VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2018 Roland Dávidík

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

PROGRAM PRO DETEKCI ÚTOKŮ NA TLS PROTOKOL
DETECTION OF ATTACKS TARGETED AT TLS PROTOCOL

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Roland Dávidík**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Zdeněk Martinásek, Ph.D.**
SUPERVISOR

BRNO 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Informační bezpečnost**
Ústav telekomunikací

Student: Roland Dávidík ***ID:*** 185924
Ročník: 3 ***Akademický rok:*** 2017/18

NÁZEV TÉMATU:

Program pro detekci útoků na TLS protokol

POKYNY PRO VYPRACOVÁNÍ:

Cílem bakalářské práce je analyzovat známé útoky na protokol TLS (Transport Layer Security). Analyzujte současný stav problematiky a zaměřte se na útoky HeartBleed, Poodle a BEAST attack. Detailně popište využití zranitelnosti při útoku. Popište způsob obrany proti analyzovaným útokům. Navrhněte a implementujte nástroj v programovacím jazyku Python pro detekci výše zmíněných útoků. Výsledkem bakalářské práce bude funkční program detekující tyto útoky a dále video prezentující funkčnost útoku na experimentálním pracovišti.

DOPORUČENÁ LITERATURA:

[1] DURUMERIC, Zakir, et al. The matter of heartbleed. In: Proceedings of the 2014 Conference on Internet Measurement Conference. ACM, 2014. p. 475-488.

[2] GUJRATHI, Siddharth. Heartbleed bug: AnOpenSSL heartbeat vulnerability. International Journal of Computer Science and Engine ter Science and Engineering, 2014, 2.5: 61-64.

Termín zadání: 5.2.2018 ***Termín odevzdání:*** 29.5.2018

Vedoucí práce: Ing. Zdeněk Martinásek, Ph.D.
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Táto bakalárska práca popisuje princíp útokov HeartBleed, POODLE a BEAST na TLS protokol. Následne sú tieto útoky zrealizované v experimentálnych pracoviskách. Ďalej táto práca popisuje TLS protokol a ochranu voči zmieneným útokom. Výstupom tejto bakalárskej práce je program na detekciu útokov HeartBleed, POODLE a BEAST a dve videa zobrazujúce útok HeartBleed a POODLE.

KĽÚČOVÉ SLOVÁ

TLS, SSL, HeartBleed, POODLE, BEAST, program, útok, video, CBC, detekcia.

ABSTRACT

This Bachelor thesis describes principles of HeartBleed, POODLE and BEAST attack in TLS protocol. Consequently, these attacks are implemented in experimental workplaces. Further, this work describes the TLS protocol and protection against these attacks. Output of this bachelor thesis is a program for detection HeartBleed, POODLE and BEAST attacks and two videos showing HeartBleed and POODLE attacks.

KEYWORDS

TLS, SSL, HeartBleed, POODLE, BEAST, program, attack, video, CBC, detection.

DÁVIDÍK, R. *Program pro detekci útoků na TLS protokol*. Brno, 2018, 48 s. Bakalárska práca. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedúci práce: Ing. Zdeněk Martinásek, Ph.D.

VYHLÁSENIE

Vyhlasujem, že som svoju bakalársku prácu na tému „Program pro detekci útoků na TLS protokol“ vypracoval(a) samostatne pod vedením vedúceho bakalárskej práce, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor(ka) uvedenej bakalárskej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto bakalárskej práce som neporušil(a) autorské práva tretích osôb, najmä som nezasiahol(-la) nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý(-á) následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákoníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora(-ky)

POĎAKOVANIE

Rád by som poďakoval vedúcemu bakalárskej práce pánovi Ing. Zdeňku Martináskovi, Ph.D. za odborné vedenie, konzultácie a podnetné návrhy k práci. Ďalej by som rád poďakoval svojej rodine a priateľke za podporu, bez ktorej by som to nedokázal. Na záver by som rád poďakoval vybraným spolužiakom a kamarátom za pomoc počas celého bakalárskeho štúdia.

Brno ....

podpis autora(-ky)

OBSAH

Úvod **10**

1 TLS protokol **11**
1.1 Podrobný popis celej komunikácie medzi klientom a serverom 12
1.1.1 Popis komunikácie s klientom bez dôveryhodného certifikátu: . 12
1.1.2 Popis komunikácie s klientom s dôveryhodným certifikátom: . 15

2 Útok HeartBleed **17**
2.1 Popis HeartBleed 17
2.2 Popis útoku 17
2.3 Ochrana proti útoku 19

3 Útok POODLE **20**
3.1 Popis útoku 20
3.2 Ochrana proti útoku 21

4 Útok BEAST **22**
4.1 Popis blokových šifier s CBC módom 22
4.2 Požiadavky a princíp útoku BEAST 23
4.2.1 Princíp útoku BEAST 23
4.3 Ochrana proti útoku 24

5 Experimentálne pracovisko **25**
5.1 Experimentálne pracovisko č.1 25
5.1.1 Nastavenie servera 25
5.1.2 Nastavenie monitorovacej stanice 26
5.1.3 Nastavenie útočníka 27
5.1.4 Praktická realizácia útoku HeartBleed 27
5.2 Experimentálne pracovisko č.2 28
5.2.1 Nastavenie servera 29
5.2.2 Nastavenie monitorovacej stanice a útočníka 29
5.2.3 Nastavenie klienta 29
5.2.4 Praktická realizácia útoku POODLE 30
5.2.5 Praktická realizácia útoku BEAST 30

6 Program na zachytenie útoku HeartBleed, POODLE a BEAST **32**
6.1 Popis programu 33
6.1.1 Časť HeartBleed 33

6.1.2 Časť POODLE 34
6.1.3 Časť BEAST 35

7 Videá prezentujúce funkčnosť útokov na experimentálnom pracovisku 38
7.1 Video: Útok HeartBleed 38
7.2 Video: Útok POODLE 38

8 Záver 40

Literatúra 41

Zoznam symbolov, veličín a skratiek 45

Zoznam príloh 47

A Obsah priloženého CD 48

ZOZNAM OBRÁZKOV

1.1 Zaradenie v modele TCP/IP. 11
1.2 Popis komunikácie medzi klientom a serverom. 14
1.3 Štruktúra Record protokolu. 15
2.1 Správa HeartBeatRequest. 17
2.2 Znázornenie útoku HeartBleed. 18
2.3 Správa HeartBeatRequest poslaná útočníkom. 18
2.4 Správa HeartBeatResponse poslaná obeťou. 19
3.1 Znázornenie útoku POODLE. 20
4.1 Znázornenie CBC módu v TLS 1.0. 22
4.2 Zostavená správa, modrá = známe znaky, žltá = cookie. 24
4.3 Nová zostavená správa, modrá = známe znaky, zelená = zistený znak, žltá = cookie. 24
5.1 Experimentálne pracovisko č.1. 25
5.2 Získané dáta. 28
5.3 Experimentálne pracovisko č.2. 29
5.4 Získané cookie. 31
6.1 Záznam v súbore. 32
6.2 Zadávanie parametrov do programu. 33
6.3 Schéma časti HeartBleed. 34
6.4 Schéma časti POODLE. 36
6.5 Schéma časti BEAST. 37

ZOZNAM TABULIEK

5.1 Nastavenie staníc. 26
5.2 Nastavenie staníc. 29

ÚVOD

S rozvojom internetu sa ukázalo, že protokol HTTP (Hypertext Transfer Protocol), ktorý slúži najčastejšie na komunikáciu medzi webovým prehliadačom a webovým serverom, nie je bezpečnostne vyhovujúci. Jedným z týchto dôvodov je aj fakt, že dáta sú posielané v otvorenej forme. Preto vznikol protokol SSL (neskôr TLS), ktorý slúži na zabezpečenie týchto bezpečnostných slabín, a vďaka nemu vznikol protokol HTTPS (Hypertext Transfer Protocol Secure). HTTPS je v podstate HTTP zabezpečené pomocou SSL protokolu. Postupom času sa našli bezpečnostné slabiny aj v SSL/TLS protokole, ale tie boli následne vo vyšších verziách ošetrené.

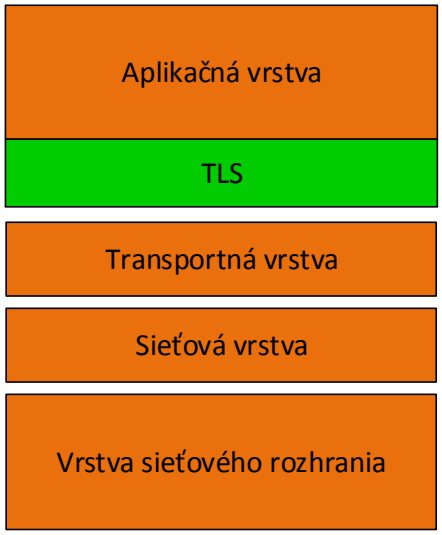
Aby sme lepšie porozumeli danej tématike musíme si určiť niekoľko služieb, ktoré SSL/TLS zabezpečuje:

- integrita dát - schopnosť rozoznať, či bola správa pozmenená pri prenose,
- autentizácia - overenie autora správy,
- dôvernosť - schopnosť utajenia informácie pred neoprávneným užívateľom.

Hlavným cieľom mojej bakalárskej práce je analyzovať útoky HeartBleed, POODLE a BEAST. Aby bolo možné lepšie pochopiť princípy týchto útokov, tak sa v bakalárskej práci budem podrobnejšie zaoberať vysvetlením štruktúry a funkčnosti TLS protokolu, pretože na tento protokol sú dané útoky cielené. Ďalším cieľom mojej práce je popísať spôsoby ochrany proti týmto útokom, a navrhnúť a zostrojiť program v programovacom jazyku Python, ktorý detekuje vyššie uvedené útoky na operačnom systéme Linux. Ďalší cieľ mojej práce je zostrojiť funkčné experimentálne pracoviská na simuláciu daných útokov, ktoré mi potom poslúžia pri výrobe videí prezentujúcich funkčnosť útokov. Moje ciele vychádzajú zo zadania bakalárskej práce.

1 TLS PROTOKOL

TLS (Transport Layer Security) protokol je protokol, ktorý slúži na bezpečné prenášanie dát medzi dvomi aplikáciami na dvoch rôznych počítačoch. Patrí do aplikačnej vrstvy modelu TCP/IP (obr. 1.1).



Obr. 1.1: Zaradenie v modele TCP/IP.

TLS protokol vznikol z SSL 3.0 (Secure Sockets Layer) protokolu, ktorý vytvorila firma Netscape Comunication. TLS protokol používa klient/server architektúru, kde ten, čo zahajuje komunikáciu, je klient.

TLS sa skladá zo štyroch protokolov:

- Handshake Protocol nazývaný aj zahajovací protokol, slúži na vzájomnú autentizáciu oboch komunikujúcich, dohodnutie sa na použití šifier, ustanovenie a výmenu kľúčov. Tieto kľúče sa ďalej používajú v prenosovom protokole (Record protocol).
- Change Cipher Spec Protocol slúži na prechod z Handshake Protocol na Record Protocol. Akonáhle si oba komunikujúci dohodnú kľúče v Handshake Protocol, zašle sa správa o veľkosti 1 bajt, ktorá signalizuje, že sa má prejsť do Record Protocol. Táto správa je Change Cipher Spec Protocol.
- Alert Protocol slúži na predávanie informácii o chybách v priebehu celého spojenia. Akonáhle príde k chybe, pošle sa správa (Alert Protocol). Ak je táto chyba menej závažná, komunikácia pokračuje. Ak je ale závažnosť chyby vysoká, spojenie sa ukončí. Správu môžu poslať obaja komunikujúci.

- Record Protocol nazývaný aj prenosový protokol, slúži na šifrovanie komunikácie, kontrolný súčet a prenášanie šifrovaných dát.

1.1 Podrobný popis celej komunikácie medzi klientom a serverom

Existujú dve možnosti, s ktorými sa môžeme stretnúť. Prvá možnosť je, že klient má svoj vlastný certifikát vydaný dôveryhodnou autoritou. Druhá možnosť je, že klient takýto certifikát nemá a pristupuje anonymne.

1.1.1 Popis komunikácie s klientom bez dôveryhodného certifikátu:

Komunikácia je vždy zahajovaná klientom. Klient pošle ClientHello (CH) správu na server. Správa sa skladá z unikátu klienta (U_k) a kryptografických kombinácií (KK). U_k sa skladá z dvoch častí. Jedna časť je aktuálny čas a dátum plus počet sekúnd od 1.1.1970. To má veľkosť 32 bitov. Druhá časť je 28 bitové náhodné číslo. Kryptografické kombinácie obsahujú zoznam šifier a hešovacích funkcií, ktoré je klient schopný používať.

Akonáhle dorazí CH správa na server, server odošle Server Hello (SH) správu. Táto správa obsahuje verziu protokolu TLS, unikát servera (U_S - aktuálny čas a dátum plus počet sekúnd od rovnakého dátumu ako u CH plus náhodné 28 bitové číslo) a kryptografickú kombináciu (KK_S), ktorú si server vybral. Môže sa stať, že server si nevyberie žiadnu z kryptografických kombinácií, ktoré mu pošle klient a vtedy spojenie ukončí.

Ďalej server pošle Server Certificate (SC) správu, ktorá je certifikát serveru. Certifikát obsahuje verejný kľúč servera (K_{PS}), ktorý slúži na šifrovanie správ v asymetrickej kryptografii. Keď klient obdrží SH a SC, overí certifikát, uloží si do pamäti U_S a vygeneruje si náhodné tajné číslo PMS (PreMaster-Secret) o dĺžke 48 bitov. Akonáhle klient vykoná všetky tieto kroky, zašifruje náhodné tajné číslo PMS pomocou šifrovacej funkcie, ktorá je uvedená v KK_S a verejného kľúča serveru, ktorý získal z certifikátu. Takto zašifrované PMS potom pošle serveru. Táto správa sa nazýva Client Key Exchange Message. Tu dochádza v podstate k autentizácii serveru, pretože ak na druhej strane je niekto iný a vydáva sa za server, tak nedokáže dešifrovať správu s PMS, pretože nevlastní súkromný kľúč, ktorý sa používa na dešifrovanie.

Obe strany si z U_K, U_S a PMS vypočítajú pomocou pseudonáhodnej funkcie tajnú hodnotu MS (Master Secret) a kryptografické kľúče. Kryptografické kľúče sú

iné pre oba smery komunikácie, teda od klienta k serveru a od serveru ku klientovi. Oba tieto smery majú svoj šifrovací kľúč (K_E). Ďalší kľúč, ktorý si obe strany vypočítajú je HMAC kľúč (K_{HMAC}). Funkcia HMAC je hešovacia funkcia, ktorá slúži na zaistenie integrity správ. Funguje tak, že vstupujú do nej dáta, ktorým chceme zabezpečiť integritu a kľúč pre HMAC. Funkcia HMAC vypočíta z dát a kľúča výslednú hodnotu s konštantou dĺžkou, ktorá sa priloží potom k dátam a celá takáto správa sa zašifruje. Zaistenie integrity tu spočíva v tom, že druhá strana, ktorá takúto správu obdrží, dešifruje ju a vloží dáta z nej do funkcie HMAC spolu s kľúčom pre HMAC. Výslednú hodnotu potom porovná s priloženou hodnotou k obdržaným dátam. Ak sa hodnoty zhodujú, vie, že správa nebola počas prenosu pozmenená. Ak by útočník pozmenil zašifrovanú správu, nezhodovala by sa u príjemcu jeho vypočítaná hodnota HMAC funkcie s prijatou hodnotou, pretože kľuč k HMAC pozná len príjemca a odosielateľa.

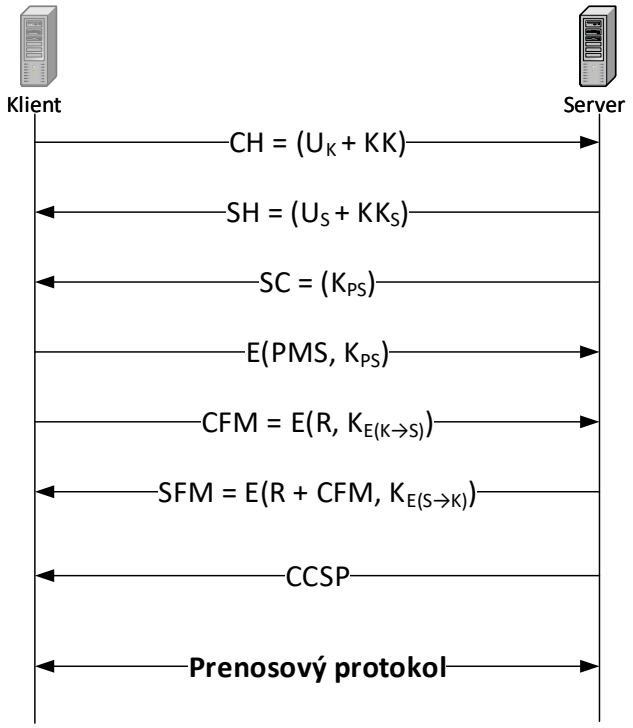
Potom klient pošle správu Client Finished Message (CFM). Táto správa je zašifrovaná šifrovacím kľúčom pre smer od klienta k serveru. Obsahom správy sú všetky doposiaľ poslané správy medzi klientom a serverom (R) a k nim výstup HMAC funkcie. Ako kľúč k funkcii HMAC je použitá hodnota MS. Takto si klient aj server overujú, že kryptografické kľúče pre smer od klienta k serveru sú v poriadku, pretože server pozná čo má klient poslať, keďže má tajnú hodnotu MS a vie všetky správy, ktoré si poslali.

Ak sú kľúče správne a server si dešifroval správu a hodnota v nej sedí, potom server zašle správu Server Finished Message (SFM). Je to posledná správa Handshake Protocol. Správa je zašifrovaná pomocou kľúču pre smer od serveru ku klientovi. Obsahom správy sú, rovnako ako u CFM, všetky doposiaľ poslané správy aj s CFM, ku ktorým je pridaný výstup HMAC funkcie. Ako K_{HMAC} je použitá znova hodnota MS. Takto si klient aj server overia kľúče pre tento smer, a zároveň sa takto aj autentizuje server klientovi, pretože správne kľúče mohol vygenerovať len ten, kto pozná správnu hodnotu PMS, a pretože PMS posielal klient zašifrované pomocou verejného kľúča serveru, môže túto hodnotu poznať len server. Potom sa zašle správa Change Cipher Spec Protocol (CCSP) a prechádza sa z Handshake Protocol na Record Protocol (obr. 1.2).

Proces prenosu je v oboch smeroch rovnaký, jediné, čo sa mení sú kľúče pre smer prenosu. Preto budem nazývať vysielacia strana, klienta alebo server, ktorý bude prenášať dáta, a klienta alebo server, ktorý bude prijímať dáta, prijímacia strana.

V Record Protocol dostane vysielacia strana dáta od aplikácie, ktoré sa majú preniesť. Tieto dáta sa tu následne rozdelia do blokov, ktoré sa nazývajú fragmenty (F_i). Ku každému fragmentu je pridaná výplň a výstup funkcie HMAC z týchto dát. Toto tvorí reťazec. Do funkcie HMAC vstupuje vždy kľúč pre daný smer komunikácie. Výstup z nej má vždy dĺžku 160 bitov. Výplň sa generuje po bajtoch tak, že

sa určí, koľko je treba doplniť bajtov do reťazca, aby mal odpovedajúcu dĺžku a od tohto čísla sa odčíta číslo jedna. Číslo, ktoré vyjde sa doplní do každého bajtu vo výplni. Napr. ak je treba doplniť päť bajtov, tak v každom z týchto piatich bajtov bude hodnota štyri.



Obr. 1.2: Popis komunikácie medzi klientom a serverom.

CH = Client Hello, U_k = unikát klienta, KK = kryptografické kombinácie, SH = Server Hello, U_S = unikát serveru, KK_S = kryptografická kombinácia serveru, SC = Server Certificate, K_{PS} = verejný kľúč servera, E = šifrovanie, PMS = PreMaster Secret, CFM = Client Finished Message, R = všetky predchádzajúce poslané správy, $K_{E(K\rightarrow S)}$ = kryptografický kľúč pre smer od klienta k serveru, SFM = Server Finished Message, $K_{E(S\rightarrow K)}$ = kryptografický kľúč pre smer od serveru ku klientovi, CCSP = Change Cipher Spec Protocol

Keď je reťazec hotový, tak k nemu vysielacia strana vygeneruje náhodný inicializačný vektor. Potom je tento reťazec zašifrovaný šifrou z KK_S. K výslednému zašifrovanému reťazcu je následne pridaný tento inicializačný vektor a je poslaný na prijímaciu stranu.

Prijímacia strana obdrží tento reťazec, odstráni z neho inicializačný vektor a dešifruje ho. Z dešifrovanej správy odstráni zo začiatku výstup z funkcie HMAC o dĺžke 160 bitov a z konca výplň (Počet bajtov výplne dostane, tak že sa pozrie do posledného bajtu na hodnotu a pripočíta k nemu hodnotu jedna). Zo získaného fragmentu potom vypočíta výstup funkcie HMAC a porovná ho s výstupom z dešifrovanej správy. Ak sa výstupy zhodujú predá fragment aplikácii na prijímacej strane, ak nie, spojenie je prerušené (obr. 1.3).

Obr. 1.3: Štruktúra Record protokolu.

IV = inicializačný vektor, V_{HMAC} = výstup HMAC funkcie, F_i = fragment, V = výplň

Ak počas celého tohto prenosu príde server alebo klient k nejakej chybe, zašle druhej strane Alert Protocol. Podľa závažnosti Alert Protocol, následne strana, ktorá tento protokol poslala, zruší alebo ponechá spojenie.

1.1.2 Popis komunikácie s klientom s dôveryhodným certifikátom:

Toto zostavenie komunikácie sa veľmi nelíši oproti zostaveniu komunikácie s klientom bez certifikátu. Rozdiel je tu v pridaní správ na autentizáciu klienta.

Prvá takáto správa je Client Certificate Request (CCR). Túto správu zasiela server v ServerHello a žiada ňou klienta o poslanie jeho certifikátu.

Ďalšia správa je Client Certificate (CC), kde klient pošle svoj certifikát vydaný certifikačnou autoritou serveru.

Posledná takáto správa je Certificate Verify (CV). Táto správa je zaslaná pred Client Finished Message. Tu klient zašifruje pomocou svojho súkromného kľúča všetky doposiaľ prenesené správy a zašle ich serveru. Server potom zoberie verejný kľúč z certifikátu klienta a správu dešifruje. Tu dochádza k autentizácii klienta, pretože len on pozná svoj súkromný kľúč. Ak server dešifruje pomocou verejného kľúča správu a hodnoty v nej súhlasia, vie, že komunikuje s klientom. Ak nie, spojenie je prerušené.

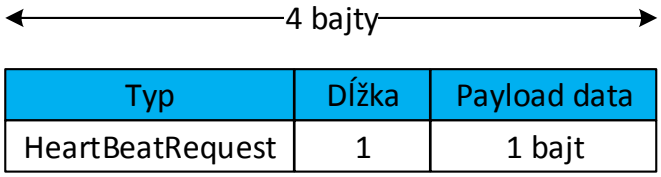
Z tohto popisu nám vyplýva, že TLS zabezpečuje: autentizáciu komunikujúcich strán, šifrovaný prenos, dôvernosť a integritu prenášaných správ. [1, 2, 3, 4, 5]

2 ÚTOK HEARTBLEED

HeartBleed je útok na rozšírenie HeartBeat, ktoré bolo pridané do TLS v standarte RFC 6520, a bolo implementované do kryptografickej knižnice OpenSSL verzie 1.0.1 až 1.0.1f. HeartBeat zaisťuje udržiavanie spojenia v šifrovanom kanále v TLS, väčšinou u nespoľahlivého UDP prenosu, keď sa práve nič nevysiela. Toto spojenie udržiava tak, že posiela provozné správy. Výhodou je, že sa nemusí vyjednávať nové spojenie.

2.1 Popis HeartBleed

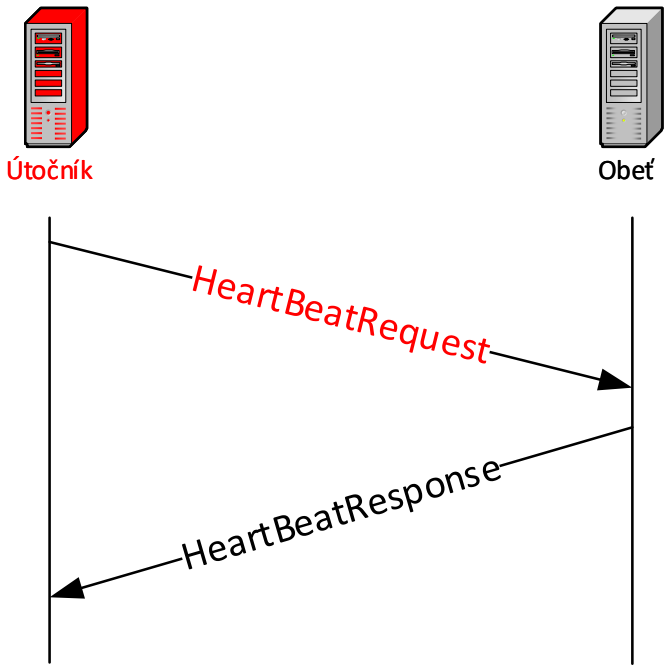
HeartBeat sa skladá z dvoch druhov správ: HeartBeatRequest a HeartBeatResponse. Ktorákoľvek strana môže poslať počas spojenia HeartBeatRequest. Táto správa sa skladá z typu správy (tu sa udáva, čo je to za typ správy, v tomto prípade teda HeartBeatRequest), z dĺžky (hodnota, ktorá udáva počet bajtov payload) a z payload, čo je pole dát. Druhá strana obdrží tuto správu a pošle správu, ktorá sa skladá z typu správy (teda HeartBeatResponse), z dĺžky, a z toľko bajtov z payload, koľko je hodnota dĺžky poslaná v HeartBeatRequest. Takto sa periodicky udržiava spojenie (obr. 2.1).



Obr. 2.1: Správa HeartBeatRequest.

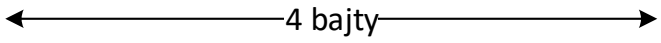
2.2 Popis útoku

Hlavná slabina HeartBeat je, že druhá strana slepo verí hodnote dĺžka a pošle presne toľko bajtov, koľko je jej hodnota. To práve využíva útok HeartBleed, ktorý je postavený na tom, že payload poslaný útočníkom väčšinou obsahuje jeden bajt a hodnota dĺžky je čo najväčšie číslo, ktoré sa dá dosiahnúť, čo je 65535. Celá správa môže mať maximálne 65538 bajtov, a pretože je typ správy jeden bajt a dĺžka dva bajty, preto útočník môže maximálne zadať 65535 (obr. 2.2, 2.3).



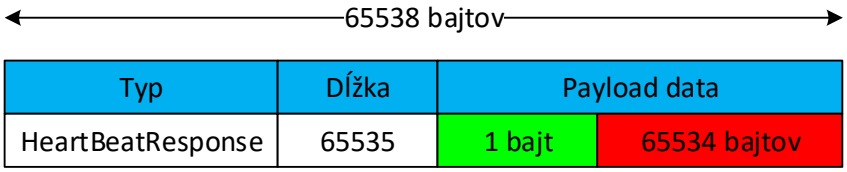
Obr. 2.2: Znázornenie útoku HeartBleed.

Akonáhle obdrží druhá strana túto správu, pošle také množstvo bajtov, aká je hodnota dĺžky. Keďže je ale payload menší ako jeho hodnota dĺžky, pošle server zbytok bajtov z pamäte, ktorá nasleduje za miestom, kde je uložený payload. Server takto pošle časť obsahu svojej pamäte. Pretože sa správy posielajú periodicky, útočník tak dostáva viac a viac dát z pamäte. V tejto pamäti môže byť čokoľvek, čo má server uložené, napríklad kryptografické kľúče, Master Secret, dokonca aj privátny kľúč k certifikátu. Útočník takto môže získať tieto dôležité hodnoty a zneužiť ich, poprípade vydávať sa za druhú stranu. Ohrozený je ako klient, tak i server (obr. 2.4).



Typ	Dĺžka	Payload data
HeartBeatRequest	65535	1 bajt

Obr. 2.3: Správa HeartBeatRequest poslaná útočníkom.



Obr. 2.4: Správa HeartBeatResponse poslaná obeťou.

2.3 Ochrana proti útoku

Ochrana proti tomuto útoku je používať novšie verzie OpenSSL, ako je uvedené vyššie, ktoré majú ošetrený tento typ útoku. Ak server alebo klient používal tieto staršie verzie a prešiel na novú, tak by si mal pre istotu nanovo vygenerovať všetky kľúče a obstarať si nový certifikát, aby neriskoval, že niekto už jeho staré kľúče má. [14, 18, 19, 20, 21, 22]

3 ÚTOK POODLE

Názov útoku vznikol ako skratka „Padding Oracle On Downgraded Legacy Encryption“. Útok POODLE objavili v roku 2014 bezpečnostný technici spoločnosti Google: Bodo Möller, Thai Duong a Krzysztof Kotowicz. Útok využíva zraniteľnosť v protokole SSL 3.0, kde útočí na výplň reťazca pri blokovom šifrovaní v CBC móde.

3.1 Popis útoku

Podmienky pre útok sú rovnaké ako pre útok BEAST. Útočník musí byť „man in the midle“, teda dokáže zachytávať správy medzi klientom a serverom. Ďalej potrebuje mať útočník kontrolu nad internetovým prehliadačom obete, aby mohol pozmeňovať správy medzi klientom a serverom. Na to môže útočník použiť napr. javascriptový kód. Pre úspešné prevedenie útoku je treba ešte dosiahnuť, aby klient a server použili kryptografické kombinácie z protokolu SSL 3.0 takže sa útočník cez obeť snaží downgradovať TLS na SSL verzie 3.0 pri dohadovaní v Handashake protokole.

Slabinu, ktorú útok využiva je že, v SSL 3.0 reťazec vstupuje do funkcie HMAC bez výplne, preto sa nedá zaistiť, že výplň nebola popri prenose pozmenená, a práve to útočník využíva.

Princíp spočíva v tom že, obeť zostrojí správu na zašifrovanie. K tej je pridaná výplň. Útočník vie zistiť aká je hodnota v poslednom bajte V_x a podľa toho vie koľko bajtov je výplň $(V + V_x)$. Túto hodnotu sa dozvie aj funkcia HMAC a tak vie koľko bajtov sú dáta, z ktorých vypočíta výstup a koľko bajtov je výplň, ktorú nechá tak. Útočník pomocou rôznych dotazov posunie dáta v správe tak aby zaplnili aj celú výplň. Po vypočítaní výstupu funkcie HMAC je výstup pridaný k dátam. K dátam sa ešte pridá IV. Reťazec vyzerá presne ako (obr. 1.3) akurát miesto výplne sú pridané dáta, ktoré chce útočník získať (Obr. 3.1).

Obr. 3.1: Znázornenie útoku POODLE.

Správa sa zašifruje a pošle na druhú stranu k príjemcovi. Po dešifrovaní sa pozrie funkcia HMAC u príjemcu na hodnotu v poslednom bajte a k hodnote, ktorá tam je pripočíta jeden. Toto číslo čo vyjde je počet bajtov výplne a toľkoto bajtov funkcia

zahodí. Ďalej oddelí IV a V_{HMAC} zo začiatku a zo zbytku vypočíta výstup funkcie HMAC. Ak posledný bajt v pozmenenej výplni útočníkom sa rovná pôvodnému poslednému bajtu výplne V_x, tak výstup funkcie HMAC u príjemcu vyjde rovnako ako výstup funkcie HMAC u odosielateľa, tak správa je vyhodnotená ako nepozmenená a spojenie sa neukončí. V opačnom prípade je spojenie ukončené. Ak teda správa prejde, útočník vie posledný bajt dát, ktoré chcel vedieť, pretože tento bajt sa rovná hodnote V_x. Toto správne doplnenie sa mu štatisticky podarí raz za 256krát. Ďalej útočník posunie výplň o jeden bajt aby tak získal predposledný bajt dát, ktoré chce vedieť. A takto pokračuje ďalej a ďalej.

3.2 Ochrana proti útoku

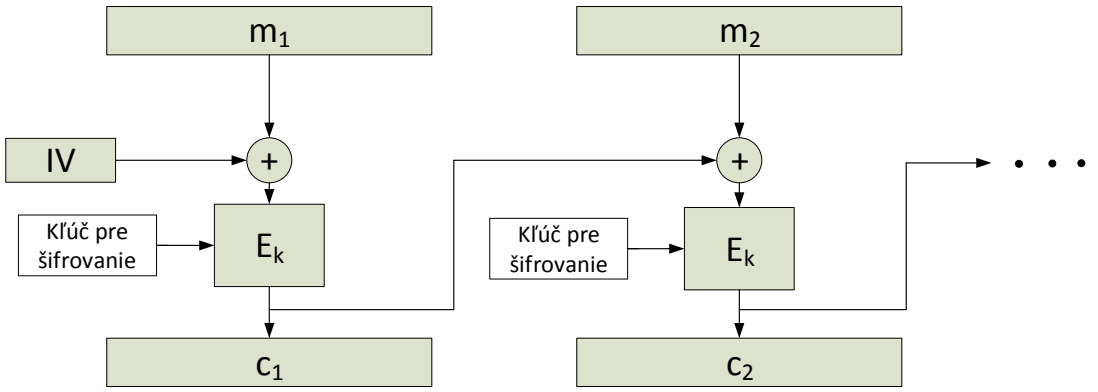
Ochrana proti tomuto útoku je zakázať podporu kryptografických kombinácii z SSL 3.0 na serveri a klientovi.[6, 14, 15, 16, 17]

4 ÚTOK BEAST

Názov BEAST vznikol ako skratka z „Browser Exploit Against SSL/TLS“. Tento útok uverejnil v roku 2011 Juliano Rizzo and Thai Duong. BEAST útočí na zlú implementáciu šifrovacieho módu CBC s predikovaným inicializačným vektorom v protokole TLS verzie 1.0 a SSL 3.0 až nižšie. V TLS verzii 1.1 je už tento útok ošetrený. Útočník môže pomocou tohto útoku získať citlivé dáta od obete.

4.1 Popis blokových šifier s CBC módom

Blokové šifry, ktoré používa TLS verzie 1.0 a SSL 3.0 až nižšie sú AES, DES a 3DES. Tieto blokové šifry, pracujú nad blokmi o 64 alebo 128 bitovej velikosti (záleží od blokovej šifry). To znamená, že ak je správa dlhšia než táto veľkosť bitov, tak je rozdelená na viacej blokov.



Obr. 4.1: Znázornenie CBC módu v TLS 1.0.

Princíp CBC módu v SSL/TLS je, že sa zoberie prvý blok dát, ktorý chceme zašifrovať (m_1) a inicializačný vektor IV (je to náhodná vygenerovaná hodnota) a prevedie sa nad nimi matematická operácia XOR. Takýto výstup sa potom zašifruje pomocou blokovej šifry s tajným kľúčom a vyjde nám prvý zašifrovaný blok (c_1). Druhý blok sa zašifruje rovnako ako predchádzajúci, akurát IV bude zašifrovaný predchádzajúci blok (c_1), takže v tomto prípade IV = c_1. Takéto bloky sú poslané na druhú stranu k príjemcovi. Tu je postup dešifrovania inverzný ako šifrovací postup (obr. 4.1).

Hlavná slabina tejto implementácie je, že keď bude nasledovať ďalšia správa, ktorá sa má zašifrovať, tak ako začiatočný IV sa použije posledný zašifrovaný blok z predchádzajúcej správy.

4.2 Požiadavky a princíp útoku BEAST

Aby bolo možné tento útok realizovať musia byť splnené určité požiadavky:

- Odposluch komunikácie - útočník musí byť schopný odpočúvať komunikáciu medzi klientom a serverom. Je to preto aby mohol zachytávať šifrované bloky, z ktorých zistí inicializačný vektor.
- Stanovenie pozície tajnej hodnoty v správe - útočník musí byť schopný určovať, kde sa budú tajné dáta, ktoré chce zistiť, nachádzať vzhľadom na hranice bloku. Týmto útočník môže vytvárať bloky, ktoré sa dajú ľahšie odhadnúť.
- Posielanie správ cez prehliadač obete - útočník musí byť schopný posielať správy cez obeť. Toho útočník docieli napr. použitím škodlivého javascriptového kódu, ktorý sa začne vykonávať na strane obete.

Útok je založený na tom, že je útočník „man in the midle“, teda dokáže zachytávať správy medzi klientom a serverom. Týmto je zabezpečený odposluch komunikácie. Aby mohol pozmeňovať spravy potrebuje nejako infikovať webový prehliadač obete. Na to útočník použije napr. javascriptový kód.

Útočník na základe zachytenej zašifrovanej komunikácie a znalosti štruktúry AES vkladá cez obeť svoj vlastný nezašifrovaný text do komunikácie a pozoruje a porovnáva šifrované bloky, ktoré z toho dostane. Na základe tohto porovnávania dokáže určiť dáta pred šifrovaním.

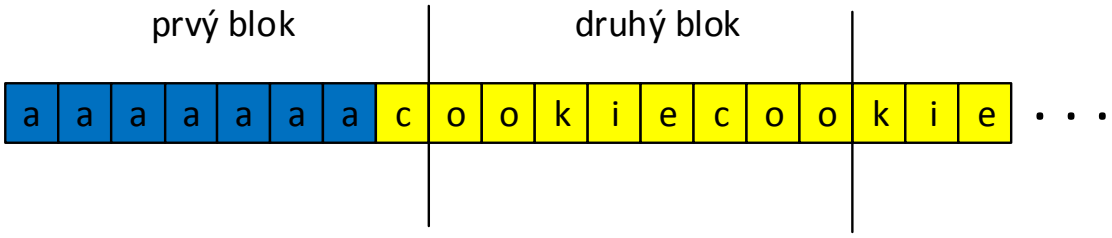
4.2.1 Princíp útoku BEAST

Zoberieme si napríklad blokovú šifru, ktorá pracuje nad blokmi o veľkosti 64 bitov. To je 8 bajtov, takže 8 znakov. Vieme že:

- šifrovanie je: $c_i = E_k(m_i \oplus c_{i-1}), c_0 = IV$, ($E_k$ = šifrovanie s kľúčom k)
- dešifrovanie je: $m_i = D_k(c_i \oplus c_{i-1}), c_0 = IV$, ($D_k$ = dešifrovanie s kľúčom k)

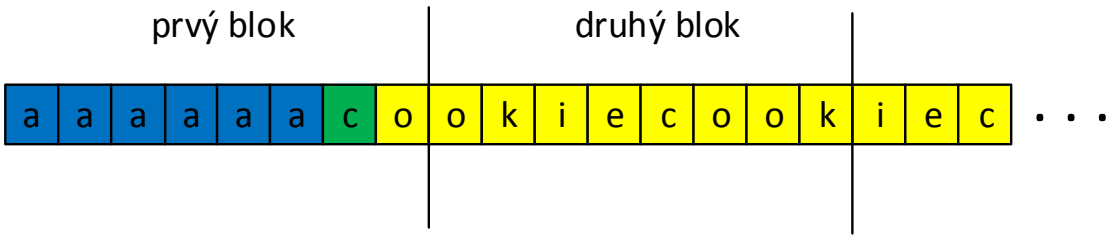
Dajme tomu, že útočník chce získať nejaké citlivé dáta napr. cookie. Útočník vie, kde sa tieto citlivé dáta nachádzajú. Útočník nevidí dáta na obeti priamo, vie len, kde sa nachádzajú a môže ich pomocou javascriptového kódu posielať na server. Následne tieto dáta vidí len v zašifrovanej podobe.

Prvý krok, ktorý spraví je, že pomocou javascriptového kódu pošle cez obeť nejakú náhodnú správu (M_r) na server. To robí preto, lebo sa potom pozrie na posledný zašifrovaný blok. Tento blok bude pri ďalšom šifrovaní použitý ako začiatočný IV. Následne útočník pošle cez obeť špecialne zostavenú správu M_1. Tá sa bude skladať z 7 známych znakov plus cookie (obr. 4.2). Keďže sa do bloku zmestí 8 znakov, tak v prvom bloku bude mať útočník 7 svojich známych znakov a prvý



Obr. 4.2: Zostavená správa, modrá = známe znaky, žltá = cookie.

znak z cookie. Útočník si zapamätá ako vyzerá prvý šifrovaný blok z zašifrovanej správy (C_1). Potom útočník znova pošle cez obeť M_r. Je to z toho dôvodu, aby pri ďalšom šifrovaní bol rovnaký IV ako pri šifrovaní M_1. Teraz útočník zostrojí správu M_2, ktorá sa bude skladať z tých rovnakých 7 známych znakov plus 1 náhodný znak. Následne pošle túto správu cez obeť. Útočník sa potom pozrie, ako vyzerá prvý šifrovaný blok z zašifrovanej správy (C_2). Ak sa prvý šifrovaný blok z C_1 a C_2 rovnajú, útočník uhádol prvý znak z cookie a vie jeho hodnotu. Pravdepodobnosť odhadnutia je 0,39 %, teda raz z 256. Keď sa tak stane útočník vytvorí novú špecialne zostavenú správu M_1, ktorá sa bude skladať z 6 známych znakov plus 1 známy znak z cookie plus 1 neznámy znak z cookie. Tak to sa opakuje postup až dokým útočník nezíska celú hodnotu cookie (obr.4.3). [6, 7, 8, 9]



Obr. 4.3: Nová zostavená správa, modrá = známe znaky, zelená = zistený znak, žltá = cookie.

4.3 Ochrana proti útoku

Ochrana proti tomuto útoku je používať vyššie verzie SSL/TLS ako tie, ktoré sú uvedené v úvode. Poprípade, ak by ste chceli zabezpečiť proti tomuto útoku zraniteľné verzie, musíte na servery zakázať používanie kryptografických kombinácii s blokovými šiframi v CBC mode.[10, 11, 12, 13]

5 EXPERIMENTÁLNE PRACOVISKO

Ďalším bodom mojej bakalárskej práce je vytvoriť experimentálne pracovisko a zrealizovať na ňom útoky HeartBleed, POODLE a BEAST, aby som mohol vytvoriť videá prezentujúce funkčnosť týchto útokov, čo je tiež súčasťou zadania mojej bakalárskej práce. Moje experimentálne pracovisko bude pozostávať z dvoch podpracovísk:

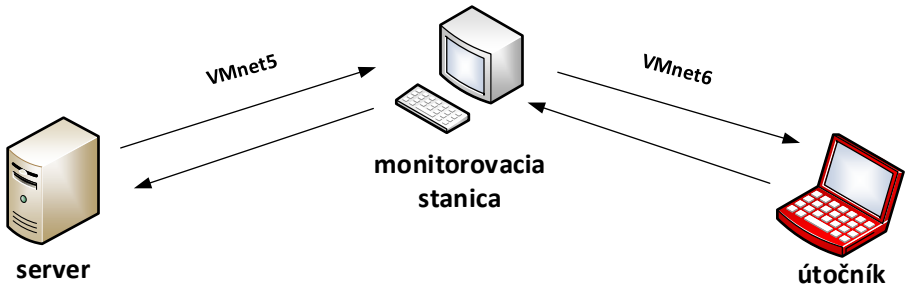
1. experimentálne pracovisko č.1,
2. experimentálne pracovisko č.2.

Ako virtualizačné prostredie pre obe pracoviská som použil VMware Workstation 12. VMware som použil preto, lebo mám s ním najväčšie skúsenosti spomedzi virtualizačných prostredí.

5.1 Experimentálne pracovisko č.1

Toto experimentálne pracovisko bude slúžiť na realizáciu útoku HeartBleed. Experimentálne pracovisko č.1 pozostáva z troch strojov (obr. 5.1):

1. server - Ubuntu 12.04,
2. monitorovacia stanica - Ubuntu 17.10,
3. útočník - Kali Linux 2017.2.



Obr. 5.1: Experimentálne pracovisko č.1.

5.1.1 Nastavenie servera

Aby sa nám tento útok podaril, musíme mať na servery openSSL (verziu 1.0.1 až 1.0.1f), pretože vo vyšších verziách je TLS ošetrené voči tomuto útoku. Ale zase nemôžeme mať staršiu verziu ako 1.0.1, pretože v starších verziách sa nenachádza

stanica	VMnet	interface	IP adresa	maska	brána
server	VMnet5	eth0	192.168.64.135	255.255.255.0	192.168.64.1
monitorovacia stanica	VMnet5	Ens33	192.168.64.1	255.255.255.0	-
	VMnet6	Ens37	192.168.0.1	255.255.255.0	-
útočník	VMnet6	eth0	192.168.0.5	255.255.255.0	192.168.0.1

Tab. 5.1: Nastavenie staníc.

doplnok HeartBeat. OpenSSL je open source knižnica, ktorá nám implementuje SSL/TLS protokol do operačného systému. Ja v mojom operačnom systéme používam openSSL 1.0.1 verziu.

Ďalej nastavíme IP adresu, masku, bránu a VMnet podľa tabuľky 5.1. Ďalším krokom je vytvoriť na servery webovú stránku, na ktorú sa potom útočník pripojí, a tak prebehne https spojenie, ktoré je zabezpečené pomocou SSL/TLS protokolu. Na vytvorenie webovej stránky som použil program Apache. Apache je softwarový webový server, ktorý je dostupný pre rôzne operačné systémy.[23]

Ďalej si musíme vytvoriť certifikát, aby server vlastnil svoj verejný a súkromný kľúč. Tie nám budú slúžiť v Handshake protokole na bezpečné prenesenie PMS, z ktorého si vypočítame MS (popis v časti 1.1.1). Po vytvorení certifikátov prepojíme certifikáty s našou vytvorenou webovou stránkou, aby sa k nej dalo pristupovať pod https. Ak nám ide otvoriť naša webová stránka v prehliadači pod https, nastavenie servera je hotové.

5.1.2 Nastavenie monitorovacej stanice

Na monitorovacej stanici nastavíme IP adresy, masky a VMnety podľa tabuľky 5.1. Potom si na monitorovaciu stanicu nahrajeme súbor s programom HPB_Detective (program je výsledkom mojej bakalárskej práce). Aby nám tento program fungoval musíme si stiahnuť a nainštalovať modul pyshark.[24]

Jeho správnu funkciu môžeme otestovať spustením programu HPB_Detective. Program je treba spustiť s administrátorskými právami, príkazom „sudo“. Na vypnutie programu potom zadáme „q“ a potvrdíme enterom. Ďalej potrebujeme preposielať komunikáciu medzi dvomi interface. Je to preto, aby všetka komunikácia čo ide na server išla cez monitorovaciu stanicu. Na to nám slúžia tieto príkazy:

```
sudo iptables -t nat -A POSTROUTING --out-interface Ens33 -j MASQUERADE
sudo iptables -A FORWARD --in-interface Ens37 -j ACCEPT
sudo iptables -t nat -A POSTROUTING --out-interface Ens37 -j MASQUERADE
sudo iptables -A FORWARD --in-interface Ens33 -j ACCEPT
```

Následne musíme ešte povoliť presmerovanie, preto si otvoríme súbor sysctl.conf príkazom:

```
nano /etc/sysctl.conf
```

V súbore odkomentujeme „net.ipv4.ip_forward = 1“ a uložíme súbor. Nakoniec zadáme príkaz:

```
sudo sysctl -p
```

Po zadaní všetkých príkazov je nastavenie monitorovacej stanice hotové.[25]

5.1.3 Nastavenie útočníka

Na nastavenie útočníka nám stačí len nastaviť IP adresa, maska, brána a VMnet podľa tabuľky 5.1, pretože Kali Linux má už všetko potrebné v sebe implementované. Správne nastavenie celého pracoviska môžeme overiť tak, že útočník otvorí v internetovom prehliadači stránku servera s https. Po potvrdení certifikátu by sa mal útočník na webovú stránku dostať. Ak nie, niečo ste nastavili zle. V takomto prípade si skúste prekontrolovať celé nastavenie.

5.1.4 Praktická realizácia útoku HeartBleed

Na monitorovacej stanici spustíme program HPB_Detective. Program je treba spustiť s administrátorskými právami, príkazom „sudo“. Pretože monitorujeme stanicu „server“, do prvej výzvy napíšeme jej IP adresu. Do druhej výzvy zadáme, kam chceme ukladať log súbor. Zadáme ľubovolnú cestu akú chceme. Ja som zadal: /home/monitoring/Plocha/Log.txt. V tretej výzve zadáme, ktorý interface chceme sledovať. Zadáme názov interface, podľa tabuľky 5.1. Po zadaní všetkých troch výziev program monitoruje sieťový prenos.

Na útočníkovy otvoríme aplikáciu Metasploit. Po načítaní programu zadáme príkaz:

```
use auxiliary/scanner/ssl/openssl_heartbleed
```

Týmto príkazom sa dostaneme do pomocného modulu pre tento útok. Ďalej musíme nastaviť, aby sa v HeartBeat choval útočník „ukecano“ a často robilo HeartBeat-Request príkazom :

```
set verbose true
```

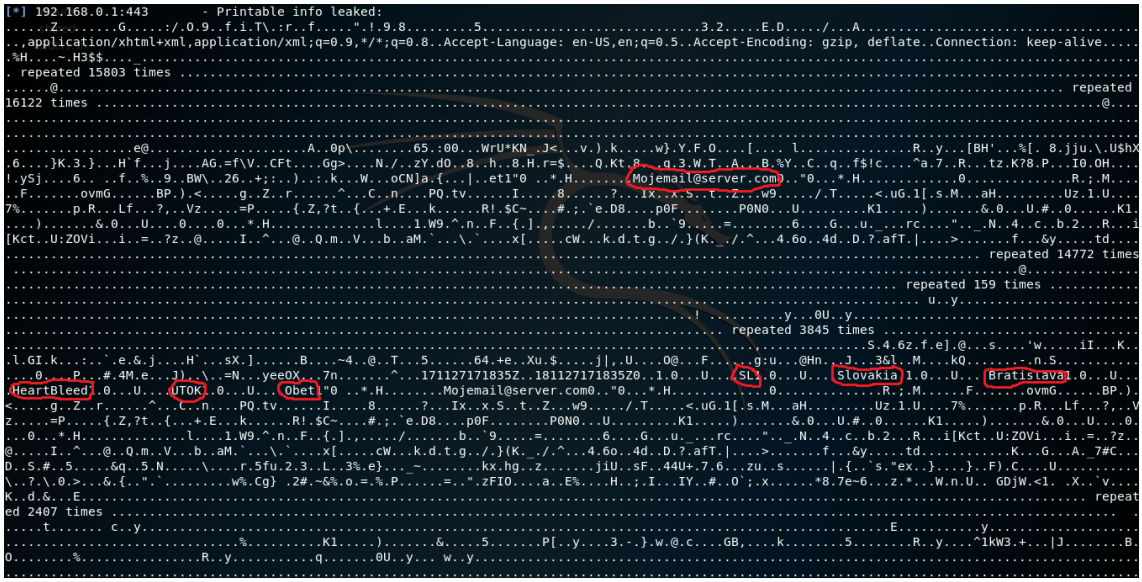
Potom nastavíme adresu obete.

```
set rhosts 192.168.64.135
```

A už nám stačí len spustiť útok.

```
run
```

Po príkaze „run“ môžeme vidieť v terminále celý priebeh útoku, od naväzovania spojenia až po samotné získané dáta. Na obrázku môžeme taktiež vidieť, že sa nám podarilo ukoristiť časť dát z certifikátu, ktoré som tam zadával (zakrúžkované časti) (obr. 5.2).



Obr. 5.2: Získané dáta.

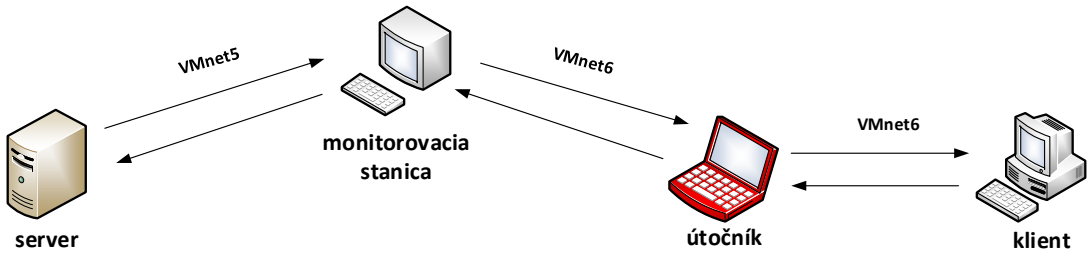
Získané dáta z nášho útoku HeartBleed majú 65535 bitov. To, ktoré dáta sa podarí útočníkovi získať závisí od toho, kde si obeť payload zo správy HeartBeatRequest ukladá. Je to preto, lebo zvyšné dáta, ktoré obeť posiela, sú tie, čo sa nachádzajú uložené v pamäti hneď za uloženým payload. Takže, ak máme nejaké citlivé dáta, v rozsahu do 65535 bitov, uložené za payload, útočník ich má tiež. [26, 27]

Po prepnutí na monitorovaciu stanicu, nájdeme v adresári, ktorý sme zvolili vo výzve programu, log súbor. Po jeho otvorení môžeme vidieť, že bol útok zaznamenaný.

5.2 Experimentálne pracovisko č.2

Toto experimentálne pracovisko bude slúžiť na realizáciu útoku POODLE a BEAST. Experimentálne pracovisko č.2 pozostáva zo štyroch strojov (obr. 5.3):

1. server - Bee-box v.1.6,
2. monitorovacia stanica - Ubuntu 17.10,
3. útočník - Kali Linux 1.1.0,
4. klient - Kali Linux 1.1.0.



Obr. 5.3: Experimentálne pracovisko č.2.

stanica	VMnet	interface	IP adresa	maska	brána
server	VMnet5	eth0	192.168.64.135	255.255.255.0	192.168.64.1
monitorovacia stanica	VMnet5	Ens33	192.168.64.1	255.255.255.0	-
	VMnet6	Ens37	192.168.0.1	255.255.255.0	-
útočník	VMnet6	eth0	192.168.0.5	255.255.255.0	192.168.0.1
klient	VMnet6	eth0	192.168.0.3	255.255.255.0	192.168.0.1

Tab. 5.2: Nastavenie staníc.

5.2.1 Nastavenie servera

Aby sa nám tento útok podaril, musíme mať na servery openSSL (verziu 0.9.8n alebo nižšiu), pretože vo vyšších verziách je TLS ošetrené voči týmto útokom.

Ja v mojom operačnom systéme používam openSSL 0.9.8g verziu. Ďalej nastavíme IP adresu, masku, bránu a VMnet podľa tabuľky 5.2. Webovú stránku s certifikátmi už nemusíme vytvárať, pretože na servery Bee-box v.1.6 sa už nachádzajú. Bee-box je linuxová distribúcia, ktorá má zámerné veľké množstvo bezpečnostných chýb. Používa sa na etický hacking a rozširovanie bezpečnostných znalostí.[28]

5.2.2 Nastavenie monitorovacej stanice a útočníka

Nastavenie monitorovacej stanice a útočníka je úplne rovnaké ako pri experimentálnom pracovisku č.1

5.2.3 Nastavenie klienta

Na klientovi nastavíme IP adresu, masku, bránu a VMnet podľa tabuľky 5.2. Ďalej nastavíme v prehliadači Iceweasel, aby prehliadač volil len SSL3, a také kryptografické kombinácie, ktoré majú blokové šifry s CBC módom. To nastavíme tak, že do adresného riadku napíšeme:

```
about:config
```

Potom potvrdíme modifikáciu nastavenia prehliadača. Ďalej do vyhľadávania napíšeme „security.tls.version“ a upravíme takto nastavenia:

- security.tls.version.max = 0,
- security.tls.version.min = 0.

Týmto sme nastavili, aby prehliadač používal len SSL3. Ďalej do vyhľadávania napíšeme „security.ssl3.*_rc4_*“.

Všetky výsledky, ktoré nám vyhľadávač vyhľadá zmeníme na „false“. Týmto sme zakázali kryptografické kombinácie, ktoré nemajú blokové šifry s CBC módom. Týmto sme dokončili nastavenie klienta.

5.2.4 Praktická realizácia útoku POODLE

Na monitorovacej stanici spustíme program HPB_Detective, tak ako to popisujem v časti 5.1.4. Potom si na stanicu útočník stiahneme priečinok Poodle-PoC [29]. Po stiahnutí otvoríme tento priečinok v terminále. Následne zadáme príkaz na spustenie skriptu „poodle.py“ s týmito parametrami:

```
sudo python3 poodle.py --start-offset 430 --target-port 443 --target
-host 192.168.64.135 https://192.168.0.5:8443/bWAPP/portal.php
```

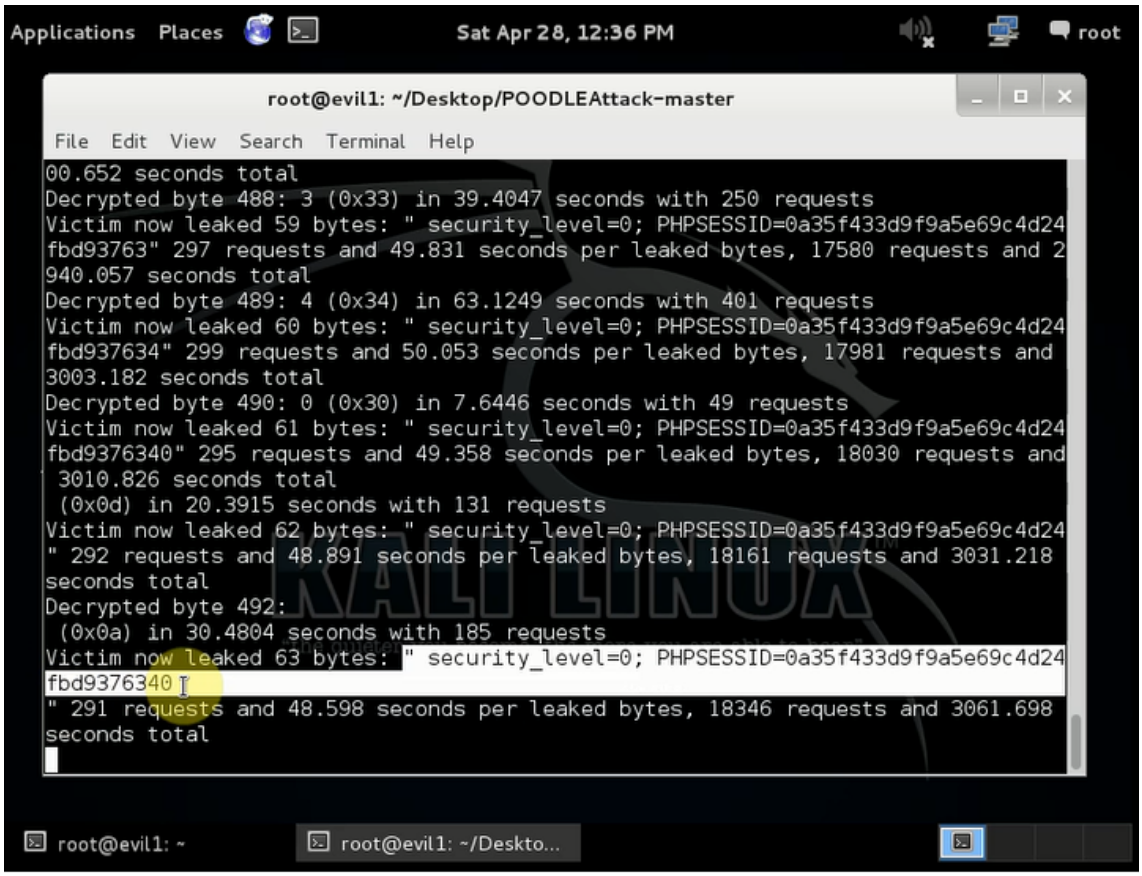
Týmto skriptom útočník zrkadlí https stránku zo servera z adresy 192.168.64.135 na adresu „https://192.168.0.5:8443/bWAPP/portal.php“. Ďalšia vec, ktorú robí tento skript je, že na adrese „192.168.0.5:8000“ vytvorí stránku s javascriptovým kódom, ktorý upravuje správy klienta tak, že veci, ktoré chce útočník zistiť posúva do výplne. Tento príkaz a ani terminál nevypíname.

Potom na klientovi otvoríme prehliadač Iceweasel a zadáme do adresného riadku stránku „https://192.168.0.5:8443/bWAPP/portal.php“. Po potvrdení bezpečnostnej výnimky u certifikátu sa prihlásime na stránke pod účtom: „bee“ a heslom: „bug“. Po prihlásení potvrdíme, aby si prehliadač zapamätal prihlasovacie údaje. Potom, v druhej záložke otvoríme stránku „192.168.0.5:8000“. Potom sa už len prepneme na útočníka a čakáme, kým nám skript nevyhodí celé cookie (obr. 5.4).

Po prepnutí na monitorovaciu stanicu, nájdeme v adresári, ktorý sme zvolili vo výzve programu, log súbor. Po jeho otvorení môžeme vidieť, že bol útok zaznamenaný.

5.2.5 Praktická realizácia útoku BEAST

Na realizáciu tohto útoku som našiel skript Beast-POC [30]. Pokúšal som sa tento skript sprovozniť na experimentálnom pracovisku č.2, pretože som vedel, že útok BEAST má veľmi podobnú konfiguráciu ako útok POODLE. Bohužiaľ sa mi tento



Obr. 5.4: Získané cookie.

útok nepodarilo nasimulovať pomocou tohto skriptu a k nemu priloženého popisu. V skripte správne nefungoval javascriptový kód, ktorý mal na strane klienta pozmeňovať správy. Pokúsil som sa aj tento kód upraviť (keďže mi to dovoľuje licencia, pod ktorou je tento kód licencovaný), aby som dosiahol tohto útoku, no bohužiaľ sa mi to nepodarilo. Keďže sa jedná už o starší útok, ďalšie užitočné zdroje, na nasimulovanie tohto útoku som nenašiel.

6 PROGRAM NA ZACHYTENIE ÚTOKU HEART-BLEED, POODLE A BEAST

Ďalším bodom mojej bakalárskej práce je napísanie programu na detekciu útokov HeartBleed, POODLE a BEAST. Tento program som nazval HPB_Detective. HPB_Detective som vytvoril v programovacom jazyku python2, podľa zadania, a ako vývojové prostredie som použil program PyCharm. HPB_Detective funguje v operačných systémoch Linux.

Môj program je zameraný na operačné systémy Linux preto, lebo sú najrozšírenejšie operačné systémy u serverov. Práve preto sú často vedené útoky na túto platformu.

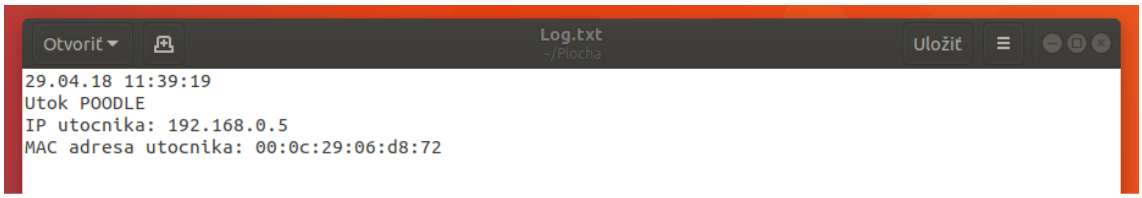
Program sa dá použiť priamo na servery, na ktorom chceme detekovať útoky, alebo sa dá použiť na monitorovacej stanici, ktorú vložíme do nami zvolenej siete, tak ako to robím ja v mojom experimentálnom pracovisku. Program monitoruje sieťový provoz a v reálnom čase vyhodnocuje, či nenastal niektorý z daných útokov.

Program je treba spustiť s administrátorskými právami, príkazom „sudo“. Po spustení programu zadáme IP adresu rozhrania (interface), na ktorom chceme útoky detekovať.

Ďalej zadávame cestu k súboru a jeho názov, v ktorom sa nám budú ukladať prípadné detekované útoky.

Do súboru sa vždy uloží (Obr. 6.1):

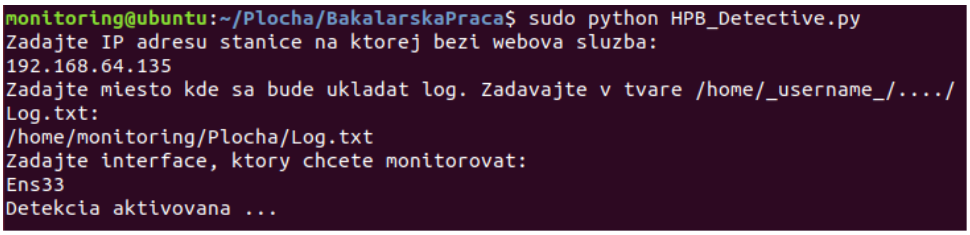
- čas, v ktorom bol zaznamenaný útok,
- druh útoku (HeartBleed, POODLE alebo BEAST),
- IP adresa útočníka,
- MAC adresa útočníka.



Obr. 6.1: Záznam v súbore.

Naposledy zadáme názov rozhrania (interface) (Obr. 6.2).

Ak chceme program vypnúť stačí zadať „q“ a potvrdiť enterom. Ukončenie programu pomocou „ctrl+c“ neodporúčam, pretože program vypíše do terminálu veľa chybových hlásení kvôli nedokončeniu krokov v bežiacej funkcii.



Obr. 6.2: Zadávanie parametrov do programu.

6.1 Popis programu

V programe používam tieto knižnice :

- pyshark - táto knižnica je použitá na monitorovanie sieťového provozu po paketoch,
- datatime - je použitá na zistenie dátumu a času,
- timeit - je použitá na stopovanie času,
- sys - je použitá na ukončenie programu.

Program monitoruje celý sieťový provoz po jednotlivých paketoch na zvolenom rozhraní a zameriava sa len na pakety, ktoré majú SSL/TLS vrstvu. Ostatné pakety nás pri detekovaní útokov nezaujímajú. Ďalej sa program delí na 3 časti:

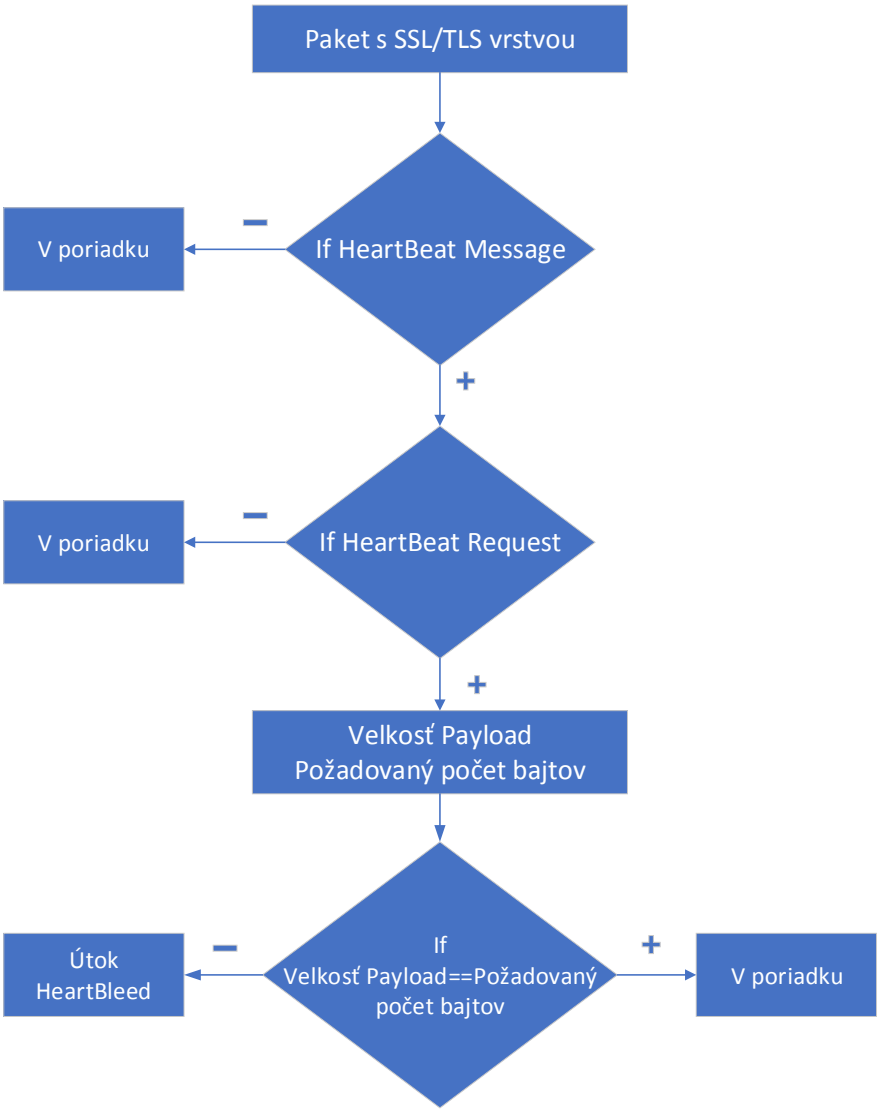
- HeartBleed,
- POODLE,
- BEAST.

Jednotlivé časti zaznamenávajú rovnako pomenované útoky.

6.1.1 Časť HeartBleed

Princíp tohto útoku spočíva v tom, že útočník pošle správu HeartBeatRequest s minimálnou veľkosťou pola payload a požaduje od príjemcu odpoveď s rozdielnou, väčšinou čo najväčšou veľkosťou pola payload (popis v časti 2).

V časti HeartBleed program porovnáva, či je paket s SSL/TLS vrstvou typu HeartBeat správa (HeartBeat Message). Ak nie, daným paketom sa už v tejto časti nezaoberá. Ak áno, porovná, či je HeartBeat Message typu HeartBeatRequest. Ak nie, zase sa týmto paketom nezaoberá. Ak áno, zaznamená si akú veľkosť má payload, ktoré poslal odosielateľ a koľko bajtov od príjemcu požaduje. Porovná tieto dve hodnoty. Ak sa hodnoty rovnajú všetko je v poriadku. Ak nie jedná sa o útok a ten zaznamená do súboru, ktorý sme zadali pri spustení programu (Obr. 6.3).



Obr. 6.3: Schéma časti HeartBleed.

6.1.2 Časť POODLE

Princíp tohto útoku spočíva v tom, že sa útočník snaží zhodnúť posledný bajt výplne a posledný bajt tajomstva, ktoré chce zistiť (popis v časti 3). Keďže sa mu toto podarí s pravdepodobnosťou 0,39 %, teda raz z 256 pokusov, vzniká veľa správ typu Alert Message. Tieto správy vznikajú aj pri zlom prenose, keď nastane chybovosť, ale nie v takom veľkom množstve ako pri útoku. A to je práve to, čo využívam na detekovanie útoku POODLE.

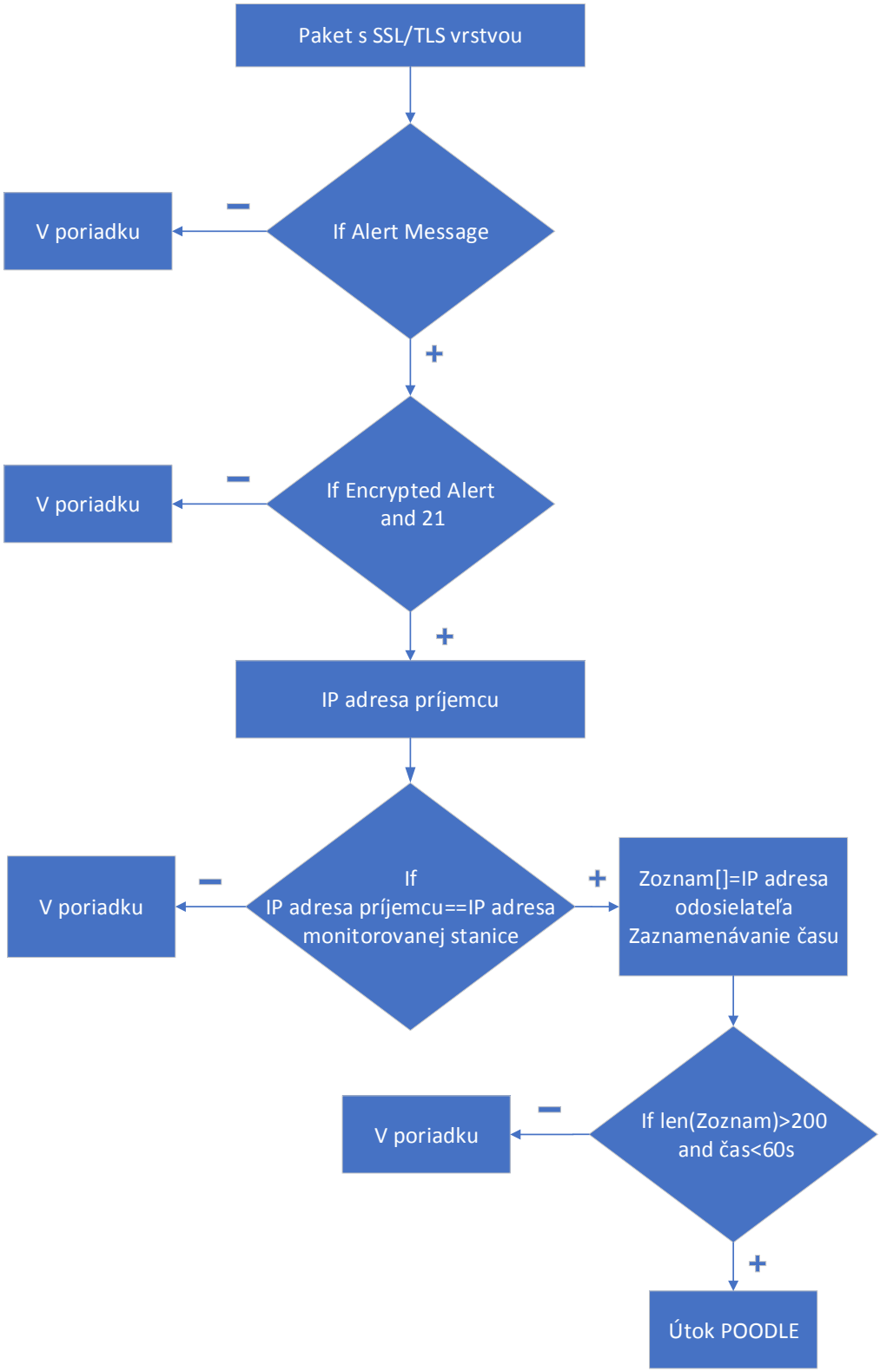
V časti POODLE program porovnáva, či je paket s SSL/TLS vrstvou typu Alert Message. Ak nie, nezaoberá sa týmto paketom. Ak áno, porovná, či je výstražná správa typu Encrypted Alert a kód chyby 21. Ak nie, tak paket nieje naším zá-

ujmom. Ak áno, zaznamená si IP adresu príjemcu paketu. To je podstatné, pretože ak nastane Alert Message príjemca odošle odosielateľovi tiež túto správu. Potom porovná, či sa IP adresa rovná s IP adresou monitorovanej stanice. Tento krok je preto, aby sme milne neoznačili monitorovanú stanicu za útočníka. Ak sa IP adresy rovnajú, pridá IP adresu odosielateľa do zoznamu a začne zaznamenávať čas. Takto sleduje aj ďalšie pakety, dokým nebude mať zoznam 200 záznamov. 200 záznamov som určil podľa sledovania. Je to množstvo, ktoré stíha poslať útočník do minúty a zároveň nemôže vzniknúť také množstvo záznamov pri chybovosti do minúty v bežnom prostredí. Keď tak nastane, prestane so záznamom času. Ak je zaznamenaný čas pod jednu minútu jedná sa o útok a program zaznamená útok do súboru, ktorý sme zadali pri spustení programu (Obr. 6.4).

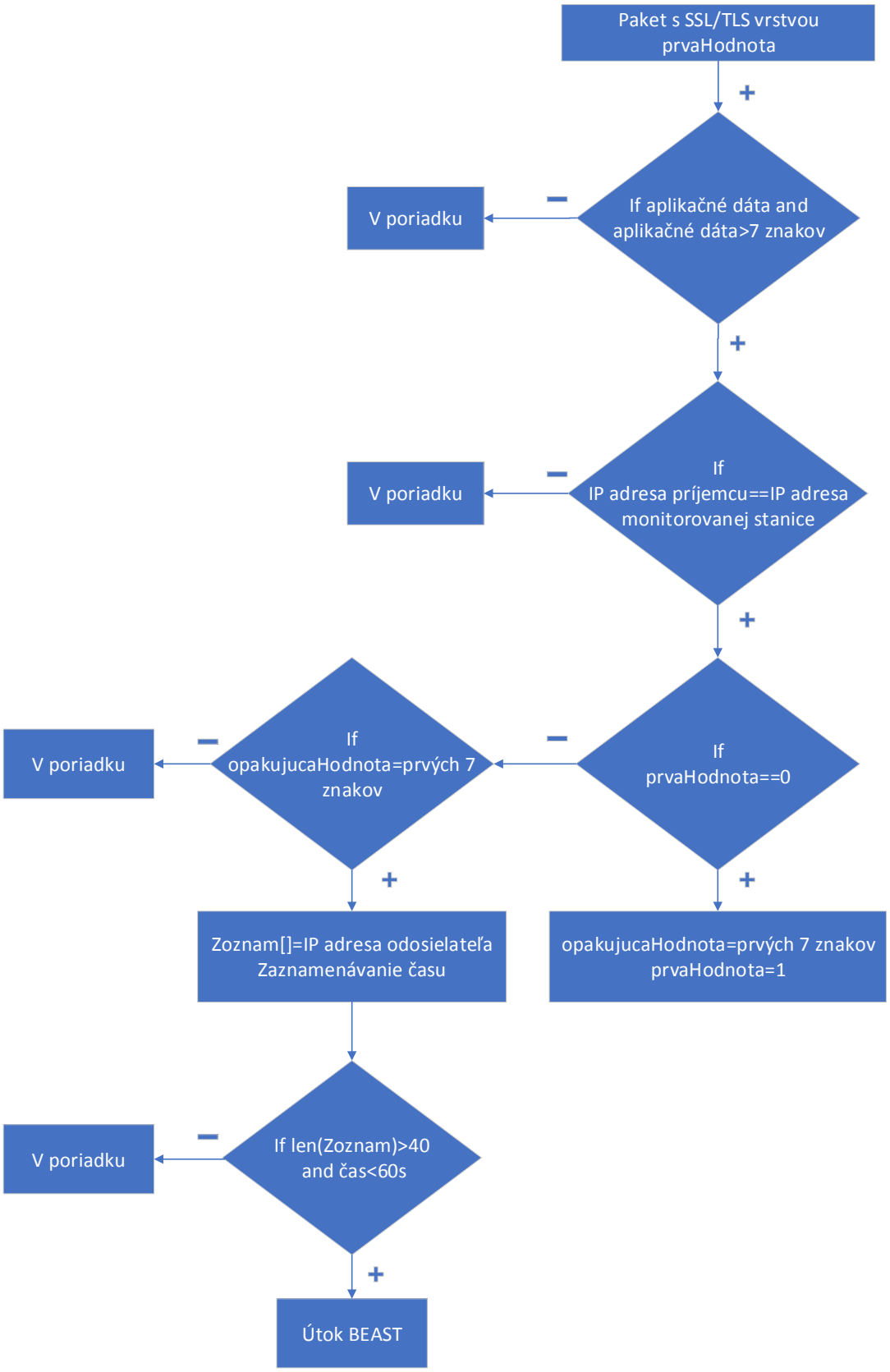
6.1.3 Časť BEAST

Princíp tohto útoku spočíva v tom, že útočník sa snaží trafiť posledný znak z bloku. To sa mu podarí s pravdepodobnosťou 0,39 %, teda raz z 256 pokusov. Správy, ktoré posiela útočník sa teda skladajú z jedného náhodného znaku a zvyšok je stále rovnaký. Minimálny blok, ktorý má bloková šifra v TLS verzie 1.0 a SSL 3.0 až nižšie má 8 znakov. Takže v tomto prípade útočník posiela 7 stále rovnakých znakov a jeden náhodný, dokiaľ netrafí ten správny (popis v časti 4). Túto znalosť využívam na detekovanie útoku.

V časti BEAST má program na začiatku premennú prvaHodnota = 0. Do časti BEAST potom vstupujú pakety. Program porovnáva, či má paket s SSL/TLS vrstvou aplikačné dáta, ktoré musia byť zároveň väčšie ako 7 znakov. Ak nie, paket ho nezaujíma. Ak áno, porovná, či je IP adresa príjemcu rovnaká ako IP adresa monitorovanej stanice. Ak áno, porovná, či sa premenná prvaHodnota rovná nule. Ak áno, uloží si prvých 7 znakov v bloku do premenej opakujucaHodnota a do premenej prvaHodnota uloží číslo 1. Potom prichádzajú ďalšie pakety. Ak ďalší paket vyhovel predchádzajúcim podmienkam a premenná prvaHodnota sa nerovná nule, tak z tohto paketu potom program zoberie prvých 7 znakov v bloku a porovná ich s premenou opakujucaHodnota. Ak sa rovnajú, pridá IP adresu odosielateľa do zoznamu a začne zaznamenávať čas. Takto sleduje aj ďalšie pakety, dokým nebude mať zoznam 40 záznamov. Hodnotu 40 som určil z pravdepodobnosti. Keď tak nastane, prestane so záznamom času. Ak je zaznamenaný čas pod jednu minútu, jedná sa o útok a program ho zaznamená do súboru, ktorý sme zadali pri spustení programu (Obr. 6.5).



Obr. 6.4: Schéma časti POODLE.



Obr. 6.5: Schéma časti BEAST.

7 VIDEÁ PREZENTUJÚCE FUNKČNOSŤ ÚTOKOV NA EXPERIMENTÁLNOM PRACOVISKU

Ďalším bodom mojej bakalárskej práce je vytvoriť videá, ktoré prezentujú funkčnosť útokov na experimentálnom pracovisku. Vytvoril som 2 videá, jedno zachytávajúce útok HeartBleed, druhé zachytávajúce útok POODLE. Útok BEAST sa mi bohužial nepodarilo nasimulovať, ako popisujem v časti 5.2.5 ,takže sa mi ani nepodarilo natočiť video k tomuto útoku. Na vytvorenie videí som použil program BB FlashBack 5 [31], ktorý ponúka 30 dňovú trial verziu. Tento program slúži na natáčanie obrazovky, editovanie záznamu a následné prevedenie do videa. Videá som natáčal na experimentálnych pracoviskách, ktoré uvádzam v kapitole 5. Na videách môžeme vidieť priebeh útokov a následné zachytenie útokov programom HPB_Detective, ktorý je výstupom mojej bakalárskej práce.

7.1 Video: Útok HeartBleed

Video, ktoré som nazval „HeartBleed“ zachytáva priebeh útoku HeartBleed. Funkčnosť tohto útoku prezentujem na experimentálnom pracovisku č.1.

Vo videu najprv zobrazujem experimentálne pracovisko so všetkými stanicami. Na každej stanici môžeme vidieť jej IP adresu. Potom na monitorovacej stanici spustím program HPB_Detective a zadám potrebné parametre na jeho spustenie. Ďalej vo videu môžeme vidieť, ako na útočníkovi zadávam do okna aplikácie Metasploit príkazy:

```
use auxiliary/scanner/ssl/openssl_heartbleed
set verbose true
set rhosts 192.168.64.135
run
```

Po týchto príkazoch je vidieť, ako sa v aplikácii zobrazia ukoristené dáta. Na konci videa zobrazujem monitorovaciu stanicu kde program vytvoril log súbor, ktorý zaznamenal útok a útočníka.

7.2 Video: Útok POODLE

Video, ktoré som nazval „POODLE“ zachytáva priebeh útoku POODLE. Funkčnosť tohto útoku prezentujem na experimentálnom pracovisku č.2.

Vo videu zase najprv zobrazujem experimentálne pracovisko so všetkými stanicami. Na každej stanici môžeme vidieť jej IP adresu. Potom na monitorovacej stanici

spustím program HPB_Detective a zadám potrebné parametre na jeho spustenie. Ďalej vo videu môžeme vidieť, ako na útočníkovi zadávam v terminále v priečinku /POODLEAttack-master príkaz:

```
sudo python3 poodle.py --start-offset 430 --target-port 443 --target
-host 192.168.64.135 https://192.168.0.5:8443/bWAPP/portal.php
```

Potom je vidieť, ako na klientovi navštívim stránku z príkazu, potvrdím certifikát a prihlásim sa na danej stránke. V druhej záložke v prehliadači si otvorím stránku: http://192.168.0.5:8000. Ďalej vo videu zobrazujem monitorovaciu stanicu kde program vytvoril log súbor, ktorý zaznamenal útok a útočníka. Potom zobrazím útočníka a je vidieť, ako útočník získava po znakoch cookie z prihláseného klienta. Tu som náhľad posunul do neskoršej doby, pretože celé vypočítanie cookie trvá približne 45 min. Na konci videa je vidieť celé cookie klienta.

8 ZÁVER

Cieľom teoretickej časti mojej bakalárskej práce bolo analyzovať známe útoky HeartBleed, POODLE a BEAST na protokol TLS a detailne popísať využitie zraniteľností pri týchto útokoch. Ďalším cieľom bolo popísanie spôsobu obrany proti týmto útokom. V teoretickej časti som podrobne naštudoval a spracoval popis TLS prokolu, pretože si myslím, že je to dôležité na pochopenie útokov HeartBleed, POODLE a BEAST. Ďalej v tejto časti podrobne popisujem tieto útoky a princípy, na ktorých fungujú. Potom vo svojej bakalárskej práci popisujem spôsob obrany pred týmito útokmi.

Cieľom praktickej časti mojej bakalárskej práce bolo navrhnutie a implementácia nástroja v programovacom jazyku Python pre detekciu týchto útokov a vyrobenie videa prezentujúce funkčnosť útokov na experimentálnom pracovisku. V praktickej časti som navrhol riešenie ako detekovať tieto útoky podľa vedomostí popísaných v teoretickej časti. Následne som podľa tohto návrhu naprogramoval funkčný program, ktorý detekuje všetky tri útoky, čo som aj následne potvrdil vo videách. Ďalej som si vyrobil dve funkčné experimentálne pracoviská pre zprovoznenie a prezentáciu vyššie zmienených útokov. Ďalej som vyrobil dve videá zachytávajúce funkčnosť útokov HeartBleed a POODLE, a ich následné detekovanie. Video zachytávajúce funkčnosť útoku BEAST som nevyrobil preto, lebo sa mi nepodarilo zprovozniť tento útok z dôvodov, ktoré popisujem v časti 5.2.5.

Na základe vyššie zmieneného môžem konštatovať, že som splnil všetky ciele bakalárskej práce, až na časť jedného bodu v praktickej časti.

LITERATÚRA

[1] *The Transport Layer Security (TLS) Protocol Version 1.2. Https://tools.ietf.org* [online]. [cit. 2017-12-14]. Dostupné z: <https://tools.ietf.org/html/rfc5246

[2] Libor DOSTÁLEK a Alena KABELOVÁ. *Velký průvodce protokoly TCP/IP a systémem DNS* [online]. Praha: Computer Press, 2000 [cit. 2017-12-14]. ISBN 80-7226-323-4. Dostupné z: <http://download.matus.in/it/Velky%20pruvodce%20protokoly%20TCP-IP%20a%20systemem%20DNS.pdf

[3] Doc. Ing. Karel Burda, CSc. *Bezpečnost informačních systémů*. Brno, 2013. Skripta. Vysoké učení technické v Brně. ISBN 978-80-214-4890-2.

[4] *SSL a TLS: Úvod do SSL a TLS. Http://zive.cz* [online]. [cit. 2017-12-14]. Dostupné z: <http://programovani.blog.zive.cz/2010/11/ssl-a-tls-uvod-do-ssl-a-tls/

[5] *SSL protokol (8) - příklady použití a jednotlivé verze protokolu. Http://www.svetsiti.cz* [online]. [cit. 2017-12-14]. Dostupné z: <http://www.svetsiti.cz/clanek.asp?cid=SSL-protokol-8-priklady-pouziti-a-jednotlive-verze-protokolu-3052002

[6] Pavel SOUKUP. *Nástroj na testování zranitelností na běžné síťové útoky* [online]. Praha, 2016 [cit. 2017-12-14]. Dostupné z: <https://dspace.cvut.cz/bitstream/handle/10467/65149/F8-DP-2016-Soukup-Pavel-thesis.pdf Diplomová práce. ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE. Vedoucí práce Ing. Tomáš Zahradnický, Ph.D.

[7] *BEAST Poc. https://mpgn.fr* [online]. 2015, 3.4.2015 [cit. 2018-05-12]. Dostupné z: <https://mpgn.fr/security/BEAST-PoC/

[8] DUONG, Thai a Juliano RIZZO. *BEAST: A Surprising Crypto Attack Against HTTPS* [online]. 12.1.2012, s. 27 [cit. 2018-05-18]. Dostupné z: <http://antoanthongtin.vn/Portals/0/TempUpload/pProceedings/2014/9/26/tetcon2012_juliano_beast.pdf

[9] DUONG, Thai a Juliano RIZZO. *Here Come The Ninjas* [online]. 13.5.2011, s. 10 [cit. 2018-05-18]. Dostupné z: <http://netifera.com/research/beast/beast_DRAFT_0621.pdf

[10] *Je komunikace přes HTTPS bezpečná? http://www.cleverandsmart.cz* [online]. [cit. 2017-12-14]. Dostupné z: <http://www.cleverandsmart.cz/lze-se-spolehnout-na-https/

[11] *BEAST útok na SSL v3.0 a TLS v1.0. Https://nethemba.com* [online]. [cit. 2017-12-14]. Dostupné z: <https://nethemba.com/cs/beast-utok-na-ssl-v3-0-a-tls-v1-0/

[12] *A diversion: BEAST Attack on TLS/SSL Encryption. Https://blog.cryptographyengineering.com* [online]. [cit. 2017-12-14]. Dostupné z: <https://blog.cryptographyengineering.com/2011/09/21/brief-diversion-beast-attack-on-tlsssl/

[13] *SSL v ohrožení: komunikaci je možné dešifrovat. Https://www.root.cz* [online]. [cit. 2017-12-14]. Dostupné z: <https://www.root.cz/clanky/ssl-v-ohrozeni-komunikaci-je-mozne-desifrovat/

[14] Martin Šindler *LABORATORNÍ ÚLOHA ÚTOKŮ NA PROTOKOL HTTPS* [online]. Brno, 2017 [cit. 2017-12-14]. Dostupné z: <https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=144942. DIPLOMOVÁ PRÁCE. VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ. Vedoucí práce Doc. Ing. Karel Burda, CSc.

[15] MÖLLER, Bodo, Thai DUONG a Krzysztof KOTOWICZ. *This POODLE Bites: Exploiting The SSL 3.0 Fallback* [online]. 2014, s. 4 [cit. 2018-05-19]. Dostupné z: <https://www.openssl.org/~bodo/ssl-poodle.pdf

[16] *Everything you need to know about the POODLE SSL bug* [online]. 21.10.2014 [cit. 2018-05-19]. Dostupné z: <https://www.troyhunt.com/everything-you-need-to-know-about/

[17] Alert (TA14-290A) SSL 3.0 Protocol Vulnerability and POODLE Attack. *Https://www.us-cert.gov* [online]. 17.10.2014 [cit. 2018-05-19]. Dostupné z: <https://www.us-cert.gov/ncas/alerts/TA14-290A

[18] *Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) Heartbeat Extension. Https://tools.ietf.org* [online]. [cit. 2017-12-14]. Dostupné z: <https://tools.ietf.org/html/rfc6520

[19] *Princíp zraniteľnosti Heartbleed v OpenSSL. https://citadelo.com* [online]. [cit. 2017-12-14]. Dostupné z: <https://citadelo.com/cs/2014/04/princip-zranitelnosti-heartbleed-v-openssl/

[20] *The Heartbleed Bug* [online]. 29.4.2014 [cit. 2018-05-19]. Dostupné z: <http://heartbleed.com/

[21] RAGAN, Steve. *Heartbleed (CVE-2014-0160): An overview of the problem and the resources needed to fix it* [online]. 11.4.2014 [cit. 2018-05-19]. Dostupné z: <https://www.csoonline.com/article/2142700/vulnerabilities/vulnerabilities-heartbleed-cve-2014-0160-an-overview-of-the-problem-and-the-resources-needed-to.html

[22] FRUHLINGER, Josh. *What is the Heartbleed bug, how does it work and how was it fixed?* [online]. 13.9.2017 [cit. 2018-05-19]. Dostupné z: <https://www.csoonline.com/article/3223203/vulnerabilities/what-is-the-heartbleed-bug-how-does-it-work-and-how-was-it-fixed.html

[23] *The Apache HTTP Server Project* [online]. [cit. 2018-05-19]. Dostupné z: <https://httpd.apache.org/download.cgi

[24] KimiNewt/pyshark. *Github* [online]. [cit. 2018-05-19]. Dostupné z: <https://github.com/KimiNewt/pyshark

[25] *Iptables forward all traffic to interface* [online]. [cit. 2018-05-19]. Dostupné z: <https://unix.stackexchange.com/questions/126595/iptables-forward-all-traffic-to-interface

[26] Alexandre BORGES. *How to perform a Heartbleed Attack* [online]. , 10 [cit. 2017-12-14]. Dostupné z: https://alexandreborgesbrazil.files.wordpress.com/2014/04/hearbleed_attack_version_a_1.pdf

[27] *Detecting and Exploiting the OpenSSL-Heartbleed Vulnerability. Https://hakin9.org* [online]. [cit. 2017-12-14]. Dostupné z: <https://hakin9.org/detecting-and-exploiting-the-openssl-heartbleed-vulnerability/

[28] *BWAPP an extremely buggy web app !* [online]. [cit. 2018-05-19]. Dostupné z: <http://www.itsecgames.com/

[29] Mpgn/poodle-PoC. *Github* [online]. [cit. 2018-05-19]. Dostupné z: <https://github.com/mpgn/poodle-PoC

[30] Mpgn/BEAST-PoC. *Github* [online]. [cit. 2018-05-19]. Dostupné z: <https://github.com/mpgn/BEAST-PoC

[31] *Flashback* [online]. [cit. 2018-05-19]. Dostupné z: <https://www.flashbackrecorder.com/

ZOZNAM SYMBOLOV, VELIČÍN A SKRATIEK

3DES	Triple Data Encryption Algorithm
AES	Advanced Encryption Standard
BEAST	Browser Exploit Against SSL/TLS
c_{i}	šifrovaný blok s indexom poradia
C_{i}	šifrovaná správa s indexom poradia vzniknutá z M_1
CBC	Cipher Block Chaining
CC	Client Certificate
CCR	Client Certificate Request
CCSP	Change Cipher Spec Protocol
CFM	Client Finished Message
CH	Client Hello
CV	Certificate Verify
D_{k}	dešifrovanie s kľúčom k
DES	Data Encryption Algorithm
E	šifrovanie
E_{k}	šifrovanie s kľúčom k
F_{i}	fragment s indexom poradia
HMAC	Keyed-hash Message Authentication Code
HTTP	Hyper Text Transfer Protocol
HTTPS	Secured Hyper Text Transfer Protocol
i	index poradia
IP	Internet Protocol
IV	inicializačný vektor
k	kľúč
K_{E}	šifrovací kľúč
$K_{\mathrm{E(K\rightarrow S)}}$	kryptografický kľúč pre smer od klienta k serveru
$K_{\mathrm{E(S\rightarrow K)}}$	kryptografický kľúč pre smer od serveru ku klientovi
K_{HMAC}	HMAC kľúč
K_{PS}	verejný kľúč servera
KK	kryptografické kombinácie
KK_{S}	kryptografická kombinácia serveru
m_{i}	blok správy s indexom poradia pred šifrovaním
M_{i}	správa s indexom poradia pred šifrovaním
M_{r}	náhodná správa
MAC	Media Access Control
MS	Master Secret
PMS	PreMaster Secret

POODLE	Padding Oracle On Downgraded Legacy Encryption
R	všetky predchádzajúce poslané správy
RFC	Request For Comment
SC	Server Certificate
SFM	Server Finished Message
SH	Server Hello
SSL	Protokol Secure Socket Layer
TCP	Transmittion Control Protocol
TLS	Transport Layer Security
U_{K}	unikát klienta
U_{S}	unikát serveru
UDP	User Datagram Protocol
V	výplň
V_{HMAC}	výstup funkcie HMAC
V_{X}	posledný bajt výplne
XOR	Exlusive OR

ZOZNAM PRÍLOH

A Obsah priloženého CD **48**

A OBSAH PRILOŽENÉHO CD

Priložené CD obsahuje hlavné súboru tejto práce.

Priečinok Program obsahuje program HPB_Detective, ktorý slúži na detekovanie útokov HeartBleed, POODLE a BEAST, a ktorý je výstupom mojej bakalárskej práce.

Priečinok Videá obsahuje dve videá: HeartBleed a POODLE, ktoré zachytávajú funkčnosť útokov HeartBleed a POODLE na experimentálnych pracoviskách a tiež sú výstupom mojej práce.

```
/ ..... koreňový priečinok priloženého CD
├── Program ..... priečinok s programom
│   └── HPB_Detective.py
└── Videá ..... priečinok s Videami
    ├── HeartBleed.mp4
    └── POODLE.mp4
```