předchozím případě, jen zde je využívána zpráva REGISTER. Princip je následující: útočník vyšle na PBX ústřednu požadavek o registraci, ústředna na daný požadavek odpoví a to jednou z výše uvedených zpráv. Tuto odpověď ovšem útočník ignoruje a neustále zasílá další požadavky na PBX systém. PBX si musí uchovat v paměti informace o již vytvořených *nonce* a zároveň musí vypočítávat nové kombinace. Tím dochází ke značnému zatěžování celého systému.



Obr. 3.6: Útok záplavou paketů pomocí metody REGISTER

Tento útok lze ovšem rozdělit do dvou částí a to na základě uživatelských jmen, které jsou k útoku využity. Pokud je pro fiktivní registraci využito neexistující uživatelské jméno je odpověď *403 Forbidden* [44], to znamená, že dané uživatelské jméno v databázi není obsaženo. Systém se tím pádem nezabývá výpočtem *nonce* a vytížení systému není tak razantní. Pokud se útočníkovi podařilo získat informace o existujících uživatelích a ty pak využije v útoku, je vytížení systému podstatně větší. V tomto případě probíhá i výpočet *nonce* pro odpověď *401* nebo *407*. Průběh komunikace je znázorněn na obrázku 3.6. Při vytížení systému pomocí existujících uživatelů se doba odpovědi podstatně prodlužuje, jelikož jsou

požadavky výpočetně náročné. Oproti tomu při použití neexistujících uživatelů, je systém vytížen, ale nedochází k tak velkému zpoždění.

Do této kategorie také spadají jiné útoky, které mají za následek vytížení systému. V rámci praktické části jsou testovány i záplavy pomocí paketů OPTIONS, ACK a BYE. Jak bude v testech dokázáno, i při záplavě pomocí paketů OPTIONS nebo BYE bylo umožněno systém odstavit minimálně po dobu útoku.

3.3.3. Útoky využívající nedokonalosti protokolů

V předchozí kapitole byly popsány útoky, které byly vytvořeny pomocí záplavy paketů. V této kapitole budou popsány útoky založené na nedokonalosti protokolů, využívají chyb, které vznikly již při návrhu systému, nebo při špatné implementaci. Útočník vygeneruje sadu speciálně upravených paketů na jednu službu a ta se začne chovat nestabilně, může nepřiměřeně vytížit procesor, či konzumovat velké množství paměti. Tyto útoky mohou využívat jak chyb operačních systémů, tak chyby samotných protokolů. Jelikož se jedná o útoky, které jsou postaveny na chybách v návrhu, nemají příliš dlouhou životnost. Většinou se jedné pouze o několik měsíců než se chybu podaří odstranit. Útoky v této kategorii jsou zde uvedeny pouze pro úplnost, většina z nich není na nových systémech příliš efektivní, speciálně pokud se jedná o UNIX systémy.

- **Ping of Death** – jedná se o poměrně známý útok, který vznikl již před několika lety. Využívá paket ICMP Echo request o velikosti vetší než 65535 bytů [31]. Takto velké pakety se v síti běžné nepřenáší, ale fragmentují se [30]. Útočník pošle tyto fragmenty k cílové stanici a po obdržení fragmentů se stanice pokusí poskládat původní paket. Některé operační systémy na to nebyly připraveny a při obdržení této zprávy kolabovaly. Stalo se tak kvůli přetečení paměti, která byla alokována pro paket. Mohlo se to stát také při přepočítávání velikosti IP datagramu, jelikož je potřeba po defragmentaci obnovit původní IP hlavičku. Právě parametr o velikosti datagramu je dvoubytový a to znamená, že datagram může být dlouhý maximálně 65535 bajtů. Pokud byl tedy datagram větší, tak při přepočítávání přetekla tato dvoubytová hodnota a systém se s tím neuměl vyrovnat [31]. V současné době je tento útok pro odstavení systému již neúčinný, jedná se ale o celkem efektivní způsob, jak zahltit komunikační kanál systému.
- **Teardrop (slza)** – jedná se o útok, který je podobný předchozímu. Útok opět spočívá v modifikaci fragmentace paketu. K oběti je poslán paket, kde se fragmenty překrývají. U dřívějších systémů mohlo při sestavování paketu dojít k nečekané chybě a systém se zhroutil. Jedná se o útok, který je v současné době již nepoužitelný [12]
- **Land Attack –** útoků tohoto typu existuje mnoho variant. Princip spočívá v modifikaci paketu, kde jako cílová a zdrojová IP adresa je použita stejná hodnota. Díky tomu se oběť ocitne v cyklu, kdy sama sobě začne zasílat pakety. To může vyústit až v pád systému. Jen pro zajímavost, tento typ útoku byl proveden na jednu ze stanic, která pracuje se systémem Windows XP, Service Pack 2. Jako varianta útoku byla zvolena záplava paketů ICMP Echo request, kde jako cílová byla zvolena IP adresa systému, stejně jako zdrojová IP adresa. Systém okamžitě přestal reagovat na jakýkoliv podnět od uživatele [66].

3.3.4. Útoky využívající nedokonalosti inicializačního protokolu SIP

Do této kategorie spadají i útoky, které využívají nedokonalosti SIP protokolu. Na rozdíl od předchozích variant, kde byl odstaven celý systém, tyto útoky jsou zaměřeny pouze na dostupnost uživatele. K tomuto útoku je ovšem vyžadováno, aby útočník měl dostatečné informace o síti a aby měl přístup ke komunikaci mezi uživateli a ústřednou. V předchozí části této kapitoly bylo popsáno jak si tento přístup zajistit, z toho důvodu bude pro další popis předpokladem, že komunikaci mezi PBX a uživateli je schopen útočník odposlechnout. Spadá sem několik typů útoků, které dokáží přerušit probíhající spojení nebo znemožnit komunikaci uživatele, jedná se o útok pomocí metody BYE, metody CENCEL a metody REGISTER.

REGISTER útok

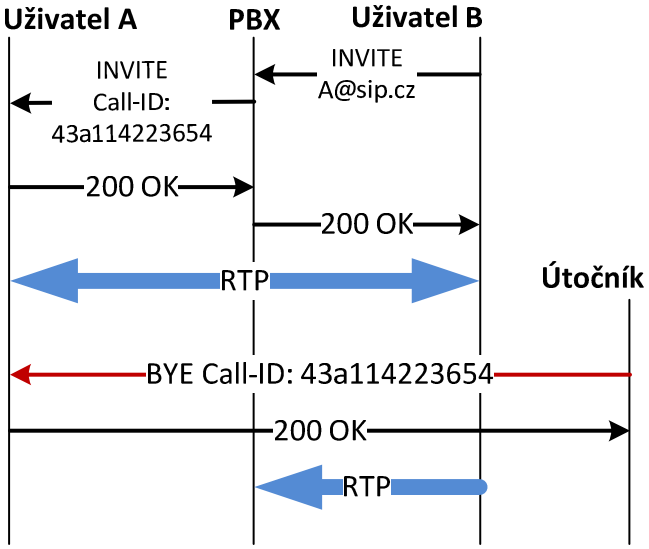
V současných VoIP systémech se registrují SIP telefony přímo k PBX ústředně. Díky tomu ústředna dokáže definovat, kam odeslat příchozí požadavek, jelikož si udržuje vazby mezi SIP účty a IP adresami [7]. Každý telefon se registruje při startu a pak v určených časových intervalech. Defaultní nastavená hodnota pro obnovení registrace je 60 minut, ověřeno pro softphone X-lite, naopak softphone Zoiper neumožňuje nastavit časový interval. Díky modifikaci SIP zprávy REGISTER může být zabráněno příchozím hovorům pro jednotlivé uživatele. K tomu, aby byla registrace odstraněna, je nutné znát identifikační číslo uživatele a jeho heslo. Dále je potřeba definovat IP adresu telefonu, kde je uživatel registrován a SIP port, na kterém naslouchá ústředna. Všechny tyto hodnoty je možné získat z odposlechu komunikace [7]. Upravená zpráva REGISTER:

```
REGISTER sip:192.168.201.148 SIP/2.0
Via: SIP/2.0/UDP 192.168.201.1:5060;branch=z9hG4bK-8296-1-0
From: "test"<sip:104@192.168.201.148>;tag=1
To: "test"<sip:104@192.168.201.148>
Call-ID: 1-8296@127.0.1.1
CSeq: 2 REGISTER
Contact: *
Expires: 0
Max-Forwards: 5
Content-Length: 0
```

Klíčovými parametry jsou *Contact: ** a *Expires: 0*, které odstraní všechny registrace pro konkrétní SIP telefon u PBX ústředny. Díky tomuto zásahu nemůže uživatel přijímat žádné příchozí hovory, dokud nebude obnovena jeho registrace. V případě telefonu X-Lite se bude jednat o 60 minut.

BYE útok

Jak již bylo uvedeno, pomocí tohoto útoku je možné přerušit již probíhající spojení mezi dvěma uživateli. Při vytvoření spojení má každý hovor své unikátní identifikační číslo, které se udržuje po celou dobu spojení. Pokud nejsou data mezi PBX a uživateli šifrována, je možné odchytit probíhající SIP komunikaci, která toto číslo obsahuje. Na obrázku 3.7 je uveden příklad, jak přerušení hovoru probíhá [62].



Obr. 3.7: Ukončení spojení BYE útok

CENCEL útok

Pomocí CENCEL útoku je možné zabránit uživateli ve vytvoření relace, pro kterou byl již odeslán INVITE požadavek. Jakmile uživatel obdrží zprávu CENCEL, zruší se vytváření spojení. Požadavek CENCEL musí obsahovat všechny důležité informace, které byly uvedeny ve zprávě INVITE. Jedná se také o způsob přerušení spojení, ovšem implementace útoku je příliš složitá a vyžaduje přesné načasování. Z toho důvodu není příliš často využíván a jedná se spíše o teoretickou možnost jak zabránit spojení [7].

3.3.5. Útoky využívající přetečení zásobníku

Jedná se o poměrně rozšířený typ útoku. Zásobník je vyrovnávací paměťový prostor, sloužící pro přechodné uložení přesouvaných dat z rychlejšího paměťového média na pomalejší, pro uložení proměnných programů, mezivýsledků, návratových adres při volání podprogramů apod. Zásobník tak plní roli jakéhosi mezičlánku (např. aby pomalejší médium mělo data stále k dispozici a aby nebylo rychlejší médium blokováno zdlouhavým přenosem) [76]. Přetečení zásobníku (*Buffer overflow*) je situace, která označuje anomálii, kdy zapíše data za konec alokované části paměti. Při přetečení zásobníku dochází k výjimce při přístupu k paměti, čímž nastává pád programu nebo spuštění nekorektní části kódu.

Existuje mnoho variant tohoto útoku. Často jsou tyto útoky zaměřeny na nové aplikace, kde existují bezpečnostní slabiny, které při vývoji nebyly objeveny. Nejčastěji k němu dochází při zpracování vstupních dat, kdy útočník posílá speciálně upravené pakety, které využívají chyb v programování, nebo nevhodného použití programovacího jazyka. Jedním z možných důsledků této chyby přitom může být přepsání nebo poškození korektních dat [62].

3.3.6. DNS zesilující útok

DNS zesilující útok (Amplification attack) patří do kategorie útoků, při němž dochází k zahlcení komunikačního kanálu oběti. Útočník odesílá velké množství DNS dotazů a jako zdrojovou adresu zvolí IP adresu oběti. Útok patří do skupiny zesilujících útoků, jelikož na dotaz je odeslána odpověď, která má 70 krát větší velikost než původní dotaz [74].

Normální DNS (Domain Name System) server pracuje s protokoly TCP a UDP. Standardně se používá protokol UDP, který umožňuje posílat DNS odpovědi do velikosti 512 bytů [5]. Útočník položí dotaz o velikosti 70 bytů na DNS server a odpověď, kterou dostane, může mít velikost až 512 bytů, dochází tak k sedminásobnému zesílení. Při použití EDNS (*Extension mechanisms for DNS*), může mít odpověď velikosti až 4 kilobyty [8]. Tímto způsobem je možné získat na relativně malý dotaz odpověď, která je až 70 krát větší.

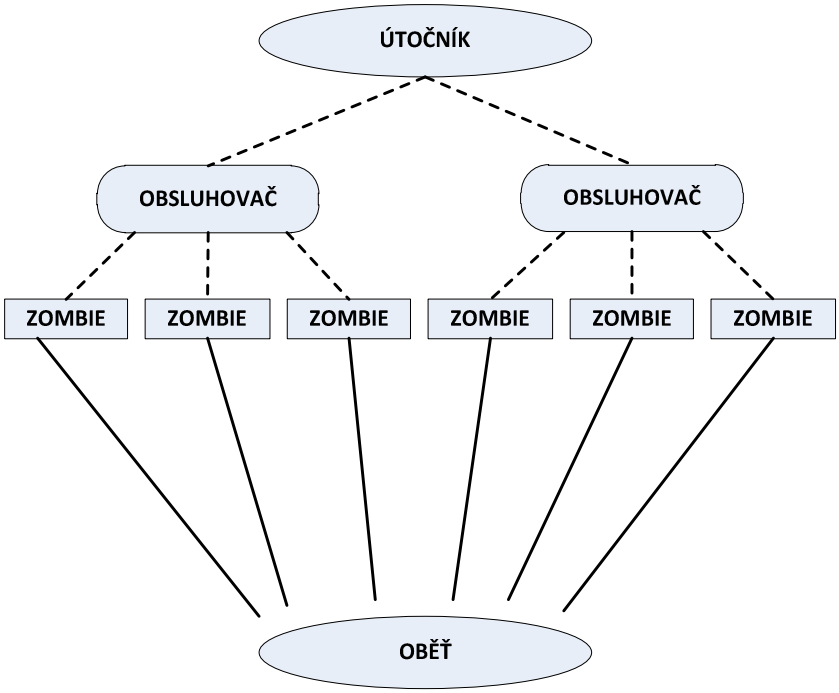
3.4. Distribuované útoky odmítnutí služby

Pro úplnost je zde uvedena i definice distribuovaných DoS útoků. DoS je útok vedený z jednoho zdroje, naopak DDoS je typ útoku, kde útočník pouze iniciuje zahájení útoku. K DDoS útoku se používá velké množství systémů s koordinovaným chováním. Tyto systémy, známé jako zombies, byly předtím kompromitovány a jsou pod kontrolou útočníků. Posíláním příkazů těmto zombies přes skryté komunikační kanály mohou útočníci zinscenovat rozsáhlé útoky.

Tyto útoky mohou být použity na jakýkoliv systém v TCP/IP architektuře, i když nejsou zaměřeny právě na SIP protokol, mohou být použity na jakoukoliv součást PBX systému a díky tomu je ovlivněna i samotná SIP komunikace [62].

3.5. Útoky zaměřené na integritu služby

RTP protokol se využívá k přenosu samotných dat mezi uživateli téměř ve všech VoIP systémech nezávisle na typu ústředny. Jeho nespornou výhodou je rychlost, jelikož pracuje na UDP protokolu. Ovšem tento protokol si sebou nese i značná bezpečnostní rizika, například odposlech komunikace pomocí běžných síťových analyzátorů je velice snadnou záležitostí.



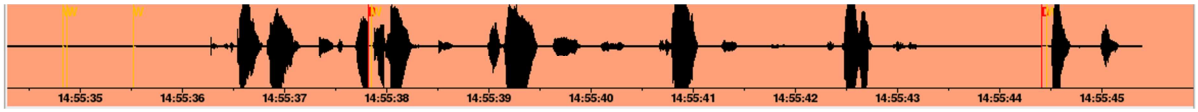
Obr. 3.8: Princip DDoS útoku

Do této kategorie spadají útoky, které jsou zaměřeny na modifikaci a odposlech již vytvořeného spojení [48]. Protokol RTP je ve své podstatě nezabezpečený a otevřený

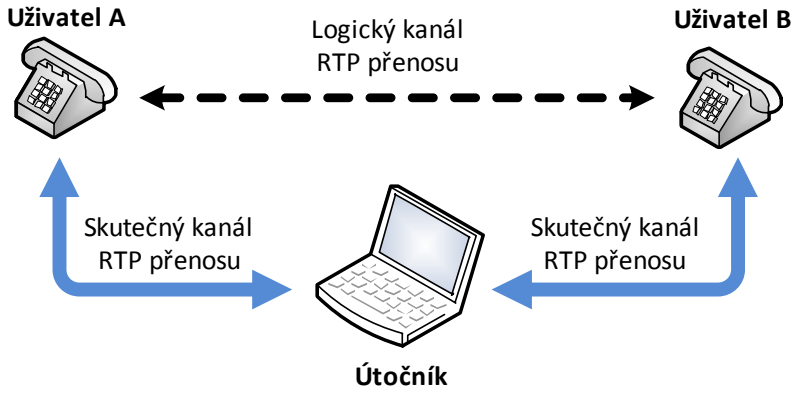
protokol, to znamená, že přenášená data jsou dostupná všem, pokud se jedná o sdílené médium, popřípadě je využita jedna z technik Man in the Middle, která byla popsána v předchozí části této kapitoly. Protokol RTP je náchylný i k dalším typům útoků jako hijacking, útoky typu DoS, spoofing a nežádoucí modifikace dat [6]. Zde budou popsány dva základní typy útoku, které souvisejí s RTP protokolem v IP telefonii.

3.5.1. Pasivní odposlech RTP dat

Protokol RTP může být odposlechnut stejným způsobem jako například telnet, nebo FTP. [6] RTP protokol přenáší audio data popřípadě video data, proto je při odposlechu důležité odchytit co nejvíce paketů. Zachycení dat a následná jejich rekonstrukce je ve VoIP prostředí poměrně snadná. K samotnému odchycení dat není potřeba znát žádné heslo nebo uživatelské jméno. K odposlechu stačí zajistit jen přístup k síti. K odchycení komunikace může posloužit jakýkoliv nástroj, který dokáže zachytit RTP pakety, například Wireshark. Z RTP streamu je následně možné získat úplné znění hovoru, které je možné přehrát či uložit, viz obrázek 3.9 a 3.10.



Obr. 3.9: Záznam hovoru



Obr. 3.10: Odposlech RTP přenosu

Celý průběh komunikace probíhá v otevřené podobě, tedy přístupné všem. Jediná část, která je podrobena zabezpečení, je autentizace účastníka, kde je heslo posíláno v zašifrované formě, metodou digest autentizace.

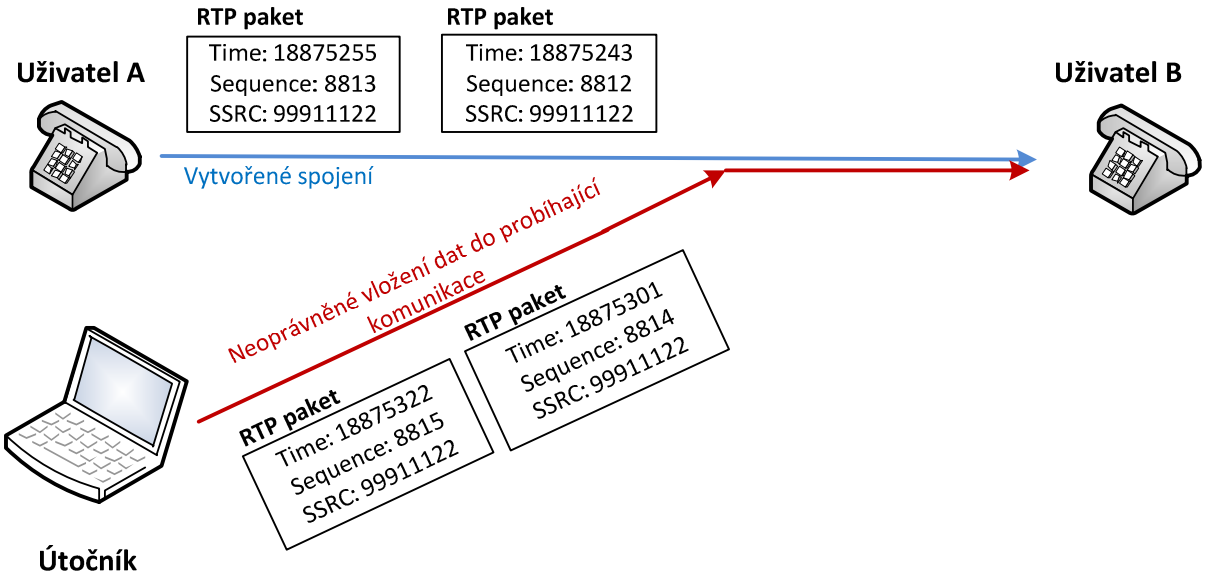
3.5.2. Aktivní modifikace přenosu RTP dat

Jedná se o útok, kdy je existující hovor neoprávněně modifikován třetí stranou. Tato modifikace může být dvojího způsobu, první je vložení dat přímo do hovoru, druhou možností je nahrazení části hovoru [6]. Využití může být různorodé, do hovoru může být vložen šum, nevhodná slova nebo například reklama. V obou případech dochází k manipulaci s časovou značkou (timestamp), informacemi o spojení, s identifikací zdroje dat (SSRC) RTP paketu.

Neoprávněné vložení dat do probíhající komunikace

Informace o spojení mezi uživateli je kontrolováno pomocí identifikací zdroje dat (SSRC), to zaručuje původce dat. Jedná se o náhodně vygenerované 32-bitové číslo. Dále je zde obsažena timestamp, která zaručuje, že data jsou přehrána ve správném pořadí [48]. Jelikož jsou informace o spojení přenášena v otevřené podobě, může je útočník odchytit. Pokud má k dispozici tyto údaje, může určit časové hodnoty, které budou následovat. Hodnota SSRC je určena dané relaci, takže je možno ji použít bez jakýchkoliv úprav [7].

Kombinace toho, že jsou informace o spojení přenášeny ve formě prostého textu, a že jediným poznávacím činitelem je timestamp, jsou vytvořeny podmínky pro RTP injection. Příklad neoprávněného vložení dat je uveden na obrázku 3.11.



Obr. 3.11 Neoprávněné vložení dat do probíhající komunikace

Neoprávněné nahrazení dat v probíhajícím hovoru

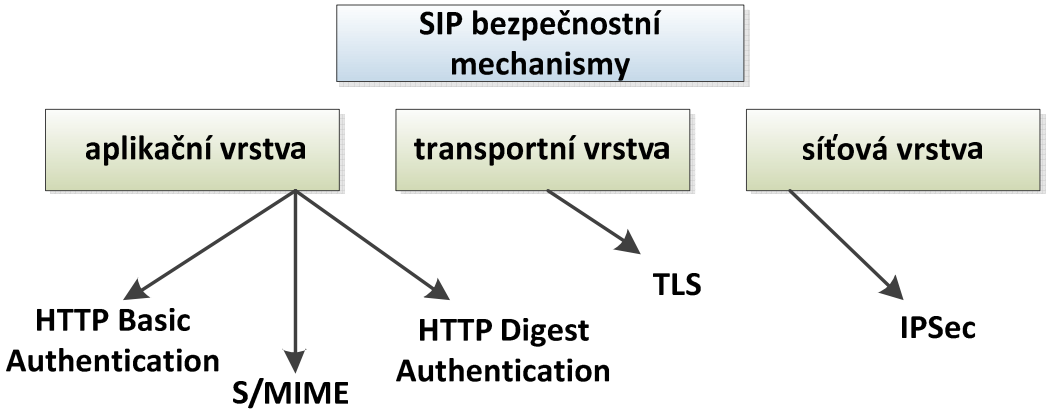
Stejně jako u předchozího útoku jsou i zde potřebné informace o SSRC, číslu sekvence (detekuje ztráty paketů) a časové značce paketu. Při tomto útoku dochází k nahrazení existujících audio dat třetí stranou. V předchozím případě se jednalo o vložení dat do probíhající komunikace, to znamená, že data byla přehrána na pozadí hovoru.

Při tomto útoku útočník odešle data k oběti, které v sobě obsahují informace o timestamp, čísle sekvence a SSRC. V porovnání s legitimním přenosem od uživatele, mají útočníkova data daleko vyšší číslo sekvence a časová značka je novější v porovnání s daty, které obdržel od uživatele. Oběť tedy dostane data jak od uživatele, tak od útočníka. Při srovnání zjistí, že data od útočníka jsou novější, obsahují vyšší číslo sekvence a aktuálnější časovou značku. Z těchto důvodů se oběť rozhodne použít právě data od útočníka, protože dle její logiky jsou data více aktuální. Pakety od legitimního uživatele budou zahozeny, protože již neobsahují aktuální informace [6].

4. Zajištění bezpečnosti open source PBX systémů

4.1. SIP security

Vytvoření, sestavení a management relace probíhá odděleně od multimediálního přenosu prostřednictvím RTP protokolu, oba tyto samostatné proudy dat musí být zabezpečeny odděleně [44]. Existuje několik různých mechanismů pro protokol SIP s cílem zajistit potřebné požadavky pro autenticitu, integritu a důvěrnost dat. SIP protokol vychází z již existujících standardů, takže podporuje opětovné použití existujících bezpečnostních mechanismů, které využívá například protokol HTTP a SMTP. Mezi technologie, které je možno použít pro zabezpečení protokolu SIP patří: HTTP autentizace, S/MIME a protokoly IPsec a TLS [62].



Obr. 4.1: Zabezpečení SIP protokolu

4.1.1. HTTP Basic Authentication

Nejméně bezpečná metoda autentizace. SIP server vyzve uživatele k autentizaci pomocí jména a hesla. Ovšem registrační údaje jsou přenášeny jako nešifrovaný text v SIP zprávě. Tato metoda vychází z předpokladu, že spojení mezi serverem a klientem je bezpečné po celou dobu přenosu [44].

4.1.2. HTTTP Digest autentizace

Jedná se o nejrozšířenější metodu autentizace. Metoda je založena na systému výzva - odpověď (challenge-response). Autorizační proces pracuje na stejném principu jako HTTP Digest Access Authentication [17]. Jakmile User Agent odešle požadavek REGISTER nebo INVITE k serveru, který vyžaduje autorizaci, dostane odpověď *401 Unauthorized* nebo *407 Proxy Authentication Required* (záleží na typu zařízení). Tyto zprávy 401 a 407 indikují, že UA se musí identifikovat [62].

Při procesu autentizace mechanismem SIP digest je zapotřebí projít následujícími kroky:

1. Uživatel odešle požadavek ke komunikaci (prostřednictvím metody REGISTER, INVITE, nebo jinou metodou).
2. Server (Registrar nebo SIP Proxy server) odpoví zprávou *401* nebo *407 Unauthorized response*, která obsahuje unikátní kombinaci *nonce*, která má být použita pro autorizační proces.
3. Uživatel musí provést následující tři akce, aby mohl odeslat správnou odpověď ve formě MD5 heše zpět na server, který má za úkol potvrdit uživatele:
    - *první krok*: vytvoří se heš kombinací username, realm a password
      MD5 (Username : Realm : Password);
    - *druhý krok*: vytvoří druhý MD5 heš skládající se z použité SIP metody a z URI
      MD5 (Method : URI);
    - *třetí krok*: User Agent vytvoří MD5 haš, který bude odeslán v poli response. Jedná se o kombinaci z předchozích dvou kroků, spolu s hodnotou nonce
      MD5 (MD5-step-3 : nonce : MD5-step-4).
4. Klient odešle výsledný heš na server jako svou odpověď.
5. Server provede stejné kroky, jako klient. Pokud odpověď od uživatele souhlasí, tak MD5 heš byl skutečně vytvořen klientem, za kterého se vydával. Server odešle potvrzení, že heslo souhlasí a uživatel bude autentizován.

Pro demonstraci, jakým způsobem tento proces ve skutečnosti funguje, je odchycena komunikace mezi uživatelem a PBX ústřednou, viz obrázek 4.2. Výpočet probíhá obdobným způsobem, jak bylo uvedeno v předchozí části.

$$
\begin{aligned}
&Response = \\
&= MD5(MD5\ (Username : Realm : Password) : nonce : MD5\ (Method : URI)) = \\
&\quad = MD5\big(MD5(101 : asterisk : Zaq12wsx) : 187a82ee : MD5(REGISTER : sip : 101@192.168.201.148)\big) \\
&\qquad\qquad = 67c28de4673938476035553adc449b06
\end{aligned}
$$

Obr. 4.2: HTTTP Digest autentizace

4.1.3. Secure Multipurpose Internet Mail Exchange (S/MIME)

Specifikace MIME (*Multipurpose Internet Mail Extensions*) [38] rozšířily formát SMTP tak, aby bylo možné přenášet zprávy obsahující více textových částí a také části binární, jako audio a video. Díky tomu je vhodný také pro VoIP komunikaci. MIME však neobsahoval žádné služby bezpečnostního charakteru, proto byl vytvořen S/MIME (*Secure MIME*). Nástavba S/MIME slouží k zabezpečení požadované služby a přitom nebude narušena vzájemná spolupráce různých implementací [50]. Opírá se o využití celé sady již existujících doporučení a norem. S/MIME využívá CMS (*Cryptographic Message Syntax*) pro digitální podpisy a šifrování a je založen na syntaxi podle PKCS#7 [39]. Toto doporučení využívá

k šifrování standardu symetrického šifrování AES (*Advanced Encryption Standard*), jehož technika šifrování je dostatečně rychlá na to, aby mohl být použit v komunikaci v reálném čase. Autentizace je zajištěna pomocí architektury veřejných klíčů. Implementace je složitější než u TLS, jelikož je potřeba, aby certifikační autorita vydávající certifikáty byla důvěryhodná pro všechny účastníky komunikace. Aby bylo možné nasadit S/MIME, musí každý User Agent vlastnit identifikační certifikát s veřejným a soukromým klíčem, který se používá k označení anebo šifrování informací obsažených v SIP paketu [44].
Pokud uživatel A chce odeslat SIP paket s použitím S/MIME uživateli B, musí zašifrovat tělo zprávy paketu použitím veřejného klíče od uživatele B. Příjemce B zprávu rozšifruje svým soukromým klíčem, který je jedinečný a je to jediný klíč, který může rozšifrovat přijatou zprávu. Díky tomu jsou data zabezpečena po celou dobu přenosu. Pro oba uživatele musí existovat zabezpečená komunikace pro vzájemnou výměnu klíčů a certifikátů, které slouží k rozšifrování uživatelské komunikace [62].

4.1.4. Transport-layer security

Protokol TLS (*Transport-layer security*) zajišťuje zabezpečení na úrovni transportní vrstvy pro spojově orientované protokoly jako je TCP. V RFC 5246 [53] je definována aktuální verze 1.2. Protokol TLS stejně jako IPSec splňuje tří základní pravidla bezpečnosti mezi dvěma koncovými body. Protokol TLS zajišťuje vzájemnou autenticitu (oba účastníci spojení jsou si jisti, s kým komunikují) prostřednictvím certifikátů, důvěrnost šifrováním.

Protokol TLS zahrnuje tři základní fáze:

- vzájemná dohoda o použitém šifrování a data-integrity algoritmu;
- výměna klíčů a vzájemná autentizace pomocí techniky veřejných klíčů;
- přenos šifrován pomocí symetrické šifry.

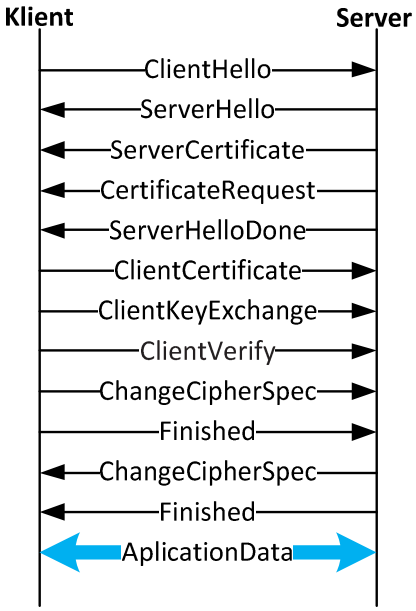
Podporované šifrovací mechanismy:

- pro kryptografii s veřejným klíčem: RSA, Diffie-Hellman, DSA;
- pro symetrické šifrování: RC2, RC4, IDEA, DES, Triple DES, AES, Camellia;
- pro jednosměrné hešování: Message-Digest algorithm (MD2, MD4, MD5), Secure Hash Algorithm (SHA-1, SHA-2).

Navázaní spojeni pomocí TLS

Průběh komunikace pomocí TLS je znázorněn na obrázku 4.3. Klient pošle zprávu *ClientHello*, kde oznamuje serveru požadavek na zahájení komunikace. Zpráva obsahuje informace o tom, jaká je nejvyšší podporovaná verze TLS, jaké je náhodné číslo, seznam doporučených šifrovacích sad a kompresních metod. Server odpoví svou zprávou *ServerHello*, kde jsou uvedeny informace, které budou použity k sestavení spojení. Server pošle svůj certifikát ve zprávě *ServerCertificate* a pokud je to nastaveno, je odeslána zpráva, které vyžaduje certifikát od klienta *CertificateRequest*. Vzápětí server odešle zprávu *HelloDone*, která signalizuje, že server dokončil úvodní relaci. Klient odpoví k serveru zprávou *ClientCertificate* se svým certifikátem a zprávu *ClientKeyExchange*, která obsahuje tajný klíč zašifrovaný pomocí RSA a veřejného klíče. Následně klient odešle zprávu *ClientVerify*, která ještě jednou ověří odesílatele, pokud je to vyžadováno. Zpráva *ChangeCipherSpec* oznamuje, že veškerá další komunikace bude šifrována. Na závěr klient pošle šifrovanou zprávu *Finished* obsahující heš a MAC předchozích iniciačních zpráv. Po

ověření je odeslána i serverem zpráva *ChangeCipherSpec* a zpráva *Finished*. V tomto okamžiku proběhly všechny ověření v pořádku a je zahájena šifrovaná komunikace požadovaným protokolem na bázi TCP spojení, v tomto případě SIP [62].



Obr. 4.3 Zahájení spojení pomocí TLS

4.1.5. Bezpečnostní rozšíření IP protokolu

IPSec (*IP security*) je sada protokolů síťové vrstvy vyvinutá organizací IETF k podpoře bezpečné výměny paketů v IP vrstvě. IPSec podporuje tří základní pravidla bezpečnosti: autenticitu, integritu a důvěrnost. Může pracovat ve dvou režimech: transportním a tunelovém. V transportním režimu zajišťuje ochranu pro protokoly vyšších vrstev (UDP, TCP a další). Tunelový režim zajišťuje ochranu pro data v IP záhlaví a využívá se k vytvoření tunelových spojení mezi dvěma stanicemi či bránami. IPSec architektura byla definována v roce 1998 RFC 2401 [41] ovšem existuje i mladší verze z roku 2005 RFC 4301 [51], zde je implementována podpora IKEv2 (*Internet Key Exchange standard version 2*) [52].

Bezpečnostní rozšíření IP protokolu (IPSec) provádí certifikaci pomocí metody veřejných klíčů, nebo přednastaveného zabezpečení, které je uloženo na zabezpečené stanici či bráně. Při implementaci certifikace metodou klíčů se používá IKE (*Internet Key Exchange*) [42] protokol, založený na Diffie-Hellmanovém algoritmu.

Příklad využití IPSec při komunikaci prostřednictvím SIP protokolu může být zabezpečené spojení mezi dvěma SIP uzly nebo dvěma SIP sítěmi. V tomto případě zabezpečené stanice či brány musí mít přístup ke klíčové infrastruktuře, nebo musí být přednastaveny kritéria zabezpečení. Dalším příkladem může být komunikace mezi UA agentem a SIP sítí. Jedná se například o 3GPP IMS (*Internet Multimedia Subsystem*). Tajný klíč je uložen na bezpečnostní kartě UICC (*Universal Integrated Circuit Card*), autentizace a klíčová dohoda je prováděna prostřednictvím AKA (*Authentication and Key Agreement*) protokolu [33].

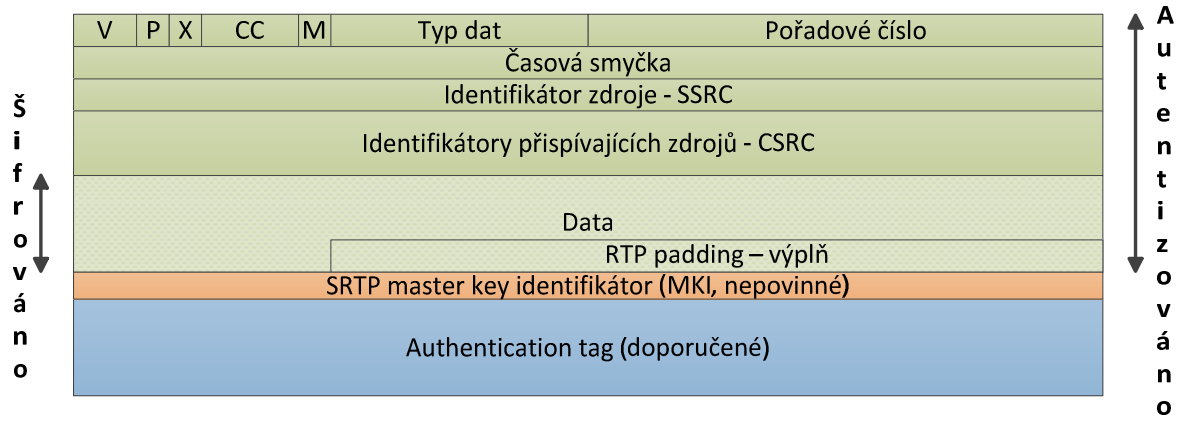
5. Obrana proti odposlechu

Odposlech hovorů probíhajících přes RTP protokol je velmi jednoduchý, k odchycení dat stačí pouze síťový analyzátor jako například Wireshark. Tento bezpečnostní problém řeší zabezpečené protokoly SRTP a SRTCP.

5.1. Zabezpečení multimediálního přenosu - SRTP/SRTCP protokol

Pro protokoly RTP nebo RTCP nejsou definovány žádné bezpečnostní mechanismy, které by zajišťovaly integritu, autentičnost a důvěrnost dat [48]. Z toho důvodu byly vytvořeny protokoly SRTP (*Secure RTP*) a SRTCP (*Secure RTCP*) [49], které zajišťují bezpečný přenos. U protokolu SRTP je šifrována pouze část, kde se nacházejí data, jak je možno vidět na obrázku 5.1 znázorňujícího strukturu SRTP paketu.

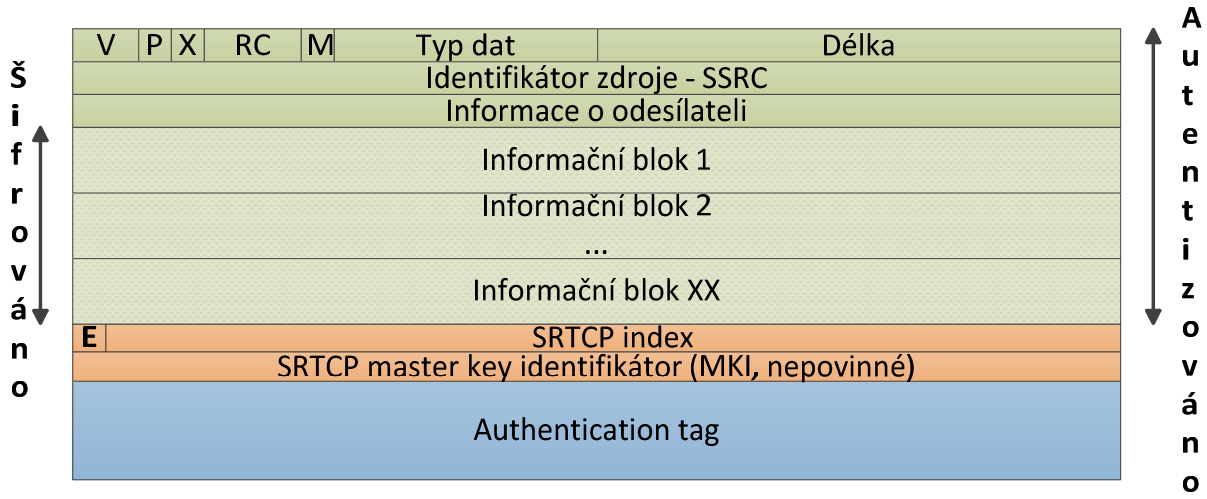
Oproti paketu RTP jsou zde navíc dvě pole – MKI (*Master Key Identifier*) a Authentication tag. Authentication tag je šifrovaný kontrolní součet hlavičky a těla RTP paketu, který zajišťuje integritu dat. Pokud by hodnota Authentication tagu neodpovídala skutečné hodnotě, znamenalo by to, že obsah dat je pozměněn. Pole MKI definuje hlavní klíč, z kterého jsou odvozeny tajné symetrické klíče. S takto odvozeným klíčem jsou následně v dané relaci data šifrována po celou dobu spojení. Na začátku relace se mezi účastníky spojení vždy dohodne, jakým klíčem budou šifrována data, a tento klíč se pak používá po celou dobu spojení. Nejdříve si ovšem komunikující strany musí vyměnit master key, pomocí kterého si pak vygenerují všechny potřebné klíče pro dané spojení [49]. K výměně master key je možno využít zabezpečeného přenosu pomocí TLS [53] nebo IPsec [21].



Obr. 5.1: Struktura SRTP paketu

Protokol SRTCP využívá stejného zabezpečení jako SRTP, ale na rozdíl od SRTP je zde pole Authentication tag povinné. Paket protokolu SRTCP navíc obsahuje pole index, které slouží jako čítač SRTCP paketů, což lze využít k obraně před opakovanými útoky. K zajištění důvěrnosti přenášených dat se používá generátor pseudonáhodných klíčů AES-CTR (counter mode). Protokol RTP ani SRTP nezaručují doručení dat v pořadí, v jakém byla data odeslána. Algoritmus AES-CTR je vhodný pro multimediální nepotvrzované přenosy [62]. K zajištění autentičnosti dat se používá šifrovací metoda HMAC-SHA-1. Tímto algoritmem je vytvořen kontrolní součet z hlavičky a obsahu SRTP paketu, tato hodnota se pak uloží do pole

authentication tag. Vzhledem k tomu, že je při přenosu kladen důraz na co nejmenší šířku přenosového pásma, je výsledný kontrolní součet zkrácen na 80 nebo 32 bitů [34].



Obr. 5.2: Struktura SRTCP paketu

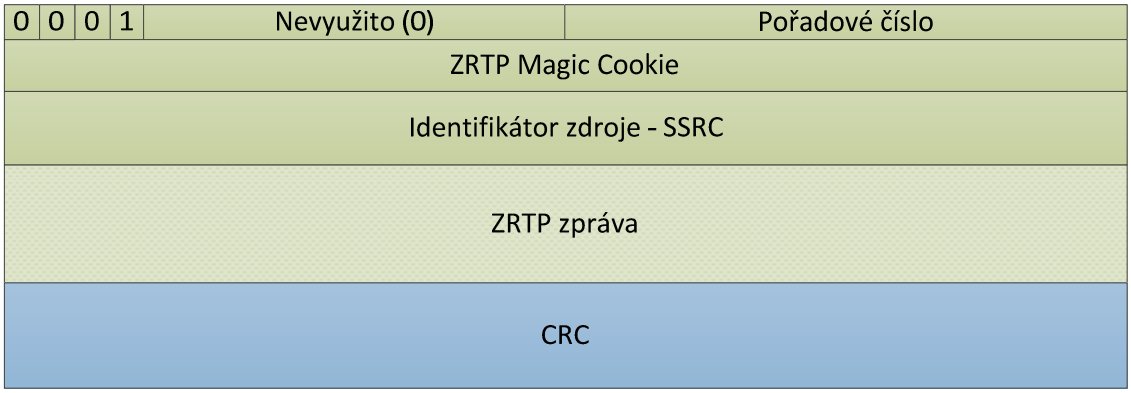
5.2. Zabezpečení multimediálního přenosu - ZRTP protokol

ZRTP je další nástavbou protokolu RTP, jedná se o protokol, který umožňuje vytvořit bezpečný kanál, kde jsou dohodnuty parametry pro spojení protokolem SRTP. Jinak řečeno nejedná se o šifrovací protokol, ZRTP pouze umožňuje bezpečný mechanizmus pro výměnu klíčů, díky tomu odpadá potřeba bezpečnostní autority [15].

K výměně symetrických klíčů je využit Diffie-Hellamnův algoritmus, díky tomu může ZRTP fungovat bez podpory PKI (*Public Key Infrastructure*). ZRTP také spojuje několik bezpečnostních mechanismů, které snižují riziko proti útoku metodou Man in the Middle. Výměna klíčů prostřednictvím ZRTP probíhá na stejném portu jako běžná RTP relace. Každá strana si před zahájením komunikace vypočte soukromý a veřejný klíč. Na základě výměny veřejných klíčů jsou dopočteny symetrické klíče, kterými jsou šifrována a dešifrována data. Pro ověření pravosti klíčů je využito krátkých ověřovacích zpráv SAS (*Short Authentication String*). Tímto způsob je zabezpečen i útok MitM, kde útočník nemá informace o klíčích a tím pádem nemůže dešifrovat data [54]. Struktura ZRTP paketu je zobrazena na obrázku 5.3. Každý paket je stejný, liší se jen pole ZRTP zpráva, kde jsou uložena samotná data.

Popis jednotlivých polí je následující:

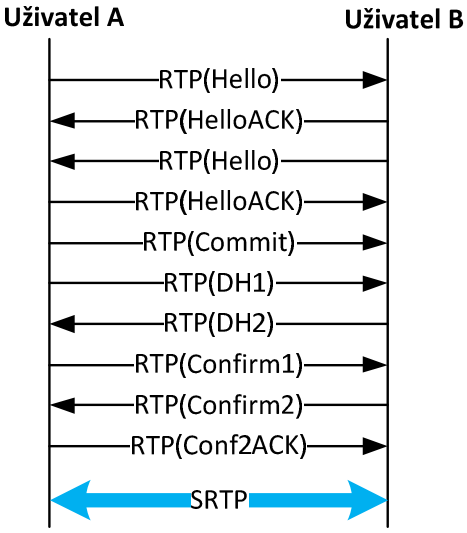
- **Pořadové číslo** - jedná se o čítač paketů, stejná funkce jako v případě RTP;
- **ZRTP Magic Cookie** - řetězec, který identifikuje ZRTP paket, jeho velikost je 32 bitů;
- **ZRTP zpráva** - obsahuje samotná data k přenosu.



Obr. 5.3: Struktura ZRTP paketu

Sestavení probíhá stejným způsobem jako při klasickém protokolu RTP/RTCP. Po sestavení spojení je odeslána zpráva ZRTP Hello, která ověří, zda oba uživatelé podporují ZRTP protokol a také se zde definují společné algoritmy. Zpráva Hello ZRTP obsahuje také všechny potřebné informace pro sestavení SRTP spojení. Po zpracování zprávy uživatelem, je odeslána odpověď ZRTP HelloACK, která slouží jako potvrzení dohodnutých parametrů. Po výměně těchto dvou zpráv začíná vyjednávání tajného klíče Master Key [15]. Princip komunikace je zobrazen na obrázku 5.4.

Nejdříve se uskuteční výměna Hello paketů, ve kterých jsou přenášeny informace o nastavených bezpečnostních parametrech. Příjem Hello paketů je potvrzen druhou stranou komunikace paketem HelloACK. Následuje výměna několika Commit paketů, ve kterých se domlouvají bezpečnostní algoritmy pro zabezpečený přenos SRTP paketů.
Poté, co se komunikující strany dohodnou na bezpečnostních parametrech, dojde k výměně paketů DH, které obsahují veřejné klíče uživatelů. Každá strana si z veřejného klíče vypočítá svůj tajný symetrický klíč a hodnotu SAS, kterou pak ve formě heše pošle druhé straně ve zprávě Confirm. Pokud se obě hodnoty SAS shodují, je odeslán paket Conf2ACK protistraně. Po úspěšné výměně symetrického klíče jsou všechny následující RTP pakety šifrovány jako SRTP [54].



Obr. 5.4: Komunikace prostřednictvím protokolu ZRTP

6. Obrana proti útoku typu DoS

6.1. Systémy detekce a prevence průniku

IDS (*Intrusion Detection System*) je systém detekce průniku, určený k monitorování provozu a odhalení podezřelých aktivit jak v síti, tak na straně systému. Podstata vychází z toho, že činnost narušitele je odlišná od běžné činnosti uživatelů. Dle definované konfigurace umožňují monitorování prakticky jakékoliv aktivity jak už v síti nebo na straně systému [11]. IDS umožňuje pouze detekci a zaznamenávání podezřelých aktivit, samo neprovádí žádnou akci k zamezení nelegitimních aktivit. IDS nedetekuje jen finálními pokusy o prolomení bezpečnosti, ale umožňuje i detekci akcí, které jim předcházejí. Mezi ně patří například skenování portů, sbírání informací potřebných k útoku, atd. Hlavním prvkem IDS je senzor, který obsahuje mechanismy pro detekci škodlivých a nebezpečných kódů. IDS systém po detekci neobvyklého chování zaznamená podezřelé chování do logu a může informovat správce sítě o detekci narušení. Sám ovšem neprovede žádnou akci k zabránění průniku[13].

K provedení rychlého a účinného opatření nebo zamezení již probíhajícího útoku existují systémy IPS (*Intrusion Prevention System*). Jedná se o systémy prevence, které na základě definovaných pravidel dokáží zamezit již probíhajícímu útoku [11]. Kombinace těchto dvou zařízení je označováno jako IDPS (*Intrusion Detection and Prevention Systems*) nebo také aktivní IDS systémy. Jedná se o prvky, které dokáží detekovat i zabezpečit jak sítě LAN, tak mohou pracovat na operačních systémech. Obecně lze systémy IDS a IPS dělit do dvou kategorií, hostitelské systémy HIDS (*Host-based Intrusion Detection Systems*) a síťové systémy NIDS (N*etwork based Intrusion Detection Systems*) [13].

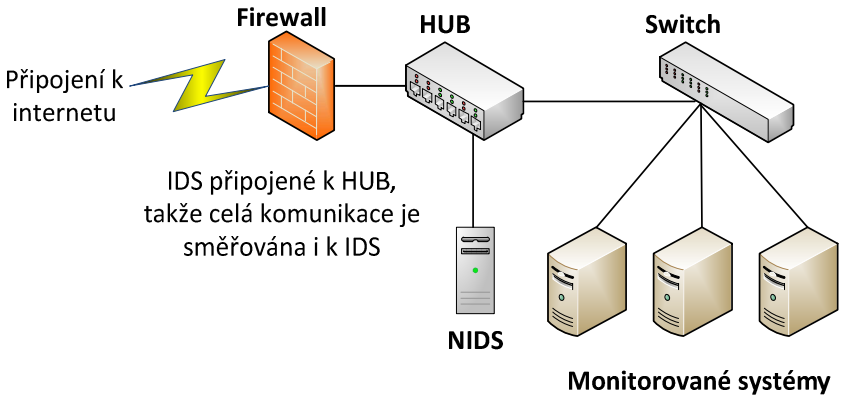
6.2. Síťové detekční systémy

NIDS jsou velmi rozšířené, jelikož dokáží nabídnout detailní analýzu provozu v síti. Tyto systémy monitorují pakety na síti a hledají neobvyklou aktivitu. Pracují na základě známých signatur útoků, nebo na základě detekce protokolových anomálií. Za stěžejní vlastnosti těchto systémů lze označit především kompletní inspekci paketů až po aplikační vrstvu, sledování provozu nejen na hranici sítě, ale i inspekci vnitřního provozu LAN a monitoring nestandardních projevů sítě bez předchozí přesné definice průniku [11].

Bezpečnostní metody systémů NIDS lze rozdělit přibližně do těchto tří hlavních oblastí:

- detailní inspekce všech paketů (ať již mezi LAN a WAN, tak i pouze v rámci LAN) dle definovaných signatur, tj. definovaných známých řetězců,
- kontrola portů / protokolů / adres,
- komplexní sledování provozu sítě.

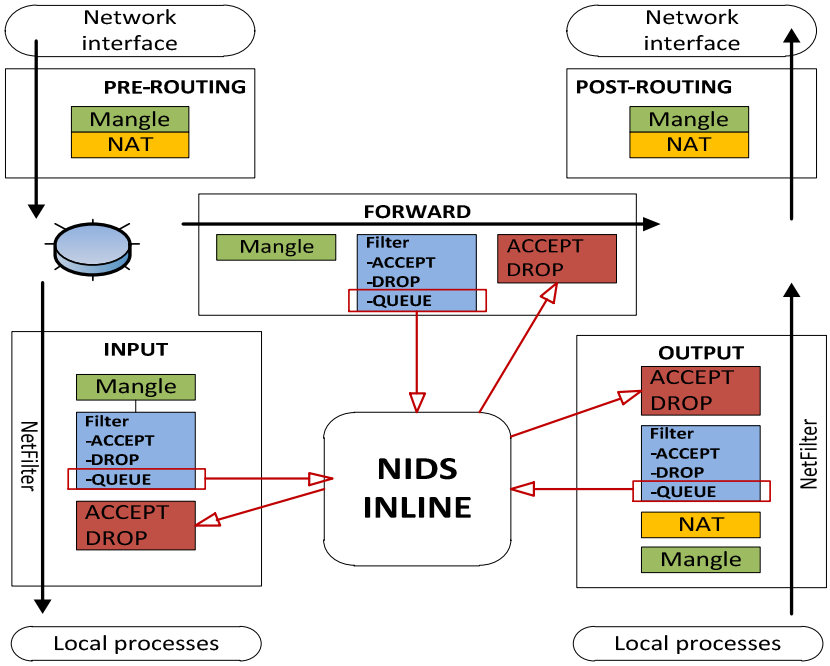
IDS může pracovat v několika různých konfiguracích. Pokud je v síti HUB je možné připojit IDS na jeden z portů a u síťové karty neuvádět žádnou IP adresu, díky tomu IDS může odposlouchávat celou komunikaci v síti, viz obrázek 6.1.



Obr. 6.1: Zapojení IDS systému v LAN síti

Další možností je připojit NIDS na switch, který obsahuje takzvaný spanning port (SPAN). Pokud switch tento port obsahuje, je veškerá komunikace v síti zrcadlena [11]. SPAN port neumožňuje připojenému zařízení komunikovat, pouze na něj kopíruje veškerou komunikaci [32].

Další možností je zapojit NIDS takzvaně inline, což znamená, že veškerá komunikace prochází právě přes NIDS, které v tomto případě pracuje jako HIDS [37]. Na obrázku 6.2 je znázorněn příklad takového zapojení, NIDS zde spolupracuje s firewallem NetFilter (iptables) [27]. Jamile NIDS obdrží paket od firewallu porovná jej s definovanými pravidly. Pokud odpovídá pravidlu, vykoná příslušnou akci a to buď paket vrátí zpět NetFilteru s příznakem DROP anebo ACCEPT, pokud paket neodpovídá žádnému pravidlu. Dále může zaznamenat informaci o paketu do logu k pozdější analýze. Existuje mnoho variant NIDS systémů jak v komerční, tak v open source sféře. Další popis bude zaměřen právě na jeden z open source NIDS systémů, Snort [26].



Obr. 6.2: Konfigurace systému NIDS jako INLINE

6.2.1. Snort – detekce a prevence průniku na úrovní sítě

V dnešní době se jedná o jeden z nejlepších IDS/IPS systémů, i při porovnání s komerční konkurencí. Snort je IDS systém, kde se k analýze provozu v síti používají přednastavená pravidla, na základě kterých jsou provedeny odpovídající akce. Snort může pracovat ve třech různých režimech: sniffer mode, packet logger mode a network intrustion detection system mode (NIDS, režim detekce narušení) [37]. V této práci bude stěžejní právě režim NIDS.

Jak již bylo uvedeno, Snort je systém založený na pravidlech. Aby mohl efektivně analyzovat síť, je třeba mít nastavené odpovídající pravidla. Každá síťová konfigurace je odlišná a v každé síti může probíhat odlišný provoz, proto je nutné nastavit pravidla pro konkrétní případ [37].

Struktura pravidel

Každé pravidlo je rozděleno do dvou sekcí: hlavička (header) a volby pravidla (options). Hlavička pravidla obsahuje akci, protokol, cílovou a zdrojovou adresu a zdrojový a cílový port. Volby (*options*) pravidla obsahuje mimo jiné zprávy určené k záznamu do logu, podmínky pravidla nebo informace, v které části paketu by se měla provést kontrola [64]. Obecný zápis pravidla:

```
action proto src_ip src_port direction dst_ip dst_port (options)
```

Tab. 7.2: Snort pravidla – možnosti nastavení [64]

options	popis
msg	zpráva, která je zaznamenána do logu
reference	jako referenci umožňuje použít jiná IDS
sid	identifikace pravidla
rev	číslo revize pravidla
classtype	označuje klasifikaci události
priority	číslo definuje prioritu útoku
logto	zaznamená paket do uživatelem definovaného souboru
ttl	parametr životnosti paketu
id	testuje ID pole z hlavičky IP fragmentu
dsize: [>|<]	velikost dat paketu, umožňuje porovnat velikost
content	hledá konkrétní hodnotu v paketu
nocase	nerozlišovat malá a velká písmena
offset	posunutí počáteční pozice pro hledání vzoru
depth	hloubka datové části k hledání
flags	položka flags v TCP hlavičce
seq	testuje TCP číselné sekvence na specifickou hodnotu
itype	testuje ICMP typ pole na specifickou hodnotu

Tab. 6.1: Snort pravidla - seznam možných akcí [64]

akce	popis
alert	vytvoří a zaznamená upozornění a na základě zvolené metody vytvoří akci
log	paket je zaznamenán v logu
pass	ignoruje paket
activate	vytvoří upozornění a použije dynamické pravidlo
dynamic	zadrží paket, dokud nebude provedena příslušná akce a pak je paket zaznamenán do logu
drop	zahodí paket a je vytvořen záznam v logu
reject	blokuje paket, zaznamená informaci do logu a pošle zprávu *TCP RTS*, nebo *ICMP Destination Unreachable*

Podporované protokoly : TCP, UDP, ICMP, IP
Orientace provozu : -> , <>

Příkladem může být následující pravidlo. Hlavička určuje, kdo paket poslal, kam ho poslal, o jaký typ paketu se jednalo a jaká akce má být vykonána. Volba pravidla obsahuje zprávu, která se má v paketu hledat (*content*) a zprávu (*msg*), která bude zaznamenána do logu společně s výstrahou [64].

```
alert tcp any any -> any 80 (msg:"EXPLOIT ntpdx overflow"; \
dsize:>128; classtype:attempted-admin; priority:10 );
```

SnortSam

Jak bylo uvedeno v předchozí části, NIDS systémy mohou pracovat v rozdílných konfiguracích. Snort může pracovat ve všech zmíněných, ovšem existuje zde i možnost, kdy Snort podle svých pravidel dokáže modifikovat firewall konkrétního systému. K tomu účelu je využit SnortSam, který dokáže propojit Snort s firewallem [65]. Kombinace Snort a SnortSam vytvoří aktivní NIDS (NIPS), který dokáže reagovat na nežádoucí komunikace prakticky okamžitě. Díky tomu vznikne efektivní ochrana před nežádoucí komunikací, jako je například DoS útok [37]. SnortSam se používá jako agent na straně klienta, který naslouchá příchozím požadavkům od centrálního prvku Snort. Pokud nějaká aktivita v síti odpovídá pravidlům v konfiguraci Snort systému, je provedena odpovídající akce. SnortSam v době psaní podporoval následující firewally [65]:

- Checkpoint Firewall-1
- Cisco PIX firewall
- Cisco Router
- Former Netscreen, nyní Juniper firewalls
- IP Filter (ipf)
- FreeBSD's ipfw2 (verze 5.x)
- OpenBSD's Packet Filter (pf)
- Linux IPchain
- Linux IPtable
- Linux EBtable
- WatchGuard Firebox firewall
- 8signs firewalls (WIN)
- MS ISA Server firewall/proxy (WIN)
- CHX packet filter

6.3. Detekce průniku na úrovni operačního systému

K detekci průniku na úrovni operačního systému slouží zařízení označované jako HIDS (*Host-based Intrusion Detection Systems*). Tyto zařízení pracují obdobným způsobem jako NIDS, ovšem s tím rozdílem, že monitorovaným prostředím jsou přímo operační systémy. Díky tomu je možné mít centrální přehled o všech událostech, jež se v systému odehrály [11]. Dalším rozdílem může být monitorování šifrovaného přenosu, které NIDS systémy nemohu monitorovat. Oproti tomu HIDS zařízení může mít senzor, který dokáže data analyzovat před šifrováním a následně po rozšifrování. Tímto způsobem je zajištěna integrita dat, jelikož HIDS systém může dle nastavených pravidel kontrolovat, zda nedošlo při přenosu ke změně. Díky HIDS systému je možné monitorovat nejen operační systémy, ale nabízí se také možnost monitorování registrů, logů, detekce rootkitů a monitoring podezřelé aktivity [13].

Kontrola Integrity

Každý soubor v operačním systému generuje svůj unikátní digitální otisk (*fingerprint*), který se nazývá kryptografický heš. Tato unikátní značka je generována na základě názvu souboru a jeho obsahu [10]. Pokud je nutné zajistit, aby důležitá data nebyla kompromitována nebo pozměněna, je možné využít HIDS systém. HIDS umožňuje kontrolu právě těchto unikátních hodnot. Pokud u daného souboru dojde ke změně, je provedena definovaná akce HIDS systémem [13].

Monitorování logů

Jak již bylo zmíněno, tak HIDS systémy umožňují také monitorování informací, které jsou zaznamenány v logovacích souborech. Do systémových logů je zaznamenána každá změna v systému. Může se jednat o důležitou změnu v nastavení, kterou provedl uživatel nebo správce systému. Také jsou zde zaznamenány informace o tom, kdy se uživatel odhlásil či byl systém restartován [69]. HIDS umožňují monitorování většinou výstupních logů, ať už se jedná o log databáze, systému nebo například PBX ústředny. [11] HIDS systém může monitorovat klíčové logy a kontrolovat, zda se v systému neděje nějaká podezřelá aktivita.

Detekce rootkitu

Rootkit je sada počítačových programů a technologií, pomocí kterých lze maskovat přítomnost zákeřného software v počítači, například přítomnost virů, trojských koní, spyware a podobně. Rootkit technologie maskuje přítomnost zákeřných nástrojů tak, aby přítomnost softwaru nebyla běžně dostupnými systémovými prostředky odhalitelná [55]. Detekce rootkitu probíhá vyhledáváním jednak známých rootkitu, ale také jejich typických příznaků, mohou být tedy detekovány i dosud neobjevené a nepopsané typy rootkitu [11].

Prevence průniku

HIDS systémy stejně jako NIDS systémy neslouží pouze k monitorování a zaznamenávání informací o systémech, ale umožňují také aktivní ochranu systému [11]. Dle definovaných pravidel je možné zabránit již probíhající akci, nebo kontaktovat přímo administrátora, který může provést odpovídající řešení. Způsob zapojení může být obdobný, jako je uvedeno na obrázku 6.1. Pro provedení odpovídající akce je ovšem nutné nastavit konkrétní pravidla pro komunikaci mezi stanicí a HIDS systémem [13].

6.3.1. OSSEC - detekce a prevence průniku na úrovni operačního systému

Jedním z nejpoužívanějších HIDS zařízení je v současné době open source systém OSSEC. Jedná se o systém, který je dostupný na většině používaných systémů, jako je Linux, OpenBSD, FreeBSD, Mac OS X, Sun Solaris a Microsoft Windows. Umožňuje monitorování registrů, logů, detekci rootkitů a monitorování podezřelé aktivity, upozornění v reálném čase, prevenci průniku a další [29]. Jedná se o plnohodnotný HIDS systém, který je šířen pod GNU licencí (General Public Licence).

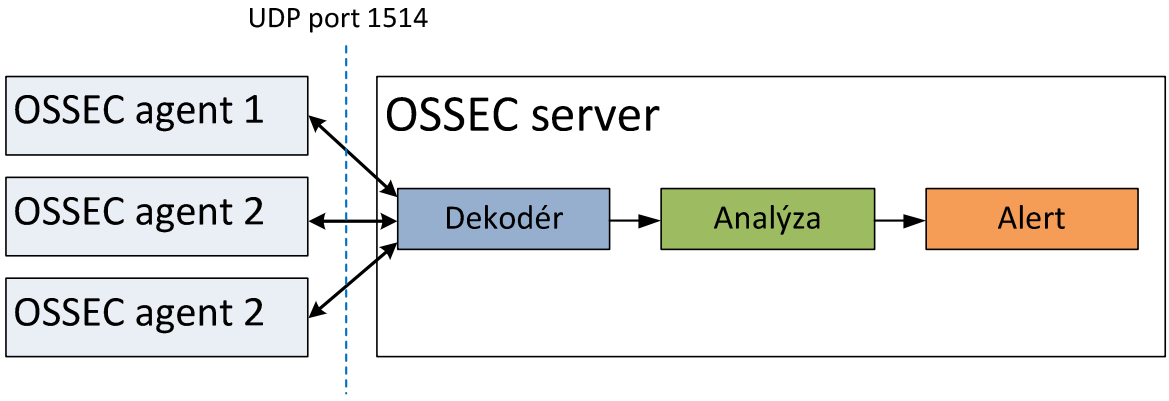
Architektura systému OSSEC

Systém OSSEC je složen ze dvou základních částí. Jedná se o centrální monitorovací prvek, který zajišťuje monitorování celého systému na základě informací, které získává od agentů [13].

Správce - jedná se o centrální prvek OSSEC architektury. Umožňuje kontrolu integrity, monitorování databáze, zaznamenává události v systému a také si uchovává informace o auditu systému. Všechna pravidla, dekodéry a podstatná část konfigurace je uložena právě v tomto centrálním prvku [2].

Agent - agent je malý program, případně kolekce programů, které jsou nainstalovány na monitorovaných systémech. Agent shromažďuje informace o systému, v reálném čase je předává centrálnímu prvku pro analýzu a provedení odpovídajících akcí [2].

Princip komunikace je zobrazen na obrázku 6.3. Každý z agentů shromažďuje data o systému podle konfigurace, která byla dána při instalaci. Tato data jsou následně zkomprimována, zašifrována pomocí předem domluveného klíče a odeslána na OSSEC server, kde běží centrální prvek - správce. Data jsou odeslána na server prostřednictvím UDP přenosu přes port 1514. Správce tato data nejprve dekóduje, analyzuje podle svých pravidel a na základě získaných dat vytvoří akci. Akce může být různého typu: aktivní akce - zablokování podezřelé IP adresy, pasivní akce - záznam upozornění nebo kontaktování administrátora. Samozřejmě je možno tyto akce kombinovat [2].



Obr. 6.3: Architektura systému OSSEC

Struktura pravidel systému OSSEC

Stejně jako u systému NIDS, jsou pravidla stěžejní konfigurací každého HIDS zařízení včetně OCCES. Každé pravidlo je definováno samostatně ve formě XML a je společně s dalšími pravidly v adresáři */var/ossec/rules/*. V základu disponuje OSSEC několika tisíci pravidly, které jsou rozděleny do různých kategorií [13]. Pro tuto práci je stěžejní právě zabezpečení PBX, z toho důvodu budou v práci využita pravidla pro ústřednu Asterisk. Pro ústřednu Asterisk jsou pravidla umístěna v souboru *asterisk.xml*, který obsahuje několik desítek pravidel.

```
<group name="syslog,asterisk,">
<rule id="6250" level="10" frequency="6" timeframe="300">
<description>Multiple failed logins (user enumeration in \
process).</description>
</rule>
</group>
```

Jedná se o jedno ze sady pravidel, které jsou využity k detekci útočníka, který se pokouší uhodnout uživatelské jméno a heslo. Syntaxe pravidel je vždy stejná. Každé pravidlo musí být přiděleno do určité skupiny pomocí *<group></group>*, v tomto případě se jedná o skupinu *syslog* a *asterisk*. Každá skupina může obsahovat neomezené množství pravidel *<rule></rule>*, ovšem každé pravidlo musí být identifikovatelné. K tomu slouží položka id. Položka *level* udává, o jak závažný problém se jedná. Pole *frequency* udává počet opakování v době *timeframe*. V tomto případě se jedná o šest opakování za 300 sekund. Pole *<description></description>* je v každém pravidle povinné a slouží k popisu funkce pravidla. Možnosti pro definování pravidel jsou téměř neomezené, stejně tak je možno nastavit velké množství parametrů [13].

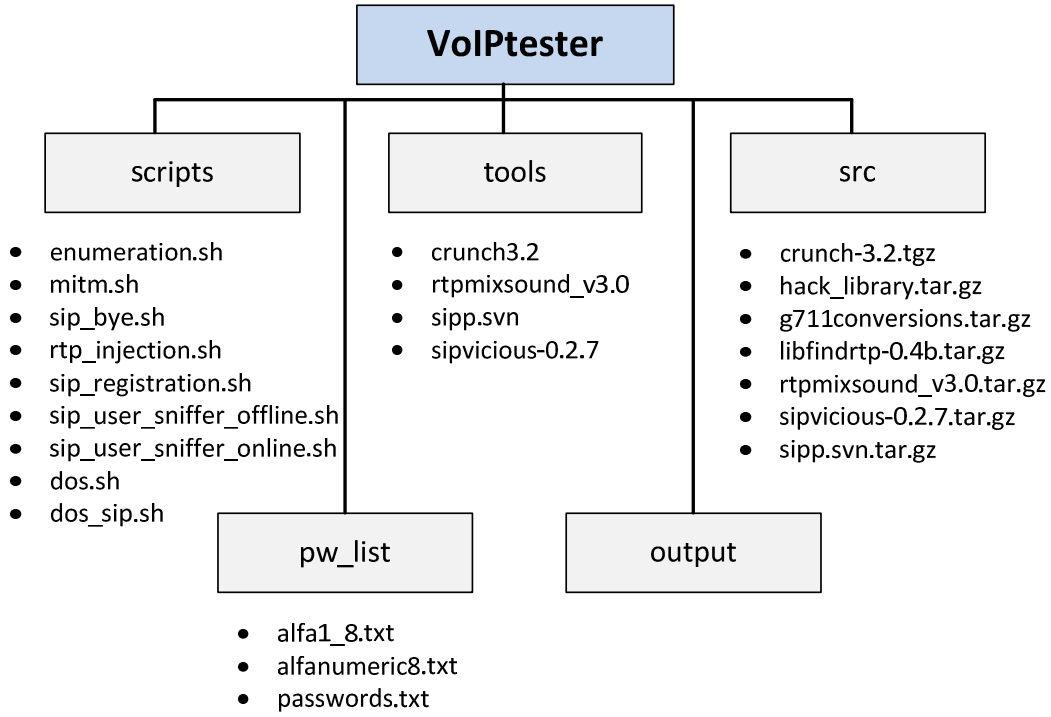
7. Praktická část

7.1. Generátor útoků proti open source PBX

Hlavní náplní této práce je otestovat open source PBX ústřednu proti různorodým útokům. K tomu účelu byl vytvořen generátor útoků, který bude dále nazýván VoIPtester. VoIPtester je složen z několika částí, jak je uvedeno na obrázku 7.1. Stěžejní součást je adresář *scripts*, kde jsou umístěny rozhraní pro jednotlivé nástroje. V adresáři *output* budou umístěny výstupní logy k jednotlivým skriptům. Každý test je zaznamenán do logu, který obsahuje informace o průběhu testu spolu s jeho výsledkem.

Adresář *pw_list* obsahuje již několik vytvořených slovníků pro útoky určené k prolomení hesla. Slovníky obsahují jak vygenerované kombinace hesel o velikosti čtyři až osm znaků, tak je zde slovník skutečných hesel. Jedná se o databázi, které obsahuje téměř pět miliónů záznamů. V databázi jsou obsažena nejčastěji používaná hesla, tato databáze byla vytvořena v roce 2002.

Adresář *src* obsahuje zdrojové soubory pro kompilaci jednotlivých nástrojů, jelikož ne všechny nástroje jsou dostupné přímo jako balíčky pro jednotlivé distribuce Linuxu. Seznam všech nástrojů, které byly využity, je zobrazen v tabulce 7.1, tyto nástroje jsou volně dostupné na internetu, seznam zdrojů je uveden v přílohách. Dalším adresářem je *tools*, kde jsou umístěny již rozbalené a zkompilované nástroje. Seznam těchto nástrojů je uveden na obrázku 7.1. Stěžejní součást je adresář *scripts*, kde jsou připraveny skripty pro testování SIP systému.



Obr. 7.1: VoIPtester – struktura

Tab. 7.1: Seznam použitých nástrojů

nástroj / verze	dostupnost	popis
nmap 5.51	*	port scanner. Pro zjištění validních IP adres je použit protokol ARP. Podporuje také TCP i UDP a dokáže detekovat i operační systém
ettercap NG-0.7.3	*	disponuje řadou nástrojů, které lze využít k odposlechu komunikace. Je vhodný pro útok typu Man in the Middle, dokáže zasahovat do probíhající komunikace bez přerušení spojení, nebo také umožňuje odchyt hesel řady protokolů
crunch 3.2	-	worldlist generátor, který umožňuje vytvořit libovolný slovník na základě předdefinované znakové sady, nebo na základě parametrů, které určí uživatel
rtpmixsound 3.0	-	umožňuje vkládání nebo nahrazení RTP paketů v již probíhajícím spojení mezi dvěma účastníky. Jako zdroj dat používá wav soubor, který v reálném čase převede do RTP paketů a ty jsou vloženy do existující relace. Uživatel pak slyší vložené audio na pozadí [7]
Sipp.svn v3.2	-	testovací nástroj pro SIP protokol, umožňuje simulovat kompletní hovory či odesílat libovolné SIP zprávy. Pracuje se scénáři v XML [60]
SipVicious 0.2.7	-	sada skriptů, jejímž primárním účelem je testování bezpečnosti SIP systémů [61]
arpspoof 2.4	*	umožňuje modifikaci ARP tabulky cílových IP adres.
sipcrack 0.2	*	obsahuje dva základní nástroje SIPDump a SIPCrack. SIPDump umožňuje zaznamenávat probíhající SIP komunikaci v síti, která obsahuje *SIP authentication digest respons*. SIPCrack slouží k prolomení hesla z dat odchycených pomocí SIPDump
hping 3.0.0-alpha-2	*	generátor a analyzátor paketů pro TCP/IP protokol. Určen pro bezpečnostní prověrky, testování firewallů a sítí.

***** dostupné jako samostatné balíčky pro UNIX systémy
- zdrojové soubory uloženy v adresáři src

7.1.1. Shromažďování informací - enumeration.sh

Skenování sítě a prvků v síti, je klíčovou součástí každého útoku popřípadě testu zabezpečení. K tomu, aby mohl být proveden úspěšný útok na PBX je potřeba mít dostatečné informace o systému a o službách, které na něm běží. Pokud má útočník k dispozici dostatek informací, provedení útoku, jako například DoS, je pak už velmi snadné. K základnímu skenu IP ústředny slouží enumeration.sh. Jedná se v podstatě o velmi jednoduché rozhraní na vložení parametrů pro síťový analyzátor Nmap a pro svmap.py , jenž je z kolekce SipVicious.

Vstupní parametry

IP adresa nebo rozsah IP adres	SIP port	Port nebo rozsah (volitelné)

Nmap nejprve definuje dostupné IP adresy z daného rozsahu a pak na základě předdefinovaných parametrů skenuje jednotlivé zařízení. Celý průběh je zaznamenán v logovacím souboru v adresáři */voip/output*.

7.1.2. Odposlech sítě - mitm.sh

Jednou z možností odposlechu dat je využití sniffovacích nástrojů, jako Ettercap. Ettercap disponuje řadou možností, které lze využít k odposlechu komunikace. Je vhodný pro útok typu Man in the Middle, dokáže zasahovat do probíhající komunikace bez přerušení spojení, nebo také umožňuje odchyt hesel řady protokolů. K tomuto účelu je využit ARP spoofing, viz kapitola 3. K modifikaci komunikace pomocí techniky Man in the Middle [57] slouží skript mitm.sh. Opět se jedná a poměrně jednoduché rozhraní pro zadání informací pro Ettercap.

Vstupní parametry – odposlech sítě

Interface	Cíl 1 - ústředna	Cíl 2 - uživatel (volitelné)

Díky malé modifikaci konfiguračních souborů */etc/etter.conf* pro Ettercap, je možné odchytit i hesla zabezpečená pomocí https. Úprava v konfiguračním souboru je vidět níže a je potřeba ji provést před zahájením odposlechu. Po zadání parametrů do mitm.sh stačí jen počkat, než se k webovému rozhraní přihlásí uživatel nebo správce ústředny. Po ukončení skenování jsou data zaznamenána opět do výstupního adresáře *output*, jedná se o soubory pro analýzu pomocí Etterlogu (*mitm.ecp, mitm.eci*), tak kompletní záznam přenosu - *mitm.pcap*.

Možnosti skenovaní sítě a získávání informací o prvcích sítě jsou poměrně rozsáhlé, ovšem k získání základních informací jsou nástroje mitm.sh a enumeration.sh dostatečné.

```
#Definice práv k provedení cleanup scriptu pro firewall
[privs]
ec_uid = 0
ec_gid = 0
#Modifikace firewallu IPTables,
#pro provedení odposlechu zabezpečené komunikace

redir_command_on = "iptables -t nat -A PREROUTING -i %iface -p tcp --dport
%port -j REDIRECT --to-port %rport"
redir_command_off = "iptables -t nat -D PREROUTING -i %iface -p tcp --dport
%port -j REDIRECT --to-port %rport"
```

7.1.3. Detekce účtů online - sip_user_sniffer_online.sh

Dalším nástrojem, který je součástí testovacího balíku, je sip_user_sniffer_online.sh. Jak již název napovídá, umožňuje vyhledávání uživatelů, kteří jsou na ústředně vytvořeny. Pokud je uživatel ze zadaného rozsahu nalezen, umožňuje také prolomení hesla. K tomu účelu je možné využít slovníkový útok popřípadě použít hesla vytvořená na základě číselného rozsahu. Ke své činnosti využívá tři nástroje z balíku SipVicious. Jedná se o svmap.py – detekce ústředny, svwar.py – vyhledání registrovaných uživatelů, svcrack.py – prolomení hesla [61]. Jako uživatelský vstup vyžaduje pouze IP adresu ústředny a rozsah uživatelů, pro které se má provést sken.

Vstupní parametry

IP adresa ústředny	Číselný rozsah uživatelů

První část detekce ústředny a detekce SIP uživatelů je poměrně rychlá. Jakmile je ústředna nalezena, okamžitě se začnou vyhledávat uživatelé z daného rozsahu. Pokud nebylo použito žádného šifrování, tak by hesla byla přenášena jako volný text [17], výsledkem by byly

informace jak o účtech, tak o samotných heslech. V současné době je snad již automaticky používána šifrovaná autentizace pomocí hesla, takže možnost kdy je autentizace nešifrována nebude dále zmiňována.

Pokud jsou nalezeny nějaké účty, nabízí se možnost prolomit heslo. Z nalezených uživatelů je možné vybrat pouze jednoho. Jako zdroj dat pro generování hesla se nabízejí dvě možnosti. Za prvé je možné vybrat již existující slovník nebo za druhé zvolit číselný rozsah. Po zvolení slovníku je testování spuštěno automaticky. Mezi serverem a útočníkem dochází k výměně zpráv SIP na základě DIGEST autentizace [17]. Slabinou tohoto procesu je neustála komunikace mezi PBX ústřednou a útočníkem. Na ústředně lze nastavit firewall, aby jednoduše blokoval požadavky z konkrétní IP adresy po překročení určitého limitu. Princip tohoto nastavení bude popsán v dalších kapitolách.

7.1.4. Detekce účtů offline - sip_user_sniffer_offline.sh

Tento nástroj má stejný účel jako předchozí skript, ovšem podstatný rozdíl je v tom, že prolomení hesla neprobíhá online. Pomocí SIPDump je odchycena komunikace na síti, která obsahuje *SIP authentication digest respons*, viz kapitola 4. To stačí k tomu, aby mohlo být využito slovníkového útoku k prolomení hesla. K modifikaci komunikace v síti je využit nástroj arpspoof, jenž umožňuje *ARP Cache poisoning*, viz kapitola 3. Opět je potřeba zadat několik vstupních parametrů. Jedná se o interface, na kterém budou odchycena data a dále IP adresy cílů. Pokud nebude definována alespoň jedna IP adresa, bude odchycena celá komunikace v síti.

Vstupní parametry – odposlech sítě

Interface	Cíl 1 - ústředna	Cíl 2 - uživatel (volitelné)

Po zadání parametrů je zahájen odposlech sítě a zaznamenána jsou pouze data, jež obsahují výše zmíněnou digest autentizaci. Po ukončení odposlechu sítě se nabídne možnost prolomení hesla offline, opět na základě slovníků, které jsou dostupné v adresáři pw_list. Po ukončení výběru se automaticky spustí SIPCrack, který slouží k prolomení hesla offline. Opět je možné vybrat pouze jeden odchycený učet.

7.1.5. Přerušení hovoru - sip_bye.sh

Další součástí testovacího rozhraní je skript bye.sh, který umožňuje přerušit libovolný hovor v síti, viz kapitola 3. Je ale nezbytné modifikovat přenos v síti, aby útočník mohl zaznamenat procházející komunikaci. K tomuto účelu slouží Ettercap. Pro přerušení spojení je odeslána zpráva SIP BYE k ústředně s odposlechnutým Call-ID a dalšími fiktivními parametry.

Vstupní parametry – odposlech sítě

Interface	Cíl 1 - ústředna	Cíl 2 - uživatel (volitelné)

Jakmile je zaznamenána komunikace mezi uživateli, je možné Ettercap ukončit. Z odchycené komunikace jsou automaticky vybrány pouze informace o Call-ID. Celý průběh komunikace je zaznamenán do logu v adresáři *output*. Pomocí skriptu je možno odeslat požadavek jak na konkrétní hovor, tak se nabízí možnost ukončit všechny hovory, které byly zaznamenány.

Po zadání Call-ID je vytvořena zpráva BYE a ta je odeslána PBX ústředně. Pokud bude hovor ukončen, PBX odešle potvrzující zprávu 200 OK přímo útočníkovi. Call-ID patří

pouze relaci mezi uživatelem a ústřednou. Přenos je přerušen pouze pro jednoho uživatele, druhý, popřípadě další uživatelé nezjistí ukončení spojení, z jeho pohledu je relace stále otevřena.

Pokud by hovor v době odeslání SIP zprávy již neexistoval, tak by útočník obdržel zprávu *SIP/2.0 481 Call Leg/Transaction Does Not Exist*. Obdobným způsobem pracuje i možnost ukončení všech hovorů. Díky tomu jsou přerušeny všechny existující relace s ústřednou. Z pohledu útočníka nebylo potřeba znát ani heslo ani uživatelské jméno, pouze informace o Call-ID.

7.1.6. Modifikace registrace uživatele - sip_registration.sh

Nástroj sip_registration.sh umožňuje zrušení registrace uživatele u PBX ústředny zasláním specifické zprávy REGISTER. Díky tomu uživatel nemůže přijímat žádné příchozí hovory. Po spuštění sip_registration.sh je položen dotaz, zda má být odposlechnuta síť pomocí nástroje Ettercap. K tomu, aby byla registrace odstraněna, je nutné znát identifikační číslo uživatele a jeho heslo. To je možné získat pomocí sip_user_sniffer, jak bylo již uvedeno.

Vstupní parametry

Interface	IP adresa ústředny	IP adresa telefonu	Uživatelské jméno	Uživatelské heslo	SIP port ústředny

Po zadání všech parametrů je ověřeno, zda jsou hodnoty zadány ve správném formátu a následně je odeslána zpráva REGISTER. Pokud bude požadavek úspěšný, je odesláno potvrzení 200 OK přímo útočníkovi. Celý průběh je zaznamenán ve výstupním logu, kde je uložen i výpis SIP komunikace mezi útočníkem a ústřednou.

```
REGISTER sip:192.168.201.148 SIP/2.0
Via: SIP/2.0/UDP 192.168.201.1:5060;branch=z9hG4bK-8296-1-0
From: "test"<sip:104@192.168.201.148>;tag=1
To: "test"<sip:104@192.168.201.148>
Call-ID: 1-8296@127.0.1.1
CSeq: 2 REGISTER
Contact: *
Expires: 0
Max-Forwards: 5
Content-Length: 0
```

Klíčovými parametry jsou **Contact: *** a **Expires: 0** ve zprávě REGISTER, které odstraní všechny registrace pro konkrétní SIP telefon u PBX ústředny.
Další součástí nástroje sip_registration.sh je možnost převést hovory, které jsou určeny konkrétnímu účastníkovi na jinou IP adresu, než na které se uživatel skutečně nachází. To je umožněno opět pomocí již získaných dat.

Vstupní parametry

IP adresa ústředny	IP adresa telefonu	Uživatelské jméno	Uživatelské heslo	Fiktivní uživatele	SIP port ústředny

Ve výstupním logu je zaznamenán celý průběh SIP komunikace, pro názornost je níže zobrazen požadavek REGISTER na server. Jako uživatele určeného k autentizaci je využito uživatele 101, ovšem jako kontaktní údaj je uveden uživatel 104, IP adresa 192.168.201.147, port 5060. SIP server bude od této chvíle veškerou komunikaci uživatele 101 přeposílat uživateli 104.

```
REGISTER sip:192.168.201.148 SIP/2.0
Via: SIP/2.0/UDP 192.168.201.147:5060;branch=z9hG4bK-10213-1-0
From: "test"<sip:101@192.168.201.148>;tag=1
To: "test"<sip:101@192.168.201.148>
Call-ID: 1-10213@127.0.1.1
CSeq: 2 REGISTER
Contact: <sip:104@192.168.201.147:5060>
Max-Forwards: 5
Expires: 3600
Content-Length: 0
```

7.1.7. Modifikace hovoru - rtp_injection.sh

Jak již bylo uvedeno v kapitole 3, přenos pomocí RTP není nijak zabezpečen a z toho důvodu je možné do již probíhajícího hovoru vkládat data, která jsou přehrána na pozadí samotné relace. V rámci tohoto projektu je vytvořen nástroj rtp_injection.sh, který je postaven na rtpmixsound. Jedná se jen o ukázkové použití, jelikož rtpmixsound má velmi omezené možnosti. Data nesmí být delší než deset sekund, zvukový soubor vložený do RTP streamu musí být „wav“ soubor zakódovaný PCM (*Pulse-code modulation*) se vzorkovací frekvencí 8 kHz [7]. Skript rtp_injection.sh k úspěšné modifikaci RTP streamu vyžaduje několik parametrů, které je opět možné získat pomocí odposlechu sítě. Hodnoty RTP portů a IP adresy ústředny jsou automaticky zjištěny přímo z odposlechu sítě.

Vstupní parametry – odposlech sítě

Interface	Cíl 1 - ústředna	Cíl 2 - uživatel (volitelné)

Vstupní parametry

IP adresa ústředny	IP adresa telefonu	RTP port ústředny	RTP port telefonu

Po zadání parametrů se automaticky spustí rtpmixsound, který umožní vložení dat do RTP streamu. Průběh je opět zaznamenán do výstupního logu. Pokud byla data úspěšně vložena, je zobrazena potvrzující informace z rtpmixsound.

7.1.8. Znepřístupnění webového rozhraní - slowloris

Jen jako doplňující útok je zde uvedena možnost znepřístupnění webového rozhraní ústředny, nebo jakékoliv jiné domény, která nevyužívá load balancing a jako webový server využívá Apache. Tento typ útoku je označován jako slowloris [63]. Princip spočívá v otevření dlouhodobého spojení mezi útočníkem a webovým serverem. Webový server obvykle otevře spojení a čeká na doručení celého HTTP požadavku, na který bude odpovídat. Ovšem

slowloris posílá data velmi pomalu po částech a těsně před vypršením časového limitu. Díky tomu server udržuje nekonečné spojení s útočníkem a tím nemůže odpovídat na další požadavky [23].

Vstupní parametr
IP adresa ústředny

Tento útok je zaměřen převážně na webový server Apache první a druhé generace a pár méně známých alternativ [23]. Komunikace se serverem vypadá následovně. Nejprve je serveru odeslána naprosto legitimní část hlavičky:

```
GET / HTTP/1.1\r\n
Host: host\r\n
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0;
.NET CLR 1.1.4322; .NET CLR 2.0.503l3; .NET CLR 3.0.4506.2152; .NET CLR
3.5.30729; MSOffice 12)\r\n
Content-Length: 42\r\n
```

Celý obsah je v pořádku, jen chybí poslední řádek, který by potvrdil, že hlavička je kompletní. V tuto chvíli web server čeká na dokončení hlavičky, ovšem místo toho je mu doručen další nic neříkající řádek:

```
X-a: b\r\n
```

Nejedná se o chybný řádek, ale také nenese žádnou informaci, jen donutí web server v tomto případě Apache, aby vynuloval čítač a čekal na zbývající informace. Tento proces se neustále opakuje a díky tomu web server není schopen odpovídat na příchozí požadavky [23]. Jelikož se jedná o velmi nízký tok dat, detekční systémy tento útok pravděpodobně nezaznamenají a tím pádem je velmi těžké ho zpětně odhalit i při pozdější analýze. Tímto způsobem je možné webové rozhraní znepřístupnit na neomezeně dlouhou dobu.

7.1.9. Útok typu Denial of Service - dos.sh

Pomocí dos.sh lze vytvářet simulaci útoku typu DoS útoky na TCP/IP architekturu. Skript umožňuje záplavu paketů UDP, ICMP, nebo TCP SYN. Od uživatele vyžaduje několik parametrů, na které se postupně dotazuje. Za zmínku stojí parametr fiktivní IP adresa a velikost paketu, ostatní parametry jsou jasně definovány. Fiktivní IP adresa umožňuje uživateli zadat libovolnou IP adresu, která bude do paketu vložena jako zdroj dat. Pokud pole zůstane nevyplněno (tato možnost zde existuje) bude zdrojová IP adresa náhodně generována pro každý paket jednotlivě. Ovšem tímto způsobem se již nebude jednat o úmyslný útok typu DoS, ale pouze o test, jak je oběť schopna reagovat na vytížení z více zdrojů, viz kapitola 3. Další parametr, který je nutno popsat, je velikost dat paketu. Tato možnost zde existuje proto, aby mohly být simulovány i útoky záplavového typu. Velikost dat paketu může být 0 až 65000 bytů pro ICMP a 0 až 65000 bytů pro UDP. Pokud je zadána maximální velikost, je útočník schopen generovat zaplavu dat přesahující 100 Mbit/s.

Vstupní parametry

Interface	IP adresa ústředny		Fiktivní IP adresa (volitelné)	Protokol
Port zdroje	Port cíle	Velikost dat paketu	SIP port ústředny	Celková doba testu

Pokud jsou zadány všechny parametry, je provedena kontrola, zda jsou veličiny zadané ve správném formátu a je spuštěn test. Postupný průběh je vidět po spuštění skriptu a také je zaznamenán do logu k pozdější analýze. V polovině časového intervalu je odeslána SIP zpráva OPTIONS k ústředně, která má ověřit, zda je ústředna schopna odpovídat i na další požadavky. Pokud odpověď 200 OK nedorazí do tří sekund od odeslání, je ústředna považována za nedostupnou. Jelikož se jedná o UDP přenos, je zpráva OPTIONS odeslána opakovaně vždy po jedné sekundě.

Obsah logu je částečně shodný i s výpisem, který se generoval na obrazovku, ovšem obsahuje i záznamy o celkovém počtu odeslaných paketů a průběhu kontroly pomocí SIP zpráv. Jelikož Hping pracuje ve flood módu nezaznamenává ani neočekává od cíle žádné odpovědi. V zadání je navíc možno definovat zdrojovou IP adresu, takže cíl sice přijme pakety, ale odpovědi odešle na zadanou IP adresu.

7.1.10. Útok typu DoS na inicializační protokol SIP - dos_sip.sh

Pro ověření protokolu SIP je k dispozici skript dos_sip.sh, který využívá testovací nástroj SIPp [60]. Tento nástroj při vhodně navrhnutém XML scénáři a se správně definovanými parametry dokáže sloužit jako velmi účinný generátor paketů protokolu SIP.

Samotný dos_sip.sh vyžaduje definování několika parametrů. Jedna se o IP adresu cíle, definice SIP zprávy, která bude odeslána k cíli, na výběr je INVITE, REGISTER, OPTIONS, ACK a BYE. Dále je potřeba definovat port, kam budou směrovány SIP zprávy, počet SIP zpráv, které budou odeslány za jednu sekundu a celkový časový interval přenosu.

Vstupní parametry – odposlech sítě

IP adresa cíle	SIP metoda	SIP port cíle	Počet zpráv za sekundu	Celková doba testu

Opět se provede kontrola, zda byly všechny parametry zadány ve správném formátu a následně je spuštěn test. V polovině časového intervalu je odeslána SIP zpráva OPTIONS k ústředně, která má ověřit, zda je ústředna schopna odpovídat i na další požadavky. Pokud odpověď 200 OK nedorazí do tří sekund od odeslání, je ústředna považována za nedostupnou. Jelikož se jedná o UDP přenos, je zpráva OPTIONS odeslána opakovaně vždy po jedné sekundě. Pro ověření skutečnosti, že je ústředna po celou dobu testu nedostupná, je vhodné také vyzkoušet hovor mezi uživateli ústředny

V rámci testu byl opět vytvořen log, který je umístěn v adresáři */voip/output*, kde jsou zaznamenány i informace o přenesených paketech a průběhu přenosu kontrolní SIP zprávy. Stejným způsobem je možné otestovat, jak bude ústředna reagovat na záplavu SIP zpráv INVITE, REGISTER, OPTIONS, ACK nebo BYE. Poslední dvě jmenované jsou zde jen pro

úplnost, nepředpokládá se, že by i při zcela nezabezpečené ústředně měly nějaký závažnější efekt na SIP komunikaci.

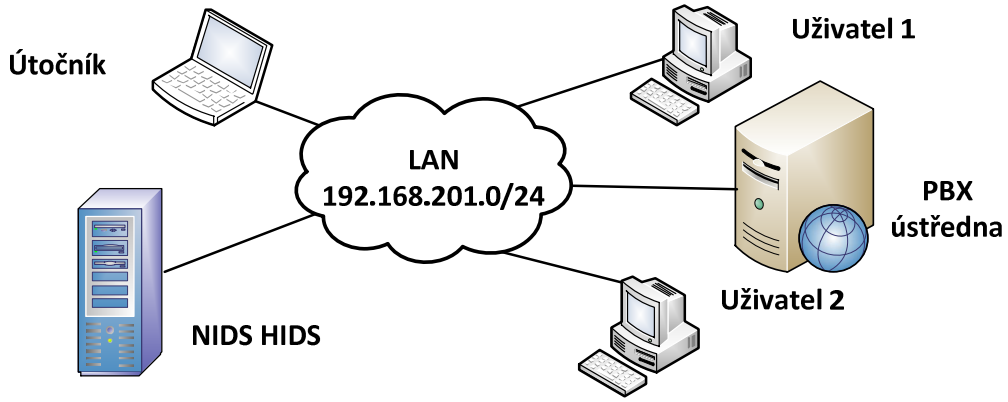
Tímto jsou vyčerpány možnosti útoků typu DoS záplavou paketů pomocí vytvořeného nástroje. Výše zmíněné útoky patří mezi nejpoužívanější a poměrně efektivní způsob jak vytížit ústřednu, popřípadě znemožnit komunikaci uživatelů.

7.2. Konfigurace experimentální sítě

K ověření útoků byla vytvořena virtuální síť v prostředí VMware. Síť obsahuje PBX ústřednu, dvě uživatelské stanice se systémem Windows, NIDS Snort & OSSEC a jako útočník je zvolena stanice s distribucí Ubuntu. Hardwarová konfigurace je znázorněna v tabulce 7.2.

- V síti budou postupně implementovány čtyři PBX ústředny, Asterisk verze 1.6.0, Asterisk verze 10, YATE 3.0 a FreeSWITCH verze 1.0. (vydána v roce 2011). Všechny uvedené ústředny jsou open source a volně dostupné.
- Uživatelské stanice jsou obě shodné, jedná se o systém Windows XP SP2, jako VoIP telefony jsou zvoleny X-Lite 4.1.
- Útočník využívá systému Unix, distribuce Ubuntu 11.10, Linux 3.0.0-17-generic. Ke generování útoků je vytvořena sada skriptů, která byla vysvětlena v předchozí kapitole.

V rámci této práce byly otestovány všechny výše zmíněné ústředny. Zapojení odpovídá obrázku 7.2. Každý z testů byl proveden třikrát, aby se zabránilo náhodným chybám. Všechny ústředny používají protokol SIP a v základním nastavení je přenos dat proveden pomocí RTP protokolu. Z toho důvodu se předpokládá, že výsledky testu budou velmi podobné. Záznamy ze skriptu, které zjistí informace o systému (enumeration.sh) jsou uvedeny v přílohách. V této části budou popsány jen skripty související s protokolem SIP a RTP. Nejprve skript, který umožňuje zjištění hesla přímo prostřednictvím ústředny, sip_user_sniffer_online.sh.



Obr. 7.2: Konfigurace sítě

Tab. 7.2: Konfigurace sítě

	Systém	**CPU**	**IP adresa**	**Operační paměť [DDR3]**	**Síťové rozhraní**
PBX ústředna	Asterisk 1.6.2.11 CentOS	Intel Core i3M370	**192.168.201.148**	1024 MB	100Mbit/s
	Asterisk 10 Ubuntu 10.4 LTS		**192.168.201.158**	1024 MB	100Mbit/s
	YATE 3.0 Mandriva		**192.168.201.155**	1024 MB	100Mbit/s
	Freeswitch 1.0 Ubuntu 10.4 LTS		**192.168.201.150**	1024 MB	100Mbit/s
Uživatelská stanice	Windows XP SP2		**192.168.201.141** **192.168.201.151**	256 MB	100Mbit/s
Útočník	Ubuntu 10.4		**192.168.201.147**	1024 MB	1Gbit/s
NIDS HIDS	**Snort & OSSEC**		**192.168.201.156**	1024 MB	1Gbit/s

7.3. Ověření jednotlivých testů

Online detekce hesla

Vstupní parametry

IP adresa ústředny	Číselný rozsah uživatelů
192.168.201.158	100-200

```
IP adresa ustredny: 192.168.201.158
Rozsah uzivatelu: 100-200

Informace o ustredne:
| SIP Device           | User Agent          | Fingerprint |
-------------------------------------------------------------
| 192.168.201.158:5060 | Asterisk PBX 10.0.0 | disabled    |

Probiha test existujicich uzivatelu...
| Extension | Authentication |
------------------------------
| 102       | reqauth        |
| 101       | reqauth        |
```

Nalezeni byli dva uživatelé, uživatel 100 a 101. V tuto chvíli se nabízí možnost zjištění hesla. Útočník má možnost zvolit pouze jednoho uživatele, v tomto případě 101. Dále je potřeba definovat, jakým způsobem bude zjištěno heslo. Na výběr jsou dvě možnosti: číselný rozsah anebo jeden ze slovníků v adresáři pw_list. V tomto případě byla vybrána slovníková metoda a jako slovník byl zvolen alfanumeric8.txt.

```
Pokusit se najit heslo  pro uzivatele ? (A - ano, N - ne)
A
Zadejte uzivatele:
101
1 - Pouzit existujici slovnik
2 - Cisleny rozsah
1
Dostupne slovniky:

alfa1_8.txt       passwords.txt  pw_list2.txt  pw_list4.txt
alfanumeric8.txt  pw_list1.txt   pw_list3.txt
```

To jsou všechny parametry, které je potřeba zadat a nyní je spuštěn samotný proces zjišťování hesla. Tento proces může trvat poměrně dlouho, záleží na velikosti slovníku a na tom, v jakém místě se heslo nachází.

```
Zvolen slovnik
Testovany uzivatel: 101
Zvolena metoda: 1

Delka testu zavisi na velikosti slovniku a robustnosti hesla
Probiha proces testovani hesla...

| Extension | Password |
------------------------
| 101       | Zaq12wsx |

Test dokoncen,vystupni informace jsou zaznamenany v
/voip/output/sip_user_sniffer_online.1336912860.log
```

Pokud se ve slovníku vyskytuje heslo, které odpovídá skutečnému heslu uživatele, je zaznamenána shoda. Nalezená kombinace je zobrazena výše.

Detekce hesla offline.

Vstupní parametry – odposlech sítě

Interface	Cíl 1 - ústředna	Cíl 2 - uživatel (volitelné)
eth0	192.168.201.158	192.168.201.151

Po zadání parametrů je zahájen odposlech sítě a zaznamenána jsou pouze data, jež obsahují výše zmíněnou digest autentizaci. Po ukončení odposlechu sítě se nabídne možnost prolomení hesla offline, opět na základě slovníků, které jsou dostupné v adresáři pw_list. Po ukončení výběru se automaticky spustí SIPCrack, který slouží k prolomení hesla. Je možné vybrat pouze jeden odchycený účet. V tomto případě se jednalo o účet 101.

```
Num     Server          Client          User    Hash|Password
1      192.168.201.151 192.168.201.158 101     d9d250d1d12f226b0577f6aec82afb87

* Generating static MD5 hash... b34630aa09e1e3f54978ebf3169d72ca
* Loaded wordlist: '/voip/pw_list/pw_list4.txt'
* Starting bruteforce against user '101'
(MD5: 'd9d250d1d12f226b0577f6aec82afb87')
* Tried 1147 passwords in 0 seconds

* Found password: 'Zaq12wsx'
```

Jak je vidět výše, bylo otestováno 1147 hesel za necelou sekundu a nalezené heslo je password. Celý proces je zaznamenán v logu, kde jsou uvedeny jak MD5 heše, tak i informace o účtech. V porovnání s přechozím případem, je tento proces mnohonásobně rychlejší a z pohledu útočníka i bezpečnější.

Ukončení hovoru

Tento útok byl také odzkoušen na všech ústřednách. Hovor byl úspěšně ukončen u ústředny Asterisk 1.6.0 a ústředny FreeSWITCH, zde stačila pouze hodnota Call-ID. Asterisk verze 10 a YATE 3.0 neumožnili útočníkovi ukončit hovor ani po zadání většiny parametrů přenosu. Při každém pokusu byla odeslána odpověď *SIP/2.0 481 Call leg/transaction does not exist.*

Z toho vyplývá, že ústředna Asterisk verze 10 vyžaduje k ukončení hovoru všechny informace o spojení, ne pouze Call-ID, jak to bylo v předchozích případech. Ústředna YATE taktéž neumožnila ukončení hovoru pouze prostřednictvím Call-ID, na každý pokus o ukončení spojení byla odeslána odpověď *401 Unauthorized*. Ústředna Asterisk verze 1.6.0 i ústředna FreeSWITCH hovor ukončili bez kontroly dalších informací. Výstupní log, který byl zaznamenán pro ústřednu Asterisk 1.6.0:

```
---------------------------------------------- 2012-04-20 11:57:36:797.708
UDP message sent (343 bytes):

BYE sip:123@192.168.201.148 SIP/2.0
Via: SIP/2.0/UDP 123.456.789.123:1234:branch=z9hG4bK-5698-1-0
Max-Forwards: 70
Contact: sip:123@123.456.789.123:1234
To: <sip:123@192.168.201.148>
From: teardown<sip:123@192.168.201.148>;tag=1
Call-ID: NTA1OWQ0ZGE2ZDAyNmNkYjZiNGNkZGMxM2FmZTQ3Yzc.
Cseq: 2 BYE
Subject: Teardown
Content-Length: 0
---------------------------------------------- 2012-04-20 11:57:36:798.333
UDP message received [435] bytes :

SIP/2.0 200 OK
Via: SIP/2.0/UDP 123.456.789.123:1234:branch=z9hG4bK-5698-1-
0;received=192.168.201.147
From: teardown<sip:123@192.168.201.148>;tag=1
To: <sip:123@192.168.201.148>;tag=as0d13dc72
```

```
Call-ID: NTA1OWQ0ZGE2ZDAyNmNkYjZiNGNkZGMxM2FmZTQ3Yzc.
CSeq: 2 BYE
User-Agent: Asterisk PBX 1.6.0.26-FONCORE-r78
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY, INFO
Supported: replaces, timer
Content-Length: 0
```

I přes to, že k ústředně byl vyslán požadavek s neexistujícími údaji ohledně IP adresy (123.456.789.123) a neexistují uživatel (123), tak byl hovor ukončen. Ústředna k přerušení spojení vyžaduje pouze Call-ID.

Manipulace s registrací uživatelů

V této části se jedná o manipulaci s uživatelskou registrací, jak bylo popsáno v kapitole 3. Vstupní parametry jsou uvedeny v tabulce níže, uživatelské jméno a heslo bylo zjištěno v předchozích případech.

Vstupní parametry

Interface	IP adresa ústředny	IP adresa telefonu	Uživatelské jméno	Uživatelské heslo	SIP port ústředny
eth0	192.168.201.158	192.168.201.151	101	Zaq12wsx	5060

I v tomto případě byla registrace zrušena úspěšně pro ústředny Asterisk verze 1.6.0, YATE 3.0. Ústředna YATE na pokus o zrušení registrace odpověděla zprávou 200 OK:

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP 192.168.201.151:5060;branch=z9hG4bK-4304-1-
4;received=192.168.201.147;rport=5060
From: "test"<sip:101@192.168.201.155>;tag=1
To: "test"<sip:101@192.168.201.155>;tag=1719852405
Call-ID: 1-4304@127.0.1.1
CSeq: 2 REGISTER
Server: YATE/3.0.0
Allow: ACK, INVITE, BYE, CANCEL, REGISTER, REFER, OPTIONS, INFO
Content-Length: 0
```

Díky této modifikaci nebylo možné vytvořit spojení s uživatelem 101, jelikož jeho telefonní přístroj nebyl u ústředny registrován.

Další testy není nutno blíže popisovat, jelikož se jedná o analogické situace. V tabulce 7.3 jsou sepsány ucelené výsledky všech testů. Pokud útok na danou ústřednu nebyl proveditelný nebo nebyl umožněn, je pole vyplněno „NE“. Test, který byl proveden s úspěšným výsledkem, je označen jako „ANO“. Testy, které se týkají útoků metodou DoS, jsou zpracovány v samostatné kapitole 7.3. Ne všechny testy byly úspěšné, to bylo způsobeno různým způsobem zpracování SIP zpráv ústřednou, ovšem většina testů byla úspěšně odzkoušena. Z dosažených výsledků vyplývá, že se nejhůře s útoky vyrovnala ústředna Asterisk 1.6.0, která nedokázala odolat ani jednomu z vytvořených testů. Bylo umožněno ukončit hovor pouze za pomoci hodnoty Call-ID. Modifikace registrace byla také úspěšně otestována, stejně jako vložení RTP paketů do již probíhajícího hovoru. Pro další ústředny

byla situace odlišná. Ústředna YATE umožnila modifikaci registrace uživatele, ale již neumožnila útočníkovi ukončit hovor. Ústředna FreeSWITCH umožnila ukončit hovor i modifikaci registrace.

Tab. 7.3: Souhrn útoků VoIPtester

	detekce hesel		**Ukončení spojení**	**Modifikace registrace**	**RTP injection**	**slowloris**
	online	**offline**				
Asterisk 1.6.0	ANO	ANO	ANO	ANO	ANO	ANO
Asterisk 10	OK	ANO	NE	NE	ANO	---
YATE 3.0	OK	ANO	NE	ANO	ANO	NE
FreeSWITCH 1.0	OK	ANO	ANO	NE	ANO	ANO

Vložení nežádoucích dat do RTP přenosu bylo umožněno ve všech případech, jelikož přenos dat probíhal prostřednictvím UDP protokolu, na tuto modifikaci ústředna neměla vliv.

Dalším testem, který byl proveden, bylo znepřístupnění webového rozhraní. Jak již bylo napsáno, tak útok typu slowloris je útok zaměřený převážně proti webovému serveru Apache. Nejedná se přímo o útok proti ústředně, ale demonstruje variantu, jak znepřístupnit webové rozhraní bez použití složitých útoků. Tento útok neměl vliv na ústřednu YATE, která disponovala webovým serverem Nginx, ten je proti útoku toho typu imunní. Test byl vynechán pro ústřednu Asterisk 10, která nedisponovala webovým rozhraním.

Výše provedené testy demonstrují, jak snadné je modifikovat hovory v IP telefonii nežádoucím způsobem, pokud nejsou k dispozici odpovídající metody zabezpečení a šifrovaní. V rámci útoku nebyl detailně rozebrán odposlech hovorů, který je umožněn prostřednictvím skriptu mitm.sh. Jedná se o velmi známou záležitost, jejíž provedení umožňuje například i Wireshark, pokud má útočník zajištěn přístup k síti. Jednotlivé obranné mechanismy byly rozebrány v teoretické části, kapitola 4.

7.4. Útoky typu odmítnutí služby

7.4.1. Útoky typu DoS na TCP/IP architekturu

V rámci této práce byly ověřeny možnosti tří ústředen Astersik, Yate a Freeswitch. Na všech byly otestovány jednotlivé útoky. Každý z testů probíhal v časovém intervalu 240 sekund, po 60 sekundách byl zahájen útok, který trval 120 sekund. V době útoku byla také ověřena možnost, zda lze uskutečnit hovor mezi účastníky. Ke každému útoku bylo zaznamenáno vytížení procesoru, množství přenesených dat během útoku a také byl ověřen fakt, zda je umožněno uskutečnit hovor mezi uživateli.

Aby byl simulován DoS útok, byla vybrána jako útočící stanice jedna konkrétní IP adresa. Samozřejmě existuje i možnost otestovat vytížení způsobem, kdy je na systém zasíláno velké množství požadavků ze stovek stanic. Nejednalo by se o DDoS, ale pouze o vytěžovací test, z toho důvodu tato možnost nebyla použita. Nejprve se jednalo o záplavy paketů prostřednictvím TCP SYN, ICMP a UDP.

Tab. 7.4: Útoky typu DoS

	port cíle	služba	port zdroje	velikost datové části [B]	měření [s]	útok [s]	IP adresa útočníka
SYN flood	80	http	1234	128	240	120	192.168.201.141
	443	https	1234	128			
ICMP flood	----		----	56			
	----		----	65000			
UDP flood	5060	SIP	1234	0			
	5060	SIP	1234	65000			

Každý z těchto útoků byl jednotlivě testován a pro ověření informací byl dvakrát opakován. Výstupní hodnoty téměř vždy odpovídali předchozí situaci. Nejprve byl proveden DoS útok záplavou SYN paketů. Jelikož SIP protokol je defaultně podporován přes UDP, tak bylo za cílový port zvoleno webové rozhraní ústředny. Z toho důvodu je žádoucí uvést i verzi a typ webové služby. Každá z těchto PBX podporovala jak http (port 80), tak https (port 443). Jedinou výjimku představuje Asterisk verze 10, kde byl dostupný pouze port 80. Pro ověření, jak se bude webové rozhraní a samotná ústředna chovat v době útoku, byla záplava paketu poslána právě na oba tyto porty. Výsledné hodnoty jsou vidět níže.

Tab. 7.5: Konfigurace ústředen

Ústředna	Verze	Rozhraní	Verze	Webový server	verze
Asterisk	1.6.0.26-FONCORE-r78	Trixbox	2.8.0.4	apache	2.2.3
Asterisk	10	---	----	apache	2.2.14
Yate	3.0.0	FreeSentral	1.2	apache	2.2.14
Freeswitch	1.0.head-git-4936b11	FusionPBX	3.0	nginx	1.0.9

V rámci této kapitoly bylo ověřeno jak se ústředna, nebo celý systém vyrovná se záplavou paketů. Každý z vytvořených testů byl proveden proti všem ústřednám, bylo zaznamenáno vytížení procesoru a celkový počet přenesených dat v rámci sítě. Také byla ověřena možnost uskutečnění hovoru mezi účastníky. Jak je zřejmé ze zaznamenaných dat, jednotlivé ústředny se vyrovnaly se záplavou paketů odlišně.

Ústředna Asterisk verze 1.6 se dokázala bez větších obtíží vyrovnat, jak se zaplavou TCP SYN paketů, tak ICMP paketů. Vytížení procesoru nedosahovalo vyšších hodnot a pohybovalo se stále okolo 10%. Maximálního vytížení procesoru dosáhla ústředna při záplavě paketů protokolu UDP, téměř po celou dobu útoku bylo vytížení procesoru 100%. Celý systém byl přetížen a díky tomu nebylo možné vytvořit hovor mezi uživateli.

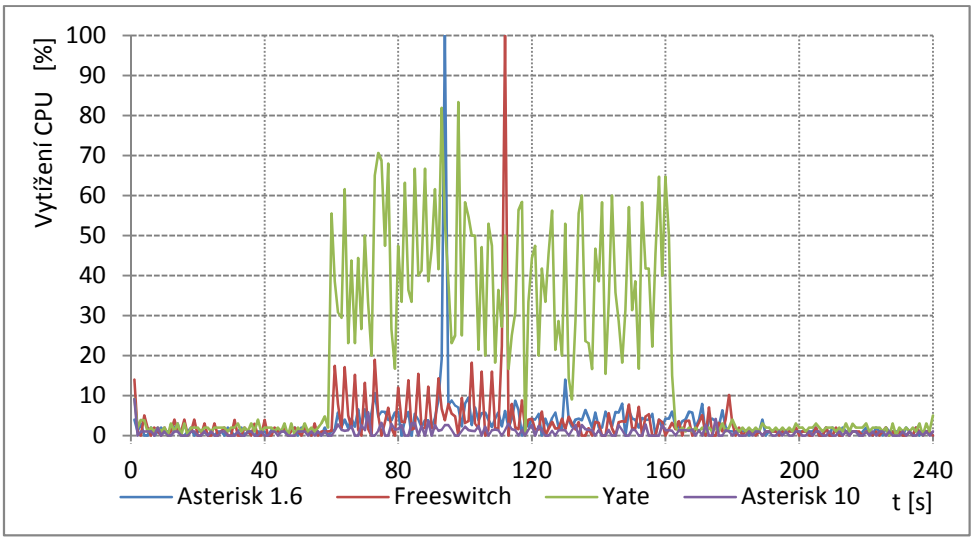
Na ústřednu Asterisk verze 10 neměly provedené útoky téměř žádný vliv, nedošlo ani k vytížení procesoru a ani nebyly omezeny funkce ústředny. Výjimku představují útoky, které měly za cíl zahltit komunikační kanál.

Ústředna YATE reagovala nejvíce ze všech ústředen na provedené útoky. Vytížení procesoru dosahovalo průměrných hodnot okolo 40 až 50 % i pří záplavě ICMP pakety. Hovor při útoku nebyl omezen. Při záplavě synchronizačních paketů dosahovalo vytížení procesoru místy až 100% a celý systém byl poměrně vytížený. Zatížení procesoru se opakovalo v cyklech, což umožnilo vytvoření hovoru mezi účastníky bez větších problémů. Stejně jako ústředna Asterisk 1.6 byl systém nejvíce zatížen pod záplavou paketů UDP,

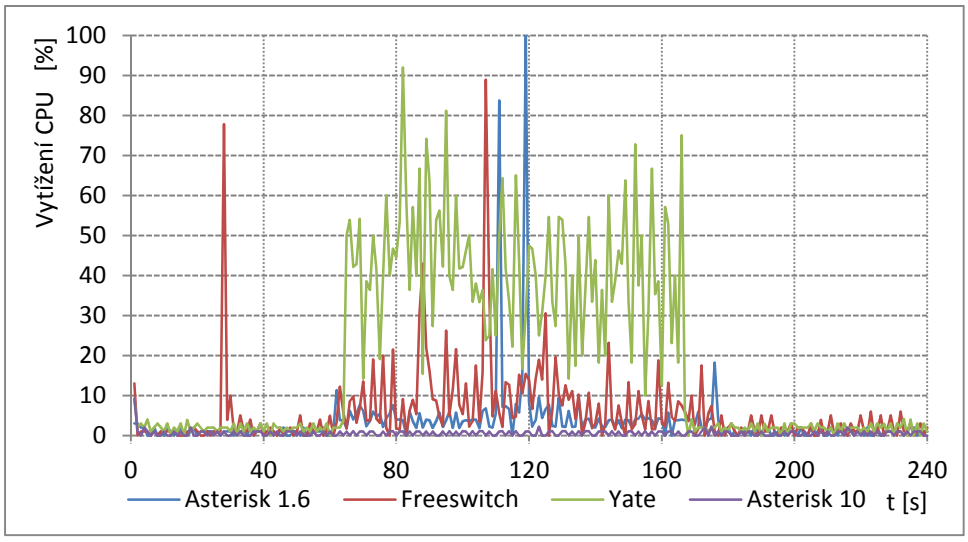
vytížení dosahovalo až 100%, ovšem na rozdíl od předchozího případu byl umožněn hovor mezi uživateli, i když s poměrně značným zpožděním.

Ústředna FreeSWITCH je podle výsledků nejvíce odolná proti útokům TCP SYN, UDP i ICMP. Naměřené hodnoty jasně ukazují, že celý systém nebyl ani v jednom z testů nějak závažně ovlivněn. Hovor byl ve všech případech uskutečněn, bez jakýchkoliv problémů.

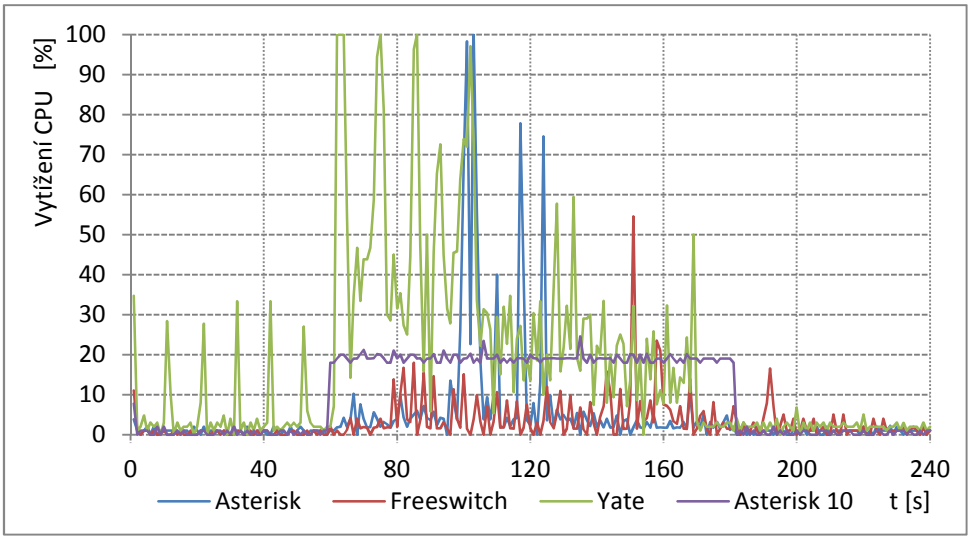
V rámci protokolu ICMP byla také ověřena záplava pakety o velikost 65000 bytů, které dokázaly vyprodukovat objem dat přesahující 100Mbit/s. Jelikož maximální šířka pásma ústředny je nastavena právě na tuto hodnotu, komunikační kanál byl přetížen a neumožnil příchozí ani odchozí hovor. Šířka pásma byla nastavena záměrně na 100Mbit/s, aby mohl být simulován útok, který dokáže překročit kapacitu linky. Analogickým způsobem funguje i útok, kdy byla nastavena velikost dat 65000 bytů pro protokol UDP, vytížení linky představovalo opět hodnoty překračující 100Mbit/s.



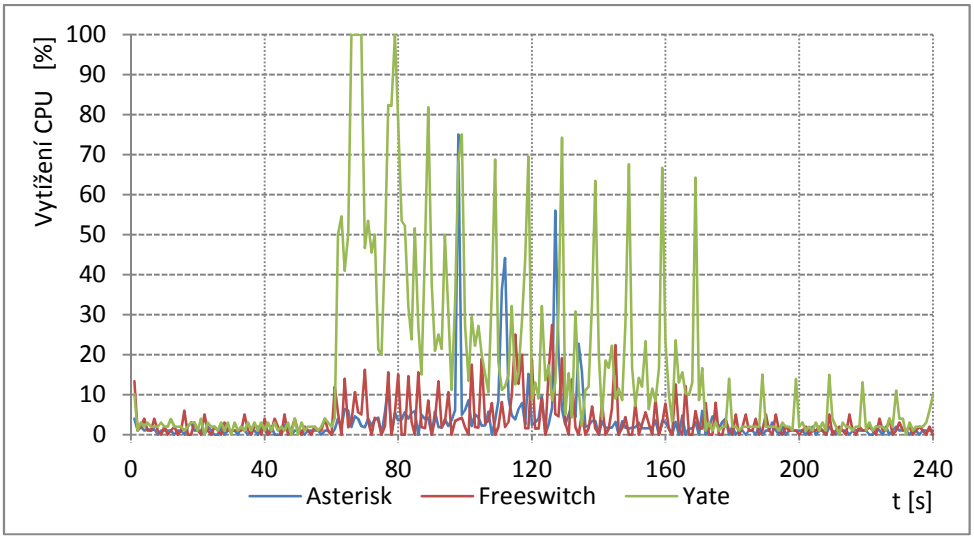
Obr. 7.3: ICMP flood velikost paketu 56 bytů



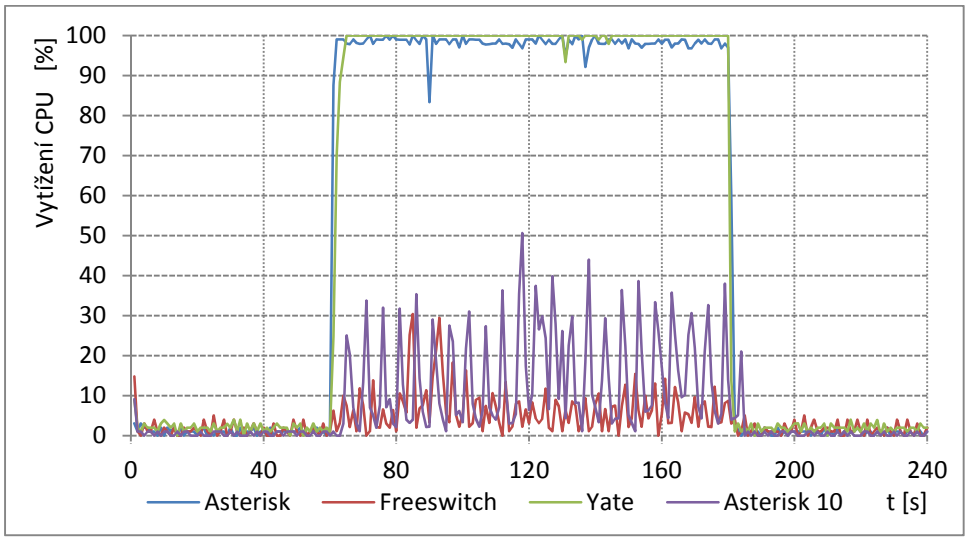
Obr. 7.4: ICMP flood velikost paketu 65535 bytů



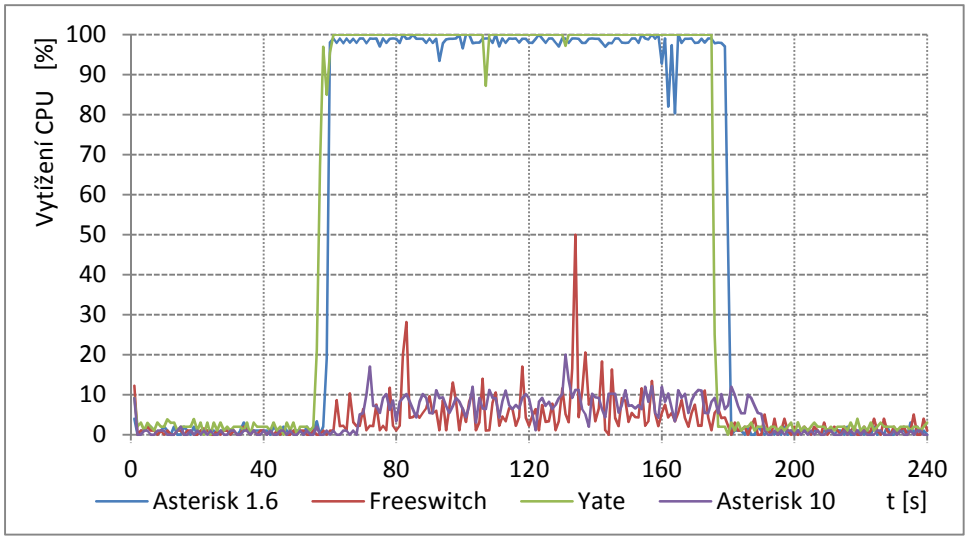
Obr. 7.5: SYN flood port 80 velikost dat 128 bytů



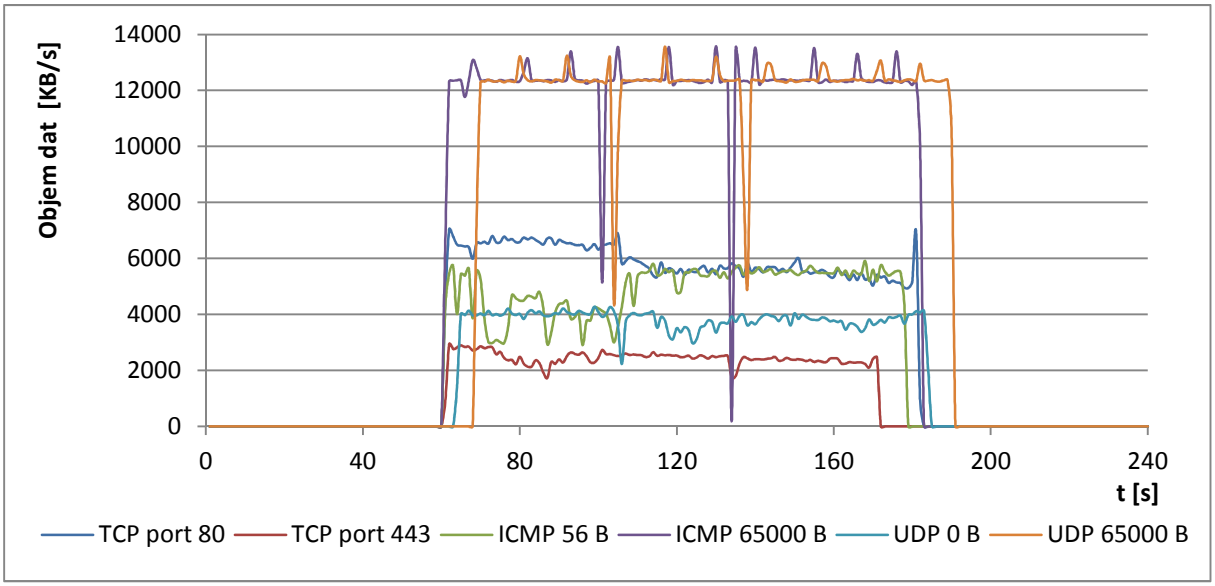
Obr. 7.6: SYN flood port 443 velikost dat 128 bytů



Obr. 7.7: UDP flood velikost dat 0 bytů



Obr. 7.8: UDP flood velikost dat 6500 bytů



Obr. 7.9: Objem zpracovaných dat na síťovém rozhraní pro jednotlivé útoky

Tab. 7.6: Shrnutí výsledků útoků typu DoS

	flood	**SYN**		**ICMP**		**UDP**	
	port	**80**	**443**			**5060**	**5060**
	velikost [B]	**128**	**128**	**56**	**65000**	**0**	**65000**
Asterisk 1.6	Hovor	ANO	ANO	ANO	NE	NE	NE
Asterisk 10	Hovor	ANO	-----	ANO	NE	ANO	NE
Yate	Hovor	ANO	ANO	ANO	NE	ANO *	NE
Freeswitch	Hovor	ANO	ANO	ANO	NE	ANO	NE

*spojení zpožděno

7.4.2. Útoky typu DoS využívající signalizačních zpráv protokolu SIP

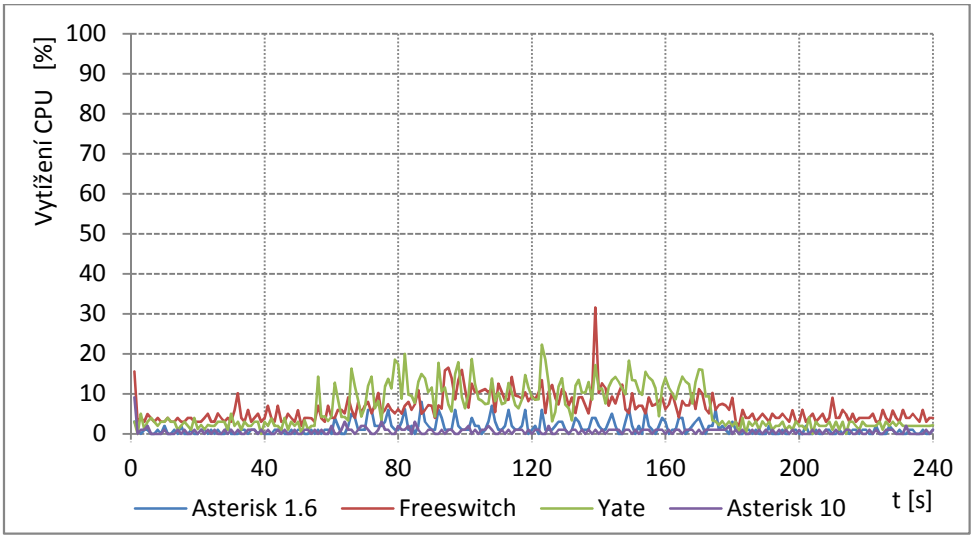
Jak již bylo uvedeno výše, mezi DoS útoky patří také kategorie, jenž využívá nedokonalosti VoIP protokolů. Princip je stejný jako v předešlých případech. Jedná se o vygenerování velkého počtu požadavků směrovaných na konkrétní služby nebo server, v tomto případě PBX systém. V rámci SIP protokolu je typickým cílem DoS útoku SIP proxy server. Na

následujících stránkách bude porovnána účinnost DoS útoku prostřednictvím jednotlivých zpráv SIP protokolu. Konkrétně se jedná o zprávy REGISTER, INVITE a OPTIONS. Cílem je odstavit PBX ústřednu, případně vytížit systém natolik, aby nebyl umožněn hovor mezi dvěma účastníky. Toho lze dosáhnout dvěma základními způsoby. Prvním z nich je zahlcení sítě velkým množstvím paketů, jak bylo znázorněno v předchozích případech. Další možností je přetížení PBX systému záplavou inicializačních zpráv, které server musí zpracovat, jelikož se může jednat o skutečné požadavky. Při zpracování dochází ke značnému zatěžování celého systému i při relativně nízké záplavě paketů. Jak již bylo uvedeno v předchozí kapitole, dos_sip.sh disponuje možností zaslat k ústředně zprávy typu INVITE, REGISTER, OPTIONS, ACK a BYE a lze definovat počet zpráv za sekundu. Postupně byly provedeny testy pro každou z uvedených SIP zpráv. Jako počáteční hodnota bylo zvoleno 500 zpráv za sekundu, následně 700, 1000 a 2000.

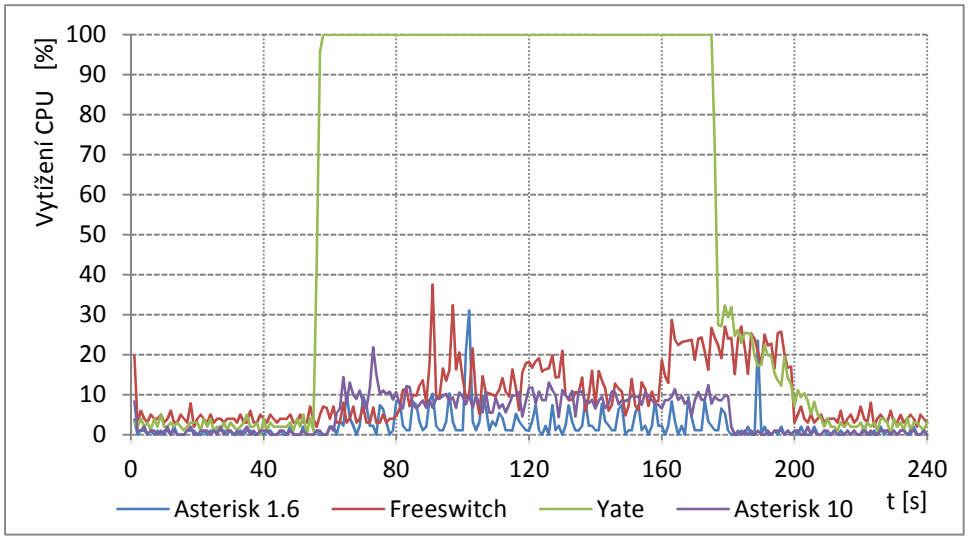
Průběh útoku probíhá stejným způsobem jako v předchozí kapitole. Útok je prováděn po dobu 120 sekund, data jsou zaznamenávána 240 sekund, 60 sekund před a 60 sekund po útoku. V době útoku je také ověřena možnost uskutečnění hovoru.

Tab. 7.7: Konfigurace testů útoků typu DoS – protokol SIP

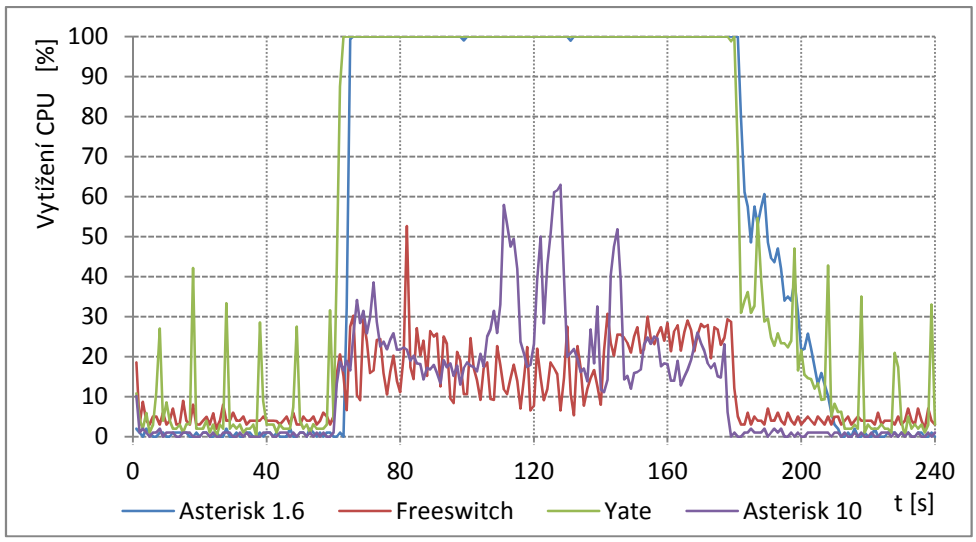
test	**I**	**II**	**III**	**IV**	**měření**	**útok**	**IP adresa útočníka**
	[zpráv/s]	**[zpráv/s]**	**[zpráv/s]**	**[zpráv/s]**	**[s]**	**[s]**	
INVITE	500	700	1000	2000	240	120	192.168.201.147
REGISTER							
OPTIONS							
ACK							
BYE							



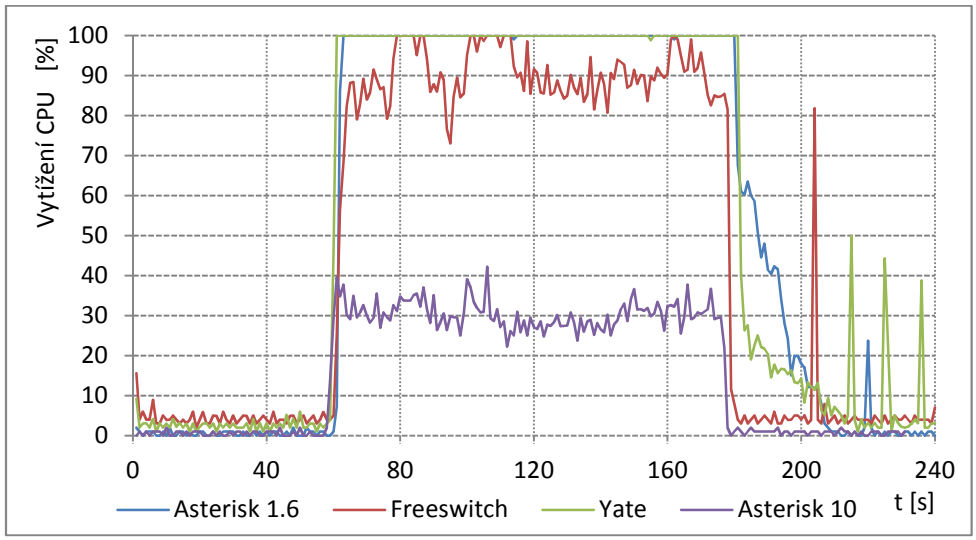
Obr. 7.10: ACK 700 zpráv/s



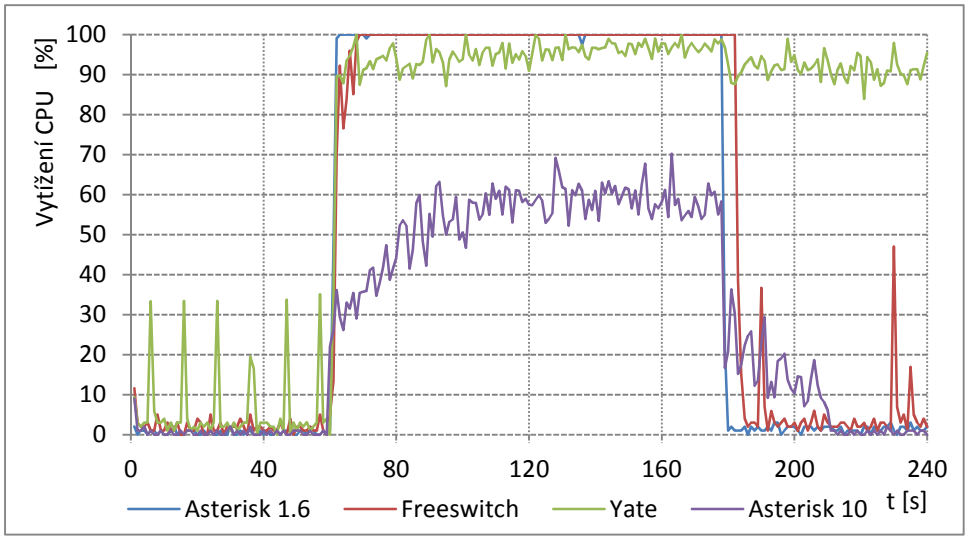
Obr. 7.11: BYE 700 zpráv/s



Obr. 7.12: OPTIONS 700 zpráv/s



Obr. 7.13: REGISTER 700 zpráv/s



Obr. 7.14: INVITE 700 zpráv/s

Jak bylo uvedeno, testy byly provedeny pro 500, 700, 1000 a 2000 zpráv za sekundu. Výše zobrazené grafy jsou naměřené hodnoty vytížení procesoru v průběhu testu pro 700 zpráv za sekundu, pro ostatní testy jsou grafy uvedeny v přílohách. Každý test byl proveden na jednotlivých ústřednách, aby bylo možné i jejich srovnání.

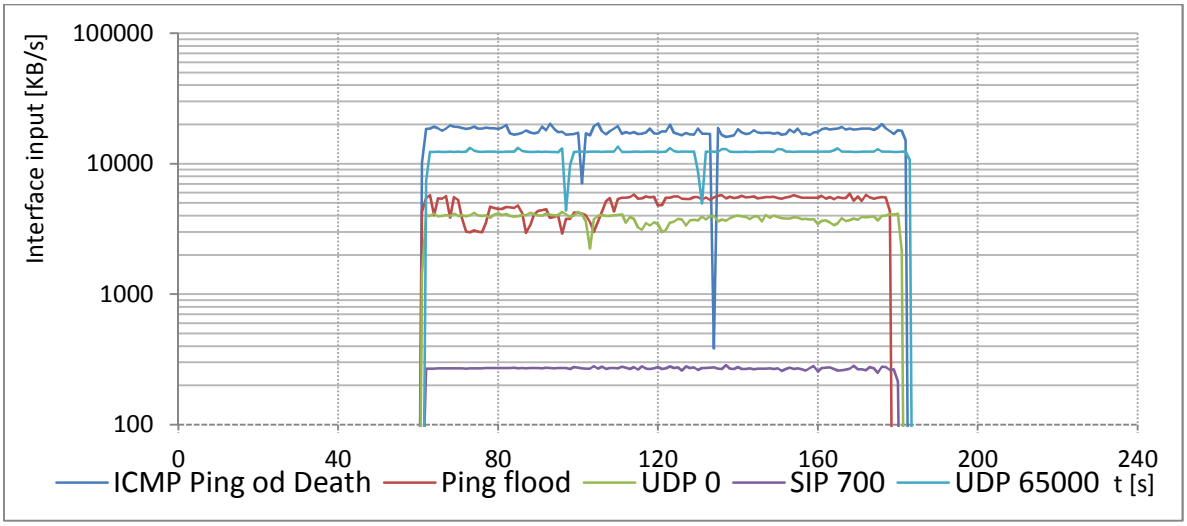
Z naměřených hodnot vyplývá, že nejmenší odolnost proti záplavě paketů má Asterisk verze 1.6, kde ústředna při záplavě 700 INVITE zpráv za sekundu přestala zcela odpovídat. Status ústředny se jevil v pořádku, ovšem neprobíhal zápis do logu a ani nebylo možné zaregistrovat jednotlivé uživatele. Z toho lze usoudit, že ústředna se nedokázala s množstvím příchozích požadavků vyrovnat a havarovala. Pro zprávu OPTIONS a REGISTER ústředna neodpovídala pouze během útoku, po uplynutí 120 sekund bylo opět možné vytvořit hovor. Zprávy ACK a BYE nezpůsobily ústředně Asterisk verze 1.6 závažnější problémy a nijak neovlivnili komunikaci uživatelů.

Útoky na ústřednu Asterisk verze 10 neměly ani v těchto testech příliš velký efekt. Vytížení procesoru bylo při testu zprávami BYE i ACK téměř na nulové hodnotě, a to i když byl systém pod záplavou 2000 zpráv za sekundu. Hovor bylo možné vytvořit, aniž by došlo ke zpoždění. V případě zprávy OPTIONS došlo ke srovnatelnému vytížení s ústřednou FreeSWITCH, kdy při útoku 700 zpráv za sekundu bylo průměrné vytížení procesoru okolo 20%. Pří záplavě zprávami REGISTER dosahovalo vytížení procesoru nejnižších hodnot ve srovnání s ostatními ústřednami, vytížení bylo pouze okolo 30% a hovor bylo možné vytvořit bez znatelného zpoždění. Dle předpokladu měla největší vliv zpráva INVITE stejně jako u ostatních ústředen. Vytížení procesoru bylo v tomto případě v rozmezí 50 až 60 %. To je opět nejnižší hodnota ve srovnání s ústřednou YATE a FreeSWITCH.

Ústředna YATE při záplavě 700 INVITE zpráv za sekundu neodpovídala stejně jako Asterisk 1.6, ovšem po ukončení testu začala odpovídat zcela normálně, i když vytížení procesoru bylo ještě několik minut téměř na 100%. Zprávy REGISTER a OPTIONS nezabránily vytvoření hovoru, ale docházelo ke značnému zpoždění. Zajímavostí je, že se ústředna nedokázala vyrovnat s příchozími SIP zprávami BYE. Tyto požadavky dokázali vytížit procesor na 100% a hovor mezi uživateli nebyl vytvořen. Po ukončení záplavy paketů se vytížení procesoru vrátilo na hodnotu okolo 20%. Zpráva ACK nedokázala ani v tomto případě nijak ovlivnit systém.

Dle dostupných zdrojů na internetu je ústředna FreeSWITCH schopna, na stejné konfiguraci jako Asterisk, obsloužit dvojnásobné množství hovorů. Tato skutečnost byla ověřena i v tomto testu, při srovnání ústředny Asterisk verze 1.6 a ústředny FreeSWITCH verze 1.0. Procesor ústředny byl vytížen na 100% po celou dobu testu, ale ani to nezabránilo vytvoření hovoru mezi uživateli. Došlo jen ke zpoždění, které bylo v řádu sekund. V rámci testovacího skriptu sice nebyla doručena odpověď na zprávu OPTIONS, ale to lze přisoudit nastavenému limitu v testu. Pokud není odpověď doručena do tří sekund, je ústředna považována za nedostupnou.

Ústředny se se záplavou paketů nedokázaly vypořádat, kromě ústředny FreeSWITCH, která dokázala odpovídat na příchozí hovory i při záplavě 2000 INVITE zpráv za sekundu. Při tomto testu docházelo ke zpoždění okolo 10 sekund, ale i tak se hovor podařilo uskutečnit. Ústředna Asterisk verze 10 nedokázala při tomto množství INVITE požadavků hovor vytvořit. V příloze B jsou uvedeny i výsledky ostatních testů pro 500 a 1000 zpráv za sekundu, výsledky zkušebního hovoru jsou shrnuty v tabulkách níže.



Obr. 7.15: Množství přenesených dat pro jednotlivé útoky

Tab. 7.8: Útok typu DoS signalizačními zprávami protokolu SIP – 500 zpráv/s

500 zpráv/s	**INVITE**	**REGISTER**	**OPTIONS**	**ACK**	**BYE**
Asterisk 1.6	NE	NE	NE	ANO	ANO
		*	*		
Asterisk 10	ANO	ANO	ANO	ANO	ANO
	bez následku	bez následku	bez následku	bez následku	bez následku
Yate	NE	ANO	ANO	ANO	NE
	*	**	**		*
Freeswitch	ANO	ANO	ANO	ANO	ANO
	spojení zpožděno	bez následku	bez následku	bez následku	bez následku

Tab. 7.9: Útok typu DoS signalizačními zprávami protokolu SIP – 700 zpráv/s

700 zpráv/s	INVITE	REGISTER	OPTIONS	ACK	BYE
Asterisk 1.6	NE	NE	NE	ANO	ANO
	ústředna havarovala	*	*		
Asterisk 10	NE	ANO	ANO	ANO	ANO
	*	bez následku	bez následku	bez následku	bez následku
Yate	NE	NE	ANO	ANO	NE
	*	**	** *		*
Freeswitch	ANO	ANO	ANO	ANO	ANO
	spojení zpožděno o 5 sekundy	bez následku	bez následku	bez následku	bez následku

Tab. 7.10: Útok typu DoS signalizačními zprávami protokolu SIP – 1000 zpráv/s

1000 zpráv/s	INVITE	REGISTER	OPTIONS	ACK	BYE
Asterisk 1.6.0	NE	NE	NE	ANO	ANO
	ústředna havarovala	*	*		
Asterisk 10	NE	ANO	ANO	ANO	ANO
	*		**		
Yate	NE	NE	ANO	ANO	NE
	*	**	***		*
Freeswitch	ANO	ANO	ANO	ANO	ANO
	spojení zpožděno o 5 sekundy	bez následku	bez následku	bez následku	bez následku

Tab. 7.11: Útok typu DoS signalizačními zprávami protokolu SIP – 2000 zpráv/s

2000 zpráv/s	INVITE	REGISTER	OPTIONS
Asterisk 1.6.0	NE	NE	NE
	ústředna havarovala	ústředna havarovala	*
Asterisk 10	NE	ANO	ANO
	*	**	**
Yate	NE	NE	NE
	*		
Freeswitch	ANO	ANO	ANO
	***	***	***

* Po ukončení útoku ústředna začala odpovídat téměř okamžitě

** Spojení zpožděno (2-3 sekundy)

*** Úspěšnost v uskutečnění spojení 50%

7.5. Implementace zabezpečení proti útokům typu DoS

Z výsledků testů v předchozí kapitole je zřejmé, že i při relativně nízké záplavě paketů je možné ústřednu vytížit natolik, aby nemohly být zpracovány legitimní požadavky. K efektivní ochraně proti DoS útokům tohoto typu slouží několik nástrojů. V tomto případě bylo využito centralizované řešení. Jak bylo popsáno v teoretické části, jako ochrana proti útokům DoS a neoprávněným aktivitám v síti je použito systému NIDS Snort.

K monitorování celé LAN sítě je využit síťový interface, který nemá nastavenou IP adresu a díky tomu nefiltruje příchozí data. Spolu s agentem SnortSam tvoří kombinaci síťového monitorovacího zařízení, které zároveň zajišťuje preventivní ochranu v reálném čase. SnortSam je nainstalován na PBX ústředně a čeká na příchozí požadavky od systému Snort. K tomu, aby SnortSam mohl vykonat nějakou akci, je potřeba přidat do Snortpravidel

specifické instrukce. Příkladem může být následující pravidlo, které slouží k zastavení INVITE flood:

```
alert ip any any -> $SIP_PROXY_IP $SIP_PORTS (msg:"INVITE message
flooding"; content:"INVITE"; \
depth:6; threshold: type both , track by_src, count 100, seconds 60; \
sid:5000004; rev:1; fwsam: src, 5 minutes;)
```

Nastavené pravidlo vygeneruje výstrahu, pokud sítí prochází více jako 100 zpráv za 60 sekund, které jsou generovány z jednoho zdroje a jsou odeslány na SIP server a definovaný port. K agentovi SnortSam bude odeslána informace, že byla detekována aktivita, která odpovídá nastavenému pravidlu. Výstraha je zapsána do logu a SnortSam provede odpovídající akci. Záznam z logu je zobrazen níže.

```
** Alert 1337012683.85886: - ids,
2012 May 14 18:24:43 ossec->/var/log/auth.log
Rule: 20101 (level 6) -> 'IDS event.'
Src IP: 192.168.201.147
Dst IP: 192.168.201.148
May 14 18:24:41 ossec snort[2910]: [1:5000004:1] INVITE message flooding
{UDP} 192.168.201.147:5060 -> 192.168.201.148:5060
```

Toto upozornění uvádí informace o tom, kdy byla data odchycena, IP adresu zdroje a cíle, port zdroje a cíle a další informace, které vycházejí z nastavení konkrétního pravidla. Ze záznamu je zřejmé, že bylo aplikováno pravidlo s identifikačním číslem 5000004, jednalo se o INVITE message flooding. SnortSam obdržel zprávu od NIDS a okamžitě zablokoval IP adresu útočníka. Blokace proběhla prostřednictvím firewallu iptables. Dle konfigurace pravidla byla veškerá komunikace s útočníkem zablokována po dobu 300 sekund a to jak příchozí, tak odchozí komunikace.

```
2012/05/14, 18:24:41, 192.168.201.156, 3, snortsam, Accepted connection
from 192.168.201.156.

2012/05/14, 18:24:41, 192.168.201.156, 2, snortsam, Blocking host
192.168.201.147 completely for 300 seconds (Sig_ID: 5000004).

2012/05/14, 18:24:41, -, 3, iptables, Info: Blocking ip 192.168.201.147

2012/05/14, 18:24:41, -, 3, iptables, Info: Command /sbin/iptables -I
FORWARD -i eth1  -s 192.168.201.147 -j DROP Executed Successfully

2012/05/14, 18:24:41, -, 3, iptables, Info: Command2 /sbin/iptables -I
INPUT -i eth1  -s 192.168.201.147 -j DROP Executed Successfully

2012/05/14, 18:24:41, -, 3, iptables, Info: Command /sbin/iptables -I
FORWARD -i eth1  -d 192.168.201.147 -j DROP Executed Successfully
```

```
2012/05/14, 18:24:41, -, 3, iptables, Info: Command2 /sbin/iptables -I
INPUT -i eth1  -d 192.168.201.147 -j DROP Executed Successfully
```

Pravidla jsou aplikovatelná na všech testovaných ústřednách. Z důvodu analogických výsledků zobrazují následující grafy vytížení procesoru před zabezpečením a po implementaci zabezpečení pouze pro ústřednu, které dopadla v testech z předchozí kapitoly nejhůře. Pro jednotlivé útoky byly nastaveny stejné podmínky, tedy systém je monitorován po dobu 240 sekund, samotný útok je veden po dobu 120 sekund.

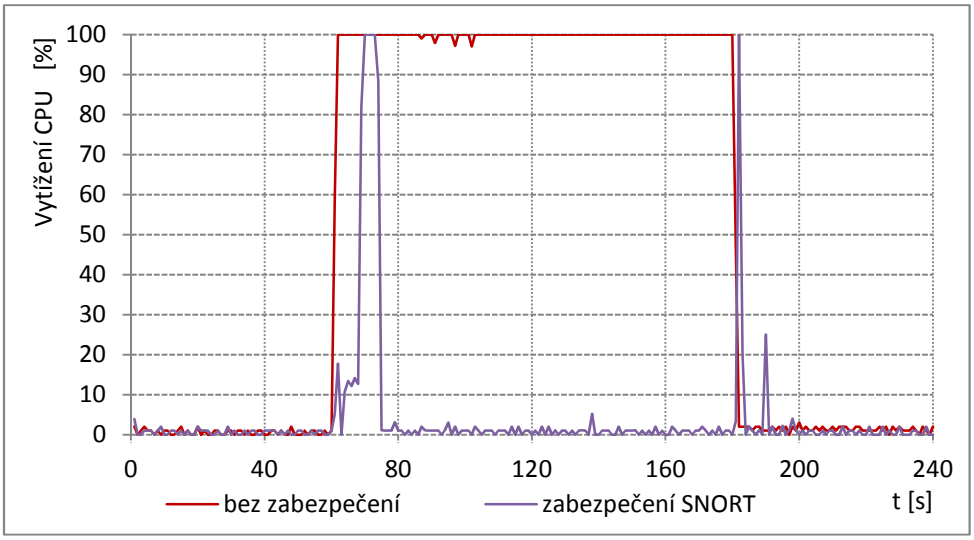
Na obrázku 7.16 -7.18 je zaznamenáno vytížení procesoru před a po zabezpečení proti provedeným útokům, v tomto případě se jednalo o ústřednu Asterisk verze 1.6.0. Pokud byl útok proveden před zabezpečením, ústředna nebyla schopna vytvořit žádné spojení mezi uživateli, jelikož byla přetížena zpracováním zpráv ze simulovaného útoku, procesor byl vytížen na 100% po celou dobu útoku. Po aplikaci zabezpečení pomocí nástroje Snort, bylo vytížení procesoru po celou minimální, i při spojení hovoru. To je způsobeno nastavenou hardwarovou konfigurací, takže v grafech není znatelný ani vytvořený hovor. Je ovšem nutno podotknout, že konfigurace byla stejná i před zabezpečením.

Před zabezpečením, při tomto testu, ústředna havarovala, po implementaci zabezpečení bylo vytížení procesoru minimální, kromě momentu, než bylo aplikované Snort pravidlo. V tomto bodě se podařilo procesor ústředny vytížit na 100%, ovšem jen po dobu jedné sekundy. To bylo pravděpodobně způsobeno zpožděním při aplikaci pravidla. Jednalo se o tak krátký časový interval, že se s tímto objemem dat dokázala ústředna vyrovnat bez větších obtíží.

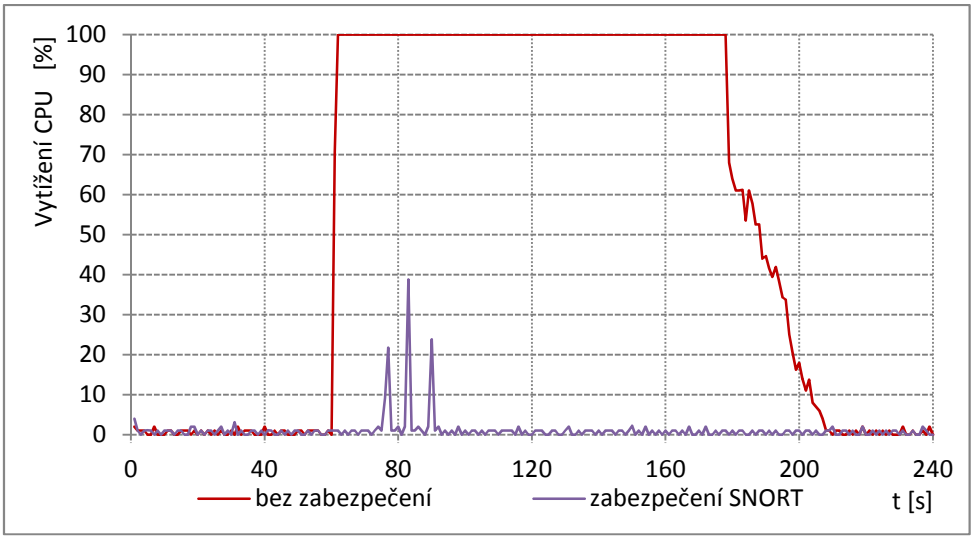
Tab. 7.12: Výsledky zabezpečení pomoci Snort– Asterisk 1.60

1000 zpráv/s	**INVITE**	**REGISTER**	**OPTIONS**
Asterisk 1.6.0 nezabezpečeno	NE	NE	NE
	ústředna havarovala	*	*
Asterisk 1.6.0 nezabezpečeno SNORT	ANO	ANO	ANO
	bez následku	bez následku	bez následku

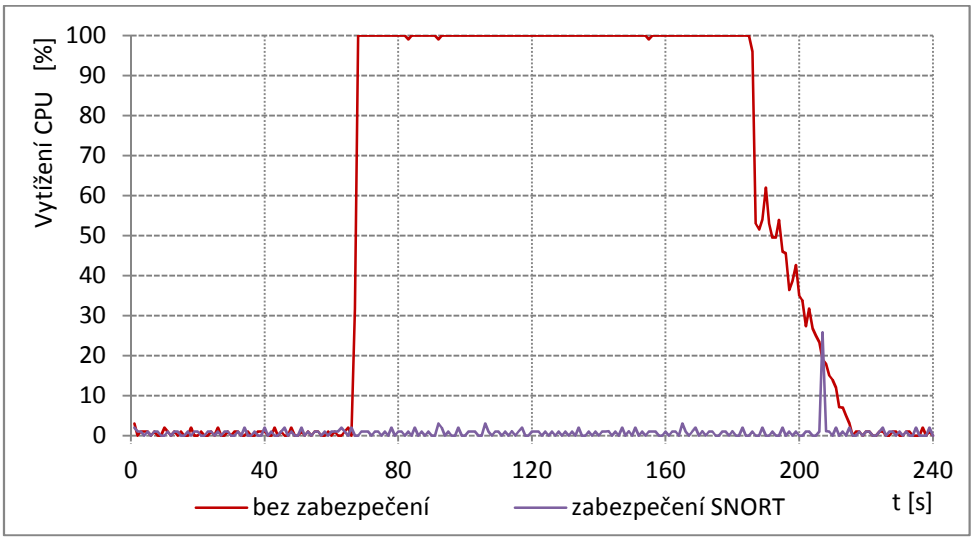
Pro další dva testované útoky byla situace stejná. Před zabezpečením ústředna nedokázala odpovídat na žádné příchozí požadavky, naproti tomu po zabezpečení byla komunikace mezi uživateli vytvořena bez jakéhokoliv zpoždění. V příloze C jsou uvedena všechna pravidla, která byla pro zabezpečení vytvořena. Princip dalších útoků pomocí záplavy signalizačních zpráv je obdobný. Pravidla jsou nastavena tak, že potencionální útočník nemůže generovat vetší množství jednotlivých zpráv než je 100 zpráv za sekundu. V rámci testované virtuální sítě je toto nastavení dostatečné. Pokud by však bylo toto zabezpečení implementováno v rozsáhlejších LAN sítích, bylo by pravděpodobně nutné limity upravit konkrétním potřebám. To lze provést jednoduchým způsobem, zvýšením hodnoty *count* v daném pravidle.



Obr. 7.16: Asterisk 1.6.0 – INVITE 1000 z/s – zabezpečení Snort



Obr. 7.17: Asterisk 1.6.0 – OPTIONS 1000 z/s – zabezpečení Snort

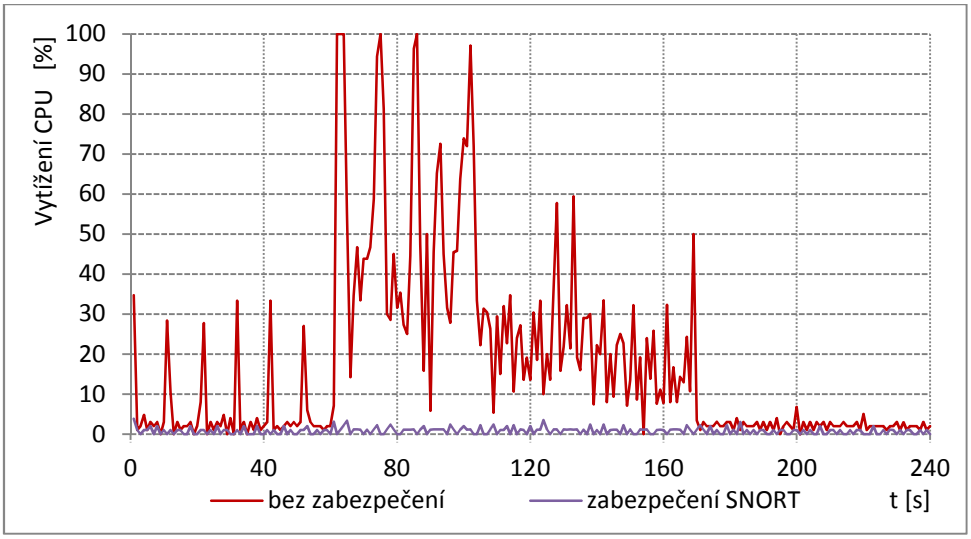


Obr. 7.18: Asterisk 1.6.0 – REGISTER 1000 z/s – zabezpečení Snort

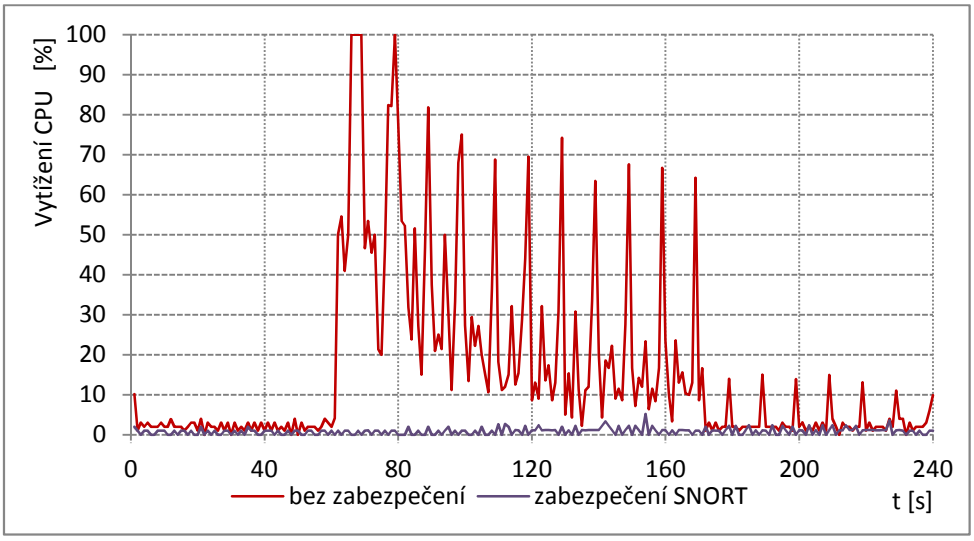
V rámci testovacích útoků byla ověřena i možnost, jak se jednotlivé ústředny vyrovnají se záplavou paketů protokolu UDP, ICMP a TCP synchronizačního paketu. V těchto testech byla pomyslným vítězem vyhodnocena ústředna YATE 3.0 a pro protokol UDP ústředna Asterisk verze 1.6.0. Tyto ústředny reagovaly na útoky, v porovnání s ostatními systémy, nejhůře. Opět bylo zaznamenáno vytížení procesoru před a po zabezpečení ústředen. Srovnaní je vidět v jednotlivých grafech 7.19-7.21.

Tab. 7.13: Výsledky zabezpečení proti DoS útokům pomocí Snort

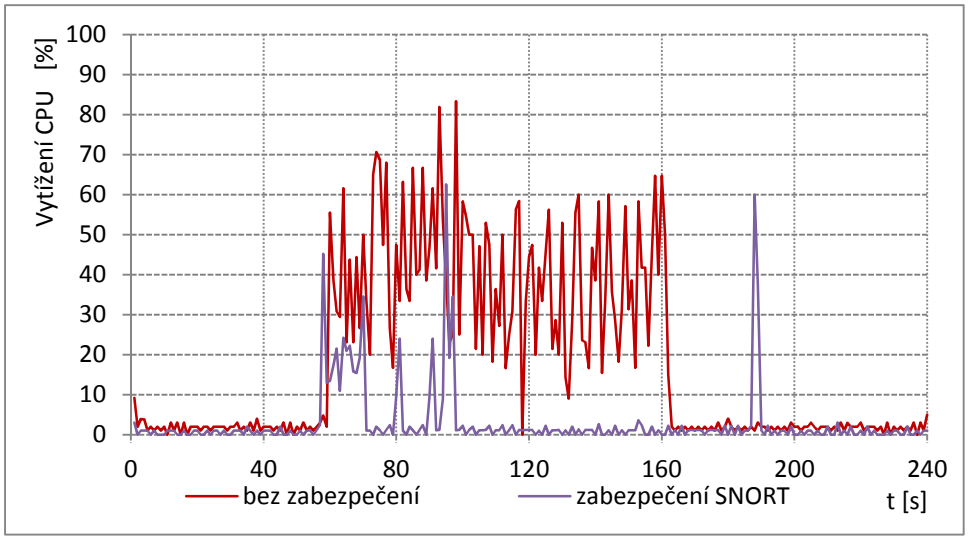
ústředna		**YATE 3.0**			**Asterisk 1.6.0**
flood		**SYN**		**ICMP**	**UDP**
port		**80**	**443**	**----**	**5060**
velikost[b]		**128**	**128**	**56**	**0**
nezabezpečeno	**hovor**	ANO	ANO	ANO	NE
zabezpečeno SNORT	**hovor**	ANO	ANO	ANO	ANO



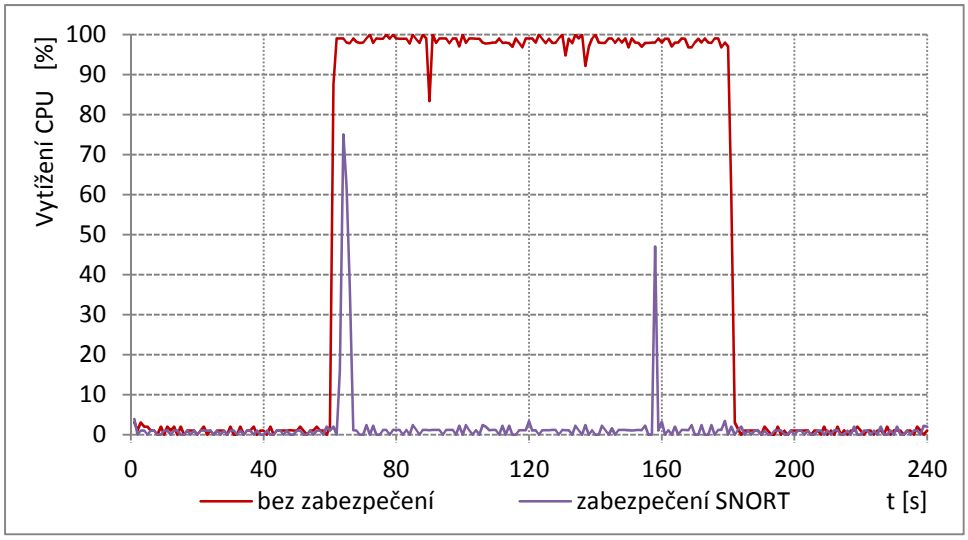
Obr. 7.19: YATE 3.0 - SYN flood port 80 – zabezpečení Snort



Obr. 7.20: YATE 3.0 - SYN flood port 443 – zabezpečení Snort



Obr. 7.21: YATE 3.0 – ICMP flood – zabezpečení Snort



Obr. 7.22: Asterisk 1.6.0 – UDP flood port 5060 – zabezpečení Snort

Po zabezpečení se vytížení procesoru znatelně snížilo, a i když bylo zaznamenáno určité vytížení systému, nejednalo se o tak závažný problém jako v případě, kdy ústředna nebyla zabezpečena. Při záplavě UDP paketů nebyly zaznamenány žádné potíže ani při sestavení hovoru.

V rámci testovaných útoků byl proveden i test omezující šířku pásma komunikačního kanálu. Jednotlivé ústředny mají nastaveno síťové rozhraní, které umožňuje přenést maximálně 100 Mbit/s. Díky útokům s pakety přesahujících 65000 bytů bylo možné vytvořit tok dat o velikost 120 Mbit/s, to umožnilo zahltit linku natolik, že nemohl být proveden hovor mezi uživateli. V rámci virtuální sítě nebylo možné zabezpečení útoku efektivně otestovat, jelikož Snort by v tomto případě vyžadoval propojení s firewallem, který by dokázal filtrovat příchozí komunikaci do interní sítě. Pravidlo, které by mělo zamezit tomuto typu útoku, musí vycházet z faktu, že data ICMP paketu anebo UDP paketu jsou fragmentována v protokolu IP. Tím pádem není možné vytvořit pravidlo pro protokol ICMP nebo protokol UDP. Ukázka části paketu je vidět níže.

```
0000   00 0c 29 70 1d 72 00 0c 29 92 8f e7 08 00 45 00  ..)p.r..).....E.
0010   05 dc 0e 88 2d bb 40 01 24 65 c0 a8 c9 94 c0 a8  ....-.@.$e......
0020   c9 93 58 58 58 58 58 58 58 58 58 58 58 58 58 58  ..XXXXXXXXXXXXXX
0030   58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58  XXXXXXXXXXXXXXXX
……
……
05e0   58 58 58 58 58 58 58 58 58 58                    XXXXXXXXXX
```

Pro test je podstatná ta část paketu, které obsahuje data. Nástroj hping3, který je použit při testování, vyplní pole data stejnou hodnotou od začátku až do konce paketu. To se dá použít jako ukazatel na nelegitimní paket, ale není to zcela ideální řešení. V tomto případě by bylo vhodnější nastavit statické pravidlo přímo pro firewall. Testovací síť je postavena na virtuálním prostředí, z toho důvodu nejsou nastavena pravidla pro Snort proti tomuto útoku.

V této kapitole byly ověřeny možnosti zabezpečení PBX ústředny proti DoS útokům. Všem útokům, které byly provedeny, bylo úspěšně zabráněno. Jedinou výjimkou jsou útoky typu DoS omezující šířku pásma, ovšem zabezpečení bylo popsáno teoreticky.

7.6. Implementace zabezpečení proti skenování LAN sítě.

V rámci VoIPtesteru je k dispozici také nástroj enumeration.sh, ten umožňuje získat cenné informace o systémech a aplikacích, které na něm běží, jak je popsáno v kapitole 3. Jedná se o techniku, která je těžce odhalitelná v rámci síťového provozu, je možné ji ale odhalit z logů přímo v systému. Jako ochrana může být použit právě detekční systém OSSEC, který umožňuje monitorování systému. Díky nastaveným pravidlům je možné zamezit útočníkovi v získávání informací prostřednictvím sítě, k zamezení přístupu slouží opět kombinace agent a správce. Pomocí agenta je možné nastavit, jaké logy budou monitorovány. Nastavení se provádí na straně agenta konfiguračním souborem *ossec.conf*. Teoretická část popisující architekturu systému byla popsána v kapitole 5. Jednotlivá pravidla jsou definována na straně serveru, kam jsou odesílány i informace od jednotlivých agentů. Server postupně tyto data prochází a porovnává je se svými pravidly. Pokud nastane shoda je vygenerováno upozornění, příkladem může být následující zpráva.

```
** Alert 1337109166.42876: - web,accesslog,attack,
2012 May 15 21:12:46 (asterisk1.6) 192.168.201.148-
>/var/log/httpd/access_log
Rule: 31104 (level 6) -> 'Common web attack.'
Src IP: 192.168.201.147
192.168.201.147 - - [15/May/2012:15:12:44 -0400] "GET
/sdk/%2E%2E/%2E%2E/%2E%2E/%2E%2E/%2E%2E/%2E%2E//etc/vmware/hostd/vmInventor
y.xml HTTP/1.1" 404 310 "-" "Mozilla/5.0 (compatible; Nmap Scripting
Engine; http://nmap.org/book/nse.html)"
```

Toto upozornění bylo vytvořeno v době, kdy se nástroj nmap pokoušel získat informace o nastavení webového serveru Apache. Z této zprávy je zřejmé, že při kontrole souboru */var/log/httpd/access_log* byl nalezen záznam, který odpovídá pravidlu s identifikačním číslem 31104, úroveň výstrahy 6. Dále jsou tam uvedeny následující informace: kdy bylo upozornění zaznamenáno, IP adresa zdroje a IP adresa cíle a další informace v závislosti na nastavení pravidla. Pravidlo, které vytvořilo tento záznam, je uvedeno níže.

```
<rule id="31104" level="6">
     <if_sid>31100</if_sid>
     <url>%027|%00|%01|%7f|%2E%2E|%0A|%0D|../..|..\..|echo;|..|</url>
     <url>cmd.exe|root.exe|_mem_bin|msadc|/winnt/|</url>
     <url>/x90/|default.ida|/sumthin|nsiislog.dll|chmod%|wget%|cd%20|</url>
     <url>cat%20|exec%20|rm%20</url>
     <description>Common web attack.</description>
     <group>attack,</group>
</rule>
```

Pravidla jsou implementována automaticky v rámci detekčního sytému OSSEC. Jak již bylo uvedeno v teoretické části, samotný OSSEC disponuje mnoha pravidly, které jsou již otestovány a pravidelně aktualizovány na webových stránkách tohoto nástroje. Díky těmto pravidlům je nejen možné získat informace o skenování, ale je možné nastavit OSSEC, aby zamezil zdrojové IP adrese v pokračování nelegitimní činnosti. K tomu účelu slouží následující nastavení na straně serveru v konfiguračním souboru *ossec.conf*.

```
<active-response>
     <disabled>no</disabled>
     <command>firewall-drop</command>
     <location>local</location>
     <level>6</level>
     <rules_id>31104</rules_id>
     <timeout>600</timeout>
</active-response>
```

Tímto způsobem bude útočník, který se pokoušel získat informace o systému, blokován po dobu 300 sekund. Pravidlo, které umožní blokaci je *firewall-drop*. K definici kde použít pravidlo slouží lokace v tomto případě se jedno o *local* , to znamená, že akce bude provedena na systému, kde byla výstraha generována

7.7. Zabezpečení proti sniffování hesla

Jak bylo rozebráno v kapitole 7.3, pomocí nástroje user_sniffer_online.sh je možné nalézt uživatel existující uživatele na PBX ústředně a následně se pro ně zjistit heslo. I když je tento test poněkud časově náročnější, tak při dostatečně dlouhém časovém intervalu je možné nalézt správnou kombinaci. Nabízí se několik variant jak tomuto útoku zabránit, jednou z nich může být nastavení registračního limitu pro jednotlivé uživatele. To znamená, že pokud uživatel zadá opakovaně za sebou špatné heslo, bude jeho účet zablokován, buď na určitou dobu nebo dokud správce systému účet neodblokuje. Toto řešení má ale značnou nevýhodu, jednoduchou úpravou skriptu user_sniffer_online.sh by bylo možné zablokovat všechny uživatele registrované k PBX ústředně.

V rámci této práce je vytvořeno opatření pomocí nástroje OSSEC, opět je využito pravidel, které jsou definované v souboru *asterisk.xml*, tato pravidla jsou uvedena také v příloze D. Systém OSSEC monitoruje aktivitu na ústředně a pokud zaznamená podezřelou aktivitu, vytvoří odpovídající akci. Výstupní log z ústředny je vidět níže:

```
[May 22 10:30:38] NOTICE[1086] chan_sip.c: Call from ''
(192.168.201.147:5060) to extension '120' rejected because extension not
found in context 'default'.
[May 22 10:30:38] NOTICE[1086] chan_sip.c: Call from ''
(192.168.201.147:5060) to extension '121' rejected because extension not
found in context 'default'.
```

Na základě výstupního logu OSSEC server vyhodnotí situaci, jako pokus o uhodnutí uživatelského jména a zablokuje IP adresu, odkud požadavky přicházely. Blokace proběhne podle definované akce v konfiguračním souboru *ossec.conf* na straně serveru.

```
<active-response>
<disabled>no</disabled>
<command>firewall-drop</command>
<location>local</location>
<level>5</level>
<rules_group>asterisk</rules_group>
<timeout>600</timeout>
</active-response>
```

Toto nastavení je podobné jako v předchozí kapitole o zabezpečení skenování proti LAN síti, jediným rozdílem, je že aktivní ochrana není definována pro jedno pravidlo, ale pro celou skupinu asterisk. Blokace útočníkovi IP adresu proběhne stejně jako v předchozím případě.

Tímto způsobem je možno zajistit, aby PBX ústředna byla chráněna před nelegitimními pokusy zjišťování uživatelů. Analogicky je možné zablokovat i pokusy o uhodnutí hesla, jen je použito jiných pravidel, ale výsledek je stejný, po překročení stanoveného limu je IP adresa blokována. Tato pravidla jsou vytvořena přímo pro PBX ústřednu Asterisk, ovšem podobným způsobem je možné je vytvořit i pro ústředny FreeSWITCH a YATE.

Další variantou je využití systému Snort, kde jsou nastavena pravidla i pro zprávu REGISTER. Útok, kterým je detekováno heslo generuje poměrně velké množství požadavků na PBX ústřednu, díky tomu může být tento útok detekován síťovými analyzátory, v tomto případě Snort. Níže uvedené pravidlo je analogické s pravidlem, které bylo vytvořeno pro obranu před útoky typu DoS, při překročení limitu 100 zpráv za 60 sekund je zdrojová IP adresa blokována po dobu 5 minut

```
alert ip any any -> $SIP_PROXY_IP $SIP_PORTS (msg:"REGISTER message
flooding"; content:"REGISTER"; depth:8; threshold: type both , track
by_src, count 100, seconds 60; sid:5000005; rev:1; fwsam: src, 5 minutes;)
```

Závěr

Open source PBX ústředna je klíčovým prvkem, ale zároveň se jedná i o nejzranitelnější místo celé VoIP sítě. Jedná se o zařízení, kterým prochází veškerá komunikace v případě B2BUA serveru, nebo alespoň signalizační komunikace v případě proxy serveru. Je proto nutné vytvořit dostatečně kvalitní zabezpečení, aby dokázalo tento systém chránit před útoky, které by mohli mít negativní vliv na komunikaci uživatelů nebo dokonce na celý PBX systém.

Profesionální komerční řešení od firem jako je CISCO, Siemens anebo AVAYA, nabízí kompletní řešení včetně zabezpečení systému, které dokáže zabránit většině známých útoků. Ovšem takovéto kompletní řešení často představuje vysoké pořizovací náklady a další prostředky na údržbu systému. Na druhé straně stojí open source PBX systémy, jedná se o zařízení, které dokáže nabídnout stejně kvalitní služby jako komerční PBX za zlomek ceny, ne-li zcela zdarma. Na rozdíl od komerčních řešení open source PBX, jsou často závislé na produktech třetích stran nebo na ochotě jednotlivých komunit vytvořit požadované nástroje či chybějící moduly. Open source PBX je čím dál více oblíbenější jak mezi IT experty, tak mezi uživateli. Podniky se snahou snížit náklady na provoz se obracejí právě na tyto systémy, jelikož jim dokáží nabídnout srovnatelné služby, i když za cenu složitější implementace.

Existuje mnoho variant útoků zaměřené na VoIP systémy, v této práci byly detailně popsány ty nejznámější a zároveň nejzávažnější útoky. Útoky lze rozdělit do dvou základních kategorií, útoky záplavového typu a útoky využívající chyb signalizačního protokolu SIP. Útoky záplavového typu jsou relativně snadno proveditelné, v porovnání s tím jaké dokážou způsobit následky. V praktické části bylo prokázáno, že i při relativně malém vytížení je možné systém zcela vyřadit z provozu. Útoky využívající chyb signalizačního protokolu SIP jsou útoky, které jsou spíše zaměřené na jednotlivé uživatele, než na celý PBX systém. Ovšem v rámci PBX systému je možné jim zabránit použitím vhodných bezpečnostních protokolů.

Pro bezpečnou komunikaci mezi PBX a uživateli existují tři varianty zabezpečení. Jedná se o protokol S/MIME, IPsec a TLS. Protokol S/MIME je primárně zaměřen na emailovou komunikaci, ovšem existují i varianty, jak tento protokol využít v rámci signalizačního protokolu SIP. Implementace S/MIME je poměrně složitá a není příliš často využívána, i když umožňuje poměrně kvalitní zabezpečení. Značným problémem je malá podpora ze strany výrobců ústředen i telefonů [62]. Dalším teoreticky popsaným protokolem byl IPsec, který umožňuje zabezpečení na síťové vrstvě, díky tomu neovlivňuje běh SIP aplikací. Z toho ale plyne i několik technologických problémů, mezi které patří zejména neschopnost SIP aplikací detekovat, zda je toto zabezpečení použito. Celková komplikace při integrování služeb také zabraňuje jeho častějšímu použití. Posledním jmenovaným je protokol TLS. Lze říci, že se jedná o jakýsi standard pro zabezpečenou SIP komunikaci. Je podporován každou z testovaných ústředen a je doporučován i od řady IT expertů [7],[62],[6]. Umožňuje šifrování na transportní vrstvě na základě výměny certifikátů. Ke svému přenosu vyžaduje spolehlivý protokol, proto podporuje pouze TCP přenos. Jelikož se jedná již o standard v SIP komunikaci, je podporován jak ze strany výrobců ústředen, tak ze strany výrobců telefonů. Implementace toho protokolu není nijak složitá ve srovnání s předchozími protokoly, například ústředna. Obdobné řešení nabízejí i ústředny YATE a FreeSWITCH. Ovšem pomocí TLS je možné zabezpečit pouze signalizaci, samotná data je potřeba zabezpečit

vlastním protokolem. K tomuto účelu slouží SRTP protokol. Protokol SRTP vznikl v roce 2004 jako odpověď na nezabezpečený protokol RTP.

Praktická část této diplomové práce byla zaměřena na vytvoření generátoru útoků proti PBX ústředně. V rámci generátoru jsou vytvořeny nástroje, které umožňují otestovat jak útoky vycházející z nedokonalosti SIP protokolu, tak útoky, které jsou zaměřeny na vytížení systému ústředny. Jednotlivé útoky byly ověřeny na třech ústřednách FreeSWITCH 1.0, YATE 3.0 a Asterisk verze 1.6 a verze 10. Každá z těchto ústředen má odlišnou architekturu a liší se i způsob zapracování některých SIP zpráv. Testy zaměřené na SIP protokol jsou detailně popsány v kapitole 7.3 včetně zpracovaných výsledků. Zde je možné konstatovat, že ne všechny testy byly úspěšně provedeny. Každá ústředna se zachovala odlišně, jedinou výjimkou je ústředna Asterisk 1.6, která nedokázala vzdorovat ani jednomu z připravených útoků. Zbývající ústředny v některých testech obstály a vytvořené útoky na ně byly neúčinné.

Jako ochranu proti těmto útokům lze použít zabezpečení pomocí LTS a SRTP protokolu, které je podporováno všemi ústřednami. I když tento fakt nebyl v rámci práce prakticky ověřen, existuje dostatek odborné literatury, která popisuje jak implementaci LTS a SRTP, tak poukazuje na tuto dvojici jako na zabezpečení proti většině útoků využívajících protokolu SIP. V rámci praktické části bylo ověřeno, jak se ústředny dokáží vyrovnat s útoky typu DoS záplavového typu. Byly provedeny testy, které zahrnovaly záplavu paketů TCP SYN, UDP, ICMP a také záplavu paketů signalizačními zprávami protokolu SIP.

Záplava paketů UDP měla překvapivě negativní vliv na ústřednu Asterisk 1.6 a YATE 3.0, při testu byl procesor vytížen na maximální hodnotu 100%. V případě ústředny Asterisk nebylo ani možné vytvořit hovor mezi uživateli. Procesor ústředny YATE byl vytížen také na 100% po celou dobu útoku, ale spojení hovoru proběhlo v pořádku. Dále byly provedeny testy záplavou paketů inicializačního protokolu SIP. Dle teoretického předpokladu největší potíže vytvořila záplava paketů INVITE, jediná ústředna, která se dokázala vyrovnat i se záplavou 2000 zpráv za sekundu byla ústředna FreeSWITCH. Procesor byl vytížen na maximální hodnotu po celou dobu útoku, ale i přesto bylo možné vytvořit hovor mezi uživateli. Ústředna Asterisk 1.6 havarovala již při záplavě 700 zpráv za sekundu, kdy ani po uplynutí několika minut nebyla schopna odpovídat na žádný požadavek. Zajímavostí je, že ústředna YATE si nedokázala poradit se záplavou zpráv BYE, na rozdíl od ostatních ústředen, tato zpráva vytížila procesor na 100% a znemožnila komunikaci mezi uživateli.

Útoky, které jsou popisovány v kapitole 7.4, umožnily i při relativně nízké záplavě paketů ústřednu vytížit natolik, aby nemohly být zpracovány legitimní požadavky. V rámci této práce bylo vytvořeno zabezpečení, které využívá síťový nástroj umožňující detekci nežádoucího provozu v síti, jedná se o NIDS Snort. Snort v kombinaci s agentem SnortSam vytváří účinné zabezpečení systému. Byla vytvořena pravidla, na základě kterých Snort dokáže detekovat nežádoucí aktivitu a vytvořit opatření, které dokáže ochránit systém před útoky DoS záplavového typu. Pravidla jsou vytvořena pro všechny typy provedených záplavových útoků DoS, jedinou výjimku představují útoky omezující šířku pásma komunikačního kanálu. Všechna pravidla byla otestována a data jsou zobrazena v grafech, které znázorňují rozdíl ve vytížení procesoru před a po zabezpečení. Již na první pohled je zřejmé, jak byla ochrana účinná. Jako příklad je možno uvést obrázek 7.16, jedná se o případ, kdy ústředna havarovala bez zabezpečení při záplavou 1000 paketů za sekundu. Jakmile bylo implementováno zabezpečení, vytížení procesoru se pohybovalo okolo 2% a spojení hovoru proběhlo zcela bez komplikací. Analogicky na tom byly i ostatní útoky.

Poslední dvě kapitoly praktické části demonstrují využití systémového detekčního nástroje OSSEC, který byl použit pro analýzu výstupních logů sytému Asterisk. OSSEC obsahuje již vytvořená pravidla, která umožňují detekci podezřelé aktivity v systému a na základě definovaných parametrů vytvoří odpovídající akci. V kapitole 7.6 byl popsán příklad, kdy systém OSSEC detekoval skenovaní systému nástrojem Nmap. Na základě konfigurace OSSEC zablokoval IP adresu, která tento sken prováděla a tím bylo znemožněno jeho dokončení, útočník tak nezískal potřebná data o systému. Kapitola 7.7 popisovala příklad, kdy se útočník pokoušel získat uživatelské jméno a heslo, analogickým způsobem byla zablokována IP adresa útočníka, aniž by dokázal zjistit uživatele nebo jeho heslo. Tím byly popsány možnosti zabezpečení systémem OSSEC.

Je zřejmé, že existuje poměrně velké množství útoků, které lze proti PBX ústředně provést ať už se jedná o útoky zaměřené na SIP signalizaci nebo útoky, které mají za cíl vyřadit systém z provozu. Je potřeba si uvědomit, že v open source sféře neexistuje jedno kompletní řešení, které by bylo možné implementovat do jakéhokoliv provozu a tím vyřešit všechny problémy bezpečnosti ve VoIP technologii. Ochrana každého systému se skládá z mnoha samostatných prvků, které je nutno konfigurovat na základě jednotlivých požadavků daného systému. Problematiku bezpečné komunikace ve VoIP systémech řeší protokoly LTS a SRTP, které zajišťují integritu, autenticitu a důvěryhodnost, problémem ovšem je, že v open source sféře stále nejsou brány tyto protokoly jako standartní způsob komunikace. Dalším nebezpečným aspektem pro PBX ústředny jsou útoky typu DoS, které mají v IP telefonii mnohem rychlejší průběh než v případě jiných systémů. Díky moderním technologiím, kdy člověk považuje dostupnost telefonu jako zcela automatickou, se jeví ochrana před těmito útoky jako zvláště důležitá. Zabezpečení pomocí systému Snort v kombinaci se systémem OSSEC je dle názoru autora jedním z nejlepších řešení. Tyto systémy patří mezi nejlepší ve své kategorii, nabízejí rozsáhlé možnosti konfigurace a co je na tom nejlepší, jedná se o open source.

Použitá literatura

[1] *Aircrack-ng* [online]. 2009, 2012 [cit. 2012-05-20]. Dostupné z: http://www.aircrack-ng.org/

[2] B. CID, Daniel. Log Analysis using OSSEC. In: *OSSEC*: *Open Source Host-based Intrusion Detection System* [online]. 2008-2010 [cit. 2012-05-20]. Dostupné z: http://www.ossec.net/ossec-docs/auscert-2007-dcid.pdf

[3] Configuration Files. *YATE* [online]. 2012 [cit. 2012-05-20]. Dostupné z: http://yate.null.ro/pmwiki/index.php?n=Main.ConfigurationFiles

[4] Documentation. In: *FreeSWITCH* [online]. 2012 [cit. 2012-05-20]. Dostupné z: http://wiki.freeswitch.org/wiki/Main_Page

[5] Domain Name System. In: *Wikipedia*: *the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2010, 15. 4. 2012 [cit. 2012-05-20]. Dostupné z: http://cs.wikipedia.org/wiki/Domain_Name_System

[6] DWIVEDI, Himanshu. *Hacking VoIP*: *protocols, attacks, and countermeasures*. San Francisco: No Starch Press, 2009, xiii, 211 s. ISBN 978-1-59327-163-3.

[7] ENDLER, David a Mark D COLLIER. *Hacking exposed VoIP*: *voice over IP security secrets*. New Yrok: McGraw-Hill, 2007, xxvii, 539 s. ISBN 978-0-07-226364-0.

[8] Extension mechanisms for DNS. In: *Wikipedia*: *the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2010, 6.5.2012 [cit. 2012-05-20]. Dostupné z: http://en.wikipedia.org/wiki/Extension_mechanisms_for_DNS

[9] Features Requests. *YATE* [online]. 2012 [cit. 2012-05-20]. Dostupné z: http://yate.null.ro/pmwiki/index.php?n=Main.FeaturesRequests

[10] Fingerprint (computing). In: *Wikipedia*: *the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2010, 1.1.2012 [cit. 2012-05-20]. Dostupné z: http://en.wikipedia.org/wiki/Fingerprint_(computing)

[11] *Guide to Intrusion Detection and Prevention Systems (IDPS)*: *Recommendations of the National Institute of Standards and Technology* [online]. Gaithersburg, 2007, 127 s. [cit. 12.5.2012]. Dostupné z: http://csrc.nist.gov/publications/nistpubs/800-94/SP800-94.pdf

[12] MCCLURE, Stuart, Joel SCAMBRAY a George KURTZ. *Hacking exposed*: *network security secrets and solutions*. 3rd ed. New York: Osborne/McGraw-Hill, 2001, xxviii, 729 p. ISBN 00-721-9381-6.

[13] HAY, Andrew, Daniel CID a Rory BRAY. *OSSEC host-based intrusion detection guide*. Burlington, MA: Syngress Pub., c2008, xxvii, 307 p. ISBN 15-974-9240-X.

[14] History. *FreeSWITCH* [online]. 2012 [cit. 2012-05-20]. Dostupné z: http://en.wikipedia.org/wiki/FreeSWITCH#History

[15] HOŠEK, Jiří a Martin KOUTNÝ. Zabezpečení technologie VoIP pomocí protokolu ZRTP. *Elektrorevue - Internetový časopis*[online]. 2008, roč. 2008, č. 11, s. 11 [cit. 2012-05-20]. ISSN 1213-1539. Dostupné z: http://www.elektrorevue.cz/cz/clanky/komunikacni-technologie/0/

zabezpeceni-technologie-voip-pomoci-protokolu-zrtp/

[16] How does FreeSWITCH compare to Asterisk?. *FreeSWITCH* [online]. 2008 [cit. 2012-05-20]. Dostupné z: http://www.freeswitch.org/node/117

[17] *HTTP Authentication: Basic and Digest Access Authentication*. [s.l.] : Internet Engineering Task Force (IETF), 1999. 34 s. Dostupné z WWW: <http://www.ietf.org/rfc/rfc2617.txt>.

[18] *Přednáška předmětu MTIS*: *Singalizace 2 VoIP signalizace vazba na PSTN signalizace* [online]. Brno: Vysoké učení technické v Brně, 2012, 54 s. [cit. 15-05-2012]. Dostupné z: http://anca.utko.feec.vutbr.cz/vyuka_data/mtis/P6_MTIS.pdf

[19] CHEN, Eric. *A WHITELIST APPROACH TO PROTECT SIP SERVERS FROM FLOODING ATTACKS*. NTT Information Sharing Platform Laboratories, 2010, 22 s. Dostupné z: http://www.ieee-cqr.org/2010/Day%202/Technical%20Session%20-%202/(3)EricChen.pdf

[20] INGATE SYSTEMS. Solving the Firewall/NAT: Traversal Issue of SIP: Who Should Control Your Security Infrastructure? [online]. 2007, 12 s. [cit. 12.05.2012]. Dostupné z: http://www.ingate.com/files/Solving_Firewall-NAT_Traversal.pdf

[21] IPsec. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 2002, last modified on 5.12.2011 [cit. 2011-12-13]. Dostupné z WWW: <http://en.wikipedia.org/wiki/IPsec>.

[22] JACKSON, Ben; LONG , Johnny . *Asterisk Hacking*. Burlington (Massachusetts) : Syngress Publishing, Inc., 2007. 253 s. ISBN 1597491519.

[23] KRČMÁŘ, Petr. Útok Slowloris aneb plíživé nebezpečí pro web servery. In: *Root.cz - informační server* [online]. 1998 – 2011, 2011-05-11 [cit. 2012-05-21]. Dostupné z: http://www.root.cz/clanky/utok-slowloris-aneb-plizive-nebezpeci-pro-web-servery/

[24] KUNTZE, Nicolai a Andreas U. SCHMIDT. TECHNICAL UNIVERSITY DARMSTADT. *SPAM OVER INTERNET TELEPHONY AND HOW TO DEAL WITH IT* [online]. Darmstadt, 2008, 25 s. [cit. 03.05.2012]. Dostupné z: http://icsa.cs.up.ac.za/issa/2008/Proceedings/Full/67.pdf

[25] LEMON, Jonathan. *Resisting SYN flood DoS attacks with a SYN cache* [online]. 2001, 9 s. [cit. 01.02.2012]. Dostupné z: http://people.freebsd.org/~jlemon/papers/syncache.pdf

[26] Netfilter & Snort_Inline. In: *OPENMANIAK*: *THE LEADER IN OPEN SOURCE NETWORK AND SECURITY TUTORIAL*. [online]. 2007 [cit. 2012-05-20]. Dostupné z: http://openmaniak.com/inline_final.ph

[27] *Netfilter* [online]. 1999, 2010 [cit. 2012-05-20]. Dostupné z: http://www.netfilter.org/

[28] *Nmap.org : Nmap Reference Guide* [online]. 2004, 12. 2. 2011 [cit. 2011-12-13]. Dostupné z WWW: <http://nmap.org/book/man.html>

[29] *OSSEC*: *Open Source Host-based Intrusion Detection System* [online]. 2008-2010, 23.10.2011 [cit. 2012-05-20]. Dostupné z: http://www.ossec.net/

[30] PETERKA, Jiří. Protokol IP. *Archiv článků a přednášek Jiřího Peterky* [online]. 1992, roč. 92, č. 48 [cit. 2012-05-20]. Dostupné z: http://www.earchiv.cz/a92/a248c110.php3

[31] Ping of death. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 2005, last modified on 2011 [cit. 2011-12-13]. Dostupné z WWW: <http://en.wikipedia.org/wiki/Ping_of_death>.

[32] Port mirroring. In: *Wikipedia*: *the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2008, 2.5.2012 [cit. 2012-05-20]. Dostupné z: http://en.wikipedia.org/wiki/Port_mirroring

[33] PORTER, Thomas, et al. *Practical VoIP Security*. Rockland (Massachusetts) : Syngress Publishing, Inc., 2006. 592 s. ISBN 1597490601.

[34] Princip zabezpečení protokolem SRTP. *Zabezpečení přenosu multimediálních dat v reálném čase* [online]. 2009 [cit. 2012-05-20]. Dostupné z: http://vosec.wz.cz/index.php?stav=srtp

[35] PROTECT IP Act. In: *Wikipedia*: *the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 20012 [cit. 2012-05-20]. Dostupné z: http://cs.wikipedia.org/wiki/PROTECT_IP_Act

[36] RADVISION. *BACK-TO-BACK USER AGENT (B2BUA) SIP SERVERS*: *POWERING NEXT GENERATION NETWORKS* [online]. 2007, 29 s. [cit. 2012-05-10]. Dostupné z: http://www.radvision.com/NR/rdonlyres/733FD7B3-053E-409B-A385-4820BEA0EDFD/0/B2BUASIPServersWhitePaper.pdf

[37] REHMAN, Rafeeq Ur. *Intrusion detection systems with Snort*: *advanced IDS techniques using Snort, Apache, MySQL, PHP, and ACID*. Upper Saddle River, N.J.: Prentice Hall PTR, c2003, xii, 263 p. ISBN 01-314-0733-3.

[38] RFC 1521. *MIME (Multipurpose Internet Mail Extensions) Part One*: *Mechanisms for Specifying and Describing the Format of Internet Message Bodies*. Internet Engineering Task Force. Dostupné z: http://www.ietf.org/rfc/rfc1521.txt

[39] RFC 2315. *PKCS #7: Cryptographic Message Syntax*: *Version 1.5*. Internet Engineering Task Force, 1998. Dostupné z: http://tools.ietf.org/html/rfc2315

[40] RFC 2396. *Uniform Resource Identifiers (URI)*: *Generic Syntax*. 1998. Dostupné z: http://www.ietf.org/rfc/rfc2396.txt

[41] RFC 2401. *Security Architecture for the Internet Protocol*. Internet Engineering Task Force, 1998. Dostupné z: http://www.ietf.org/rfc/rfc2401.txt

[42] RFC 2409. *The Internet Key Exchange (IKE)*. Internet Engineering Task Force, 1998. Dostupné z: http://www.ietf.org/rfc/rfc2409.txt

[43] RFC 2976. *The SIP INFO Method*. 2000. Dostupné z: http://www.ietf.org/rfc/rfc2976.txt

[44] RFC 3261. *SIP: Session Initiation Protocol*. 2002. Dostupné z: http://www.rfc-editor.org/rfc/rfc3261.txt

[45] RFC 3265. *Ession Initiation Protocol (SIP)-Specific Event Notification*. 2002. Dostupné z: http://www.ietf.org/rfc/rfc3265.txt

[46] RFC 3331. *Signaling System 7 (SS7) Message Transfer Part 2 (MTP2)*: *User Adaptation Layer*. 2002. Dostupné z: http://tools.ietf.org/rfc/rfc3331.txt

[47] RFC 3515. *The Session Initiation Protocol (SIP) Refer Method*. 2003. Dostupné z:

http://www.ietf.org/rfc/rfc3515.txt

[48] RFC 3550. *RTP: A Transport Protocol for Real-Time Applications*. Internet Engineering Task Force. Dostupné z: http://www.ietf.org/rfc/rfc3550.txt

[49] RFC 3711. *The Secure Real-time Transport Protocol (SRTP)*. 2004. Dostupné z: http://www.ietf.org/rfc/rfc3711.txt

[50] RFC 3851. *Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1*: *Message Specification*. 2004. Dostupné z: http://www.ietf.org/rfc/rfc3851.txt

[51] RFC 4301. *Security Architecture for the Internet Protocol*. Internet Engineering Task Force, 2005. Dostupné z: http://tools.ietf.org/html/rfc4301

[52] RFC 4306. *Internet Key Exchange (IKEv2) Protocol*. Internet Engineering Task Force, 2005. Dostupné z: https://tools.ietf.org/html/rfc4306

[53] RFC 5246. *The Transport Layer Security (TLS) Protocol*: *Version 1.2*. 2008. Dostupné z: http://www.rfc-editor.org/rfc/rfc5246.txt

[54] RFC 6189. *ZRTP: Media Path Key Agreement for Unicast Secure RTP*. 2011. vyd. Dostupné z: www.rfc-editor.org/rfc/rfc6189.txt

[55] Rootkit. In: *Wikipedia*: *the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 11.4.2012 [cit. 2012-05-20]. Dostupné z: http://cs.wikipedia.org/wiki/Rootkit

[56] RUSSELL, Bryant. Asterisk Versions. *Asterisk* [online]. 2012 [cit. 2012-05-20]. Dostupné z: https://wiki.asterisk.org/wiki/display/AST/Asterisk+Versions

[57] SANS INSTITUTE. *DNS Spoofing by The Man In The Middle*: *GSEC Practical Assignment* [online]. version 1.4c. 2005, 35 s. [cit. 01.05.2012]. Dostupné z: http://www.sans.org/reading_room/whitepapers/dns/dns-spoofing-man-middle_1567

[58] SIP TLS. *FreeSWITCH* [online]. 2012 [cit. 2012-05-20]. Dostupné z: http://wiki.freeswitch.org/wiki/Tls

[59] SIP. CESNET. *IpTelWiki*: *SIP* [online]. 2006, 20.03.2012 [cit. 2012-05-02]. Dostupné z: https://sip.cesnet.cz/cs/protokoly/sip

[60] *SIPp* [online]. 10/25/2010 [cit. 2011-12-13]. Dostupné z WWW: <http://sipp.sourceforge.net/>.

[61] *SIPVicious* [online]. 2010 [cit. 2011-12-13]. Dostupné z WWW: <http://code.google.com/p/sipvicious/>.

[62] SISALEM, Dorgham. *SIP security*. Chichester, U.K.: Wiley, 2009, xiv, 336 p. ISBN 04-705-1636-4.

[63] Slowloris. In: *Wikipedia*: *the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 2009-05-17 [cit. 2012-05-21]. Dostupné z: http://en.wikipedia.org/wiki/Slowloris

[64] Snort Users Manual: 2.9.2. Snort [online]. 2011 [cit. 2012-05-20]. Dostupné z: http://manual.snort.org/

[65] *SnortSam* [online]. 2007 [cit. 2012-05-20]. Dostupné z: http://www.snortsam.net/

[66] SONICWALL, Inc. *Denial of Service Attacks*: *An Emerging Vulnerability for the "Connected" Networ [online]*. 1999 [cit. 2012-05-20].. Dostupné z:

ftp://ftp.sonicwall.com/pub/info/Denial_Of_Service_Attacks.pdf

[67] STALLINGS, William. CISCO. *The Session Initiation Protocol* [online]. 2003 [cit. 2012-04-20]. Dostupné z: http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_6-1/sip.html

[68] Stop Online Piracy Act. In: *Wikipedia*: *the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 20012 [cit. 2012-05-20]. Dostupné z: http://cs.wikipedia.org/wiki/Stop_Online_Piracy_Act

[69] Syslog. In: *Wikipedia*: *the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 5.5.2012 [cit. 2012-05-20]. Dostupné z: http://en.wikipedia.org/wiki/Syslog

[70] THERNELIUS, Fredrik. *SIP, NAT, and Firewalls* [online]. New York, 2000 [cit. 2012-05-20]. Dostupné z: http://www.cs.columbia.edu/sip/drafts/Ther0005_SIP.pdf. Master's Thesis. Columbia University.

[71] Top 10 Open Source PBX Software. *VoIPToday* [online]. 2010 [cit. 2012-05-02]. Dostupné z: http://voiptoday.org/index.php?option=com_content&view=article&id=414&Itemid=101

[72] Transmission Control Protocol. In: *Wikipedia*: *the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 13.05.2012 [cit. 2012-05-20]. Dostupné z: http://en.wikipedia.org/wiki/Transmission_Control_Protocol

[73] VANĚK, Tomáš. BEZPEČNOST V OBLASTI VOIP : SIPS. *Teorie a praxe IP telefonie*. 2008, 3, s. 133-134. Dostupný také z WWW: <http://www.ip-telefon.cz/archiv/dok_osta/ipt-2008_Bezpecnost_VoIP.pdf>.

[74] VAUGHN, Randal a Evron GADI. *DNS Amplification Attacks* [online]. 2006, 19 s. [cit. 2012-05-20]. Dostupné z: http://www.isotf.org/news/DNS-Amplification-Attacks.pdf

[75] VOZŇÁK, Miroslav. TELEFONNÍ ÚSTŘEDNY ASTERISK. In: *Teorie a praxe IP telefonie* [online]. 2008 [cit. 2012-04-20]. Dostupné z: http://www.ip-telefon.cz/archiv/dok_osta/ipt-2008_Telefonni_ustredny_Asterisk.pdf

[76] Vyrovnávací paměť. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2011, 29. 2. 2012 [cit. 2012-05-20]. Dostupné z: http://cs.wikipedia.org/wiki/Vyrovnávací_paměť

[77] What's Yate?. *YATE* [online]. 2012 [cit. 2012-05-20]. Dostupné z: http://yate.null.ro/pmwiki/index.php?n=Main.WhatsYate?

[78] Yate 4. *YATE* [online]. 2012 [cit. 2012-05-20]. Dostupné z: http://yate.null.ro/pmwiki/index.php?n=Main.Yate4

[79] Yate Architecture. *YATE* [online]. 2012 [cit. 2012-05-20]. Dostupné z: http://yate.null.ro/pmwiki/index.php?n=Main.Architecture

[80] YATE: Yet Another Telephony Engine. *VoIP-Info.org*: *A reference guide to all things VoIP* [online]. 2012, 23.04.2012 [cit. 2012-04-20]. Dostupné z: http://www.voip-info.org/wiki/view/YATE

[81] Yet Another Telephony Engine. *YATE* [online]. 2012 [cit. 2012-05-20]. Dostupné z: http://yate.null.ro/pmwiki/

Seznam použitých zkratek

ACD	-	Automatic Call Distribution.
AKA	-	Authentication and Key Agreement
ARP	-	Address Resolution Protocol
B2BUA	-	Back-to-back user agent
CAM	-	Content addressable memory
DDoS	-	Distributed Denial of Service
DNS	-	Domain Name System
DoS	-	Denila of Service
EDNS	-	Extension mechanisms for DNS
HIDS	-	Host-based Intrusion Detection System
IKE	-	Internet Key Exchange
IM	-	Instant Messaging
IMS	-	Internet Multimedia Subsystem
IPS	-	Intrusion Prevention System
IPsec	-	IP security
ISDN	-	Integrated Services Digital Network
IVR	-	Interactive Voice Response
LTS	-	Long Term Support
MAC	-	Media Access Control
MKI	-	Master Key Identifier
MPL	-	Mozilla Public License
NIDS	-	Network Intrusion Detection System
P2P2P	-	Peer to Peer to Peer
PBX	-	Private Branch Exchange
PCM	-	Pulse-Code Modulation
PKI	-	Public Key Infrastructure
QoS	-	Quality of Service.
RTP	-	Real-time Transport Protocol
SIP	-	Session Initiation Protocol
SIPS	-	Session Initiation Protocol Security
SPIT	-	SPAM over IP Telephony
SRTP	-	Secure Real-time Transport Protocol
TCP	-	Transmission Control Protocol
TDM	-	Time Division Multiplex
TLS	-	Transport Layer Security
UA	-	User Agent
UAC	-	User Agent Client
UAS	-	User Agent Server
UDP	-	User Datagram Protocol
UICC	-	Universal Integrated Circuit Card
URI	-	Uniform Resource Identifier
VoIP	-	Voice over IP

Seznam příloh

A. Výstupní logy ze skriptu enumeration.sh

A.1 Výsledný sken - ústředna Freeswitch

```
Nmap scan report for 192.168.201.150
Host is up (0.00032s latency).
Not shown: 1983 closed ports
PORT      STATE         SERVICE      VERSION
22/tcp    open          ssh          OpenSSH 5.3p1 Debian 3ubuntu7
(protocol 2.0)
| ssh-hostkey: 1024 ca:ba:92:0b:af:d9:b0:dc:be:79:8f:58:51:d1:03:63 (DSA)
|_2048 0a:9b:8e:c9:a3:ed:c3:a5:87:f0:98:fd:1a:6f:1d:1b (RSA)
80/tcp    open          http         nginx 1.0.9
| http-title: 301 Moved Permanently
|_Did not follow redirect to https://192.168.201.150/
|_http-methods: No Allow or Public header in OPTIONS response (status code
301)
443/tcp   open          http         nginx 1.0.9
|_http-methods: No Allow or Public header in OPTIONS response (status code
400)
|_http-title: 400 The plain HTTP request was sent to HTTPS port
5060/tcp  open          sip          (SIP end point; Status: 200 OK)
68/udp    open|filtered dhcpc
123/udp   open          ntp          NTP v4 (unsynchronized)
| ntp-info:
|_  receive time stamp: Wed May  2 08:27:27 2012
1718/udp  open|filtered h225gatedisc
3457/udp  open|filtered vat-control
5060/udp  open          sip          (SIP end point; Status: 200 OK)
18669/udp open|filtered unknown
21247/udp open|filtered unknown
21710/udp open|filtered unknown
25240/udp open|filtered unknown
31189/udp open|filtered unknown
43686/udp open|filtered unknown
49184/udp open|filtered unknown
49185/udp open|filtered unknown
2 services unrecognized despite returning data. If you know the
service/version, please submit the following fi$
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port5060-TCP:V=5.51%I=7%D=5/2%Time=4FA0D38A%P=i686-redhat-linux-gnu%r(S
SF:IPOptions,282,"SIP/2\.0\x20200\x20OK\r\nVia:\x20SIP/2\.0/UDP\x20nm;bran
SF:ch=foo;rport=33511;received=192\.168\.201\.147\r\nFrom:\x20<sip:nm@nm>;
SF:tag=root\r\nTo:\x20<sip:nm2@nm2>;tag=ByjXtF696tF0F\r\nCall-ID:\x2050000
SF:\r\nCSeq:\x2042\x20OPTIONS\r\nContact:\x20<sip:192\.168\.201\.150>\r\nU
SF:ser-Agent:\x20FreeSWITCH-mod_sofia/1\.0\.head-git-4936b11\x202011-11-10
SF:\x2022-59-43\x20-0500\r\nAccept:\x20application/sdp\r\nAllow:\x20INVITE
SF:,\x20ACK,\x20BYE,\x20CANCEL,\x20OPTIONS,\x20MESSAGE,\x20UPDATE,\x20INFO
SF:,\x20REGISTER,\x20REFER,\x20NOTIFY,\x20PUBLISH,\x20SUBSCRIBE\r\nSupport
SF:ed:\x20timer,\x20precondition,\x20path,\x20replaces\r\nAllow-Events:\x2
SF:0talk,\x20hold,\x20presence,\x20dialog,\x20line-seize,\x20call-info,\x2
```

```
SF:0sla,\x20include-session-description,\x20presence\.winfo,\x20message-su
SF:mmary,\x20refer\r\nContent-Length:\x200\r\n\r\n");
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port5060-UDP:V=5.51%I=7%D=5/2%Time=4FA0D37F%P=i686-redhat-linux-gnu%r(S
SF:IPOptions,282,"SIP/2\.0\x20200\x20OK\r\nVia:\x20SIP/2\.0/UDP\x20nm;bran
SF:ch=foo;rport=33511;received=192\.168\.201\.147\r\nFrom:\x20<sip:nm@nm>;
SF:tag=root\r\nTo:\x20<sip:nm2@nm2>;tag=ByjXtF696tF0F\r\nCall-ID:\x2050000
SF:\r\nCSeq:\x2042\x20OPTIONS\r\nContact:\x20<sip:192\.168\.201\.150>\r\nU
SF:ser-Agent:\x20FreeSWITCH-mod_sofia/1\.0\.head-git-4936b11\x202011-11-10
SF:\x2022-59-43\x20-0500\r\nAccept:\x20application/sdp\r\nAllow:\x20INVITE
SF:,\x20ACK,\x20BYE,\x20CANCEL,\x20OPTIONS,\x20MESSAGE,\x20UPDATE,\x20INFO
SF:,\x20REGISTER,\x20REFER,\x20NOTIFY,\x20PUBLISH,\x20SUBSCRIBE\r\nSupport
SF:ed:\x20timer,\x20precondition,\x20path,\x20replaces\r\nAllow-Events:\x2
SF:0talk,\x20hold,\x20presence,\x20dialog,\x20line-seize,\x20call-info,\x2
SF:0sla,\x20include-session-description,\x20presence\.winfo,\x20message-su
SF:mmary,\x20refer\r\nContent-Length:\x200\r\n\r\n");
MAC Address: 00:0C:29:56:83:DA (VMware)
Device type: general purpose
Running: Linux 2.6.X
OS details: Linux 2.6.17 - 2.6.35
Uptime guess: 0.011 days (since Wed May  2 08:12:09 2012)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=205 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Linux
TRACEROUTE
HOP RTT     ADDRESS
1   0.32 ms 192.168.201.150

Read data files from: /usr/share/nmap
OS and Service detection performed. Please report any incorrect results at
http://nmap.org/submit/ .
# Nmap done at Wed May  2 08:27:24 2012 -- 1 IP address (1 host up) scanned
in 1070.46 seconds
Skenovani cile dokonceno.

Detekce VoIP zarizeni...

Vystup SipVicious:
| SIP Device          | User Agent                                                                          | Fingerprint |
-------------------------------------------------------------------------------------------------------------
| 192.168.201.150:5060 | FreeSWITCH-mod_sofia/1.0.head-git-4936b11 2011-11-10 | disabled    |
|                      | 22-59-43 -0500                                                                      |             |

Detekce VoiP zarizeni dokoncena.

Skenovani bylo dokonceno, vystupni informace jsou zaznamenany v
/voip/output/enumeration1335938960.log
```

A.2 Výsledný sken - ústředna Asterisk 1.6.0

```
Nmap scan report for 192.168.201.148
Host is up (0.00061s latency).
Not shown: 1975 closed ports
PORT      STATE         SERVICE       VERSION
21/tcp    open          ftp           vsftpd 2.0.5
22/tcp    open          ssh           OpenSSH 4.3 (protocol 2.0)
| ssh-hostkey: 1024 f6:9e:27:59:81:82:d4:ad:72:82:07:fd:af:4c:3b:2c (DSA)
|_2048 0e:08:28:e8:b4:e4:ea:f7:3b:60:e4:6f:9b:38:c7:7d (RSA)
80/tcp    open          http          Apache httpd 2.2.3 ((CentOS))
| http-robots.txt: 1 disallowed entry
|_/
|_http-methods: No Allow or Public header in OPTIONS response (status code
302)
|_http-favicon: Unknown favicon MD5: 1F46D824D63718B6346C4063F8204EDF
| http-title: trixbox - User Mode
|_Requested resource was http://192.168.201.148/user/
111/tcp   open          rpcbind       2 (rpc #100000)
443/tcp   open          ssl/http      Apache httpd 2.2.3 ((CentOS))
| http-robots.txt: 1 disallowed entry
|_/
|_http-methods: No Allow or Public header in OPTIONS response (status code
302)
|_http-favicon: Unknown favicon MD5: 1F46D824D63718B6346C4063F8204EDF
| http-title: trixbox - User Mode
|_Requested resource was https://192.168.201.148:443/user/
1720/tcp  open          H.323/Q.931?
2000/tcp  open          cisco-sccp?
3306/tcp  open          mysql         MySQL (unauthorized)
4445/tcp  open          upnotifyp?
68/udp    open|filtered dhcpc
69/udp    open|filtered tftp
111/udp   open          rpcbind       2 (rpc #100000)
123/udp   open          ntp           NTP v4
| ntp-info:
|   receive time stamp: Wed May  2 19:53:35 2012
|   version: ntpd 4.2.2p1@1.1570-o Fri Nov 18 13:21:16 UTC 2011 (1)
|   processor: i686
|   system: Linux/2.6.18-164.11.1.el5
|   leap: 0
|   stratum: 3
|   precision: -20
|   rootdelay: 11.976
|   rootdispersion: 22.663
|   peer: 10514
|   refid: 195.113.159.1
|   reftime: 0xd34bf2f9.a2d11754
|   poll: 6
|   clock: 0xd34bf31d.32ec104f
|   state: 4
```

```
|   offset: -0.250
|   frequency: 28.619
|   jitter: 0.845
|   noise: 0.192
|   stability: 0.050
|_  tai: 0
764/udp   open|filtered omserv
1056/udp  open|filtered vfo
3401/udp  open|filtered squid-snmp
5000/udp  open|filtered upnp
5001/udp  open|filtered commplex-link
5060/udp  open          sip           (SIP end point; Status: 200 OK)
18250/udp open|filtered unknown
18258/udp open|filtered unknown
20424/udp open|filtered unknown
22986/udp open|filtered unknown
33872/udp open|filtered unknown
49182/udp open|filtered unknown
1 service unrecognized despite returning data. If you know the
service/version, please submit the following fingerprint at
http://www.insecure.org/cgi-bin/servicefp-submit.cgi :
SF-Port5060-UDP:V=5.51%I=7%D=5/2%Time=4FA1742B%P=i686-redhat-linux-gnu%r(S
SF:IPOptions,198,"SIP/2\.0\x20200\x20OK\r\nVia:\x20SIP/2\.0/UDP\x20nm;bran
SF:ch=foo;received=192\.168\.201\.147;rport=39207\r\nFrom:\x20<sip:nm@nm>;
SF:tag=root\r\nTo:\x20<sip:nm2@nm2>;tag=as70444eb2\r\nCall-ID:\x2050000\r\
SF:nCSeq:\x2042\x20OPTIONS\r\nUser-Agent:\x20Asterisk\x20PBX\x201\.6\.0\.2
SF:6-FONCORE-r78\r\nAllow:\x20INVITE,\x20ACK,\x20CANCEL,\x20OPTIONS,\x20BY
SF:E,\x20REFER,\x20SUBSCRIBE,\x20NOTIFY,\x20INFO\r\nSupported:\x20replaces
SF:,\x20timer\r\nContact:\x20<sip:192\.168\.201\.148>\r\nAccept:\x20applic
SF:ation/sdp\r\nContent-Length:\x200\r\n\r\n");
MAC Address: 00:0C:29:92:8F:E7 (VMware)
Device type: general purpose
Running: Linux 2.6.X
OS details: Linux 2.6.9 - 2.6.30
Uptime guess: 0.010 days (since Wed May  2 19:38:41 2012)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=206 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Unix

TRACEROUTE
HOP RTT     ADDRESS
1   0.61 ms 192.168.201.148

Read data files from: /usr/share/nmap
OS and Service detection performed. Please report any incorrect results at
http://nmap.org/submit/ .
# Nmap done at Wed May  2 19:53:35 2012 -- 1 IP address (1 host up) scanned
in 1111.78 seconds
Skenovani cile dokonceno.
```

```
Detekce VoIP zarizeni...


Vystup SipVicious:
| SIP Device           | User Agent                        | Fingerprint |
-------------------------------------------------------------------------
| 192.168.201.148:5060 | Asterisk PBX 1.6.0.26-FONCORE-r78 | disabled    |
```

A.3 Výsledný sken - ústředna YATE

```
Nmap scan report for 192.168.201.155
Host is up (0.00040s latency).
Not shown: 1979 closed ports
PORT      STATE         SERVICE      VERSION
22/tcp    open          ssh          OpenSSH 5.3 (protocol 2.0)
| ssh-hostkey: 1024 cc:80:0d:85:ae:67:c2:ae:9b:a3:7a:a6:be:6b:b1:d4 (DSA)
|_2048 ec:90:56:e3:23:67:65:34:d7:49:1a:ea:2f:8b:ca:2e (RSA)
80/tcp    open          http         Apache httpd 2.2.14 ((Mandriva
Linux/PREFORK-1.2mdv2010.0))
| http-robots.txt: 8 disallowed entries
| /manual/ /manual-2.2/ /addon-modules/ /doc/ /images/
|_/all_our_e-mail_addresses /admin/ /
| http-title: FreeSentral
|_Did not follow redirect to https://192.168.201.155/index.php
|_http-methods: No Allow or Public header in OPTIONS response (status code
302)
|_http-favicon: Unknown favicon MD5: E5F95709E3C0E25ECAB61B87002D837C
111/tcp   open          rpcbind      2-4 (rpc #100000)
443/tcp   open          ssl/http     Apache httpd 2.2.14 ((Mandriva
Linux/PREFORK-1.2mdv2010.0))
| http-robots.txt: 8 disallowed entries
| /manual/ /manual-2.2/ /addon-modules/ /doc/ /images/
|_/all_our_e-mail_addresses /admin/ /
|_http-title: FreeSentral
|_http-methods: No Allow or Public header in OPTIONS response (status code
200)
|_http-favicon: Unknown favicon MD5: E5F95709E3C0E25ECAB61B87002D837C
1720/tcp  open          H.323/Q.931?
5555/tcp  open          freeciv?
68/udp    open|filtered dhcpc
111/udp   open          rpcbind      2-4 (rpc #100000)
161/udp   open          snmp         SNMPv3 server
|_snmp-win32-shares: TIMEOUT
1022/udp  open          rpcbind      2-4 (rpc #100000)
1060/udp  open|filtered polestar
5060/udp  open          sip          YATE/3.0.0 (Status: 100 Trying)
8000/udp  open|filtered irdmi
17605/udp open|filtered unknown
17638/udp open|filtered unknown
18987/udp open|filtered unknown
19682/udp open|filtered unknown
31059/udp open|filtered unknown
43094/udp open|filtered unknown
48189/udp open|filtered unknown
58640/udp open|filtered unknown
2 services unrecognized despite returning data. If you know the
service/version, please submit the following fingerprints at
http://www.insecure.org/cgi-bin/servicefp-submit.cgi :
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
```

```
SF-Port161-UDP:V=5.51%I=7%D=4/26%Time=4F990CB8%P=i686-redhat-linux-gnu%r(S
SF:NMPv3GetRequest,58,"0V\x02\x01\x030\x0f\x02\x02Ji\x02\x03\0\xff\xe3\x04
SF:\x01\0\x02\x01\x03\x04\x160\x14\x04\x05\x80\0\x86\xc5\x04\x02\x01\x02\x
SF:02\x02\x0b\x93\x04\0\x04\0\x04\x000\(\x04\x05\x80\0\x86\xc5\x04\x04\0\x
SF:a8\x1d\x02\x027\xf0\x02\x01\0\x02\x01\x000\x110\x0f\x06\n\+\x06\x01\x06
SF:\x03\x0f\x01\x01\x04\0A\x01\x02");
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port5060-UDP:V=5.51%I=7%D=4/26%Time=4F990CB5%P=i686-redhat-linux-gnu%r(
SF:SIPOptions,1F2,"SIP/2\.0\x20100\x20Trying\r\nVia:\x20SIP/2\.0/UDP\x20nm
SF:;branch=foo;rport=35462;received=192\.168\.201\.147\r\nFrom:\x20<sip:nm
SF:@nm>;tag=root\r\nTo:\x20<sip:nm2@nm2>\r\nCall-ID:\x2050000\r\nCSeq:\x20
SF:42\x20OPTIONS\r\nServer:\x20YATE/3\.0\.0\r\nContent-Length:\x200\r\n\r\
SF:nSIP/2\.0\x20200\x20OK\r\nVia:\x20SIP/2\.0/UDP\x20nm;branch=foo;rport=3
SF:5462;received=192\.168\.201\.147\r\nFrom:\x20<sip:nm@nm>;tag=root\r\nTo
SF::\x20<sip:nm2@nm2>;tag=1246110083\r\nCall-ID:\x2050000\r\nCSeq:\x2042\x
SF:20OPTIONS\r\nServer:\x20YATE/3\.0\.0\r\nAllow:\x20ACK,\x20INVITE,\x20BY
SF:E,\x20CANCEL,\x20REGISTER,\x20REFER,\x20OPTIONS,\x20INFO\r\nContent-Len
SF:gth:\x200\r\n\r\n");
MAC Address: 00:0C:29:80:AB:F3 (VMware)
Device type: general purpose
Running: Linux 2.6.X
OS details: Linux 2.6.9 - 2.6.30
Uptime guess: 0.032 days (since Thu Apr 26 10:07:10 2012)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=198 (Good luck!)
IP ID Sequence Generation: All zeros

TRACEROUTE
HOP RTT     ADDRESS
1   0.40 ms 192.168.201.155

Read data files from: /usr/share/nmap
OS and Service detection performed. Please report any incorrect results at
http://nmap.org/submit/ .
# Nmap done at Thu Apr 26 10:53:22 2012 -- 1 IP address (1 host up) scanned
in 1060.48 seconds
Skenovani cile dokonceno.

Detekce VoIP zarizeni...


Vystup SipVicious:
| SIP Device           | User Agent | Fingerprint |
-------------------------------------------------
| 192.168.201.155:5060 | YATE/3.0.0 | disabled    |
```

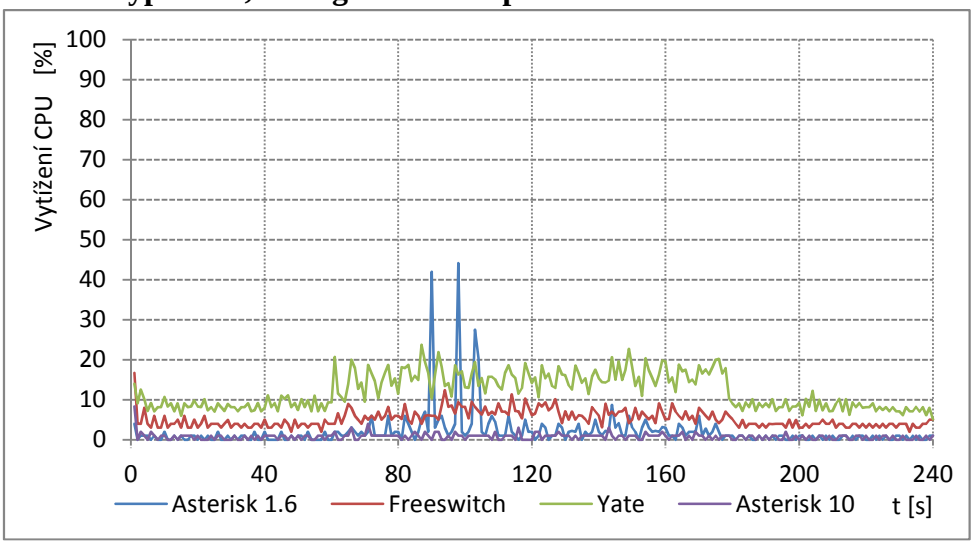
A.4 Výsledný sken - ústředna Asterisk 10

```
Nmap scan report for 192.168.201.158
Host is up (0.00034s latency).
Not shown: 1983 closed ports
PORT      STATE         SERVICE         VERSION
22/tcp    open          ssh             OpenSSH 5.3p1 Debian 3ubuntu7
(protocol 2.0)
| ssh-hostkey: 1024 20:a0:00:d1:5a:df:40:66:e3:9e:15:fa:03:b9:70:ca (DSA)
|_2048 42:58:eb:a8:a4:96:e3:aa:2a:6f:51:fd:88:0f:1f:d2 (RSA)
80/tcp    open          http            Apache httpd 2.2.14 ((Ubuntu))
|_http-methods: GET HEAD POST OPTIONS
|_http-title: Site doesn't have a title (text/html).
2000/tcp  open          cisco-sccp?
68/udp    open|filtered dhcpc
389/udp   open|filtered ldap
1214/udp  open|filtered fasttrack
3283/udp  open|filtered netassistant
5000/udp  open|filtered upnp
5060/udp  open          sip             Asterisk PBX 10.0.0 (Status: 404
Not Found)
18958/udp open|filtered unknown
19695/udp open|filtered unknown
19933/udp open|filtered unknown
20126/udp open|filtered unknown
32774/udp open|filtered sometimes-rpc12
38412/udp open|filtered unknown
49152/udp open|filtered unknown
49200/udp open|filtered unknown
1 service unrecognized despite returning data. If you know the
service/version, please submit the following fingerprint at
http://www.insecure.org/cgi-bin/servicefp-submit.cgi :
SF-Port5060-UDP:V=5.51%I=7%D=5/3%Time=4FA2D877%P=i686-redhat-linux-gnu%r(S
SF:IPOptions,176,"SIP/2\.0\x20404\x20Not\x20Found\r\nVia:\x20SIP/2\.0/UDP\
SF:x20nm;branch=foo;received=192\.168\.201\.147;rport=49377\r\nFrom:\x20<s
SF:ip:nm@nm>;tag=root\r\nTo:\x20<sip:nm2@nm2>;tag=as76656a18\r\nCall-ID:\x
SF:2050000\r\nCSeq:\x2042\x20OPTIONS\r\nServer:\x20Asterisk\x20PBX\x2010\.
SF:0\.0\r\nAllow:\x20INVITE,\x20ACK,\x20CANCEL,\x20OPTIONS,\x20BYE,\x20REF
SF:ER,\x20SUBSCRIBE,\x20NOTIFY,\x20INFO,\x20PUBLISH\r\nSupported:\x20repla
SF:ces,\x20timer\r\nAccept:\x20application/sdp\r\nContent-Length:\x200\r\n
SF:\r\n");
MAC Address: 00:0C:29:F2:CB:7B (VMware)
No exact OS matches for host (If you know what OS is running on it, see
http://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=5.51%D=5/3%OT=22%CT=1%CU=2%PV=Y%DS=1%DC=D%G=Y%M=000C29%TM=4FA2D8C
OS:A%P=i686-redhat-linux-gnu)SEQ(SP=107%GCD=1%ISR=10D%TI=Z%CI=Z%II=I%TS=8)O
OS:PS(O1=M5B4ST11NW6%O2=M5B4ST11NW6%O3=M5B4NNT11NW6%O4=M5B4ST11NW6%O5=M5B4S
OS:T11NW6%O6=M5B4ST11)WIN(W1=16A0%W2=16A0%W3=16A0%W4=16A0%W5=16A0%W6=16A0)E
OS:CN(R=Y%DF=Y%T=40%W=16D0%O=M5B4NNSNW6%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+%F
OS:=AS%RD=0%Q=)T2(R=N)T3(R=Y%DF=Y%T=40%W=16A0%S=O%A=S+%F=AS%O=M5B4ST11NW6%R
```

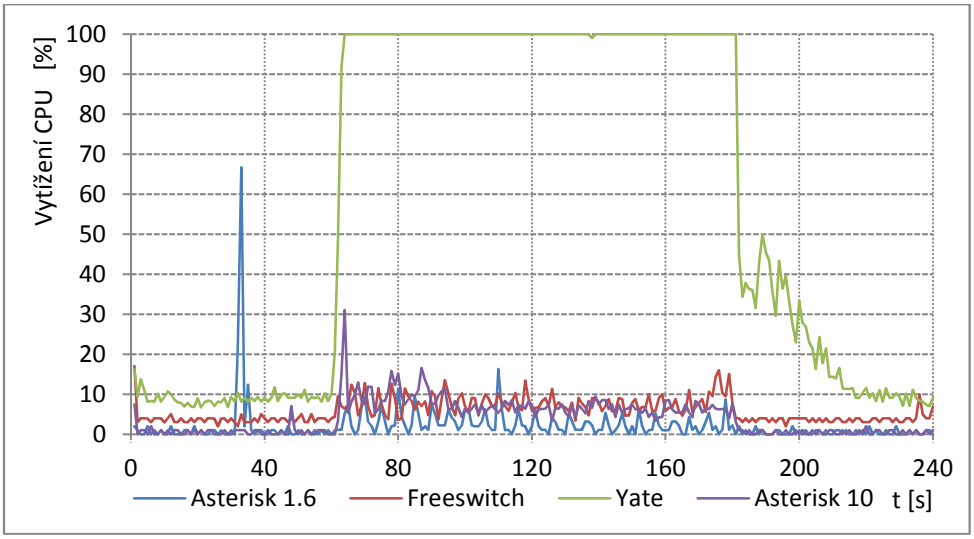
```
OS:D=0%Q=)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R=Y%DF=Y%T=40%W=0%
OS:S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T7(
OS:R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%T=40%IPL=164%UN=0
OS:%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD=S)

Uptime guess: 0.057 days (since Thu May  3 19:51:42 2012)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=263 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Linux

TRACEROUTE
HOP RTT     ADDRESS
1   0.34 ms 192.168.201.158

Read data files from: /usr/share/nmap
OS and Service detection performed. Please report any incorrect results at
http://nmap.org/submit/ .
# Nmap done at Thu May  3 21:13:14 2012 -- 1 IP address (1 host up) scanned
in 1070.91 seconds
Skenovani cile dokonceno.

Detekce VoIP zarizeni...


Vystup SipVicious:
| SIP Device           | User Agent          | Fingerprint |
-----------------------------------------------------------
| 192.168.201.158:5060 | Asterisk PBX 10.0.0 | disabled    |
```

B. Výsledné grafy pro útok typu DoS pomocí dos_sip.sh

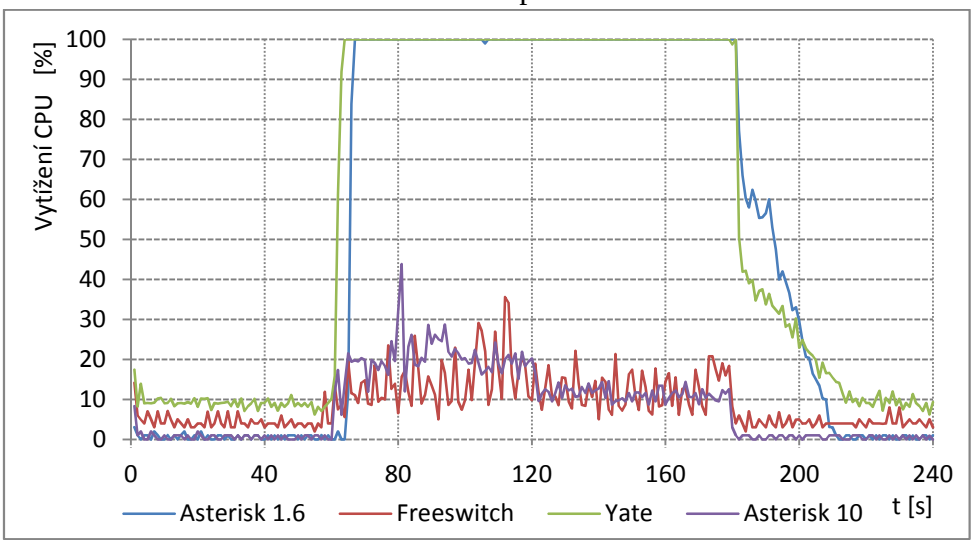
B.1 Útok typu DoS, konfigurace 500 zpráv/s



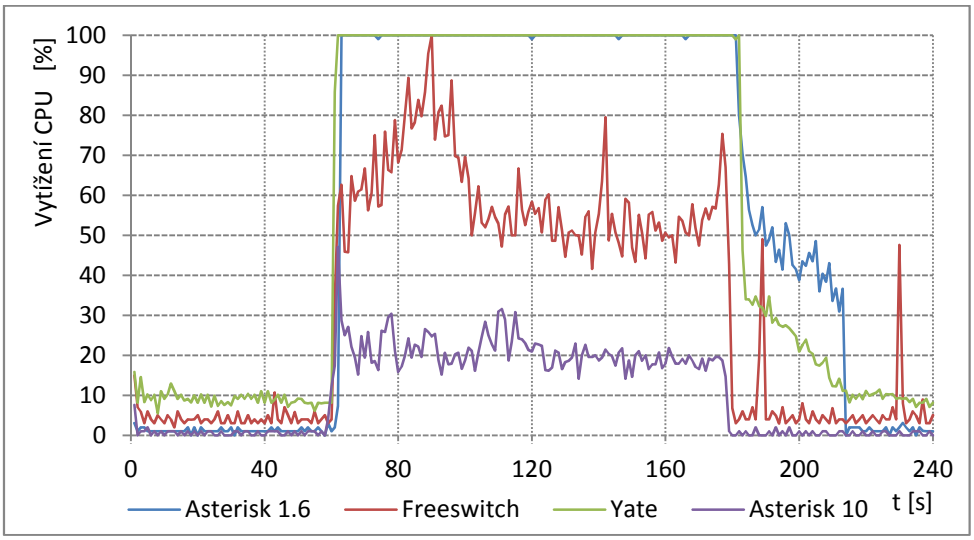
ACK 500 zpráv/s



BYE 500 zpráv/s



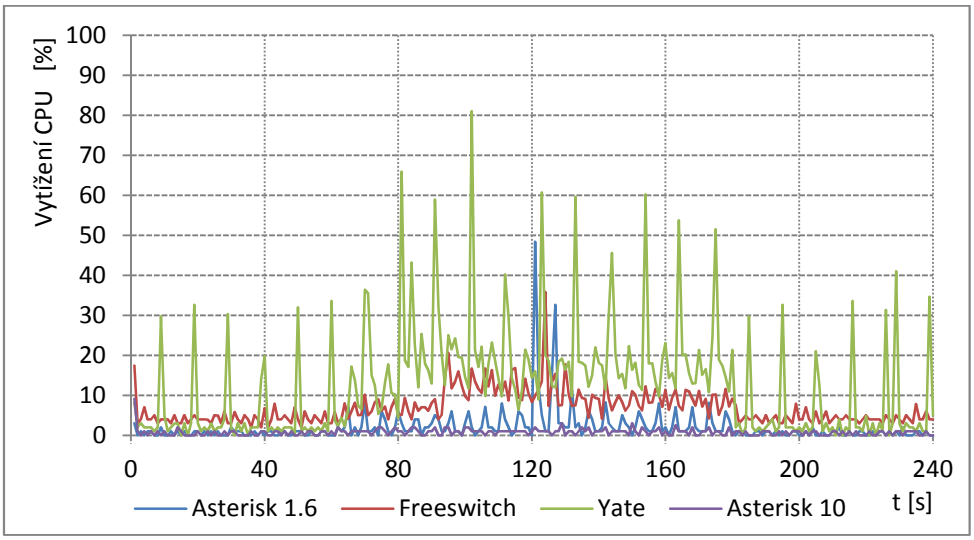
OPTIONS 500 zpráv/s



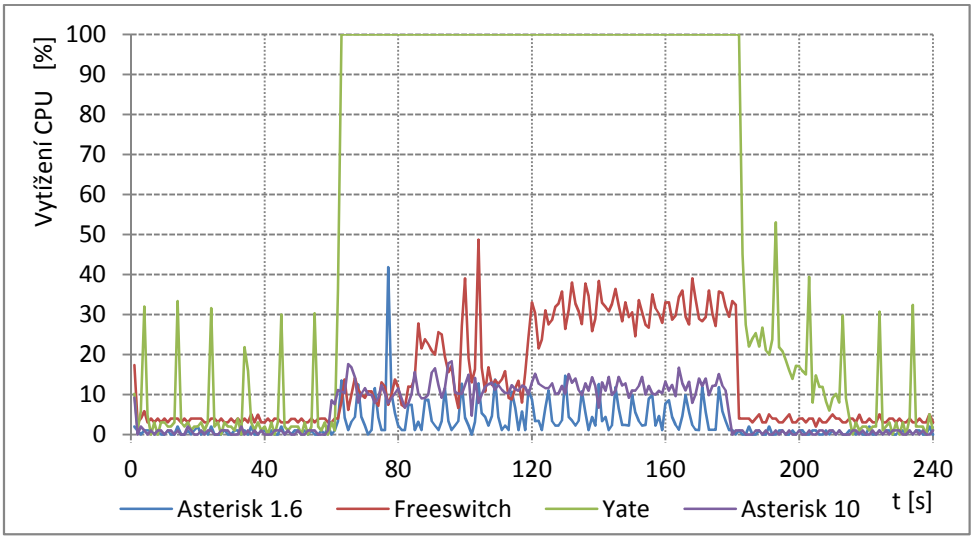
REGISTER 500 zpráv/s

INVITE 500 zpráv/s

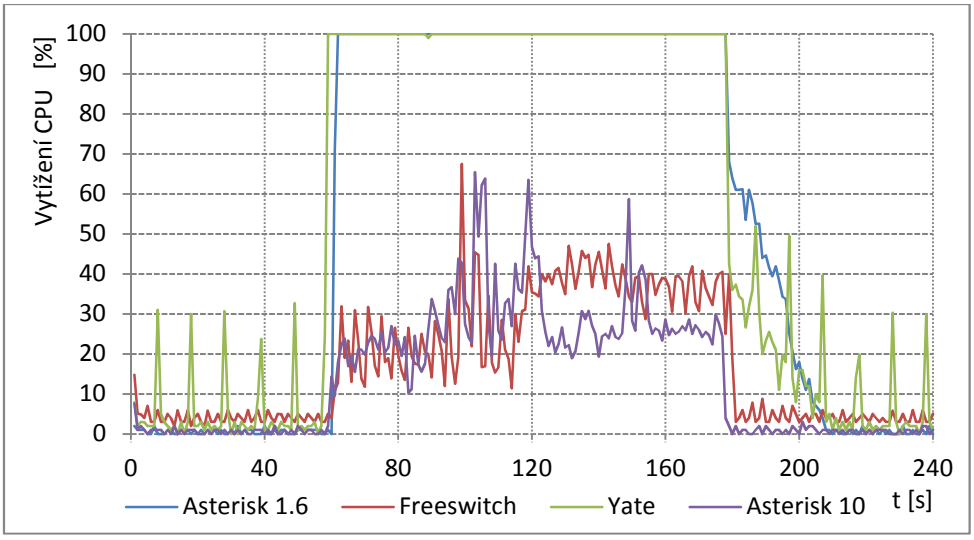
B.2 Útok typu DoS, konfigurace 1000 zpráv/s



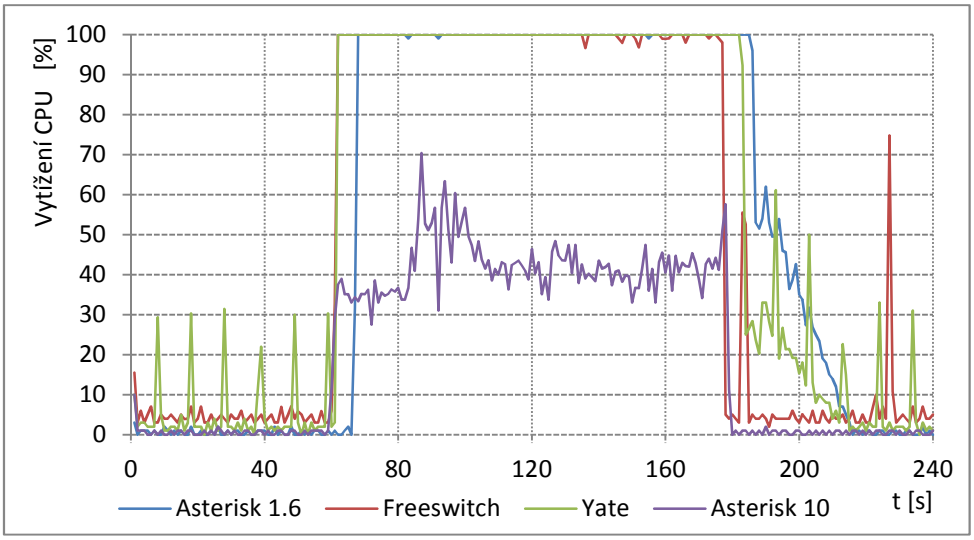
ACK 1000 zpráv/s



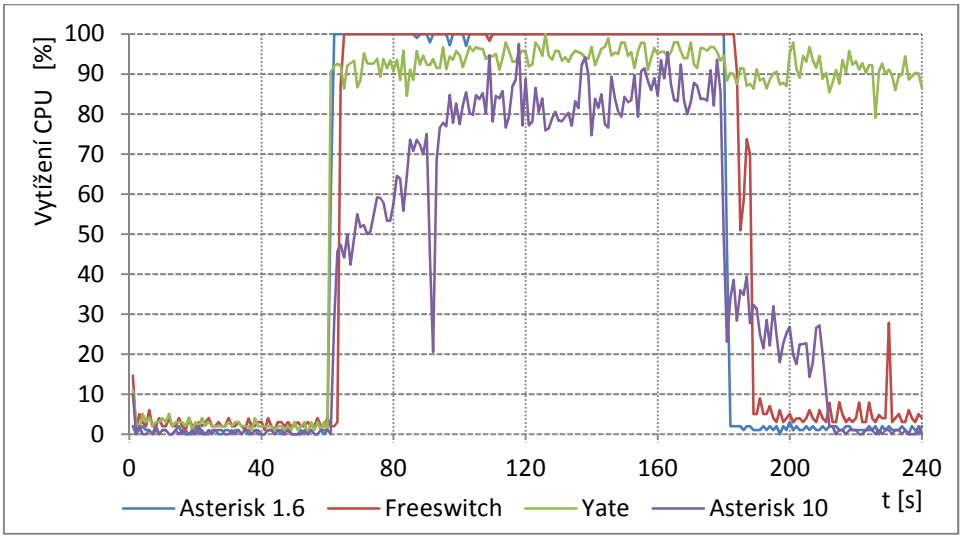
BYE 1000 zpráv/s



OPTIONS 1000 zpráv/s



REGISTER 1000 zpráv/s



INVITE 1000 zpráv/s

C. Detekční pravidla systému Snort

```
alert ip any any -> $SIP_PROXY_IP $SIP_PORTS (msg:"INVITE message
flooding"; content:"INVITE"; depth:6; threshold: type both , track by_src,
count 100, seconds 60; sid:5000004; rev:1; fwsam: src, 5 minutes;)

alert ip any any -> $SIP_PROXY_IP $SIP_PORTS (msg:"REGISTER message
flooding"; content:"REGISTER"; depth:8; threshold: type both , track
by_src, count 100, seconds 60; sid:5000005; rev:1; fwsam: src, 5 minutes;)

alert ip any any -> $SIP_PROXY_IP $SIP_PORTS (msg:"OPTIONS message
flooding"; content:"OPTIONS"; depth:7; threshold: type both , track by_src,
count 100, seconds 60; sid:5000006; rev:1; fwsam: src, 5 minutes;)

alert ip any any -> $SIP_PROXY_IP $SIP_PORTS (msg:"ACK message flooding";
content:"ACK"; depth:24; threshold: type both, track by_src, count 100,
seconds 60; sid:500007; rev:1; fwsam: src, 5 minutes;)

alert ip any any -> $SIP_PROXY_IP $SIP_PORTS (msg:"BYE message flooding";
content:"BYE"; depth:24; threshold: type both, track by_src, count 100,
seconds 60; sid:500008; rev:1; fwsam: src, 5 minutes;)

alert udp any any -> $SIP_PROXY_IP $SIP_PORTS (msg:"UDP message flooding";
sid:500009; threshold: type both, track by_src, count 2000, seconds 1;
fwsam: src, 5 minutes;)

alert icmp any any -> $SIP_PROXY_IP any (msg:"ICMP message flooding";
sid:5000010; threshold: type both, track by_src, count 1000, seconds 1;
fwsam: src, 5 minutes;)

alert tcp any any -> $SIP_PROXY_IP any (msg:"TCP message flooding";
sid:5000011; threshold: type both, track by_src, count 1000, seconds 1;
fwsam: src, 5 minutes;)
```

D. Vybrané detekční pravidla systému OSSEC

Jednotlivá pravidla se nastavují v konfiguračních souborech xml, pro tento případ byly upraveny pravidla pro asterisk v souboru *asterisk_rules.xml*

```
root@ossec:/# nano /var/ossec/rules/asterisk_rules.xml

<group name="asterisk,">
<rule id="102000" level="0">
<description>Grouping of Asterisk rules</description>
<decoded_as>asterisk</decoded_as>
</rule>

<rule id="102002" level="0">
<match>rejected because extension not found</match>
<description>an unknown username</description>
<if_sid>102000</if_sid>
</rule>

<rule id="102003" level="10" frequency="10" timeframe="600">
<if_matched_sid>102002</if_matched_sid>
<description>Enumeration of users on asterisk in process</
description>
</rule>

<rule id="102004" level="6">
<if_sid>102000</if_sid>
<match>Wrong password</match>
<description>Someone got the password wrong</description>
</rule>

<rule id="102006" level="10" frequency="10" timeframe="600">
<if_matched_sid>102004</if_matched_sid>
<description>A password cracking attack in process</description>
</rule>
</group>
```

Pro výše uvedena pravidla je nutné definovat dekoder, ten se nastaven na OSSEC serveru v souboru *decoder.xml*

```
root@ossec:/#nano /var/ossec/etc/decoder.xml

<decoder name="asterisk">
  <program_name>^asterisk</program_name>
</decoder>

<decoder name="asterisk-denied2">
  <parent>asterisk</parent>
  <prematch>Registration from </prematch>
  <regex offset="after_prematch"> \
failed for '(\d+.\d+.\d+.\d+)'\
</regex>
  <order>srcip</order>
</decoder>
```

E. Odkazy na použitý software

CRUNCH - WORDLIST GENERATOR. *Crunch 3.2* [software]. 2009, 2012-01-28 [cit. 12.5.2012]. open source. Dostupné z: http://crunch-wordlist.sourceforge.net/

DSNIFF. *Arpspoof 2.4* [software]. 2000, 2002-05-27 [cit. 12.5.2012]. 3-clause BSD License. Dostupné z: http://www.monkey.org/~dugsong/dsniff/

ETTERCAP PROJECT. *Ettercap NG-0.7.3* [software]. 2001, 2005-06-17 [cit. 12.5.2012]. open source. Dostupné z: http://ettercap.sourceforge.net/

HACKING EXPOSED VOIP. *Rtpmixsound 3.0* [software]. 2007 [cit. 12.5.2012]. Dostupné z: http://www.hackingvoip.com/sec_tools.html

HPING. *Hping3 3.0.0-alpha-2* [software]. 2005, 2011-05-17 [cit. 12.5.2012]. GNU GPL license. Dostupné z: http://www.hping.org/hping3.html

NMAP. *Nmap 5.51* [software]. 1997, 2011-02-11 [cit. 12.5.2012]. open source. Dostupné z: http://nmap.org/

SIP LOGIN DUMPER/CRACKER. *SIPcrack 0.2* [software]. 2011-05-17 [cit. 12.5.2012]. open source. Dostupné z: https://launchpad.net/ubuntu/+source/sipcrack/0.2-2build1

SIPP. *Sipp 3.2* [software]. 2004, 2010-11-16 [cit. 12.5.2012]. GNU GPL license. Dostupné z: http://sourceforge.net/projects/sipp/files/sipp/3.2/

SIPVICIOUS. *SipVicious 0.2.7* [software]. 2012-02-22 [cit. 12.5.2012]. Dostupné z: http://code.google.com/p/sipvicious/downloads/list

SLOWLORIS HTTP DOS. *Slowloris* [software]. 2009 [cit. 12.5.2012]. Dostupné z: http://ha.ckers.org/slowloris/

VMWARE. *VMware Player 4.0* [software]. 2008, 2012-05-03 [cit. 12.5.2012]. Dostupné z: https://my.vmware.com/web/vmware/info/slug/desktop_end_user_computing/vmware_player/4_0

UBUNTU. *Ubuntu 11.10* [software]. 2011, 2011-11-22 [cit. 12.5.2012]. GNU GPL license. Dostupné z: http://releases.ubuntu.cz/oneiric/

UBUNTU. *Ubuntu 10.04.4 LTS* [software]. 2011 [cit. 12.5.2012]. GNU GPL license. Dostupné z: http://releases.ubuntu.com/lucid/

FUSIONPBX. *Fusionpbx v3.0. FreeSWITCH 1.0* [software]. 2011 [cit. 12.5.2012]. GNU GPL license. Dostupné z: http://wiki.fusionpbx.com/index.php/Ubuntu_ISO

FREESENTRAL. *FreeSentral 1.2*: *YATE 3.0.0* [software]. 2011 [cit. 12.5.2012]. GNU GPL license. Dostupné z: http://www.freesentral.com/index.php/Download/Mirrors

TRIXBOX. *Trixbox CE 2.8.0.4 (Stable)*: *Asterisk 1.6.* [software]. 2011 [cit. 12.5.2012]. GNU GPL license. Dostupné z: http://fonality.com/trixbox/downloads

ASTERISK. *Asterisk 10.0.0* [software]. 2011 [cit. 12.5.2012]. GNU GPL license. Dostupné z: http://www.asterisk.org/downloads

F. Obsah přiloženého CD

1. Elektronická kopie diplomové práce
2. Textová verze jednotlivých skriptů z generátoru VoIPtester
3. Virtuální obraz útočníkova počítače s připraveným rozhraním VoIPtester
4. Návod k použití generátoru VoIPtester
5. Návod k instalaci detekčního systému Snort
6. Návod k instalaci detekční systému OSSEC