Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

FIIT-5220-8439

Bc. Tomáš Morvay

# Bezpečnosť domácej siete

*Diplomová práca*

Vedúci práce: doc. Ing. Ladislav Hudec, CSc.

Máj 2016

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

FIIT-5220-8439

Bc. Tomáš Morvay

# Bezpečnosť domácej siete

*Diplomová práca*

Študijný program: Softvérové inžinierstvo
Študijný odbor: 9.2.5 Softvérové inžinierstvo
Miesto vypracovania: Ústav počítačového inžinierstva a aplikovanej informatiky, FIIT STU Bratislava
Vedúci práce: doc. Ing. Ladislav Hudec, CSc.

Máj 2016

STU FIIT
SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ

V Bratislave, 15.2.2016

**Vec: Oznámenie o vypracovaní diplomovej práce**

Vážený pán Bc. Tomáš Morvay,

v súvislosti s tým, že ste ako študent študijného programu Počítačové a komunikačné systémy a siete prestúpili na študijný program Softvérové inžinierstvo, Vám oznamujem, že podľa podmienok prestupu pokračujete vo vypracúvaní diplomovej práce podľa pôvodného zadania, t.j. s nezmeneným názvom, opisom, vedúcim a termínom odovzdania. Diplomovú prácu budete obhajovať v študijnom odbore Softvérové inžinierstvo.

S pozdravom

prof. Ing. Mária Bieliková, PhD.
dekanka FIIT STU

Ilkovičova 2
842 16 Bratislava
www.stuba.sk

maria.bielikova@stuba.sk
T: +421 2 210 22 204
F: +421 2 654 20 587

# Zadanie diplomovej práce

*Meno študenta:* **Bc. Tomáš Morvay**

*Študijný program:* Počítačové a komunikačné systémy a siete

*Študijný odbor:* Počítačové inžinierstvo

*Názov práce:* **Bezpečnosť domácej siete**

Samostatnou výskumnou a vývojovou činnosťou v rámci predmetov Diplomový projekt I, II, III vypracujte diplomovú prácu na tému, vyjadrenú vyššie uvedeným názvom tak, aby ste dosiahli tieto ciele:

*Všeobecný cieľ:*

Vypracovaním diplomovej práce preukážte, ako ste si osvojili metódy a postupy riešenia relatívne rozsiahlych projektov, schopnosť samostatne a tvorivo riešiť zložité úlohy aj výskumného charakteru v súlade so súčasnými metódami a postupmi študovaného odboru využívanými v príslušnej oblasti a schopnosť samostatne, tvorivo a kriticky pristupovať k analýze možných riešení a k tvorbe modelov.

*Špecifický cieľ:*

Vytvorte riešenie, zodpovedúce návrhu textu zadania, ktorý je prílohou tohto zadania. Návrh bližšie opisuje tému vyjadrenú názvom. Tento opis je záväzný, má však rámcový charakter, aby vznikol dostatočný priestor pre Vašu tvorivosť.

Riaďte sa pokynmi Vášho vedúceho.

Pokiaľ v priebehu riešenia, opierajúc sa o hlbšie poznanie súčasného stavu v príslušnej oblasti alebo o priebežné výsledky Vášho riešenia alebo o iné závažné skutočnosti, dospejete spoločne s Vaším vedúcim k presvedčeniu, že niečo v texte zadania a/alebo v názve by sa malo zmeniť, navrhnite zmenu. Zmena je spravidla možná len pri dosiahnutí kontrolného bodu.

*Miesto vypracovania:* Ústav aplikovanej informatiky, FIIT STU Bratislava

*Vedúci práce:* **doc. Ing. Ladislav Hudec, CSc.**

*Termíny odovzdania:*

podľa harmonogramu štúdia platného pre semester, v ktorom máte príslušný predmet (Diplomový projekt I, II, III) absolvovať podľa Vášho študijného plánu

*Predmety odovzdania:*

V každom predmete dokument podľa pokynov na www.fiit.stuba.sk v časti:
home > Informácie o > štúdiu > organizácia štúdia > diplomový projekt

V Bratislave dňa 16. februára 2015

Ing. Katarína Jelemenská, PhD.
poverená vedením ústavu

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
Fakulta informatiky a informačných technológií

STU FIIT
SLOVENSKÁ TECHNICKÁ
UNIVERZITA V BRATISLAVE
FAKULTA INFORMATIKY
A INFORMAČNÝCH TECHNOLÓGIÍ

# Návrh zadania diplomovej práce

*Finálna verzia do diplomovej práce* [1]

## Študent:

**Meno, priezvisko, tituly:** Tomáš Morvay, Bc.
**Študijný program:** Počítačové a komunikačné systémy a siete
**Kontakt:** tomasmorvay@gmail.com

## Výskumník:

**Meno, priezvisko, tituly:** Ladislav Hudec, doc. Ing. CSc.

## Projekt:

**Názov:** Bezpečnosť domácej siete
**Názov v angličtine:** Home network security
**Miesto vypracovania:** Ústav aplikovanej informatiky, FIIT STU, Bratislava
**Oblasť problematiky:** Počítačová bezpečnosť

### Text návrhu zadania[2]

Súčasné technológie pripojenia koncových používateľov k sieťam poskytovateľa internetových služieb ponúkajú bezpečnosť na nízkej úrovni. Bezpečnostné mechanizmy na ochranu koncového používateľa je možné implementovať na strane poskytovateľa alebo na strane koncového používateľa. Najlepší prípad je, keď je bezpečnosť implementovaná na oboch stranách (hĺbková obrana). Téma diplomovej práce je zameraná na bezpečnosť domácich sietí koncových používateľov. Väčšina domácich používateľov pripojených do Internetu je súčasťou siete tvorenej jedným kombinovaným Wi-Fi smerovačom. Aj keď domáce smerovače poskytujú istú úroveň zabezpečenia (napr. NAT v smerovači), tieto bezpečnostné prvky však stále neposkytujú dostatočnú úroveň zabezpečenia, čo je motiváciou tejto diplomovej práce. Analyzujte existujúce bezpečnostné mechanizmy súčasných domácich smerovačov. Podrobne analyzujte metódy detekcie prienikov v sieťach. Navrhnite a implementujte model pre sledovanie štatistických parametrov sieťovej komunikácie v domácej sieti a mechanizmus detekcie prienikov v rámci domácej siete. Pri ochrane domácich smerovačov zvážte distribúciu funkcií bezpečnostnej brány. Zber údajov a pamäťové nároky na udržiavanie štatistík prispôsobte schopnostiam domácich smerovačov. Navrhnuté riešenie integrujte do voľne šíriteľného firmvéru pre domáce smerovače (napr. OpenWrt) a overte základnú funkcionalitu. Výsledné riešenie otestujte v reálnych podmienkach. Vyhodnoťte kvalitu štatistických údajov poskytnutých vytvoreným modelom.

---

[1] Vytlačiť obojstranne na jeden list papiera

[2] 150-200 slov (1200-1700 znakov), ktoré opisujú výskumný problém v kontexte súčasného stavu vrátane

**Literatúra**[3]


- STRAND, L. K.: Adaptive distributed firewall using intrusion detection, University of Oslo, Department of Informatics, 2004, http://urn.nb.no/URN:NBN:no-10126 [24.11.2014].
- Kumar, G., Kumar, K., Sachdeva, M.: The use of artificial intelligence based techniques for intrusion detection: a review. In: Artif Intell Rev, vol. 34, issue 4, 2010, pp. 369–387.


Vyššie je uvedený návrh diplomového projektu, ktorý vypracoval(a) Bc. Tomáš Morvay, konzultoval(a) a osvojil(a) si ho doc. Ing. Ladislav Hudec, CSc. a súhlasí, že bude takýto projekt viesť v prípade, že bude pridelený tomuto študentovi.

V Bratislave dňa 8.2.2015

Podpis študenta

Podpis výskumníka

**Vyjadrenie garanta predmetov Diplomový projekt I, II, III**

Návrh zadania schválený: áno / nie[4]

Dňa: 27. 2. 2015

Podpis garanta predmetov

[3] 2 vedecké zdroje, každý v samostatnej rubrike a s údajmi zodpovedajúcimi bibliografickým odkazom podľa normy STN ISO 690, ktoré sa viažu k téme zadania a preukazujú výskumnú povahu problému a jeho aktuálnosť (uveďte všetky potrebné údaje na identifikáciu zdroja, pričom uprednostnite vedecké príspevky v časopisoch a medzinárodných konferenciách)

# Anotácia

Slovenská technická univerzita v Bratislave
FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ

Študijný program: Softvérové inžinierstvo
Autor: Bc. Tomáš Morvay
Diplomová práca: BEZPEČNOSŤ DOMÁCEJ SIETE
Vedúci práce: doc. Ing. Ladislav Hudec, CSc.

Máj 2016

Predkladaný diplomový projekt analyzuje bezpečnosť domácej siete, ako i domáci smerovač v úlohe centrálneho prístupového bodu domácnosti do Internetu. V rámci analýzy bezpečnosti je uvedená definícia počítačovej bezpečnosti a jej základné vlastnosti. Ďalej je v projekte opísaná detekcia prienikov a kategorizácia systémov pre detekciu a prevenciu prienikov. Predstavené sú vybrané existujúce riešenia pre danú problematiku.

Cieľom tohto projektu je navrhnúť a implementovať model sieťového systému pre detekciu anomálií (NIDS), ktorý je ľahko nasaditeľný v domácej sieti. Model je založený na analýze stavových protokolov na sieťovej a transportnej vrstve. Pre každé zariadenie pripojené k sieti sa vytvorí profil. V sieťovej premávke sa následne detegujú anomálie na základe týchto profilov. Navrhovaný NIDS je integrovaný do nástroja s otvoreným zdrojovým kódom ucollect. V práci je taktiež opísaný distribuovaný prístup k vytváraniu pravidiel bezpečnostnej brány a použitie výpočtovej inteligenciu.

# Annotation

Slovak University of Technology Bratislava
FACULTY OF INFORMATICS AND INFORMATION TECHNOLOGIES

Degree Course: Software engineering
Author: Bc. Tomáš Morvay
Diploma thesis: HOME NETWORK SECURITY
Supervisor: doc. Ing. Ladislav Hudec, CSc.

May 2016

This diploma thesis analyses home network security, as well as home routers in the role of central home access point to the Internet. As part of analysis of security, the definition of computer security and its main properties are presented. Furthermore, the project describes intrusion detection and categorization of intrusion detection and prevention systems. Selection of existing solutions is presented.

The aim of this work is to design and implement a model for network intrusion detection system (NIDS), which can be easily deployed in home network. The model is based on stateful protocol analysis on network and transport layer. Profile is created for each device connected to network. Traffic is evaluated using these profiles in order to detect anomalies. Proposed NIDS is integrated into open source tool Ucollect. Furthermore, distributed approach to creation of firewall rules and employment of computational intelligence are discussed.

# Poďakovanie

Prostredníctvom týchto riadkov by som sa chcel poďakovať svojmu vedúcemu diplomovej práce Ing. Ladislavovi Hudecovi, CSc. za jeho ochotu, cenné rady a pripomienky, ktoré mi pomohli pri vypracovaní tejto práce. V neposlednom rade patrí veľká vďaka mojim rodičom, rodine a priateľom za ich neustálu podporu a pomoc.

Podpísaný Tomáš Morvay čestne vyhlasujem, že diplomovú prácu *Bezpečnosť domácej siete* som vypracoval samostatne, na základe poznatkov získaných počas štúdia a informácií z dostupnej literatúry uvedenej v práci.

Uvedenú prácu som vypracoval pod vedením Ing. Ladislava Hudeca, CSc.

V Bratislave 13.5.2016

Tomáš Morvay

# Obsah

# Zoznam obrázkov



# Zoznam tabuliek

# 1 Úvod

## 1.1 Motivácia

Väčšina koncových domácich používateľov pripojených do Internetu je súčasťou vlastnej siete tvorenej jedným kombinovaným *Wi-Fi* smerovačom s funkciou NAT a viacerými koncovými zariadeniami. Typický počet koncových zariadení má v poslednej dobe rastúci trend. Na jednu domácnosť v pripadá v Európe 10 zariadení pripojených do siete Internet. V Amerike je to 5,7 zariadení na domácnosť [**1**] [**2**]. Koncové zariadenia tvoria typicky stolné počítače, sieťové disky, multimediálne zariadenia, smartfóny, laptopy a iné zariadenia/spotrebiče schopné prístupu do Internetu.

Smerovače sú v takejto domácej sieti centrálnym prístupovým bodom do Internetu. Typicky poskytujú NAT funkcionalitu, lokálny DNS server, DHCP server ako aj funkcionalitu bezpečnostnej brány. S rastúcim počtom koncových zariadení pripojených do siete sa zlepšujú aj hardvérové schopnosti smerovačov a tiež ich softvérové schopnosti.

Tieto smerovače predstavujú ideálne miesto pre sledovanie sieťovej komunikácie koncových používateľov, vytváranie štatistík a aplikovanie rôznych práv alebo obmedzení pre koncové zariadenia [**3**] [**4**]. Sledovanie komunikácie z pohľadu domácich smerovačov má niekoľko výhod oproti sledovaniu komunikácie z pohľadu poskytovateľom internetového pripojenia (ISP) na jeho zariadeniach. Prvou je fakt, že funkcia NAT skrýva koncové zariadenia a z pohľadu ISP nie je možné jednoducho rozlíšiť jednotlivé zariadenia [**5**]. Pritom informácia o unikátnosti zdroja komunikácie a jeho type (smartfón/PC) môže byť dôležitá pri uplatňovaní QoS, profilovaní a vyhodnocovaní charakteru komunikácie jednotlivých zariadení [**6**]. Druhou výhodou je fakt, že pri rastúcom počte zariadení pripojených do domácej siete, je sledovanie unikátnych zariadení z pohľadu ISP pomerne náročné na výpočtové prostriedky.

## 1.2 Zoznam použitých skratiek

**CIA** - Confidentiality, Integrity, Availability

**COMPUSEC** - COMputer SECurity

**DDoS** - Distributed Denial of Service

**DHCP** - Dynamic Host Configuration Protocol

**DNS** - Domain Name System

**DRDoS** - Distributed Reflected Denial of Service

**DoS** - Denial of Service

**HIDS** - Host-based Intrusion Detection System

**HTTP** - HyperText Transfer Protocol

**IDPS** - Intrusion Detection and Prevention System

**IDS** - Intrusion Detection System

**IP** - Internet Protocol

**IPS** - Intrusion Prevention System

**ISP** - Internet Service Provider

**IoT** - Internet of Things

**MITM** - Man In The Middle

**NAT** - Network Address Translation

**NBA** - Network Behavior Analysis

**NIDS** - Network-based Intrusion Detection System

**NIST** - National Institute of Standards and Technology

**PC** - Personal Computer

**QoS** - Quality of Service

**SDK** – Software Development Kit

**SQL** - Structured Query Language

**SQLI** - SQL Injection

**SoC** - System on Chip

**SVM** - Support Vector Machines

**TCP** - Transmission Control Protocol

**VoIP** - Voice over IP

# 2 Bezpečnosť

## 2.1 Definícia počítačovej bezpečnosti

Čo je to **počítačová bezpečnosť**? Literatúra sa v definícii nezhoduje. *Dieter Gollman* definuje počítačovú bezpečnosť v [**7**] ako niečo, čo *„sa zaoberá prevenciou a detekciou neoprávnených akcií používateľov a počítačových systémov“*. Ministerstvo obrany Spojených štátov amerických ju v [**8**] definuje ako *„Ochrana vyplývajúca zo všetkých opatrení s cieľom odoprieť neoprávnený prístup a využívanie priateľských počítačových systémov“*. V dokumente *RFC2828* [**9**], v ktorom sa počítačová bezpečnosť nazýva aj **COMPUSEC** (z angl. ***COMPuter SECurity***), je možné nájsť ďalšiu definíciu: *„Opatrenia, ktoré implementujú a zaisťujú bezpečnostné služby v počítačovom systéme, najmä tie, ktoré zaisťujú službu riadenia prístupu*“. Inštitút *NIST* v slovníku bezpečnostných pojmov [**10**] uvádza podobnú, ale detailnejšiu definíciu: *„Opatrenia a ovládacie prvky, ktoré zaisťujú dôvernosť, integritu a dostupnosť aktív informačného systému vrátane hardvéru, softvéru, firmvéru a spracovávaných, uchovávaných a komunikovaných informácií“*. *William R. Cheswick*, spoluautor známej knihy *„Firewalls and Internet Security: Repelling the Wily Hacker*“ [**11**] má širšiu definíciu: *„Bezpečnosť je zabránenie vykonania vecí, ktoré nechcete, aby mohol na vašom počítači alebo periférnom zariadení vykonať“*.

Napriek tomu, že existuje viacero definícií počítačovej bezpečnosti, najbežnejšou je definícia obsiahnutá v dokumente *RFC2828* [**9**] uvedená vyššie (COMPUSEC).

Čo je zrejmé, je rozdiel medzi bezpečnosťou hostiteľa a sieťovou bezpečnosťou. Bezpečnosť hostiteľa zvyčajne pokrýva všetky bezpečnostné mechanizmy v rámci jedného počítačového systému. Sieťová bezpečnosť sa používa pri tých bezpečnostných mechanizmoch, ktoré sú potrebné na zaistenie zabezpečenej komunikácie medzi počítačovými systémami [**4**].

## 2.2 Bezpečnostná služba a bezpečnostné vlastnosti

Štandard RFC2828 [**9**] definuje **bezpečnostnú službu** ako *„spracovateľskú alebo komunikačnú službu, ktorá je poskytovaná systémom, aby poskytla špecifický druh ochrany prostriedkov“*. Všeobecnejšie tento pojem vymedzuje inštitút NIST v slovníku bezpečnostných pojmov [**10**] : *„Vybavenie, ktoré podporuje jeden alebo viacero bezpečnostných cieľov“*. Zjednodušene povedané, bezpečnostná služba má za úlohu chrániť istú bezpečnostnú vlastnosť systému.

Počítačová bezpečnosť je založená na troch základných vlastnostiach: dôvernosť, integrita a dostupnosť (triáda **CIA – *Confidentiality, Integrity, Availability***) [**12**]. Ďalšími bezpečnostnými vlastnosťami sú autentickosť, nepopierateľnosť a účtovateľnosť. Tieto sa však niekedy interpretujú ako súčasť integrity [**10**] [**13**]. Niektoré zdroje uvádzajú aj ďalšie vlastnosti, ako napríklad autorizáciu [**14**].

### 2.2.1 *Dôvernosť*

Armáda bola prvou oblasťou, ktorá sa zaujímala o počítačovú bezpečnosť. Vznikala potreba dôveryhodnosti a obmedzenia prístupu k citlivým údajom. Používali sa rôzne bezpečnostné mechanizmy pre skrývanie informácií a prostriedkov. Toto viedlo k mylnej informácii, že počítačová bezpečnosť je **dôvernosť** (***confidentiality***).

Dôležitou súčasťou dôvernosti je nie len ukrývanie dát, ale aj skrývanie topológie. Často je postačujúca informácia, že sa niečo deje: napríklad pasívne odpočúvanie za nepriateľskou líniou môže na základe pozorovaného zvýšenia komunikácie na vojenskej sieti viesť k záveru, že sa

pripravuje útok. V *Kahnovej* knihe *„The Breakcoders“* [**15**], ktorá sa venuje histórii kryptografie, zohrávala analýza premávky pred druhou svetovou vojnou dôležitú úlohu: *„A pretože vojenské operácie sú zvyčajne sprevádzané zvýšením komunikácie, analýzou komunikácie je možné vyvodiť bezprostrednú hrozbu takýchto operácií a to sledovaním objemu komunikácie“*. Vypĺňanie komunikácie[1] môže zabrániť tejto analýze toku informácií.

*Bishop* definuje dôvernosť ako *„ochranu vyslaných dát pred pasívnymi útokmi“* [**16**]. Štandard *X.800* [**17**] poskytuje presnejšie vymedzenie tohto pojmu: *„Vlastnosť, podľa ktorej informácie nie sú dostupné alebo odhalené neautorizovaným jednotlivcom, entitám alebo procesom“*. Ešte detailnejšia je najčastejšia definícia v publikáciách inštitútu NIST: *„Zachovanie autorizovaných obmedzení prístupu a poskytnutia informácií vrátane prostriedkov pre ochranu súkromia a majetkových informácií“* [**10**].

Šifrovanie je najčastejšie používaným mechanizmom presadenia dôvernosti [**4**].

### 2.2.2 *Integrita*

Aj keď dôvernosť zohrávala v komerčnej oblasti dôležitú úlohu (skrývanie informácií pred konkurentmi), nepredstavovala stredobod záujmu. **Integrita** (***integrity***) bola dôležitá na zabránenie neautorizovaných zmien informácií. Nech sa napríklad firma zaoberá zhromažďovaním citlivých štatistických údajov o osobách. Ak by firma odhalila nejaké údaje o istej osobe verejnosti (porušenie dôvernosti), určite by to pre firmu bola nepríjemné a mohla by si takto niekoho rozhnevať. Ak by však firma predávala štatistické údaje svojim zákazníkom, ktorý ich používajú na rôzne výpočty a údaje by náhodne chybné, mohlo by to mať zničujúce následky. Firma by mohla prísť o všetkých svojich zákazníkov z dôvodu nespoľahlivosti dát [**4**].

Aby bolo možné dôverovať dátam (alebo produktom), musí byť použitý nejaký druh kontroly integrity s cieľom predísť alebo detegovať neautorizované alebo náhodné zmeny. Integrita je dôležitým aspektom komunikácie – oba protokol TCP a IP (vo verzii 4) majú v hlavičke pole kontrolného súčtu (***checksum***) s cieľom overiť integritu paketu.

*Bishop* definuje integritu ako *„dôveryhodnosť údajov alebo zdrojov, zvyčajne v zmysle prevencie nesprávnej alebo neautorizovanej zmeny“*. Taktiež tvrdí, že integrita zahŕňa integritu dát (obsah informácií) a integritu pôvodu (zdroj údajov, často nazývaná aj autentifikácia) [**12**]. Naproti tomu štandard *X.800* tieto dva pojmy od seba oddeľuje [**17**]. Podľa slovníka bezpečnostných pojmov od inštitútu *NIST* [**10**] je integrita *„Ochrana pred nesprávnou modifikáciou alebo zničením informácie a zahŕňa zaistenie neodmietnuteľnosti a autenticity“*.

Mechanizmy integrity spadajú do dvoch tried [**12**]:

- **Mechanizmy prevencie** sa usilujú zachovať integritu dát blokovaným akýchkoľvek neautorizovaných pokusov o zmenu dát alebo akýchkoľvek pokusov o zmenu dát neoprávnenými spôsobmi. Rozdiel medzi týmito dvoma typmi pokusov je pritom dôležitý. Neautorizovaný pokus o zmenu dát nastane, keď sa používateľ pokúsi zmeniť údaje, na ktoré nemá oprávnenie pre zmenu. Pokus o zmenu dát neautorizovanými spôsobmi nastane, keď používateľ síce je autorizovaný meniť dáta istým spôsobom, ale pokúša sa ich zmeniť iným spôsobom. Majme napríklad účtovný systém bežiaci na počítači. Niekto vnikne do tohto systému a snaží sa modifikovať účtovné dáta. V tomto prípade sa neautorizovaný používateľ

[1] Vypĺňanie komunikácie (***traffic padding***) je generovanie falošnej komunikácie alebo dátových jednotiek s cieľom zamaskovať množstvo reálne odoslaných dátových jednotiek [**10**].

pokúsil narušiť integritu účtovnej databázy. Ak si však firma najme účtovníka, aby viedol účtovníctvo a účtovník sa pokúsi spreneveriť peniaze tým, že ich pošle do zámoria a skrýva transakcie, používateľ (účtovník) sa pokúsil zmeniť dáta (účtovné dáta) neautorizovaným spôsobom (presunom na účet vo švajčiarskej banke). Adekvátna autentifikácia a riadenie prístupu vo všeobecnosti zabránia vniknutiu z vonka, ale prevencia druhého typu pokusu vyžaduje odlišný prístup.

- **Mechanizmy detekcie** sa nesnažia predísť narušeniu integrity, ale jednoducho hlásia, ak integrita dát už nie je dôveryhodná. Mechanizmus detekcie môže analyzovať systémové udalosti (používateľov alebo systémové akcie) s cieľom detegovať problémy alebo (korektnejšie) môže analyzovať samotné dáta s cieľom overiť, či sú požadované alebo očakávané obmedzenia dodržané. Mechanizmy môžu hlásiť konkrétne príčiny narušenia integrity (špecifická časť súboru bola zmenená) alebo môžu jednoducho hlásiť celkové narušenie integrity.

Práca s integritou je veľmi odlišná od práce s dôvernosťou. Pri dôvernosti môžu byť dáta buď skompromitované alebo nie, ale integrita zahŕňa správnosť aj dôveryhodnosť dát. Integritu ovplyvňuje viacero faktorov: pôvod dát (ako a od koho boli získané), ako dobre boli dáta chránené pred tým, ako dorazili na aktuálne zariadenie, a ako dobre sú dáta chránené na aktuálnom zariadený. Vyhodnotenie integrity je teda často veľmi náročné, pretože sa spolieha na predpoklady o zdroji dáta a na dôveru v zdroj – dve opory zabezpečenia, ktoré sú často prehliadané [**12**].

### 2.2.3 *Dostupnosť*

**Dostupnosť** (***availability***) je podľa štandardu *X.800 „vlastnosť prístupnosti a použiteľnosti na požiadanie autorizovanou entitou.“* [**17**]. Rovnakú definíciu používa aj inštitút NIST [**10**].

Internet sa stal kritickou súčasťou obchodovania. Spoločnosti musia byť pripojené ku svojim vnútorným sieťam rovnako ako aj ku zvyšku sveta, aby sa prepojili so zákazníkmi a často aj so zamestnancami. Ak sú prostriedky spoločnosti nedostupné, zamestnanci nemusia byť schopní robiť svoju prácu a zákazníci nemusia vedieť nakupovať produkty. Preto *Bishop* v [**12**] dodáva: *„Nedostupný systém je prinajmenšom tak zlý, ako vôbec žiadny systém“*.

### 2.2.4 *Autenticita*

**Autenticita** (***authenticity***) je vlastnosť zaručujúca pravosť a umožňujúca byť overeným a dôveryhodným. Označuje dôveru v platnosť vysielania, správy alebo pôvodcu správy. Autentifikácia je proces, ktorý ju overuje [**10**].

**Autentifikácia** (***authentication***) je previazanie identity na subjekt [**16**]. Nie vždy sa jedná o používateľa (subjekt), ktorý je potrebné autentifikovať. Môže to byť aj akékoľvek periférne zariadenie. Štandard *X.800* nepoužíva výraz „autentifikácia“, ale špecifikuje dva typy [**17**]:

- **Autentifikácia klientskej entity** (***peer entity authentication***) je potvrdenie toho, že asociovaná klientska entita je tá, ktorá tvrdí že je.
- **Autentifikácia zdroja dát** (***data origin authentication***) je potvrdenie toho, že zdroj dát je ten, ktorý tvrdí že je.

### 2.2.5 *Nepopierateľnosť*

**Nepopierateľnosť** (***non-repudiation***) je záruka, že niekto nemôže poprieť niečo, čo urobil. Rozlišuje sa nepopierateľnosť zdroja (dôkaz, že správa bola v skutočnosti poslaný udávanou stranou) a nepopierateľnosť cieľa (dôkaz, že príjemca správu prijal) [**18**] [**4**].

### 2.2.6 *Účtovateľnosť*

**Účtovateľnosť** (***accountability***) je zabezpečenie toho, aby akcie entity mohli byť sledovateľné spätne vyhľadané jednoznačne pre entitu. Toto umožňuje napríklad k nepovolenej činnosti spätne priradiť používateľa, ktorý ju vykonal [**17**].

### 2.2.7 *Autorizácia*

**Autorizácia** (***authorization***) predstavuje prístupové oprávnenia udelené používateľovi alebo akt udeľovania týchto oprávnení [**10**].

## 2.3 Hrozby a útoky

**Hrozba** (***threat***) je potencionálne narušenie bezpečnosti [**17**]. Narušenie nemusí v skutočnosti nastať len preto, že je prítomná hrozba. Fakt, že narušenie môže nastať znamená, že na akcie, ktoré ho môžu spôsobiť, musí byť systém pripravený alebo pred nimi chránený. Tieto akcie sa nazývajú **útoky** (***attacks***). Tí, ktorý takéto akcie vykonajú alebo spôsobia ich vykonanie sa nazývajú **útočníci** [**12**].

### 2.3.1 *Generické hrozby*

Všeobecne je možné hrozby rozdeliť na štyri typy [**19**]:

1. **Zachytenie** (***interception***) – neoprávnený subjekt získal prístup k službe alebo dátam (k aktívam).
2. **Prerušenie** (***interruption***) – služby alebo dáta sa stanú nedostupné, nepoužiteľné, zničené alebo stratené.
3. **Modifikácia** (***modification***) – neoprávnená zmena dát alebo taká manipulácia so službou, že služba už nezachováva pôvodnú špecifikáciu.
4. **Falšovanie** (***fabrication***) – situácia, v ktorej sa neoprávnene generujú dáta alebo aktivity, ktoré by inak neexistovali.

### 2.3.2 *Model hrozieb STRIDE*

Spoločnosť *Microsoft* navrhla model pre klasifikáciu známych hrozieb s názvom *STRIDE*. Názov tvoria iniciálky jednotlivých kategórií hrozieb v anglickom jazyku. Rozdeľuje hrozby na 6 skupín podľa druhov exploitov[2], ktoré sa používajú (alebo motivácie útočníkov) [**20**] [**21**]:

1. **Falšovanie identity** (***Spoofing identity***) je zvyčajne kľúčovým rizikom pre aplikácie, ktoré majú mnoho používateľov, ale poskytujú jediný kontext vykonávania, napríklad na aplikačnej a databázovej úrovni. Dôležité je, aby sa používatelia neboli schopný stať iným používateľom alebo prevziať atribúty iného používateľa. Príkladom falšovania identity je

[2] ***Exploit*** je špeciálny program, dáta alebo sekvencia príkazov využívajúca programátorskú chybu alebo zraniteľnosť s cieľom spôsobiť neúmyselné alebo neočakávané správanie softvéru, hardvéru alebo inej, zvyčajne počítačom riadenej, elektroniky.

možnosť nelegálne získať a následné použiť autentifikačné údajov používateľa, ako sú napríklad používateľské meno a heslo.

2. **Manipulácia s dátami** (***Tampering with data***) je zlomyseľná manipulácia s dátami. Príkladom je pozmenenie komunikácie medzi dvoma počítačmi cez otvorenú sieť, ako je Internet.

3. **Popretie** (***Repudiation***) predstavuje hrozby spojené s používateľmi, ktorí popierajú vykonanie akcie, bez toho aby ostatné strany mali nejaký spôsob, ako dokázať opak. Príkladom je používateľ, ktorý by mohol vykonať nepovolenú operáciu v systéme, ktorý neumožňuje sledovať nepovolené operácie.

4. **Odhalenie informácií** (***Information disclosure***) je sprístupnenie informácií jednotlivcom, ktorí by k nim nemali mať prístup. Príkladom je schopnosť útočníka čítať údaje prenášane medzi dvomi počítačmi.

5. **Odmietnutie služby** (***Denial of service***) je hrozba, pri ktorej môže nastať nedostupnosť služby pre používateľov, ktorí sú oprávnení ju používať. Najčastejšie sa to dosiahne zahltením cieľového systému nadmerným počtom požiadaviek, ktoré systém nestíha spracovať. Príkladom je webový server, ktorý nie je zabezpečený proti útoku DoS a veľký počet požiadaviek za krátky časový interval môže spôsobiť jeho nedostupnosť alebo nepoužiteľnosť služby.

6. **Zvýšenie úrovne oprávnenia** (***Elevation of privilege***) je hrozba, pri ktorej môže neoprávnený používateľ získať privilegovaný prístup, v dôsledku čoho má prístup k celému systému. Príkladom je prípad, kedy by sa útočník mohol úspešným preniknutím cez obranu systému stať súčasťou samotného dôverovaného systému.

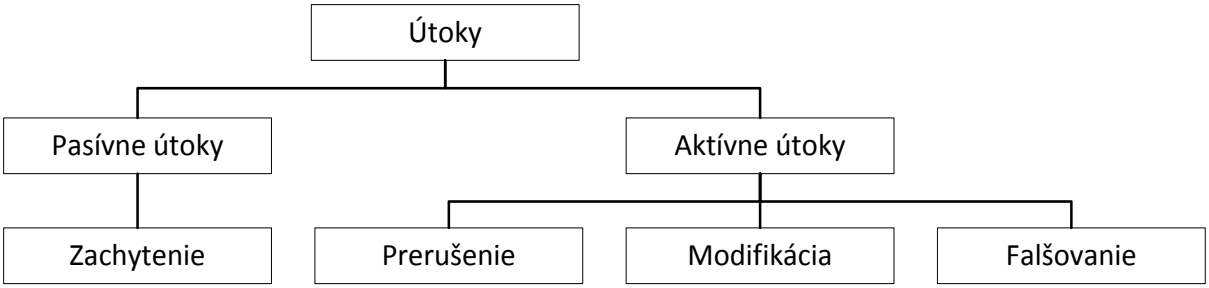
Priradenie jednotlivých typov hrozieb k bezpečnostným vlastnostiam, ktoré pred nimi chránia, obsahuje tabuľka 1.

*Tabuľka 1: Hrozby a bezpečnostné vlastnosti*

| Hrozba | Bezpečnostná vlastnosť |
|---|---|
| Falšovanie identity | Autentifikácia |
| Manipulácia s dátami | Integrita |
| Popretie | Nepopierateľnosť |
| Odhalenie informácií | Dôvernosť |
| Odmietnutie služby | Dostupnosť |
| Zvýšenie úrovne oprávnenia | Autorizácia |

### 2.3.3 *Klasifikácia útokov*

Obrázok 1 obsahuje zjednodušený diagram klasifikácie prevzatej z [**22**]. *Ahmad*, *Verma*, *Kumar* a *Shekhar* v nej rozdelili útoky na dve hlavné triedy: aktívne a pasívne. Ďalej ich členia na podtriedy podľa toho, ktorú z generických hrozieb využívajú (viď. kapitola 2.3.1 Generické hrozby). Jednotlivé triedy sú opísané nižšie.




*Obrázok 1: Klasifikácia útokov [**22**]*

### *Pasívne útoky*

**Pasívne útoky** sú také, pri ktorých je cieľom útočníkov získať informácie a pritom nechcú modifikovať obsah pôvodnej správy. Tieto útoky je ťažké detegovať, pretože dáta pri nich nie sú zmenené. Táto trieda má len jednu podtriedu – **zachytenie** (***interception***). Príkladom zachytenia môžu byť nasledovné útoky [**22**]:

- **Uvoľnenie obsahu správy** (***release of message***) – monitorovanie obsahu prenosu, ktorý je dôverný
- **Analýza premávky** (***traffic analysis***) – proces zachytávania a skúmania správ s cieľom vyvodiť informácie z pozorovaných vzorov v komunikácii.
- **Odpočúvanie** (***sniffing***) – zachytávanie a zaznamenávanie prenášaných dát bez povolenie odosielateľa.
- **Zaznamenávač kláves** (***keylogger***) – softvér alebo hardvér, ktorý zaznamenáva všetky stlačenia kláves. Zaznamenané stlačenia kláves sú skryté v zariadení pre neskoršie získanie alebo odoslané útočníkovi. Útočník v nich následne informácie, ktoré by umožnili kompromitáciu systému (napr. heslá) alebo by mohli byť použité pri útoku založenom sociálnom inžinierstve (napr. obsah e-mailu).

### *Aktívne útoky*

**Aktívne útoky** sú útoky, ktoré modifikujú pôvodnú správu alebo vytvárajú nejakú falošnú správu. Ďalej sa dajú rozdeliť do 3 podtried: **prerušenie** (***interception***), **modifikácia** (***modification***) a **falšovanie** (***fabrication***). Príklady aktívny útokov [**22**]:

- **Prerušenie**
  - **Odmietnutie služby** (***DoS – Denial of Service***) – technika útoku navrhnutá tak, aby zabraňovala autorizovanému prístupu k prostriedkom alebo oddialila časovo kritické operácie [**10**].
  - **Distribuované odmietnutie služby** (***DDoS – Distributed Denial of Service***) – technika odmietnutia služby, ktorá používa mnoho hostiteľov na vykonanie útoku [**10**].
  - **Distribuované odrazené odmietnutie služby** (***DRDoS – Distributed Reflective Denial of Service***) – špeciálny prípad DDOS, pri ktorom sa použijú falošné požiadavky odoslané v mene obete na veľké množstvo serverov. Odpoveď

serverov, ktorá je zvyčajne rádovo väčšia ako požiadavka, následne smeruje už na obeť. Najčastejšie sa na tento účel zneužívajú služby založené na protokole UDP (napr. DNS) [**23**].

- **Injekcia SQL kódu** (***SQLI - SQL injection***) – útok založený na injekcii kódu do databázovej vrstvy služby alebo programu. Škodlivý kód je „injektovaný" do reťazcov odoslaných databázovému systému na vykonanie. Týmto môže útočník zmeniť logiku, sémantiku alebo syntax parametrizovaného SQL dopytu [**24**] [**25**].

- **Modifikácia**

  - **Útok „man-in-the-middle"** (***MITM***) – voľne možno preložiť ako „človek uprostred". Ide o útok, ktorý sa pokúša zachytiť, prečítať a zmeniť informácie, pričom sa nachádza medzi používateľom verejnej siete a nejakou službou, ku ktorej pristupuje.

- **Falšovanie**

  - **Útok prehrávaním** (***replay attack***) – typ útoku, pri ktorom je platný prenos dát opakovaný alebo oneskorený so zlým úmyslom.

  - **Maškaráda** (***masquerading***) – technika útoku, pri ktorej jeden systém predstiera identitu iného. Používa sa na predstieranie autorizovanej osoby s cieľom získať prístup k dôverným informáciám nelegálnym spôsobom.

## 2.4 Bezpečnostná politika

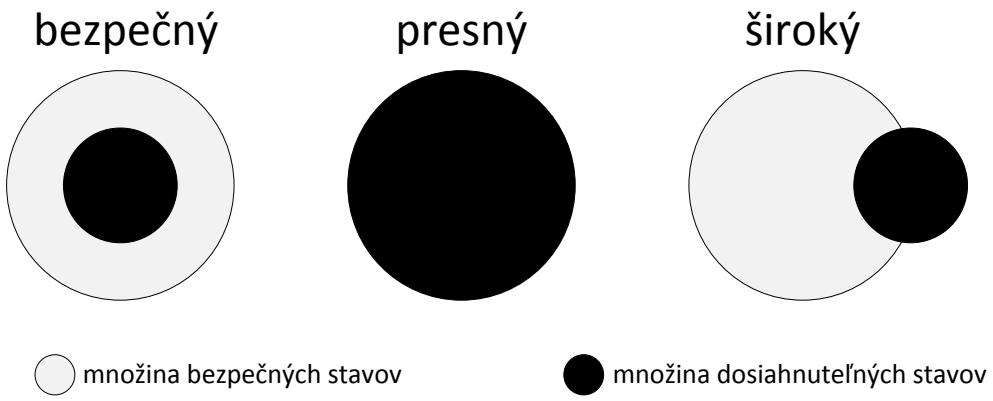
*„Bezpečnostná politika je súpis všetkého, čo je a čo nie je povolené"* uvádza *Bishop* v [**12**]. Štandard RFC2828 [**9**] obsahuje detailnejšie vymedzenie pojmu: *„Súbor pravidiel a postupov, ktoré presne určujú alebo regulujú, ako systém alebo organizácia poskytuje bezpečnostné služby, aby sa ochránili citlivé a kritické systémové prostriedky"*.

Politika môže byť vyjadrená matematickými termínmi alebo bežným jazykom. Určuje, kedy je systém v „bezpečnom" stave a čo dostane systém mimo tohto stavu. Bezpečnostná politika môže byť aj špecifickejšia: Politika dôvernosti, často nazývaná aj „politika toku informácií" je politika, ktorá sa zaoberá výmenou informácií. Najslávnejšia politika dôvernosti je *Model Bell-LaPadula*, ktorá sa riadi klasifikáciou vo vojenskom štýle. V komerčnej sfére je dôležitá politika integrity. Medzi najznámejšie politiky integrity patria Model *Biba*, *Lipnerov* model matice integrity a *Clark-Wilsonov Model Integrity* [**4**].

## 2.5 Bezpečnostný mechanizmus

**Bezpečnostný mechanizmus** je implementácia bezpečnostnej politiky. Zaisťuje dodržiavanie bezpečnostnej politiky. Podrobnejšiu definíciu je možné nájsť v štandarde RFC2828 : *„Proces (alebo zariadenie zahŕňajúce takýto proces), ktorý môže byť v systéme použitý na realizáciu bezpečnostnej služby, ktorá je poskytovaná mimo alebo vo vnútri systému"* [**9**].

Bezpečnostný mechanizmus môže, ale nie vždy musí spĺňať bezpečnostnú politiku. Mechanizmus je **bezpečný** (***secure***), ak všetky dosiahnuteľné stavy sú podmnožinou bezpečných stavov. Ak je množina dosiahnuteľných stavov rovná množine bezpečných stavov, ide o **presný** (***precise***) mechanizmus. **Široký** (***broad***) bezpečnostný mechanizmus pokrýva niektoré dosiahnuteľné stavy, ale nie všetky. Toto rozdelenie bezpečnostných mechanizmov ilustruje Obrázok 2 prevzatý od *Stranda* [**4**].




*Obrázok 2: Typy bezpečnostných mechanizmov [**4**]*

Z technického hľadiska sú často používanými bezpečnostnými mechanizmami nasledovné [**26**]:

- Fyzické zabezpečenie
- Autentifikácia
- Autorizácia
- Účtovateľnosť
- Šifrovanie
- Bezpečnostné brány
- Systémy pre detekciu a prevenciu prienikov

Bezpečnostné mechanizmy môžu mať aj netechnický charakter, napríklad požadovanie dôkazu identity pred zmenou hesla. V skutočnosti bezpečnostné politiky často požadujú určité procedurálne mechanizmy, ktoré sa len technológiou nedajú vynútiť [**12**].

# 3 Domáca sieť

Na začiatku éry využívania osobných počítačov bol jeden počítač len v málo domácnostiach bez akéhokoľvek spojenia s okolitým svetom. Dnes je bežné mať v domácnosti jeden alebo viac počítačov pripojených do Internetu. Ďalším krokom, ktorý sa už deje, je pripojenie nie jedného počítača, ale veľkej siete zariadení v domácnosti. V súčasnosti nie je nezvyčajné, ak je k domácej sieti okrem počítačov pripojených aj mnoho iných zariadení, ako sú inteligentné telefóny (smartfóny), tablety, NAS úložiská, inteligentné televízie a iné multimediálne zariadenia [**27**].

V nevzdialenej budúcnosti nám **internet vecí** (***IoT – Internet of Things***) umožní napríklad „vygoogliť naše topánky“, ak ich ráno nevieme ihneď nájsť. Niekde v sieti existuje služba, ktorá vie, kde topánky sú alebo kde ich hľadať. Táto služba musí byť nájdená potenciálnym klientom, čo je úlohou mechanizmu objavovania služieb. Je jasné že služba automatické objavovanie služieb je nevyhnutnou súčasťou dynamického prostredia, kde nie je prítomný žiadny správca. Naše domovy sú toho perfektným príkladom [**28**].

Dnes je väčšina koncových domácich používateľov pripojená do Internetu súčasťou vlastnej siete tvorenej jedným kombinovaným Wi-Fi smerovačom s funkciou NAT. Smerovač je v takejto domácej sieti centrálnym prístupovým bodom do Internetu.

## 3.1 Smerovač v domácej sieti

Smerovače pre domáce siete typicky poskytujú NAT funkcionalitu, lokálny DNS server, DHCP server, ako aj funkcionalitu bezpečnostnej brány. S rastúcim počtom koncových zariadení pripojených do siete sa zlepšujú aj hardvérové schopnosti smerovačov a tiež ich softvérové schopnosti.

Väčšinou sa jedná o riešenia využívajúce platformu *ARM*, *MIPS* alebo *Power*, zriedkavejšie sa používa architektúra *x86* [**29**]. Súčasťou smerovača môžu byť aj špecializované moduly umožňujúce **hardvérovú akceleráciu** časovo náročnejších úloh a týmto urýchliť spracovanie sieťovej premávky. V smerovačoch sa taktiež začínajú používať viacvláknové a viacjadrové procesory, ktoré môžu byť spolu so špecializovanými modulmi pre hardvérovú akceleráciu umiestnené v jednom fyzickom čipe (***System on Chip – SoC***) [**30**].

Čoraz viac modelov domácich smerovačov má predinštalovaný alebo umožňuje použiť firmvér[3] s otvoreným zdrojovým kódom na báze *OS Linux* ako napr. *OpenWrt* alebo *DD-WRT* [**31**] [**32**].

### 3.1.1 *Firmvéry pre domáce smerovače*

Firmvéry používané v smerovačoch pre domáce siete možno na základe otvorenosti zdrojového kódu rozdeliť na 2 skupiny: proprietárne firmvéry a firmvéry s otvoreným zdrojovým kódom.

Prvé smerovače určené pre domácnosti mali **proprietárne firmvéry**, pre ktoré výrobcovia zverejňovali len binárne verzie firmvéru. Bez dostupnosti zdrojových kódov a dokumentácie o architektúre firmvéru bolo vytvorenie upravených alebo rozšírených verzií firmvérov (napr. s cieľom rozšíriť funkcionalitu) veľmi náročné.

[3] **Firmvér** je softvérový program naprogramovaný do hardvérového zariadenia. Stanovuje inštrukcie, ktoré určujú ako sa zariadenie správa a ako komunikuje s okolitým hardvérom.

V roku 2002 spoločnosť *Linksys*, dcérska spoločnosť spoločnosti *Cisco*, uviedla do predaja smerovač *WRT54G* s firmvérom založeným na operačnom systéme *Linux*. Pretože je *Linux* vyvíjaný pod licenciou *GPL*, podmienky licencie zaväzovali spoločnosť *Linksys* k zverejneniu zdrojového kódu tohto firmvéru. Spoločnosť tak pod vonkajším tlakom v júli 2003 aj spravila. Toto umožnilo nezávislým vývojárom vytvoriť odvodené verzie. Takto bolo možné pridať funkcionalitu, ktorú poskytovali len drahé smerovače vyššej triedy. Postupom času začali vznikať viaceré linuxové distribúcie s otvoreným zdrojovým kódom a pribudla podpora viacerých zariadení, ktorá sa čoraz viac rozrastá [**33**].

V nasledujúcej časti sú stručne opísané vybrané **firmvéry s otvoreným zdrojovým kódom** používané v domácich smerovačoch.

### *OpenWrt*

*OpenWrt* je linuxová distribúcia určená pre vnorené systémy. Jej hlavnou črtou je, že sa nesnaží vytvoriť jeden statický firmvér. Namiesto toho poskytuje súborový systém s možnosťou zápisu a správcu balíčkov. Toto umožňuje slobodu pri výbere a konfigurácii aplikácií poskytnutých výrobcom a možnosť prispôsobenia pomocou balíčkov. *OpenWrt* umožňuje základ pre vytvorenie aplikácie bez potreby vytvorenia kompletného firmvéru okolo nej. Dôsledkom tohto je veľká miera prispôsobiteľnosti s cieľom použiť zariadenia mnohými novými spôsobmi [**31**].

Výhodou tohto firmvéru je veľké množstvo rozšírení pomocou balíčkov a veľa možností konfigurácie, čo však vyžaduje viac vedomostí ako v prípade väčšiny ostatných firmvérov [**34**]. K dispozícii je aj súbor nástrojov pre vývoj softvéru (SDK) [**35**] a šablóna pre zjednodušenie vytvárania balíčkov [**36**].

### *DD-WRT*

*DD-WRT* je najznámejší alternatívny firmvér vhodný pre širokú škálu bezdrôtových smerovačov a vnorených systémov. Má taktiež otvorený zdrojový kód, je založený na operačnom systéme *Linux*. Na rozdiel od *OpenWrt* kladie dôraz na poskytnutie čo najjednoduchšieho možného ovládania, pričom poskytuje veľké množstvo funkcií a jednoduchú inštaláciu [**32**] [**34**].

### *Tomato*

*Tomato* je taktiež firmvér s otvoreným zdrojovým kódom určený pre smerovače založené na čipoch *Broadcom*. Oproti firmvérom *OpenWrt* a *DD-WRT* poskytuje menej funkcií, čo však nahrádza prepracovaným používateľským rozhraním. Taktiež poskytuje pokročilé funkcie ako napríklad QoS alebo grafické monitorovanie sieťovej premávky [**37**] [**34**].

## 3.1.2 *Sledovanie premávky v domácej sieti*

Domáce smerovače predstavujú ideálne miesto pre sledovanie sieťovej komunikácie koncových používateľov, vytváranie štatistík a aplikovanie rôznych práv alebo obmedzení pre koncové zariadenia [**3**] [**4**]. Sledovanie komunikácie z pohľadu domácich smerovačov má niekoľko výhod oproti sledovaniu komunikácie z pohľadu ISP na jeho zariadeniach. Prvou je fakt, že funkcia NAT skrýva koncové zariadenia a z pohľadu ISP nie je možné jednoducho rozlíšiť jednotlivé zariadenia [**5**] [6]. Pritom informácia o unikátnosti zdroja komunikácie a jeho type (smartfón/PC) môže byť dôležitá pri uplatňovaní QoS, profilovaní a vyhodnocovaní charakteru komunikácie jednotlivých zariadení [**6**]. Druhou výhodou je fakt, že pri rastúcom počte zariadení pripojených do domácej siete, je sledovanie unikátnych zariadení z pohľadu ISP pomerne náročné na výpočtové prostriedky.

## 3.2 Bezpečnosť domácej siete

Počítače v domácnosti sú v súčasnosti už bežne pripojené k Internetu. Pre ich veľký (a stále zvyšujúci sa) počet sa stali častým terčom útokov, a preto musia byť zabezpečené. Tento proces je už relatívne dobre známy. Existuje napríklad mnoho antimalvérových riešení a bezpečnostných brán určených pre osobné počítače. Dodnes však nemáme perfektné riešenie a pravdepodobne ho ani nikdy nebudeme mať. Medzi tým sa však domáce výpočtové prostredie rozrastá do domácej siete mnohých zariadení, ktoré je tiež potrebné zabezpečiť. S ochranou týchto sietí však máme len málo skúsenosti a tejto oblasti sa nevenuje veľká pozornosť [**27**].

Objavovanie služieb a zabezpečené a bezpečné používanie služieb sú základnými prvkami pri nasadzovaní domácich a osobných sietí. Pretože v tomto prostredí nie je prítomný systémový administrátor, nastavenie a každodenná prevádzka takejto siete musia byť automatizované v čo najvyššej miere, pričom sa musí dbať aj na používateľskú prívetivosť. Na dosiahnutie tohto cieľa mnoho systémov obetuje bezpečnosť a súkromie, a preto môžu byť služby objavené a použité neautorizovane alebo môže byť narušené súkromie jednotlivca [**28**].

### 3.2.1 *Bezpečnosť domácich smerovačov*

Aj keď majú domáce smerovače s bezpečnosťou rovnaké problémy ako bežné počítače, nie sú vybavené ochranným softvérom, ako je napríklad antimalvér a nie sú aktualizované. Okrem toho je dôležitosť týchto zariadení zanedbávaná. Prechádza nimi všetka sieťová premávka, ovládajú službu DNS v sieti a vo väčšine prípadov sú online počas celého dňa. Pridávaním čoraz viacerých „neinternetových“ funkcií, ako sú sieťové úložisko a VoIP telefónia, sa ich úloha stáva špecifickejšou a vzniká mnoho obáv o bezpečnosť.

Tieto smerovače sú zákazníkom väčšinou poskytované ISP a veľa z nich je predkonfigurovaných, aby boli pripravené na použitie. Po nastavení prístupových údajov a pripojení domácej siete k smerovaču, môže používateľ ihneď pripojiť k Internetu. Typicky používateľ nevykoná žiadnu ďalšiu konfiguráciu, a to buď z dôvodu sa mu to zdá alebo je to pre neho náročné alebo uprednostňuje okamžité používanie služby pred prekopaním všetkých pokročilých aspektov konfigurácie smerovača. Bohužiaľ, predvolená konfigurácia väčšiny smerovačov poskytuje slabú bezpečnosť, napríklad najmä u starších bezdrôtových smerovačov je veľmi bežné, že šifrovanie bezdrôtovej siete je vypnuté. Rovnaké zariadenia sa však používajú aj v podnikovom prostredí, najmä v menších organizáciách, ktoré nie sú ochotné platiť za drahú infraštruktúru pripojenia k Internetu [**38**].

# 4 Detekcia prienikov

Vyvinúť absolútne bezpečný systém je extrémne náročné, ak nie všeobecne nemožné. Väčšina existujúcich systémov obsahuje bezpečnostné chyby, ktoré predstavujú hrozbu a robia ich náchylnými k prienikom a iný typom zneužitia. Nájdenie a opravenie všetkých takýchto nedostatkov nie je reálne z technických a ekonomických príčin. Existujúce systémy so známymi chybami nie je ľahké nahradiť bezpečnejšími systémami, pretože často chýba požadované funkcie alebo to nedovoľujú ekonomické dôvody. Aj najbezpečnejší systém je náchylný k zneužitie internými pracovníkmi, ktorí zneužijú svoje práva. Tieto fakty sú hlavnou motiváciou pre vytvorenie systémov pre detekciu a prevenciu prienikov [**39**].

**Detekcia prenikov** (***intrusion detection***) je proces monitorovania udalostí vyskytujúcich sa v počítačovom systéme alebo v sieti a ich analýzy s cieľom nájsť znaky možných narušení alebo bezprostredných hrozieb narušenia politík počítačovej bezpečnosti, pravidiel prípustného použitia alebo štandardných bezpečnostných praktík. Takéto udalosti majú mnoho príčin, ako sú malvér (napr. červy, spyware), získanie neoprávneného prístupu útočníkom z Internetu, autorizácia používateľov, ktorí zneužijú ich práva v systéme alebo sa pokúsia získať dodatočné práva, na ktoré nemajú oprávnenie. Aj keď mnoho takých udalostí skutočne je škodlivých, mnoho nie je – napríklad osoba by mohla spraviť preklep v adrese počítača a nechtiac sa pokúsiť o prihlásenie do iného systému bez autorizácie [**40**].

**Systém pre detekciu prienikov** (***Intrusion Detection System – IDS***) je softvér, ktorý automatizuje proces detekciu prienikov. **Systém pre prevenciu prienikov** (***Intrusion Prevention System – IPS***) je softvér, ktorý má schopnosti IDS a taktiež sa môže pokúsiť zastaviť potenciálne nebezpečné udalosti Oba tieto systémy sú primárne zamerané na identifikáciu potenciálne nebezpečných udalostí, zaznamenanie informácií o nich a snahu o ich zastavenie a ich ohlásenie správcom bezpečnosti. Okrem tohto organizácie používajú IDS a IPS na ďalšie účely ako sú identifikácia problémov s bezpečnostnou politikou, dokumentáciu existujúcich hrozieb a odradenie jednotlivcov od narušenia bezpečnostnej politiky [**40**].

Technológie IDS a IPS majú mnoho podobný schopností. Administrátori zvyčajne môžu vypnúť funkciu prevencie v produktoch IPS, ktoré potom fungujú ako systémy IDS. Pre skrátenie sa technológie IDS a IPS spoločne označujú ako **systémy pre detekciu a prevenciu prienikov** (***Intrusion Detection and Prevention Systems – IDPS***) [**40**].

## 4.1 Typické komponenty IDPS

Typické komponenty riešení IDPS sú [**40**]:

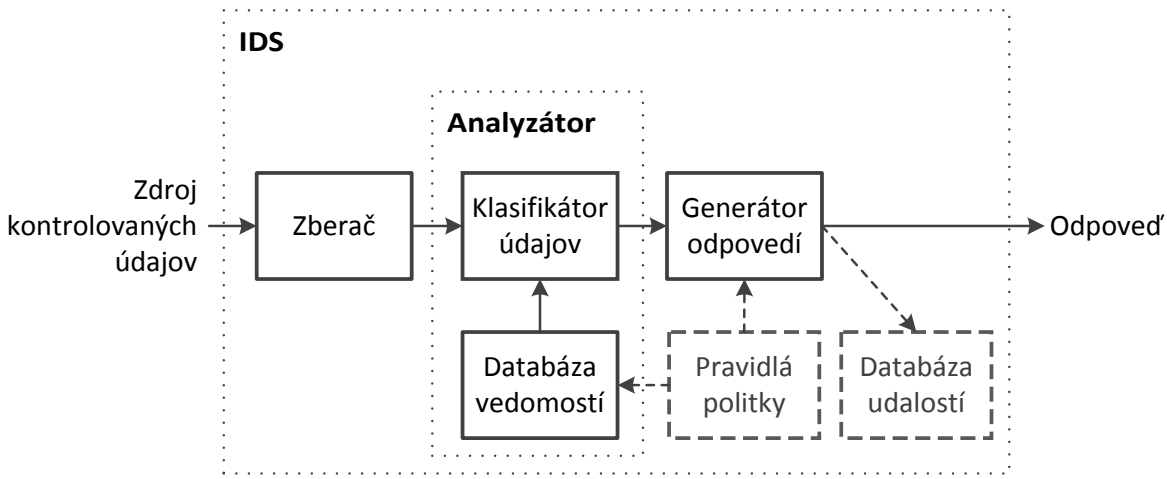
- **Senzor alebo agent** – monitorujú a analyzujú aktivitu. Pojem senzor je typicky používaný pre systémy IDPS, ktoré monitorujú siete. Pojem agent je typicky používaný pre hostiteľské IDPS.
- **Server manažmentu** – centralizované zariadenie, ktoré prijíma informácie od senzorov a agentov a vyhodnocuje ich.
- **Databázový server** – zaznamenáva informácie, ktoré zaznamenali senzory, agenti alebo server manažmentu.
- **Konzola** – poskytuje rozhranie medzi IDPS a administrátormi a používateľmi.

Takéto rozdelenie komponentov sa typicky používa v podnikových IDPS. Avšak niektoré typy IDPS senzorov a agentov možno nasadiť samostatne a riadiť a monitorovať ich priamo administrátorom bez použitia servera manažmentu.

### 4.1.1 *Model IDS*

Existuje veľké množstvo prístupov k detekcii anomálií. Všeobecne ale väčšina z nich nasleduje jeden základný prístup: Na vstupe sú údaje, ktoré je potrebné do potrebnej internej formy, aby mohli byť ďalej analyzované. Potom sa pri procese analýzy rozhodne, či dané vstupné údaje prejavujú nejaké útoku, prípadne anomálie. Pre zistené potenciálne anomálie/útoky sa vytvorí udalosť, ktorá sa pošle jednotke pre generátoru odpovedí. Ten podľa vlastností prijatej udalosti zvolí reakciu na detegovanú potenciálny prienik. Na základe tohto zaviedli *Arvidson* a *Carlbark* **model IDS**, ktorý znázorňuje základné komponenty a toky údajov v systémoch IDS [**41**].

Obrázok 3 zobrazuje model IDS prevzatý od *Arvidsona* a *Carlbarka* [**41**]. Každý rámček reprezentuje komponent, ktorý spravidla vykonáva jednu úlohu. Tok informácií medzi týmito komponentami je znázornený šípkami. Rámčeky s prerušovaným a bledším obrysom znázorňujú že daný komponent nie je nevyhnutnou súčasťou každého IDS. Systém IDS môže fungovať aj bez nich. Väčšina dnes nasadených systémov IDS ich ale používa. V ďalšej časti dokumentu sa nachádza podrobnejší opis jednotlivých komponentov [**24**].




*Obrázok 3: Arvidsonov a Carlbackov model IDS [**41**]*

#### *Zdroj kontrolovaných údajov*

Zdroj kontrolovaných údajov je vstupom IDS. Jedná sa o surové dáta získané z rôznych zdrojov. Ich charakter môže závisieť od typu IDS, ale aj od umiestnenia systému IDS. Príkladom zdroja kontrolovaných údajov sú IP pakety zo sieťovej premávky, logovacie súbory aplikácií alebo výstupy iných systémov IDS.

#### *Zberač*

Úlohou zberača je vzorkovanie vstupných údajov získaných zo zdroja kontrolovaných údajov. Taktiež môže vykonávať predspracovanie. Predspracovanie predstavuje transformáciu vzorkovaných dát do interného formátu, ktorý používa analyzátor. Zberač môže tieto úlohy

vykonávať priebežne v reálnom čase alebo periodicky (napr. v časových intervaloch). Ak je sledovaných niektorý sieťový protokol, môže byť použitá pre účely rekonštrukcie relácie vyrovnávacia pamäť. Súčasťou predspracovania môže byť aj redukcia alebo agregácia údajov. Príkladom je zoskupovanie rovnakých položiek v logovacích súboroch.

### *Analyzátor*

Analyzátor pozostáva z dvoch základných častí - kvalifikátora údajov a databázy vedomostí. Analyzátor analyzuje údaje prevzaté od zberača s cieľom rozhodnúť, či predstavujú potencionálny utok alebo anomáliu. V prípade detekcie potencionálneho útoku sa vytvorí záznam s bližšími informáciami o danej udalosti a odovzdá sa generátoru odpovedí.

### *Klasifikátor údajov*

Klasifikátor udajov sa snaží rozhodnúť, či dáta prijaté zo zberača vykazujú nejaké dôkazy o útoku. Vo všeobecnosti sa toto docieli porovnávaním dát zo zberača s informáciami v databáza vedomostí. Využíva sa pritom jedna alebo viacero metód detekcie. Ak sa zistia náznaky útoku, vytvorí sa záznam obsahujúci informácie o tejto udalosti. Takáto udalosť sa môže klasifikovať podľa jej celkovej závažnosti a spravidla sa oznámi generátoru odpovedí.

### *Databáza vedomostí*

Databáza vedomostí predstavuje perzistentnú pamäť IDS. Môže obsahovať rôzne informácie o útokoch. Obsah tejto databázy závisí od typu IDS a taktiež od nasadených metód.

### *Generátor odpovedí*

Generátor odpovedí rozhoduje, aké akcie budú vykonané na základe udalostí prijatých od klasifikátora. Možné reakcia závisia od typu IDS.

### *Pravidlá politiky*

Pravidlá politiky umožňujú konfiguráciu systému IDS. Na základe pravidiel politiky sa rozhodne, ako bude systém detegovať útoky a ako bude na prípadné detekcie útokov. Takáto konfigurácia systému umožňuje použiť len podmnožinu celej databázy vedomostí, ktorú používa klasifikátor pri identifikácii potencionálnych útokov. Pravidlá politiky môžu taktiež určovať akciu, ktorú vykoná generátor odpovedí pri istej udalosti. Pravidlá politiky nie sú povinnou súčasťou IDS. Ak nie sú prítomné systém IDS aplikuje celú databázu vedomostí a na udalosti bude reagovať preddefinovanými akciami.

### *Databáza udalostí*

Databáza udalostí predstavuje úložisko udalostí, ktoré sú pri prevádzke systému vygenerované. Politika prevzatá od generátora odpovedí môže bližšie určovať, ktoré udalosti bude do databázy zaznamenané. Údaje z tejto databázy môžu byť použité na rôzne účely. Príkladom je vytvorenie štatistickej správy o detegovaných útokoch. Môže sa napríklad využiť pri skúmaní útokov, ktorých prejavy bolo možné registrovať až s odstupom času [**41**] [**24**].

## 4.2 Detekčné metodiky

Na detekciu incidentov sa používa viacero mechanizmov. Tri základné triedy detekčných mechanizmov používajú tieto prístupy [**40**] [**41**] :

- **Detekcia na základe príznakov** (***signature-based detection***) – IDS má databázu informácií opisujúcich útok, podľa ktorých rozoznáva útoky. Všetko, čo nie je identifikované na základe týchto príznakov ako útok, je považované za normálne. Väčšinou sa používajú príznaky, ktoré popisujú určitý typ známeho útoku. Táto metóda by mala mať v závislosti od konkrétnosti príznaku vysokú presnosť, a preto by mala produkovať nízky počet falošných detekcií útokov. Nedokáže však reagovať na neznáme hrozby/útoky.

- **Detekcia anomálií** (***Anomaly-based detection***) – používa vytváranie profilov na základe monitorovania charakteristík typickej aktivity počas istej časovej periódy. Systém potom porovnáva charakteristiky aktuálnej aktivity s hraničnou hodnotou určenou pri vytváraní profilu. Pri tejto metóde detekcie je väčšia možnosť prítomnosti falošných detekcií útokov. Závisí to hlavne od nastavenia hraničných hodnôt detekcie útoku v IDS. Výhodou detekcie na základe anomálií je však ich schopnosť učiť sa a preto vie zareagovať aj na nové typy útokov.

- **Analýza stavových protokolov** (***stateful protocol analysis***) – sleduje všeobecne známe stavy rôznych protokolov a prechody medzi nimi, pričom v sledovaných udalostiach identifikuje odchýlky od štandardnej aktivity. Bližšie tento prístup rozoberá kapitola 4.5.

## 4.3 Základné typy IDPS

Vo všeobecnosti sa technológie IDPS podľa typu udalostí, ktoré monitorujú a miest ich nasadenia rozdeľujú na 4 základné typy [**40**]:

- **Sieťové** (***network-based***) – monitoruje sieťovú premávku na určitom sieťovom segmente alebo zariadení a analyzuje aktivitu sieťových a aplikačných protokolov s cieľom identifikovať podozrivé správanie.

- **Bezdrôtové** (***wireless***) – monitoruje premávku na bezdrôtovej sieti a analyzuje bezdrôtové sieťové protokoly s cieľom identifikovať podozrivé činnosti.

- **Analýza sieťového správania** (***NBA - Network Behavior Analysis***) – skúma sieťovú premávku alebo štatistiky sieťovej premávky s cieľom identifikovať nezvyčajné toky, ako napríklad DDoS útoky, niektoré formy malvéru (červy a zadné vrátka) a porušenia politiky (napr. klientsky systém poskytujúci služby iným systémom).

- **Hostové** (***host-based***) – sleduje charakteristiky jedného hostiteľského systému a udalosti, ktoré na ňom vznikajú s cieľom identifikovať podozrivé správanie.

Ešte všeobecnejšie, len na základe umiestnenia zdroja analyzovaných údajov, sa systémy IDPS rozdeľujú na 2 základné skupiny [**40**]:

- **Sieťový IDS** (***Network-based IDS – NIDS***) – je založený na sledovaní sieťovej premávky pre konkrétny sieťový segment alebo pre sieťové zariadenie. Zachytená sieťová premávka sa analyzuje a identifikujú sa podozrivé aktivity.

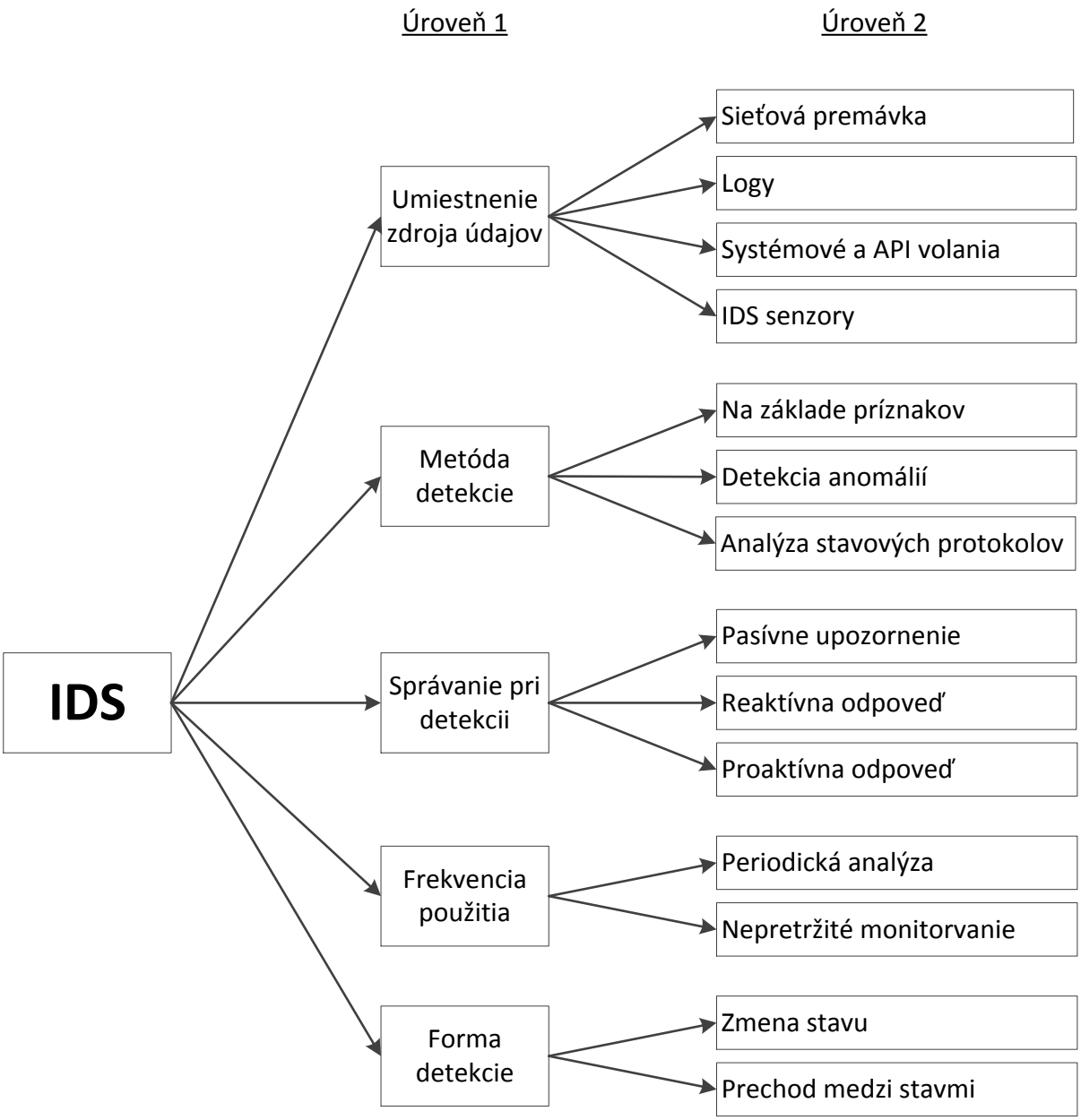
- **Hostiteľský IDS** (***Host-based IDS – HIDS***) – monitoruje vlastnosti na jednom hostiteľskom zariadení a udalosti nastávajúce na tomto zariadení, pričom sa snaží identifikovať podozrivé aktivity. Príkladom je sledovanie systémových záznamov, bežiacich procesov, súborového systému a pod.

## 4.4 Kategorizácia IDPS

### 4.4.1 *Kategorizácia IDS*

Rôzne IDPS sa líšia mnohými parametrami a vlastnosťami. Preto je vhodné zaviesť kategorizáciu týchto systémov. Na obrázku č.4 je kategorizácia prevzatá od *Arvidsona* a *Carlbarka* [**41**]. Keďže nezahŕňala metódu detekcie analýzou stavových protokolov zavedenú vyššie, bola táto metóda pridaná. Taktiež bola zmenená kategória umiestnenia zdroja s názvom „sieťové pakety“ na „sieťová premávka“, keďže pojmom paket sa spravidla označuje dátová jednotka na tretej vrstve TCP/IP modelu. Modifikovanú kategorizáciu zachytáva obrázok 4.

Podľa tejto kategorizácie sa dá zaradiť do kategórií systém pre detekciu útokov na webové servery ale aj akýkoľvek iný IDPS. Každý IDPS by sa mal dať zaradiť v každej kategórii na úrovni 1. Napríklad IDPS by mal mať metódu detekcie útokov. Existujú iba tri metódy detekcie útokov: na základe príznakov, na základe analýzy stavových protokolov a na základe anomálií. Pokiaľ sa neobjaví nová technológia, malo by byť možné bez problémov určiť, či konkrétny IDPS používa metódu detekcie na základe príznakov, na základe anomálií alebo na základe analýzy stavových protokolov (alebo kombináciu dvoch alebo troch týchto prístupov). Každý IDPS sa teda dá zaradiť do aspoň do jednej kategórie úrovne 2 pre každú z kategórií na úrovni 1. Jednotlivé kategórie na úrovni 1 a úrovni 2 sú opísané v nasledujúcej časti dokumentu [**24**].




*Obrázok 4: Kategorizácia IDPS*

### *Umiestnenie zdroja údajov*

IDS môžu zbierať vstupné dáta na spracovanie na rôznych miestach. Vstupom môže byť [**41**] [**24**]:

- **Sieťová premávka** – V prípade, že IDS zbiera pakety zo siete, ide o **sieťový systém pre detekciu útokov** (angl. ***Network-based Intrusion Detection System – NIDS***). Takýto zber údajov sa spravidla docieli pridaním sieťového rozhrania napraveného do promiskuitného režimu[4] do, na ktorom je nasadený nástroj IDS. Takýto prístup umožní NIDS sledovať

[4] V promiskuitnom režime sieťové rozhranie prijíma všetky rámce linkovej vrstvy vrátane tých, ktoré sú určené pre iné sieťové rozhranie.

všetku komunikáciu v rámci sieťového segmentu pripojeného, ku ktorému je táto sieťová karta pripojená. Príkladom môže byť rekonštrukcia HTTP požiadaviek smerovaných na webový server.

- **Logy** - **hostiteľské systémy pre detekciu útokov** (angl. ***Host-based Intrusion Detection Systems - HIDS***) na rozdiel od NIDS zbierajú vstupné údaje na spracovanie len z jedného hostiteľského stroja. Takéto zberače údajov sa nazývajú **agenti**. Pre systémy HIDS je často vstupom logovací súbor, ktorý vytvára operačný systém alebo konkrétna aplikácia. Obyčajne sa takýto prístup používa pri dôležitých serveroch. Môže však byť nasadený napríklad aj pri sledovanie logovacích výstupov bezpečnostnej brány. Pri zistení potenciálne neželanej aktivity môže odoslať napríklad upozornenie odborníkom zodpovedným za bezpečnosť, ktorí danú aktivitu bližšie prešetria.

- **Systémové a API volania** – Ďalším typom zdroja údajov pre systémy HIDS a API a systémové volania. V prípade monitorovania systémových volaní HIDS zbiera údaje napríklad na rozhraní medzi aplikáciou a operačným systémom (napr. jeho jadrom). Volania sú vyhodnotené a v prípade podozrivej aktivity sa môže vykonanie systémového volania zamietnuť. Obdobným prístupom je možné získať vstup pre HIDS z API rozhrania, ktoré poskytuje napríklad služba bežiaca na danom systéme.

- **IDS senzory** – Ďalším typom vstupu pre IDS sú udalosti z iných IDS rozmiestnených v rôznych častiach siete. Takéto zberače sa nazývajú **senzory**. Vo väčších systémoch spravidla centrálne IDS zbiera údaje od niekoľkých senzorov a centralizovane vyhodnocuje celkové riziko možného prieniku.

### *Metóda detekcie*

Metóda detekcie stanovuje, akým prístupom sú v klasifikátore údajov potenciálne prieniky . Existujú tri druhy metód detekcie útokov: detekcia na základe príznakov, detekcia anomálií a analýza stavových protokolov. Jednotlivé druhy metód sú opísané v kapitole 4.2. Metóda detekcie určuje, aké informácie sú obsahom databázy vedomostí [**41**] [**24**].

### *Správanie pri detekcii*

Správanie pri detekcii predstavuje akciu alebo odpoveď vykonanú systémom IDS (v generátore odpovedí) pri detekcii prieniku. Podľa tento vlastnosti je systémy IDS možné rozdeliť na **aktívne** a **pasívne** [**42**]. Pri zavedení pojmu systém pre prevenciu útokov (IPS) bola aktívna detekcia ďalej rozdelené na **reaktívnu** a **proaktívnu**. Generátor odpovedí môže v rámci generovania odpovede pridať záznam do databázy udalostí [**41**] [**42**] [**24**] :

- **Pasívne upozornenie** – Cieľom pasívneho upozornenia je distribuovať informácie o vzniknutej udalosti. Nazýva sa taktiež ja pasívna odpoveď. Pasívne upozornenie môže byť zasielanie mailov zodpovedným pracovníkom, zobrazenie udalostí na konzole alebo či akýkoľvek spôsob informovania kompetentnej osoby o vzniknutej udalosti. Odpoveď je realizovaná po detekcii útoku. Pri tomto type odpovede neprichádza bezprostredne k prerušeniu či zastaveniu aktivity, ktorú systém vyhodnotil ako potenciálny útok.

- **Reaktívna odpoveď** - Reaktívna odpoveď je druh aktívnej odpovede. Pri takejto odpovedi je pozmenené okolité prostredie na systéme alebo sieti, v rámci ktorého bola zaznamenaná detegovaná podozrivá aktivita. Reaktívna odpoveď môže zmeniť okolité prostredie po detegovaní útoku. Cieľom je zamedzenie ďalšieho prístupu útočníka k prostriedkom, na

ktoré sa detegoval potencionálny útok. Príkladom takejto odpovede môže byť prerušenie sieťového spojenia, ktoré obsahovalo útok. Nie je nutné zahadzovať aj paket, ktorý v rámci ktorého bol detegovaný prienik. Takýto paket pri reaktívnej odpovedi môže svoj cieľ dosiahnuť.

- **Proaktívna odpoveď** – Na rozdiel od reaktívnych odpovedí sa proaktívne odpovede pokúsia útoky prerušiť už počas ich detekcie. Príkladom takejto odpovede môže byť zahodenie paketu alebo odmietnutie obsluhy pre systémové volanie. Hlavný rozdiel oproti reaktívnej odpovedi je v tom, že proaktívna odpoveď sa vykoná už počas detekcie útoku a preto môže zabrániť útoku v plnej miere. Príkladom môže byť bezprostredné zahodenie sieťového paketu, ktorý prejavoval znaky útoku.

### *Frekvencia použitia*

Prostredie, ktoré systém IDS monitoruje, môže sledované periodickým preskúmavaním v pravidelných časových intervaloch alebo permanentne reálnom [**41**] [**24**]

- **Periodická analýza** – Niektoré systémy IDS analyzujú monitorované prostriedky len v nastavených časových intervaloch. Príkladom môže byť IDS analyzujúci logovacie záznamy aplikácie periodicky v pravidelných intervaloch.
- **Nepretržité monitorovanie** – Pri nepretržitom monitorovaní sleduje systém IDS monitorované prostriedky nepretržite (často aj v reálnom čase) a priebežne vyhodnocuje riziká.

### *Forma detekcie*

Následnosť udalostí, ktorá predstavuje útok môže byť analyzovaná dvoma spôsobmi [**41**] [**24**]:

- **Detekcia na základe zmeny stavu** - Pri detekcii na základe zmeny stavu systém IDS analyzuje stavy monitorovaného systému a zisťuje, či sa systém nedostal do chybového stavu, napr. či nebol zmenený obsah systémového súboru alebo overuje či neprišlo k neoprávnenému prístupu k systému. Týmto spôsobom je možné detegovať útok, ktorý už nastal pred detekciou. Z tohto dôvodu nie možné daný útok zastaviť, ako sa detegujú jeho prejavy. Možné je však zamedziť pokračovaniu útoku
- **Detekcia pri prechode medzi stavmi** - Pri Detekcii pri prechode medzi stavmi sa rozpoznávajú prechod medzi jednotlivými stavmi. To, že došlo k prechodu medzi stavmi znamená že sa v systéme niečo práve deje. Systémy IDS analyzujú samotné prechody. Toto dovoľuje prerušiť prebiehajúci útok pred tým, ako bol dokončený.

## 4.5 Analýza stavových protokolov

**Analýza stavových protokolov** je proces porovnávania vopred daných profilov všeobecne uznávaných definícií neškodnej aktivity pre každý stav protokolu s pozorovanými udalosťami s cieľom identifikovať odchýlky. Na rozdiel od detekcie anomálií, pri ktorej sa používa profil špecifický pre hostiteľa alebo sieť, sa analýza stavových protokolov spolieha na univerzálne profily vyvinuté dodávateľom, ktoré určujú, ako konkrétne protokoly majú a nemajú byť použité. Slovo „stavový“ v analýze stavových protokolov znamená, že IDPS je schopné pochopiť a sledovať stav siete, prenosu alebo aplikačného protokolu, ktorý má pojem stavu. Napríklad pri vytvorení FTP relácie, je relácia zo začiatku v neautentifikovanom stave. Neautentifikovaný používatelia by mali

v tomto stave vykonávať len niekoľko príkazov, napríklad poskytnutie používateľského mena a hesla. Vykonanie iných príkazov v tomto stave možno považovať za podozrivé [**40**].

Metódy stavovej analýzy protokolov používajú **modely protokolov**, ktoré sú typicky založené primárne na protokolových štandardoch od dodávateľov softvéru a organizácií pre štandardizáciu (napr. *Internet Engineering Task Force* [*IETF*]). Pre nestavové protokoly je možné v niektorých prípadoch vytvoriť umelé stavy. Modely protokolov zvyčajne berú do úvahy aj rozdiely rôznych implementácií protokolu. Protokoly niekedy nie sú špecifikované jednoznačne a niektorí dodávatelia nedodržujú vždy štandard alebo pridávajú vlastné črty. [**40**].

Hlavnou nevýhodou metód analýzy stavových protokolov je, že bývajú náročne na prostriedky kvôli zložitosti analýzy a réžii potrebnej pri sledovaní veľkého množstva súbežných spojení/relácií. Ďalším problémom je, že tento prístup nedokáže detegovať útoky, ktoré neporušujú štandardné správanie protokolov. Ďalším problémom býva, že modely protokolov použité systémom IDPS môžu byť v rozpore so spôsobom, akým je protokol implementovaný v špecifickej verzii aplikácie alebo operačného systému [**40**].

### 4.5.1 *Kombinácia s detekciou anomálií*

Zdroje informácií ako sú sieťové pakety predstavujú významnú výzvu pre techniky detekcie anomálií z dvoch hlavných dôvodov [**43**]:

- Objem dát a teda aj priestor pre možné dôležité štatistické vlastnosti je extrémne veľký.
- Surový (***raw***) sieťový paket má tendenciu byť neštruktúrovaný, čo sťažuje rozlišovanie zmysluplných informácií od „šumu na pozadí“.

Na riešenie tohto problému sú surové dáta zvyčajne spracované s cieľom extrahovať dôležité vlastnosti. Tento proces značne znižuje množstvo dát, ktoré majú byť spracované systémom pre detekciu anomálií. Dôležitosť výberu vhodných vlastností je potvrdená väčšinou výskumníkov v oblasti detekcie anomálií. V súčasnosti je výber týchto vlastností poháňaný vedomostiach expertov a ľudským úsudkom, ktorý berie do úvahy „užitočné informácie“ pre detekciu anomálií. Toto riešenie ani zďaleka nie je vždy spoľahlivé. Často je totiž predstava užitočných vlastností ovplyvnená znalosťou známych útokov. V dôsledku tohto nemusia vybrať vlastnosti, ktoré sú užitočné pri detegovaní neznámych útokov [**43**].

*Sekar* a jeho spolupracovníci preto prišli s myšlienkou zvýšiť automatizáciu tohto procesu výberu vlastností. Navrhli mapovanie vlastností postupností paketov (štatistických) do vlastností spojených s prechodmi v stavovom automate. Keďže je počet prechodov v porovnaní s možný počtom kombinácií sieťových paketov relatívne menší, toto mapovanie redukuje priestor možných vlastností [**43**].

## 4.6 Distribuovaný systém pre detekciu prienikov

V súčasnosti sa začínajú čoraz viac používať aj ***distribuované systémy pre detekciu prienikov*** (***Distributed Intrusion Detection Systems – DIDS***). Tie pozostávajú s viacerých IDS väčšinou umiestnených vo väčšej sieti. Jednotlivé IDS pritom komunikujú medzi sebou alebo s centrálnym serverom, ktorý poskytuje pokročilé sledovanie siete, analýzu udalostí a aktuálne informácie o útokoch. Systém DIDS umožňuje spoločnosti efektívne spravovať zdroje pre analýzu incidentov centralizovaním záznamov o útokoch a poskytnutím rýchleho a jednoduchého prehľadu trendov a vzorov analytikovi, ktorý takto môže získať globálny prehľad. Taktiež umožňujú identifikáciu hrozieb pre siete, ktoré zahŕňajú viaceré sieťové segmenty [**44**].

# 5 Existujúce riešenia

V tejto kapitole sú opísané vybrané riešenia, ktoré majú podobný cieľ alebo prístup ako tento projekt.

## 5.1 Projekt Turris

Projekt *Turris* (novší názov *TurRys*)je služba, ktorá pomáha používateľom s ochranou domácej siete pomocou špeciálneho smerovača. Jedná sa o neziskový výskumný projekt združenia *CZ.NIC*, ktoré je okrem iného aj správcom českej domény *.cz* [**3**]. Projekt má vlastný otvorený softvér postavený na *OpenWrt* a taktiež vlastný otvorený hardvér.

Smerovač okrem bežnej funkcionality dokáže taktiež **analyzovať premávku medzi Internetom a domácou sieťou** a identifikovať podozrivé dátové toky. Pri detekcii sa možný útok oznámi na centrálu *Turris*. Centrála systému ma na premávku globálny pohľad a umožňuje porovnávať dáta z mnohých pripojených smerovačov a vyhodnotiť nebezpečnosť detegovanej premávky. V prípade, že je odhalený útok, vytvoria sa aktualizácie, ktoré sú distribuované do celej siete *Turris* [**3**]..

Zariadenie je vzhľadom k výskumnej povahe dlhodobo prenajímané za symbolickú cenu 1 Kč. Podmienkou pre účasť v projekte však je záväzok používať smerovač *Turris* po predom stanovenú dobu vo funkcii hlavného bodu pripojenia siete k Internetu a nezasahovať do prevádzkovaného zberu dát [**3**].

### 5.1.1 *Hardvér*

Projekt má vlastný hardvér, ktorého dizajn je uvoľnený pod open source licenciou. Hardvér je oproti bežne dostupným domácim smerovačom výkonnejší. Hlavným dôvodom pre rozhodnutie vývoja vlastného hardvéru bola potreba použitia zariadenia, ktoré bude dostatočne výkonné na to, aby zvládlo analýzu sieťovej premávky pri súčasných, ale aj budúcich rýchlostiach domáceho pripojenia. Druhý model (verzia 1.1) ma nasledujúce základné vlastnosti [**45**]:

- Procesor: Freescale P2020, 2 x 1200MHz
- RAM: 2 GB DDR3 v SO-DIMM slote
- Úložisko: 16 MB NOR a 256 MB NAND flash pamäti
- Ethernet: gigabitový WAN a 5 gigabitových LAN portov
- Wi-Fi: 802.11a/b/g/n 3x3 MIMO s odnímateľnou anténou
- USB: 2 x USB 2.0, 1 x USB 3.0
- MiniPCIe: 1 voľný slot a 1 slot obsadený Wi-Fi kartou
- Rozhrania: UART, SPI, I2C, debug konzola napojená cez interný microUSB port
- SIM slot: áno
- Spotreba: 9,5 W bez záťaže, 14 W pri maximálnej záťaži.

Pre používateľov, ktorý sa pripájajú pomocou technológie DSL (ADSL, VDSL) bol vytvorený separátny *DSL bridge*, napájaný z USB a určený výhradne pre prevádzku so smerovačom *Turris*.

### 5.1.2 *Softvér*

Základom vlastného operačného systému *Turris OS* je systém *OpenWrt*. Do tohto systému boli pridané niektoré vylepšenia, napríklad vlastný monitor sieťovej prevádzky *uCollect*, systém *nuci* pre vzdialenú správu a automatické aktualizácie. Vzhľadom na použitie *OpenWrt* ako základu sú používateľom k dispozícii balíčky všetkých programov, ktoré sú dostupné pre túto distribúciu [**46**].

#### *Majordomo*

Nástroj *Majordomo* slúži na pre sledovanie komunikácie LAN zariadení s Internetom. Okrem toho, že umožňuje užívateľovi získať si prehľad o správaní počítačov v jeho lokálnej sieti [**46**].

#### *Pasívny monitoring rýchlosti*

*Turris* priebežne sleduje množstvo dát prenesených routerom za krátke intervaly (2 sekundy) a určuje tak okamžitú rýchlosť pripojenia. Z týchto dát sa potom vytvára štatistika, ktorá umožňuje jednotlivým používateľom sledovať využitie ich linky a zároveň aj globálnu štatistiku maximálnych dosiahnutých rýchlostí, ktorá umožňuje odhadnúť skutočnú prenosovú rýchlosť linky [**46**].

#### *Netflows*

Sonda *flows* zbiera informácie ako sú adresy, porty, časy a množstvo prenesených dát v jednotlivých podozrivých spojeniach. Zber dát sa spúšťa automaticky v nadväznosti na detekciu špecifickej prevádzky, anomálie alebo komunikácia s podozrivou adresou [**46**].

### 5.1.3 *Bezpečnosť*

Ochranu, ktorú projekt poskytuje možno rozdeliť na pasívnu a aktívnu. Do prvej kategórie spadá bezpečný operačný systém s podporou automatických aktualizácií. V oblasti aktívnej ochrany je hlavným prvkom distribuovaný adaptívny firewall. Ten je tvorený nástrojmi, ktoré spoločne vytvárajú systém ochrany schopný reagovať na bezpečnostné hrozby. Skladá sa z nasledujúcich častí [**47**]:

- **Zber** vstupných dát – predovšetkým monitoring prevádzky koncových zariadení.
- **Analýza** získaných dát na centrálnom serveri.
- **Tvorba** pravidiel pre bezpečnostnú bránu na základe analýzy dát.
- **Aktualizácia** koncových zariadení – okrem dát získaných z koncových zariadení v bode 1 sa pri implementácii pravidiel pre bezpečnostnú bránu používajú informácie aj z verejných a špecializovaných zdrojov.

#### *Monitorovanie v reálnom čase*

Pre monitorovanie sieťovej prevádzky v reálnom čase bol vytvorený framework *uCollect*, ktorý spracováva všetku premávku na určenom sieťovom rozhraní a používa systém zásuvných modulov pre monitorovanie jednotlivých funkcií.

Pre túto platformu je implementovaných viacero špecializovaných sond, napríklad sonda pre zber štatistických dát sieťovej prevádzky, sonda pre detekciu anomálií, sonda pre zber dát podozrivých spojení, či sonda pre monitoring nevytvorených spojení. Sumarizované výsledky sú potom pravidelne odosielané na analytický server, kde sa následne spracovávajú [**47**].

### *Monitorovanie bezpečnostnej brány*

Okrem sledovania prevádzky v reálnom čase je ďalším zdrojom log bezpečnostnej brány. Ten ukazuje nevyžiadanú premávku prichádzajúcu z vonkajšej siete a môže sa použiť ako doplnkový zdroj informácií pri analýze anomálií [**47**].

## 5.1.4 *Zbierané dáta*

Na smerovači sa zbierajú nasledovné dáta [**48**]:

- Základné štatistiky
- Štatistiky PCAP – počty prijatých a zahodených paketov
- Dáta pre detekciu anomálií
- Logy z bezpečnostnej brány
- Logy z automatických aktualizácií

# 6 Špecifikácia požiadaviek

Vyvíjané riešenie by malo umožňovať sledovanie štatistických parametrov sieťovej komunikácie v domácej sieti a na základe týchto parametrov detegovať prieniky v rámci domácej siete. Pôjde teda o sieťové IDS (NIDS). Detekcia bude prebiehať v reálnom čase. Model poskytujúci štatistické údaje o priebehu a charaktere sieťovej komunikácie koncových zariadení bude poskytovať:

- Údaje o priebehu spojení na sieťovej a transportnej vrstve, čiže sa bude jednať o multiprotokolové sledovanie sieťovej komunikácie
- Údaje budú sledované vo viacerých časových intervaloch (1ms – 1h) pre pokrytie rôznych druhov komunikácie.
- Modely pre jednotlivé protokoly budú popisovať priebeh spojenia podľa špecifikácie RFC. Napr. počet polootvorených, polouzavretých, nadviazaných, prerušených alebo neúspešne nadviazaných spojení z pohľadu rôznych protokolov.
- Získané údaje budú analyzované štatistickými a numerickými metódami, prípadne aj využitím algoritmov umelej inteligencie pre fázu učenia – tvorby profilov správania pre koncové zariadenia.
- Z programového hľadiska pôjde o vytvorenie rozšírenia pre existujúce riešenie (napr. IDS *Snort*). Cieľom je využiť existujúcu funkcionalitu a zároveň využiť existujúcu používateľskú základňu pre spätnú väzbu, testovanie a verejné zhodnotenie.
- Integrácia do firmvéru pre domáce smerovače s otvoreným zdrojovým kódom (napr. *OpenWrt*) má rovnaký motív ako predošlý bod.
- Lokálne ako aj agregované výstupy by mali umožniť ľahké zhodnotené z pohľadu:
    - schopnosti vytvorenia profilov rôznych zariadení a rôzneho spôsobu využívania služieb v Internete.
    - schopnosti identifikovať anomálie a prípadné útoky na základe štatistického profilovania koncových zariadení a ich používania.
    - schopnosti poskytnúť informácie pre priamu tvorbu ACL pre dané smerovače s cieľom potlačiť niektoré neželané typy sieťovej komunikácie.
    - pridanej hodnoty distribuovaného zberu a agregovaného vyhodnocovania údajov o správaní koncových zariadení pri prístupe do Internetu.
    - schopnosti sledovania mobility koncových používateľov/zariadení pri presune medzi jednotlivými sieťami, resp. schopnosti identifikácie daného zariadenia/jeho typu na základe priradenia do existujúceho profilu správania.

Detekčný mechanizmu by mal v čo najväčšej možnej miere eliminovať **falošné detekcie útokov** (***false positives***) a **prehliadnutia útokov (*false negatives***).

Zber údajov a pamäťové nároky na udržiavanie štatistík by mali byť prispôsobené schopnostiam domácich smerovačov. Riešenie by malo byť ľahko integrovateľné do voľne šíriteľného firmvéru pre domáce smerovače (napr. *OpenWrt*).

Riešenie by malo umožňovať získané údaje centrálne vyhodnocovať tak, aby bolo možné na základe takéhoto globálneho pohľadu vytváranie pravidiel pre distribuovanú adaptívnu bezpečnostnú bránu.

# 7 Návrh

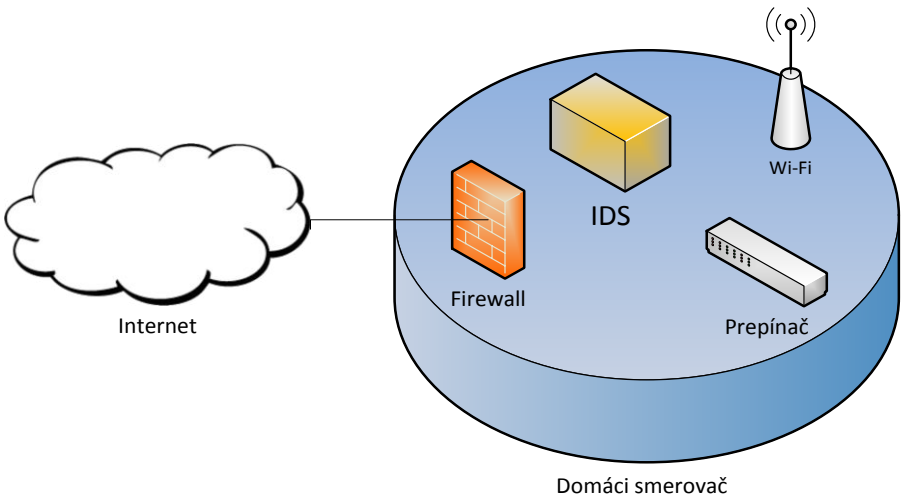
Detekciu prienikov bude systém vykonávať prístupom, ktorý kombinuje analýzu stavových protokolov a detekciu anomálií a je opísanej v kapitole 4.5.1. Vlastnosti postupností paketov (štatistických) budú mapované do vlastností spojených s prechodmi v stavovom automate. Následne sa bude vykonávať detekcia anomálií nad vzniknutými štatistickými parametrami.

## 7.1 Integrácia IDS do domácej siete

Pre nasadenie boli navrhnuté dva prístupy: integrácia IDS do smerovača a umiestnenie IDS na separátne zariadenie. Druhý prístup bude použitý v prípade, že výkon smerovača nebude postačujúci. Tieto prístupy sú opísané ďalšej časti dokumentu

### 7.1.1 *Integrácia IDS do smerovača*

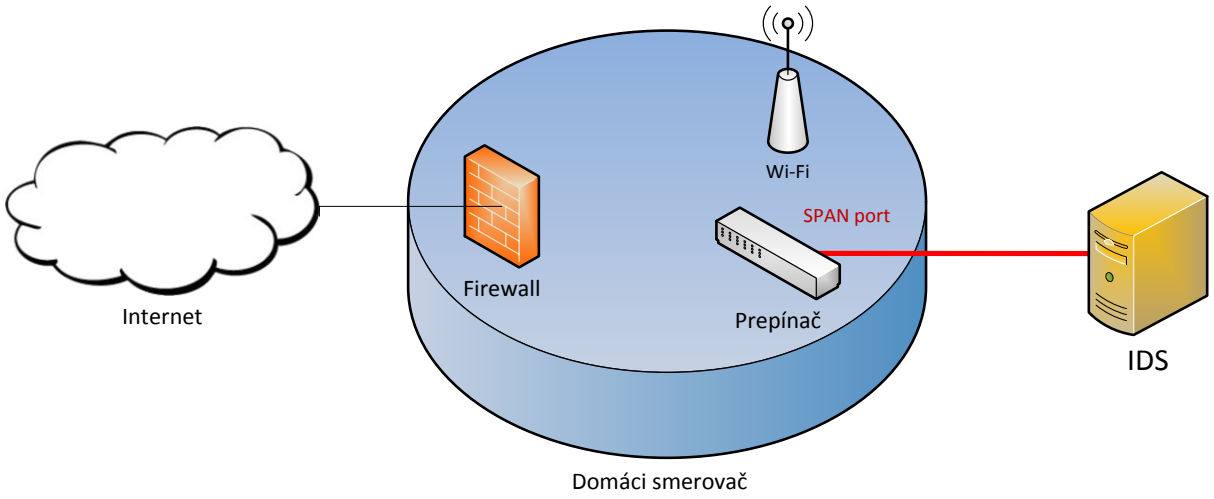
Obrázok 5 znázorňuje integráciu navrhovaného IDS do domáceho smerovača. IDS bude spustený ako softvérové riešenie priamo na operačnom systéme domáceho smerovača (*OpenWrt*).




*Obrázok 5: Schematické znázornenie integrácie IDS do smerovača*

### 7.1.2 *IDS na separátnom zariadení*

Obrázok 6 znázorňuje umiestnenie samotného IDS mimo domáceho smerovača. Takéto riešenie sa použije v prípade, že výkon smerovača nebude postačovať na beh detekcie v reálnom čase. Aby IDS mohlo spracovávať všetku premávku, je potrebné, aby bola preposielaná zo smerovača. Toto sa dosiahne nastavením portu na smerovači do režimu, v ktorom prepošle všetku premávku, ktorá cez smerovač prechádza. Takýto port sa nazýva aj ***SPAN port*** a tento prístup sa tiež nazýva ***port mirroring***.




*Obrázok 6: Schematické znázornenie umiestnenia IDS na separátnom zariadení*

## 7.2 Metóda detekcie

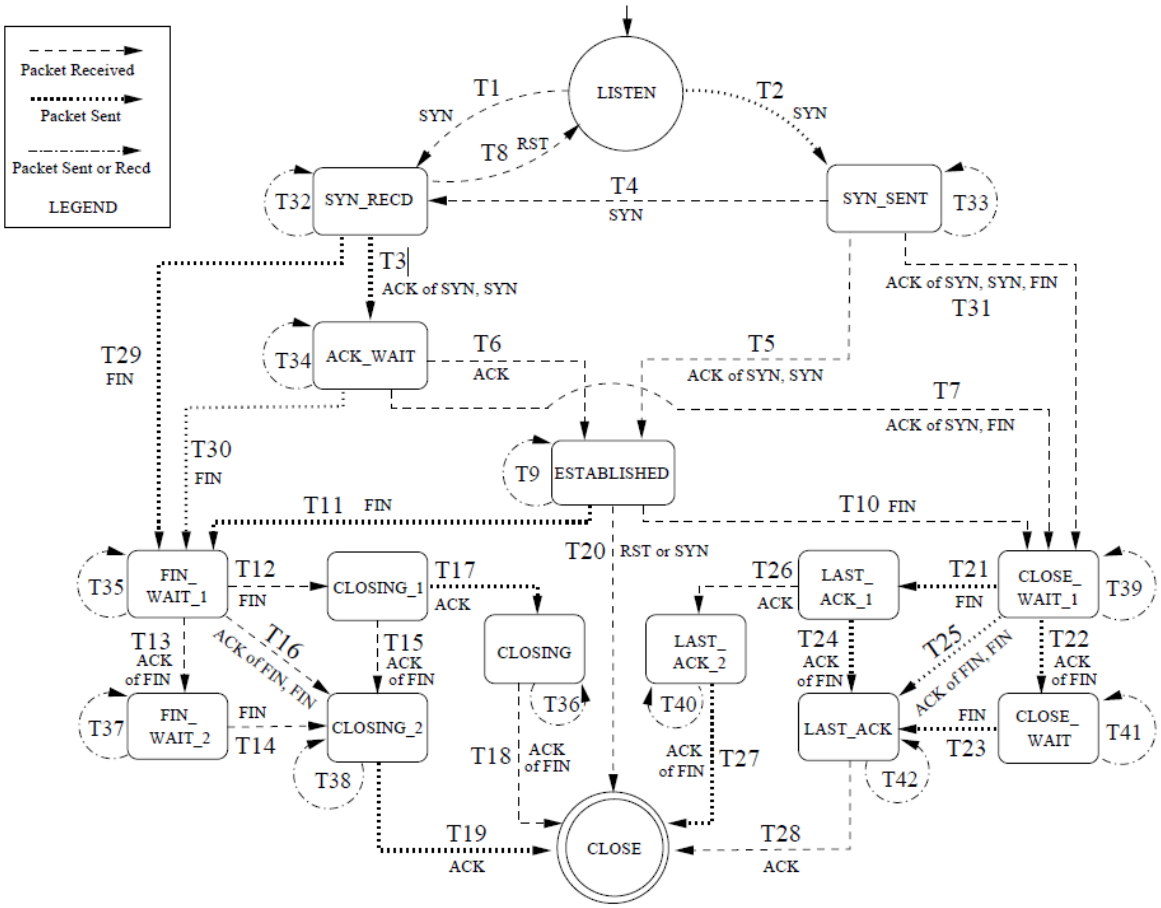
Navrhované IDS bude detegovať útoky a anomálie v sieťovej premávke použitím **analýzy stavových protokolov** v spojení s detekciou anomálií. Stavové automaty budú vytvorené podľa špecifikácií sledovaných protokolov, pričom bude cieľom aj použitie funkcionalít existujúcich riešení a možnosť integrácie do nich.

### 7.2.1 *_Stavový automat pre protokol TCP*

Stavový automat pre protokol TCP bude vychádzať z automatu, ktorý navrhli *Sekar* a jeho spolupracovníci v [**43**]. Tento stavový automat vychádza zo štandardov RFC, snaží sa ju však abstrahovať. Abstrakcia zjednoduší návrh a implementáciu daného automatu. Jej hlavnou výhodou však je, že pri striktnom dodržaní štandardov je väčšia pravdepodobnosť označenia bežnej premávky ako útoku, a to kvôli nepatrnému rozdielu v interpretácii daného protokolu. Toto riziko sa ďalej zvyšuje aj vďaka skutočnosti, že sa používajú rôzne implementácie protokolu TCP.

Obrázok 7 zobrazuje stavový automat pre protokol TCP prevzatý od *Sekara* a jeho spolupracovníkov. Jednotlivé prechody sú očíslované. Pre zníženie náročnosti na výpočtové prostriedky sa zváži verzia, ktorá nebude obsahovať prechody ústiace späť do stavu, z ktorého vychádzajú (slučky). Toto by malo značne znížiť nároky, keďže väčšina spojení po riadnom nadviazaní zotrváva najdlhšie v stave `ESTABLISHED`, kedy sa prenášajú užitočné dáta.

Prechody a udalosti, ktoré nie sú zobrazené, budú považované za narušenie štandardnej postupnosti prechodov a budú vyhodnotené prerušenie prechodov v stavovom automate.




*Obrázok 7: Stavový automat pre protokol TCP [**43**].*

### 7.2.2 *Stavové automaty pre bezspojové protokoly*

Pre protokoly, ktoré nie sú spojovo-orientované (a teda neumožňujú prirodzené sledovanie stavu „spojenia“), budú vytvorené umelé stavy. Napríklad pre protokol **UDP**, ktorý je bezspojový, je možné umelo zadefinovať nasledovné stavy:

- `NONE` – pre komunikáciu neboli zaznamenané žiadne pakety.
- `CLIENT_SEEN` – pakety pre danú komunikáciu boli zaznamenané len v jednom smere (od klienta smerom k serveru).
- `ESTABLISHED` – boli zachytené pakety v oboch smeroch (od klienta k serveru aj naopak).
- `TIMEDOUT` – vypršal časový limit spojenia t.j. po stanovenú dobu nebol pre danú komunikáciu zaznamenaný žiadny paket.

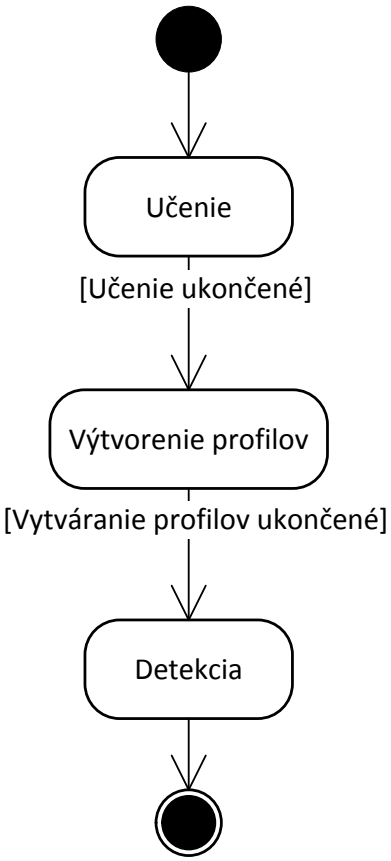
## 7.3 Sledovanie prechodov vo viacerých časových intervaloch

Distribúcie prechodov budú sledované vo viacerých časových intervaloch (napr. 1ms – 1h) pre pokrytie rôznych druhov komunikácie. Toto umožní systému reagovať na anomálie krátkodobého ale aj dlhodobého charakteru.

## 7.4 Stavový diagram režimov prevádzky

Na obrázku 8 je stavový diagram režimov prevádzky navrhovaného systému. Systém sa môže nachádzať v 2 základných režimoch: režim učenia a režim detekcie. Pri prechode z režimu učenia do

režimu detekcie však na čas potrebný pre vytvorenie profilov správania prejde systém do stavu vytvárania profilov.




*Obrázok 8:Stavový diagram režimov prevádzky navrhovaného IDS*

## 7.5 Vyhodnotenie anomálie

Základným spôsobom vyhodnocovania anomálií bude vytvorenie štatistického profilu pre každé zariadenie na lokálnej sieti. Tento profil bude obsahovať priemernú distribúciu prechodov. Na vyhodnotenie sa použijú štatistické prístupy – určí sa hraničná odchýlka od priemeru, ktorú ak aktuálna komunikácia prekročí, bude označená ako anomália.

### 7.5.1 *Štatistické vyhodnotenie anomálie*

Pri štatistickom vyhodnotení bude profil zariadenia obsahovať pre každý časový interval a každý sledovaný prechod priemer a varianciu.

V režime učenia sa určí stredná hodnota (**priemer**) $\mu$ a **variancia** $\sigma^2$ priemerného počtu vykonania daného prechodu pre daný časový interval.

V režime detekcii sa použije **Čebyševova nerovnosť**. Tá určí hornú hranicu pravdepodobnosti, že rozdiel medzi náhodnou premennou $x$ (aktuálne priemerný počet vykonaní daného prechdu) a priemerom $\mu$ prekročí danú hraničnú hodnotu $t$ pre rozdelenie s varianciou $\sigma^2$ a priemerom $\mu$ [**49**] .

### 7.5.2 *Rozšírený spôsob vyhodnotenia anomálie*

Rozšírený spôsob vyhodnocovania anomálií bude využívať strojové učenie. Pre detekciu anomálií sa overia nasledujúce prístupy v strojovom učení:

- One class support vector machines
- Klastrovanie

Jednotlivé metódy je potrebné otestovať a vyhodnotiť, ako účinné sú. V režime učenia sa budú vytvárať profily pre jednotlivé stanice použitím strojového učenia. Výstupom strojového učenia budú modely, ktoré budú predstavovať profily, ktoré budú ďalej použité vo fáze detekcie.

# 8 Experimentálne overenie

Táto časť dokumentu obsahuje experimentálne overenie modelu pre sledovanie prechodov v stavovom automate pre protokol TCP opísaného v časti návrh. Pre overenie bola použitý nástroj *Snort* s modifikovaným rozšírením *heuristate*, ktorého autorom je *Peter Marko* [**50**]. Modifikáciou rozšírenia sa dosiahlo podrobnejšie zaznamenávanie údajov do logovacieho súboru. Zozbierané údaje sa vyhodnocujú pomocou strojového učenia. Pre časť strojového učenia je použitý jazyk *R*. Pri overení bol použitý dataset *UNB ISCX IDS* [**51**].

## 8.1 IDS Snort

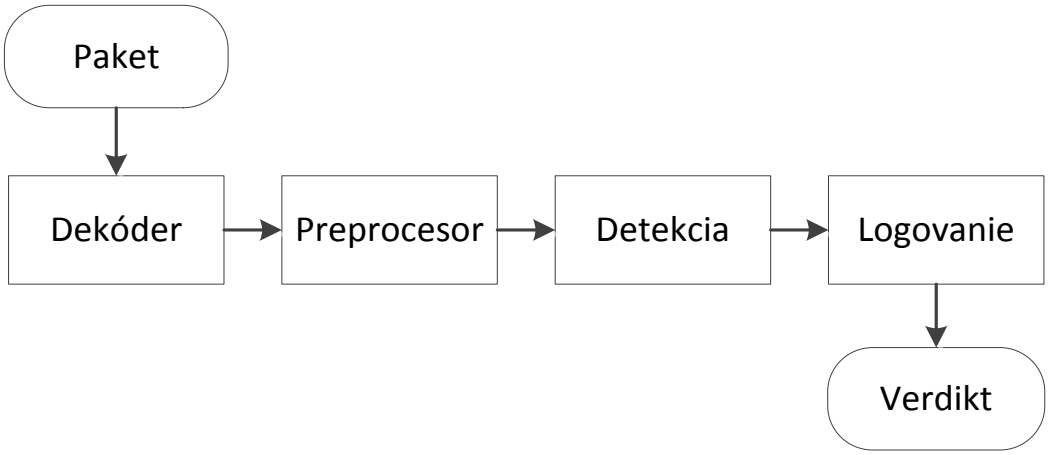
Ako základ pre extrakciu informácií o prechodoch v stavových automatoch je použitý IDS *Snort* [**52**]. Snort analyzuje živú premávku alebo záznam premávky (súbor `.pcap`). Následne umožňuje v režime IDS dekódovanie premávky a detekciu útokov na základe príznakov podľa sady nakonfigurovaných pravidiel. Pravidlá pre známe a opísané útoky sú voľne dostupne a priebežne aktualizované.

Pomocou rozšírení umožňuje Snort pokročilú analýzu premávky. Jedným typom rozšírení sú **preprocesory** (***preprocessors***). Ich účelom je umožniť ľahké rozšírenie funkcionality pomocou modulárnych rozšírení. Kód preprocesoru je volaný pred samotnou detekciou, ale po tom čo bol paket dekódovaný. V základnej distribúcii nástroja Snort je k dispozícii viacero takýchto preprocesorov, napríklad nasledovné [**53**]:

- `frag3` – modul pre defragmentáciu IP premávky.
- `stream5` – preprocesor pre rekonštrukciu TCP spojení.
- `HTTP Inspect` – generický HTTP dekóder.
- `SMTP Preprocessor` – dekóder SMTP protokolu pre používateľskú aplikáciu.

Okrem tých to pribalených preprocesorov je možné vytvoriť aj vlastný proprocesor. Snort pre tento účel poskytuje rozhranie pre preprocesory. Prerocesor môže taktiež používať existujúcu funkcionalitu iného preprocesora.

Po tom čo premávku spracujú aktívne preprocesory, odovzdá sa ďalej detektoru. Ten vykoná detekciu na základe príznakov. Výsledok sa zaznamená spravidla do logovacieho súboru. Obrázok 9 znázorňuje celý tento proces [**54**].




*Obrázok 9: Priebeh spracovania paketov nástrojom Snort [**54**]*

### 8.1.1 *Preprocesor Heuristate*

Pre zaznamenávanie prechodov v stavových automatoch protokolov bol vytvorený preprocesor *Heuristate*. Jeho autorom je *Peter Marko* [**50**]. Toto riešenie bude rozšírené a použité ako základ pre extrakciu čŕt, na základe ktorých sa budú určovať anomálie. Tento modul používa služby preprocesora `stream5`. V režime učenia zaznamenáva štatistiky prechodov a štatistickými metódami určuje hraničné hodnoty, ktoré majú byť použité vo fáze detekcie. Cieľom rozšírenia je preskúmať možnosti nahradenia štatistických metód sofistikovanejšími metódami strojového učenia, ktoré by mohli odhaliť aj neznáme útoky, ktoré sú týmto metódami modelovateľné.

Preprocesor *Heuristate* pri každom prechode sleduje starý a nový stav v stavovom automate a na základe tohto pozorovania určí, o ktorý prechod ide. Pre protokol TCP sú v module definované nasledujúce stavy:

- `NONE` – počiatočný stav.
- `LISTEN` – čakanie na spojenie od vzdialenej strany.
- `SYN_SENT` – čakanie na zodpovedajúcu požiadavku (SYN+ACK) po odoslaní požiadavky (SYN).
- `SYN_RCVD` – čakanie na potvrdenie spojenia (***acknowledgement***) po prijatí správ SYN v oboch smeroch.
- `ESTABLISHED` – reprezentuje otvorené spojenie, normálny stav pre fázu prenosu údajov.
- `FIN_WAIT_1` – čakanie na požiadavku o ukončenie spojenia od vzdialenej strany alebo potvrdenie predtým odoslanej požiadavky o ukončenie spojenia.
- `FIN_WAIT_2` – čakanie na požiadavku o ukončenie spojenia zo vzdialenej strany
- `CLOSE_WAIT` – čakanie na požiadavku o ukončenie spojenia z lokálnej strany.
- `CLOSING` – čakanie na potvrdenie ukončenia spojenia od vzdialenej strany.
- `LAST_ACK` – čakanie na konečné potvrdenie ukončenia spojenia.
- `TIME_WAIT` – čakanie po dostatočne dlhú dobu na to, aby sa zaistilo prijatie potvrdenia ukončenia na vzdialenej strane.

- `CLOSED` – ukončené spojenie, nepredstavuje žiadny stav spojenia.

Tieto stavy zodpovedajú stavom v štandarde *RFC 793* [**55**]. Pre protokol UDP sa sledujú nasledovné umelo vytvorené stavy:

- `NONE` – pre komunikáciu neboli zaznamenané žiadne pakety.
- `CLIENT_SEEN` – pakety pre danú komunikáciu boli zaznamenané len v jednom smere (od klienta smerom k serveru).
- `ESTABLISHED` – boli zachytené pakety v oboch smeroch (od klienta k serveru aj naopak).
- `TIMEDOUT` – vypršal časový limit spojenia t.j. po stanovenú dobu nebol pre danú komunikáciu zaznamenaný žiadny paket.

Pre oba protokoly je vytvorené matica prechodov, v ktorej prvý rozmer (riadok) predstavuje starý stav a druhý rozmer (stĺpec) reprezentuje nový stav. Stav spojenia sa vždy sleduje pre oba konce komunikácie.

Výstupom preprocesoru v režime učenia je súbor `heuristatat.log` obsahujúci priemerné počty jednotlivých prechodov v 7 rôzny časových úsekoch (od `1ms` po `1 000 000 ms`). Ďalším výstupným súborom je `learnt_thresholds.csv`, do ktorého sa zaznamenávajú vypočítané hraničné miery odchýlok.

Preprocesor *Heuristate* je rozšírený o zaznamenávanie všetkých počtov prechodov v rámci všetkých intervalov do súboru `tukabel.log`. Ku každému časovému intervalu sa okrem jeho dĺžky v ms ukladá aj časTento výstup je pre ľahšie ladenie a testovanie ďalej spracovaný v oddelenom nástroji, kde sa aplikujú metódy strojového učenia pre vytvorenie profilov pre jednotlivé stanice.

## 8.2 Prvotná analýza extrahovaných čŕt komunikácie

Pred použitím extrahovaných **čŕt** (***features***) komunikácie je vhodne overiť ich výpovednú hodnotu. Na tento účel boli použité vybrané druhy útokov z datasetu *UNB ISCX IDS* [**51**].

### 8.2.1 *Dataset UNB ISCX IDS*

Pre demonštratívne overenie vhodnosti extrahovaných **čŕt** (***features***) komunikácie bol použitý **dataset UNB ISCX IDS** *(University of New Brunswick, Information Security Centre of Excellence, Intrusion Detection Evaluation DataSet*) [**51**]. Princípy, na ktoré detailnejšie popisujú tento dataset a spôsob jeho generovania, sú zhrnuté v článku od *Shivariho* a jeho spolupracovníkov [**56**].

Hlavnou myšlienkou tohto datasetu je vytvoriť realistickú premávku, ktorá je označená a je teda možné rozlíšiť anomálnu premávku od normálnej. Dataset pozostáva z `.pcap` súborov, ktoré zachytávajú kompletnú sieťovú premávku a XML súborov, ktoré k jednotlivým komunikáciám (spojeniam) **označenie** (***label***), ktoré určuje či ide o anomália a taktiež o aký typ útoku sa jedná.

Dataset je rozdelený na 7 dní, pričom 3 dni obsahujú len normálnu premávku. Zvyšné 4 dni obsahujú okrem normálnej premávky aj anomálnu premávku s rôznymi útokmi. Jednotlivé dni sú opísané v tabuľke nižšie.

*Tabuľka 2: Obsah datasetu UNB ISCX IDS [**51**]*

| Deň | Dátum | Popis | Veľkosť (GB) |
|---|---|---|---|
| Piatok | 11.6.2010 | Normálna premávka | 16,1 |
| Sobota | 12.6.2010 | Normálna premávka | 4,22 |
| Nedeľa | 13.6.2010 | Infiltrácia siete z vnútra + Normálna premávka | 3,95 |
| Pondelok | 14.6.2010 | HTTP DoS + Normálna premávka | 6,85 |
| Utorok | 15.6.2010 | DDoS s použitím IRC botnentu + Normálna premávka | 23,4 |
| Streda | 16.6.2010 | Normálna premávka | 17,6 |
| Štvrtok | 17.6.2010 | Útok hrubou silou na SSH + Normálna premávka | 12,3 |

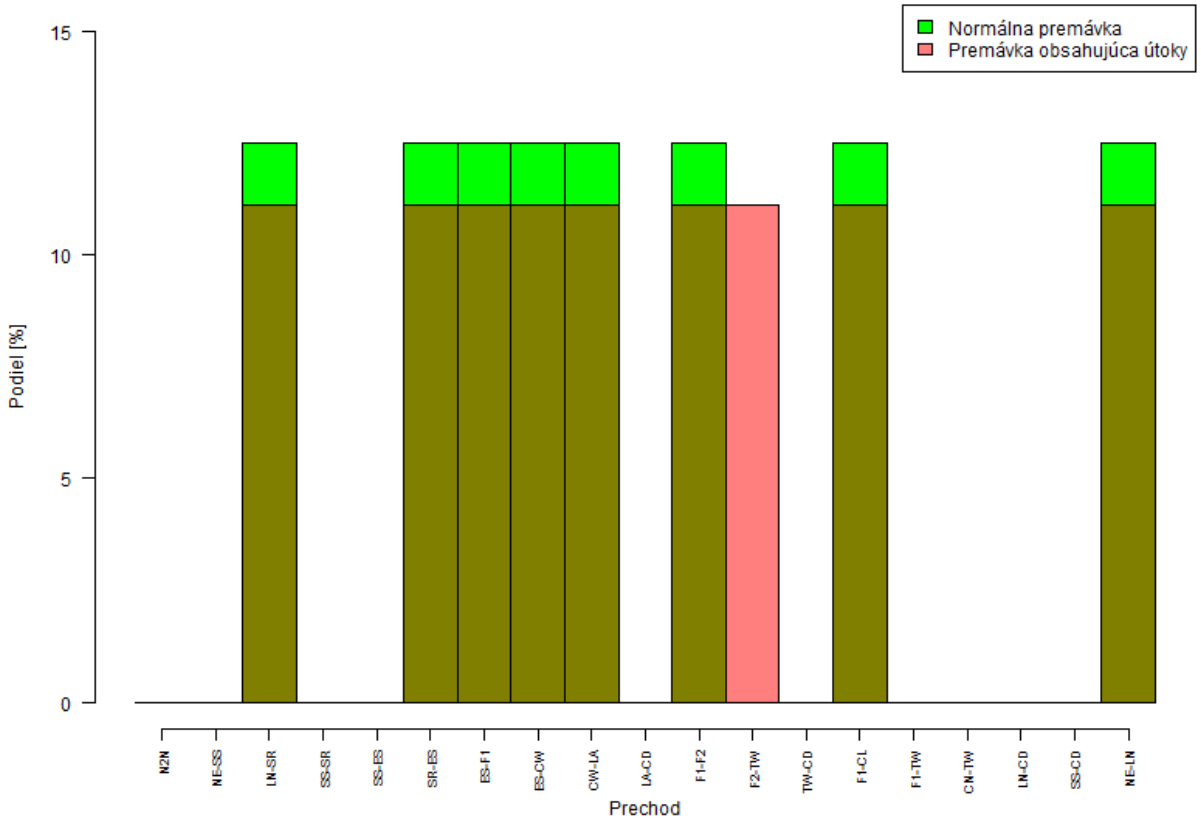
### 8.2.2 *Základné vyhodnotenie distribúcie prechodov pre normálnu a anomálnu premávku*

Záznamy premávky pre dni 11 (normálna premávka) a 15 (obsahuje aj anomálnie) boli analyzované pôvodným preprocesorom *Heuristate*. Jeho výstupom sú profily pre každú IP adresu obsahujúce priemery distribúcií prechodov pre jednotlivé časové úseky (od `1ms` po `1 000 000 ms`). Výsledná distribúcia bola zobrazená vo forme histogramu pre každú IP adresu oddelene. Nižšie (Obrázok 10 až Obrázok 13) sú zobrazené výsledky pre primárny webový server (cieľová IP adresa `192.168.5.122`) počas dňa 11 (normálna premávka, svetlo zelená) a dňa 15 (útok na webový server, ružová).

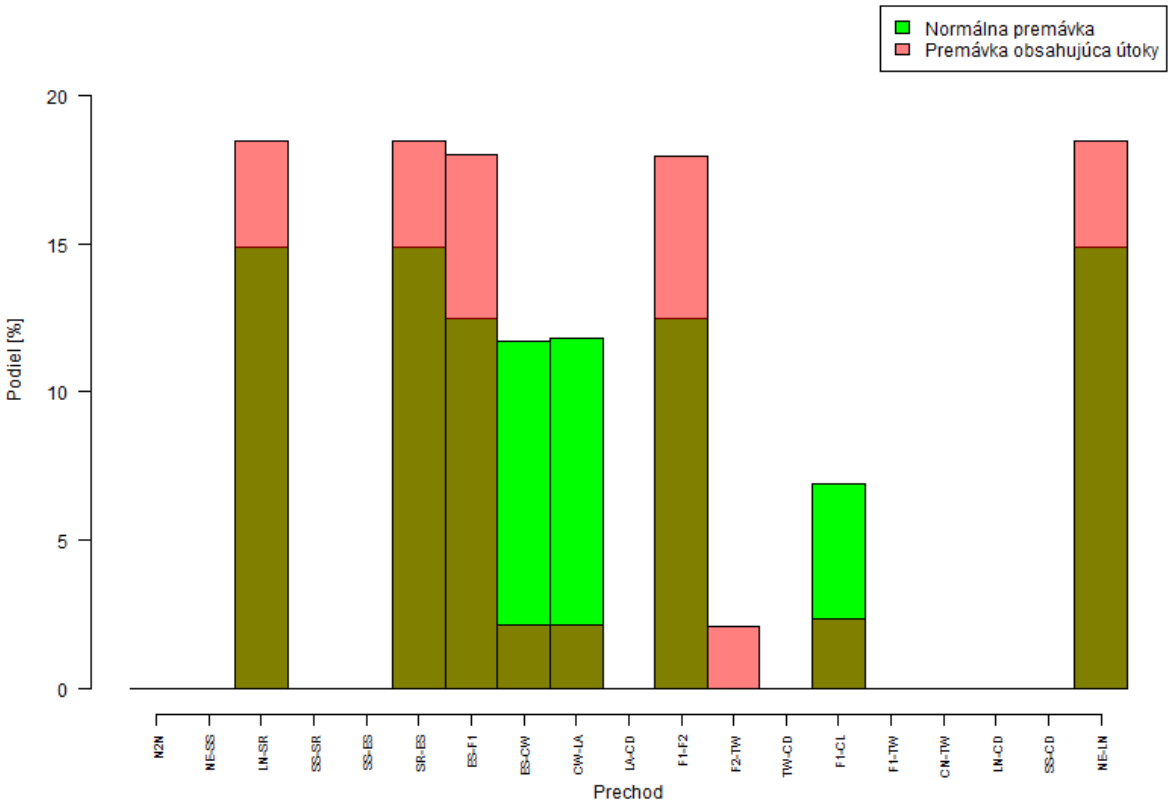
Svetlo zelené stĺpce predstavujú normálnu premávku, ružové predstavujú abnormálnu premávku a olivová farba je ich prienikom. Keďže bola distribúcia pre časové intervaly `1 ms`, `10 ms`, `100 ms` a `1 000 ms` takmer identická, uvedený je len histogram prvý z týchto časových intervalov.

Z histogramov je možné usúdiť, že pre každý časový interval existuje badateľný rozdiel v distribúcii priemerného počtu prechodov medzi stavmi v TCP stavovom automate. Napríklad pre časové intervaly `1 ms` až `1000 ms` sa podiel prechodu `F2-TW` zvýšil z nuly na vyše 10 percent na úkor ostatných prechodov. Toto možno považovať za odchýlku od normálneho správania. Obdobné zmeny možno pozorovať aj pri ostatných časových intervaloch.

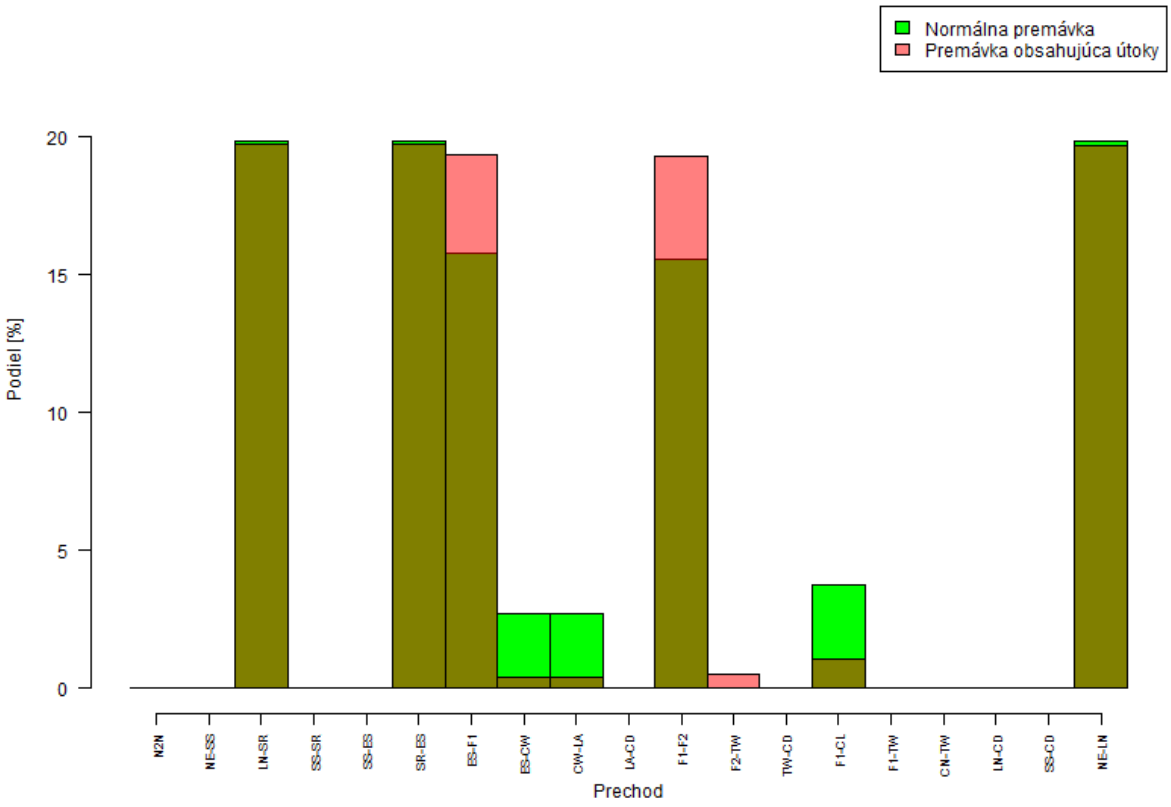
Z týchto pozorovaní vyplýva, že sledovanie distribúcie prechodov môže byť vhodnou črtou pre detekciu anomálií v sieťovej premávke.



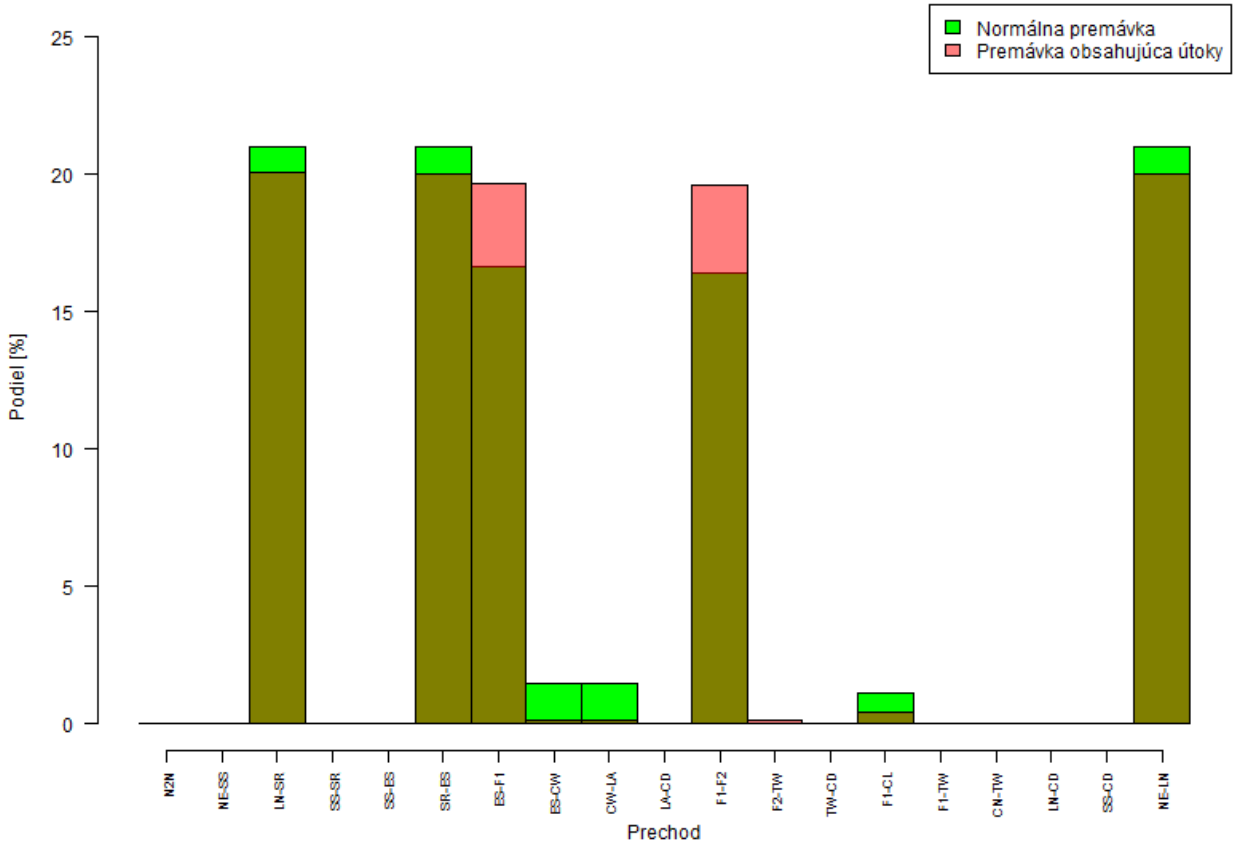

*Obrázok 10: Priemerné distribúcie prechodov TCP pre webový server – interval 1 ms*




*Obrázok 11: Priemerné distribúcie prechodov TCP pre webový server – interval 10 000 ms*




*Obrázok 12: Priemerné distribúcie prechodov TCP pre webový server – interval 100 000 ms*



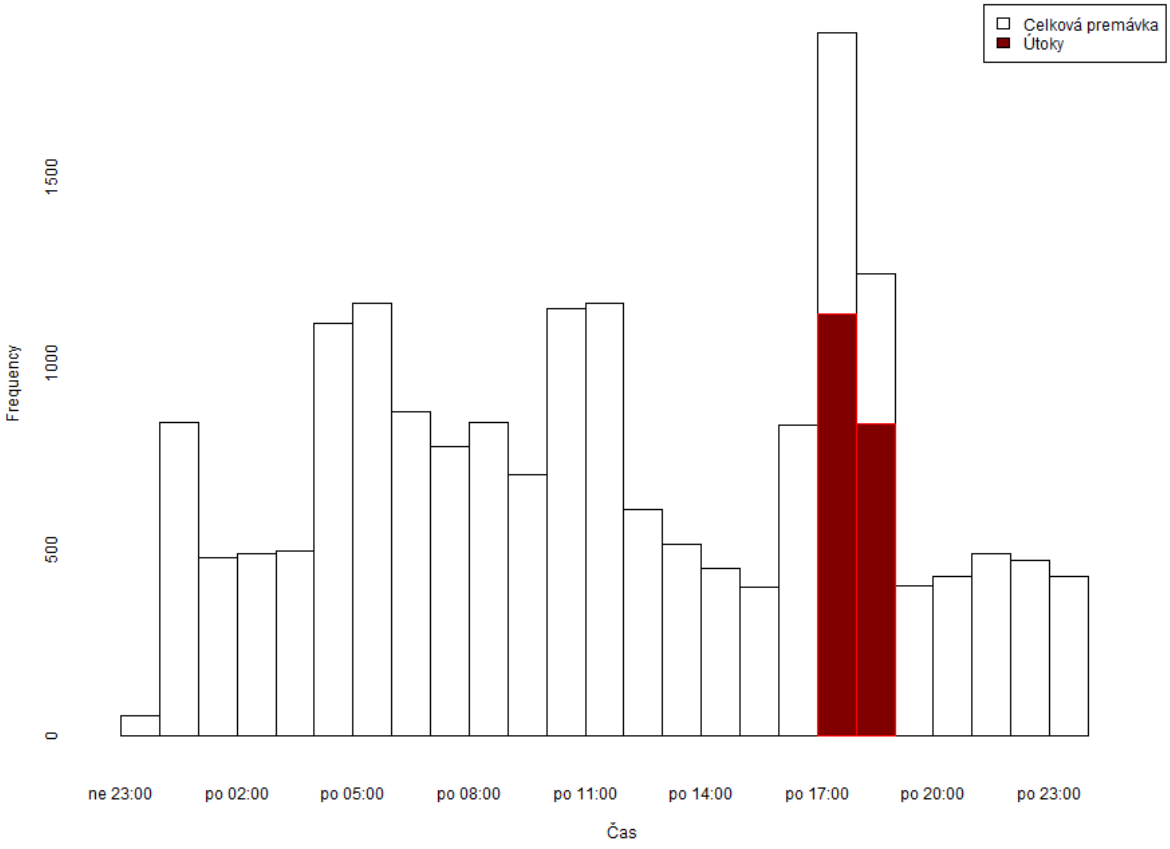

*Obrázok 13: Priemerné distribúcie prechodov TCP pre webový server – interval 1 000 000 ms*

### 8.2.3 *Identifikácia prechodov predstavujúcich anomálnu komunikáciu*

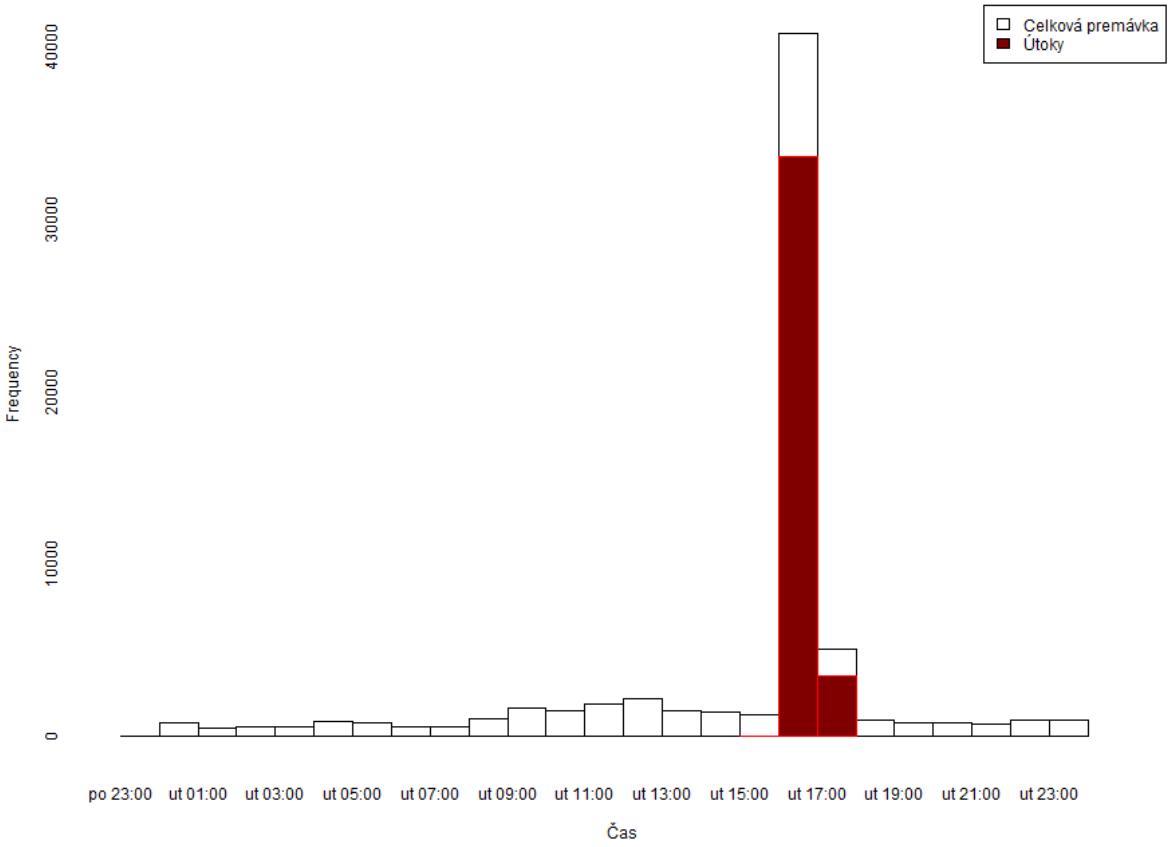
Pre overenie vhodnosti čŕt boli zvolené dni 14 (HTTP DoS) a 15 (DDoS). Keďže je k dispozícii priradenie označení ku komunikáciám a našim cieľom je označiť extrahované črty reprezentujúce počty prechodov medzi stavmi v stavovom automate. Pre lepšiu demonštráciu sa v tejto podkapitole bude brať do úvahy iba spojenia, ktorých cieľom je hlavný webový server, na ktorý sa v oboch prípadoch útočilo (cieľová IP adresa `192.168.5.122`) .

Pre tento cieľ boli na základe priradených označení v XML súbore datasetu vytvorené histogramy zobrazujúce rozloženie spojení v priebehu dňa (Obrázok 14 a Obrázok 15 nižšie). Biele stĺpce predstavujú celkový počet komunikácií v danú hodinu a červené stĺpce reprezentujú počet útokov v danú hodinu.

Pre oba dni sú takmer všetky útoky sústredené do intervalu 2 hodín, pričom jeden z intervalov obsahuje výraz väčší počet spojení.




*Obrázok 14: UNB ISCX IDS – rozloženie normálnych a anomálnych spojení v priebehu dňa 14.*

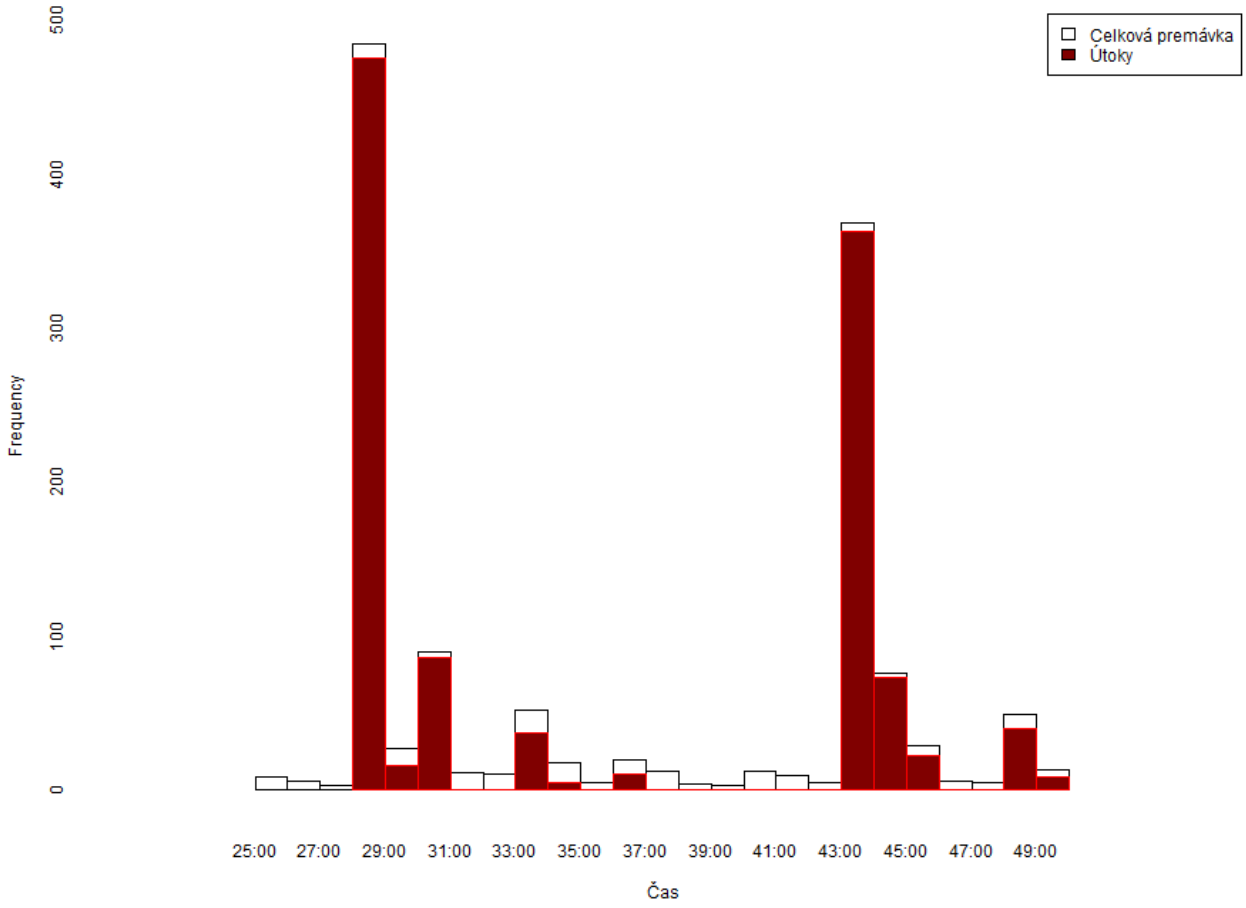



*Obrázok 15: UNB ISCX IDS – rozloženie normálnych a anomálnych spojení v priebehu dňa 15.*

Pre deň 15 (DDoS) v hodinovom intervale od `16:00` po `17:00` je pomer útokov k celkovej premávke `82,40%`. Tento interval by mal postačovať ako reprezentatívna vzorka anomálnej premávky pre demonštráciu vhodnosti čŕt.

Pre deň 14 (HTTP DoS) je tento pomer pre hodinu s najväčším počtom útokov (`17:00` – `18:00`) nižší (`60,02%`). Preto bola v rámci tohto hodinového intervalu zvolená minúta s najväčším počtom útokov (Obrázok 16). Minútu s najväčším počtom útokov predstavuje časový interval `17:28` – `17:29`. Pomer útokov k celkovej premávke reprezentuje `97,94%` , čo už je vhodná reprezentatívna vzorka.

Pre vyhodnotenie a vytvorenie grafov bol použitý programovací jazyk *R* a vývojové prostredie *RStudio*.




*Obrázok 16: UNB ISCX IDS – rozloženie normálnych a anomálnych spojení v minútových intervaloch pre deň 14 v čase od 17:25 do 17:50.*

## 8.3 Vytváranie profilov

Premávka pre deň 15 bola analyzovaná nástrojom *Snort* s modifikovaným preprocesorom *Heuristate*. Z výsledkov boli extrahované záznamy pre TCP spojenia smerujúce na primárny webový server (cieľová IP adresa `192.168.5.122`) .

Na základe zvolených časových intervalov reprezentujúcich anomálnu premávku zvolených v predchádzajúcej podkapitole boli vyčlenené črty pre premávku v výraznou prevahou anomálnych spojení. Taktiež bola obdobným spôsobom zvolená čas čŕt, ktoré reprezentujú normálnu premávku. Normálne dáta boli použití pre fázu učenia a pre overenie sa použila kombinácia normálnych a anomálnych dát.

### 8.3.1 *SVM*

Pre vytvorenie profilov sa použil balík *e1071* pre programovací jazyk *R*, ktorého súčasťou je metóda učenia **SVM** (***Support Vector Machines***). Balík *e1071* je rozhraním pre knižnicu *libsvm* v jazyku *R*.

Základným princípom metódy SVM je hľadanie nadroviny, ktorá v priestore príznakov optimálne rozdeľuje trénovacie údaje. Optimálna nadrovina je potom taká, že body ležia v opačných polopriestoroch a hodnota minima vzdialeností bodov od roviny je čo najväčšia. Inými slovami sa maximalizuje odstup (***maximal margin***). Táto metóda používa trik nazývaný **jadrový trik** (***kernel***

***trick***), ktorý umožňuje priestor príznakov previesť do priestoru vyššej dimenzie. Toto umožní previesť lineárne neseparovateľnú úlohu na lineárne separovateľnú.

Špeciálnym prípadom SVM je **detekcia noviniek** (***novelty detection***), taktiež nazývaná ***outlier detection*** alebo ***one-class-classification***. Táto metóda deteguje položky, ktoré ležia mimo datasetu (***outliers***). SVM detekcia noviniek vytvára sférickú rozhodovaciu hranicu okolo súboru dátových bodov pomocou súboru podporných vektorov (***support vectors***) opisujúcich sférickú hranicu. Prvotný optimalizačný problém pre SVM detekciu noviniek je nasledovný [**57**]:

$$\min \quad t(w, \xi, \rho) = \frac{1}{2}||wk|| - \rho + \frac{1}{m\nu}\sum_{i=1}^{m}\xi i$$

$$\text{s.t.} \ \langle \Phi(x_i), w \rangle + b \geq \rho - \xi i \qquad (i = 1, \ldots, m)$$

$$\xi i \geq 0 \qquad (i = 1, \ldots, m).$$

Kde *v* je parameter používaný na reguláciu objemu gule a v dôsledku toho aj počtu položiek ležiacich mimo. Hodnota *v* taktiež určuje hornú hranicu podielu položiek ležiacich mimo (***outliers***).

### 8.3.2 *Aplikácia metódy SVM*

Model pre údaje extrahované vyššie opísaným spôsobom bol vytvorený pre rôzne kombinácie parametrov (napr. rôzne jadrové funkcie, hodnoty parametra *v*). Bolo otestovaných 400 kombinácií parametrov, pričom sa sledovali nasledujúce miery výkonnosti:

- **Presnosť** (***accuracy***) = `(TP + TN) / (TN + FN + TP + FP)`
- **Senzitivita** (***sensitivity***) = `TP / (TP + FN)`
- **Specificita** (***specificity***) = `TN / (TN + FP)`
- **Precíznosť** (***precision***) = `TP / (TP + FP)`

kde:

- `TP` = ***true positive***
- `TN` = ***true negative***
- `FP` = ***false positive***
- `FN` = ***false negative***

Pre každú mieru výkonnosti bola do nasledujúcej tabuľky zvolená kombinácia parametrov, v ktorom bola maximálna.

*Tabuľka 3: Vyhodnotenie použtia metóda SVM detekcia noviniek*

| **Maximalizovaná vlastnosť** | **Presnosť [%]** | **Senzitivita [%]** | **Specificita [%]** | **Precíznosť [%]** |
|---|---|---|---|---|
| Presnosť | 78,41 | 0,00 | 99,97 | 0,00 |
| Senzitivita | 21,71 | 98,54 | 0,58 | 21,42 |
| Specificita | 78,41 | 0,00 | 99,97 | 0,00 |
| Precíznosť | 35,37 | 78,4 | 23,54 | 21,99 |

Miery precíznosť, senzitivita a specificita sa podarilo dosiahnuť relatívne vysoké, nie však súčasne všetky. Nízka maximálna hodnota precíznosti 21,99% znamená že daný model by mal veľmi veľký počet falošných detekcií (***false positives***). Pre tento prípad model preto nie je vhodný.

Tento problém by mohla vyriešiť analýza a testovanie dodatočných parametrov metódy SVM detekcia noviniek v balíčku *e1071* a/alebo vytvorenie pokročilých agregovaných čŕt opisujúcich priebeh distribúcie prechodov medzi stavmi v stavovom automate.

# 9 Implementácia

V tejto časti dokumentu sú zhrnuté detaily implementácie a testovania/hodnotenia zvolených riešení. Riešenie je postavené na nástroji Ucollect [**58**], ktorý je súčasťou projektu Turris [**3**]. Samotne riešenie je implementované ako zásuvný modul pre nástroj Ucollect. Vytvorený zásuvný modul je pomenovaný *StateTrans*.

## 9.1 Voľba implementačného prostredia

Výsledné riešenie by v ideálnom prípade malo dať nasadiť priamo na domáci smerovač. Domáce smerovače aj v porovnaní s počítačom nízkej triedy k dispozícii menej výpočtových prostriedkov. Zahltenie týchto prostriedkov na smerovači by mohlo spôsobiť veľké oneskorenie v sieťovej komunikácii alebo, v horšom prípade, aj úplné prerušenie sieťovej komunikácie. Preto je dôležité, aby zvolené implementačné prostredie využívalo zdroje efektívne. Vhodným programovacím jazykom preto bude kompilovaný jazyk napr. *C* alebo *C++*. Kompilované programovacie jazyky oproti interpretovaným spravidla efektívnejšie využívajú prostriedky.

Do úvahy prichádzajú dva nástroje, ktoré by bolo možné rozšíriť: *Snort* [**52**] a *Ucollect* [**58**].

Pre nástroj *Snort* je k dispozícii veľké množstvo rozšírení. Ich súčasťou je napríklad aj stavový automat pre sledovanie stavu spojenia/konverzácie. Tento stavový automat používa aj preprocesor *heuristate* použitý v kapitole 8. Aj keď je nástroj Snort označovaný ako ľahký (***lightweight***), jeho nasadenie na smerovač s cieľom sledovať stavový automat spojení by stále nebolo dostatočne efektívne. Jednotlivé rozšírenia (preprocesory) a takisto aj jadro nástroja totiž obsahujú a vykonávajú komponenty, ktoré sledujú aj prebytočné informácie.

Naproti tomu nástroj *Ucollect* bol vyvinutý s cieľom nasadiť ho na domáci smerovač. Umožňuje rozšírenie pomocou zásuvných modulov. Pre účely nástroja vyvíjaného v tejto práci je *Ucollect* postačujúci ako základ, avšak zároveň využíva prostriedky efektívne. Jeho nasadenie na otvorené systému (ako napr. *OpenWrt*) by taktiež malo byť jednoduchšie.

Pre horeuvedené dôvody bol ako základ pre implementáciu navrhnutého nástroja zvolený nástroj **Ucollect**.

## 9.2 Nástroj Ucollect

Nástroj Ucollect je démon[5] určený na zber a analýzu sieťovej premávky. Je napísaný v jazyku *C* a umožňuje rozšírenie prostredníctvom zásuvných modulov (***plugins***). Samotný nastroj po skompilovaní poskytuje dve verzie: jedna umožňuje komunikáciu zásuvných modulov so serverovou časťou a druhá je určená výhradne na lokálne spustenie.

Nástroj taktiež obsahuje serverovú časť (jazyk *python*). Tá taktiež umožňuje rozšírenie prostredníctvom zásuvných modulov. Jadro klientskej časti a jadro serverovej časti poskytujú zásuvným modulom rozhranie, prostredníctvom ktorého môže modul na klientskej časti komunikovať s rovnomenným zásuvné modulom na serverovej časti.

Autori nástroja v dokumentácii priloženej ku zdrojovému kódu (vo forme textových súborov) vyzdvihli fakt, že nástroj z implementačného hľadiska využíva miestami špecifické prístupy. Preto je potrebné pri vytváraní zásuvných modulov dodržiavať niekoľko pravidiel a miestami sa

[5] Démon (***daemon***) je proces, ktorý beží na pozadí a nie je pod priamou kontrolou používateľa.

prispôsobiť sa filozofii nástroja. V ďalšej časti sú popísané špecifické črty nástroja a pravidlá, ktoré je potrebné/vhodné pri vývoji zásuvných modulov dodržiavať. Počas celého vývoja zásuvného modulu sa kládol dôraz na to, aby boli tieto pravidlá a prístupy v čo najväčšej miere dodržané.

### 9.2.1 *Filozofia nástroja*

Verzia nástroja *Ucollect* nasadzovaná na smerovače v rámci projektu *Turris*, sleduje pakety na rozhraní do Internetu (WAN). Obsahuje niekoľko zásuvných modulov, ktoré zbierajú rôzne štatistiky o sieťovej premávky, analyzujú logy z bezpečnostnej brány a aktualizácií firmvéru. Tieto údaje sa centralizovane vyhodnocujú a slúžia jednak na zlepšenie bezpečnostných nastavení ale aj odladenie chýb v samotnom systéme projektu *Turris*.

Pre zber dát sa používajú iba informácie z hlavičiek. Neskúmajú sa samotné užitočné dáta (***payload***). Tento prístup sa bude snažiť nasledovať vyvíjaný aj implementovaný zásuvný modul.

Pre zvýšenie zabezpečenia komunikácie medzi klientom a serverom je smerovači vyvinutom v rámci projektu *Turris* umiestnený šifrovací čip *Atmel ATSHA204*. Aby bolo možné nástroj *Ucollect* plnohodnotne využívať aj na iných systémoch, bola v rámci knižnice pre komunikáciu s týmto šifrovacím čipom vytvorená vrstva pre emuláciu tohto čipu. Ide o knižnicu *libatsha204,* dostupnú na repozitároch projektu *Turris* [**59**].

### 9.2.2 *Správa pamäte*

Štandardná manuálna správa pamäte v jazyku C je náchylná na chyby. Taktiež neumožňuje navzájom separovať zásuvný moduly.

Nástroj Ucollect používa namiesto tohto štandardného prístupu používa systém pamäťových blokov (***memory pools***). Každý kus pamäte je alokovaný z pamäťového bloku. Alokované kusy pamäte sa však neuvoľňujú individuálne. Jediný možný spôsob ako uvoľniť pamäť, je resetovať celý pamäťový blok (alebo ho zničiť, čo ho implicitne resetuje). Tento krok uvoľní všetku alokovanú pamäť v danom pamäťovom bloku.

Každý pamäťový blok má definovaný životnosť. K dispozícii sú pamäťové bloky, ktoré počas existencie zásuvný modulu nie sú nikdy resetované a taktiež pamäťové bloky, ktoré sú určené na krátkodobú alokáciu pamäte a po návrate riadenia zo zásuvného modulu do jadra nástroja sú resetované. Zásuvné moduly môžu vytvárať dodatočné pamäťové bloky, ktorých životnosť bude kontrolovať samy. Pre každú potrebný alokovaný kus pamäte je vhodné rozhodnúť, ako dlho bude potrebný a na základe tohto zvoliť pamäťový blok s najkratšou životnosťou.

Použitie pamäťových blokov prináša na jednej strane nevýhody (napr. nemožno uvoľniť alokované kusy pamäte individuálne. Na druhej strane však tento prístup prináša viaceré výhody:

- Sú rýchlejšie
- Majú menšie režijné požiadavky na pamäť
- Nie je potrebné explicitne uvoľňovanie alokovaných kusov pamäte. Dôsledkom tohto je, že použitie pamäťových blokov znižuje riziko úniku zdrojov (***resource leak***)
- Všetka pamäť asociovaná s zásuvným modulom je pri odstraňovaní modulu automaticky uvoľnená

### 9.2.3 *Architektúra zásuvných modulov*

Zásuvný modul pre nástroj Ucollect je zdieľaná knižnica s preddefinovanou vstupnou funkciou (nazýva sa `plugin_info`). Táto funkcia vracia štruktúru s popisom zásuvného modulu.

Popis zásuvného modulu (`struct plugin`) obsahuje informácie o zásuvnom module. Najzaujímavejšou časťou sú **funkcie spätného volania** (***callbacks***), ktoré sú volané pri rôznych udalostiach. Každá funkcia spätného volania môže byť nastavená na hodnotu `NULL`. V taktom prípade sa daná funkcia pri vzniku danej udalosti nič nevolá.

Všetky funkcie spätného volania majú spoločný parameter typu `struct context *`, ktorý predstavuje kontext aktuálne volanej funkcie. Tento kontext obsahuje užitočné informácie – permanentný a dočasný pamäťový blok, ukazovatele na niektoré objekty v jadre nástroja a ukazovateľ na používateľské dáta zásuvného modulu. Zásuvný modul by nemal meniť žiadnu položku okrem ukazovateľa na používateľské dáta, ktorý môže použiť na ľubovoľný účel.

Nasleduje stručný popis funkcií spätného volania:

- `init_callback`
  Volá sa pri inicializácii zásuvného modulu. V tejto funkcii sa očakáva nastavenie ukazovateľa na používateľské dáta zásuvného modulu (`user_data`).
- `finish_callback`
  Volá sa pred odstránením zásuvného modulu. Keďže o odstránenie pamäťových blokov je postarané automaticky, zvyčajne je nastavený na hodnotu NULL.
- `packet_callback`
  Volá sa pre každý paket zachytený nástrojom ucollect. Paket je funkcii odovzdaný na „inšpekciu“ prostredníctvom parametra. Okrem nespracovaných dát sú k dispozícii aj vypočítané hodnoty dôležitých vlastností (napr. adresa odosielateľa a prijímateľa, smer a pod.)
- `uplink_data_callback`
  Volá sa v prípade, že riadiaci server odošle dáta určené pre daný zásuvný modul. Dáta odovzdané v parametri a ich obsah je definovaný zásuvným modulom.
- `uplink_connected_callback`
  Volá sa, keď zásuvný modul nadobudne schopnosť komunikovať so serverom. Toto znamená že sa úspešne nadviazalo spojenie so serverom.
- `uplink_disconnected_callback`
  Volá sa pri stratení spojenia so serverom. Po opätovnom vytvorení spojenia sa zavolá funkcia spätného volania `uplink_connected_callback`.
- `fd_callback`
  Volá sa, keď si zásuvný modul zaregistruje sledovanie popisovača súboru a popisovať súboru sa stane pripravený na čítanie.
- `config_check_callback`
  Načítanie novej konfigurácie pre zásuvný modul. Zásuvný model skontroluje konfiguráciu, či je v poriadku. Nastavenia v nej však zatiaľ nezačne používať.

- `config_finish_callback`
  Na základe parametra sa buď začne používať nová konfigurácia načítaná pri volaní `config_finish_callback`, alebo sa táto konfigurácia zahodí.
- `child_died_callback`
  Volá sa po ukončení dcérskeho procesu nástroja ucollect. Táto funkcia je volaná aj v prípade, že ukončený dcérsky proces nebol vytvorený daným zásuvným modulom

V rámci zásuvných modulov je možné využívať knižnice pre zásuvné moduly nástroja Ucollect. Tieto knižnice definujú exportované funkcie a záduvný modul zase definuje aké funkcie importuje. Vytváranie takýchto knižníc je zjednodušené vďaka makrám pre import a export knižníc, ktoré sú definované v jadre nástroja Ucollect.

### 9.2.4 *Použité komponenty jadra*

V tejto časti sú popísané vybrané moduly jadra nástroja Ucollect, ktoré implementujú kľúčové dátové štruktúry a algoritmy použité pri implementácii. Každý modul jadra sa skladá z hlavičkového súboru a voliteľne implementácie (`.c` súbor).

#### *Hlavný cyklus loop*

Hlavný cyklus pre spracovanie udalostí `loop` poskytuje rozhranie prostredníctvom súboru `src/core/loop.h`. Spracováva nové dostupné dáta na súborových popisovačoch, nové zachytené dáta zo sieťových rozhraní a pod. Následne volá zodpovedajúce funkcie spätného volania a spravuje zásuvné moduly. Hlavný cyklus je srdcom celého nástroja.

#### *Zreťazený zoznam link_list*

Hlavičkový súbor `src/core/link_list.h` obsahuje makrá preprocesora pre vytváranie zreťazeného zoznamu. Jedná sa o špecifický typ hlavičkého súboru, pretože pri každom zahrnutí pomocou direktívy `#include` generuje nový kód.

Tento hlavičkový súbor je možné zahrnúť viackrát a zakaždým generuje nový kód. Vytvára niečo podobné ako šablóny (***templates***) v jazyku C++. Vlastnosti generovaného zoznamu sa dajú nastaviť prostredníctvom definícií pomocou direktívy `#define`.

Je možné napríklad nastaviť názov dátového typu predstavujúceho prvok v zozname, názov ukazovateľa v ňom smerujúceho na nasledujúci, prípadne prechádzajúci prvok a podobne sa dajú nastaviť ďalšie vlastnosti generovaného zreťazeného zoznamu.

Takýto zoznam je vhodný pre dynamické prepojenie záznamov, ktoré sú prechádzané najčastejšie lineárne.

#### *Správca pamäte mem_pool*

Modul `mem_pool` implementuje správu pamäte pomocou pamäťových blokov popísané v kapitole *9.2.2 Správa pamäte.* Poskytuje základne funkcie pre prácu s pamäťovými blokmi:

- `mem_pool_create()` – vytvorenie nového pamäťového bloku (***memory pool***).
- `mem_pool_destroy()` – odstránenie pamäťového bloku
- `mem_pool_alloc()` – alokovanie kusu pamäte z daného pamäťového bloku

- `mem_pool_reset()` – resetovanie daného pamäťového bloku

### *Prefixový strom trie*

Implementácia zhusteného prefixového stromu s názvom `trie`. K prvom v strom sa pristupuje prostredníctvom kľúča, ktorý ma variabilnú dĺžku. Modul poskytuje základné operácie: pre vytvorenie nového stromu, vyhľadanie/vloženie položky, prechádzanie stromu. Nie však implementovaná operácia odstránenia prvku a strom môže byť odstránený len ako celok resetovaním pamäťového bloku, v ktorom sa nachádza.

Prefixový strom je vhodný v prípadoch kde je potrebné na základe kľúča rýchlo vyhľadať/pridať prislúchajúcu položku.

### *Informácie o pakete*

Modul `packet` extrahuje zo surových dát zachytených rámcov významné vlastnosti. Extrahované vlastnosti (napr. adresy, porty, atď..) sú spolu s ukazovateľom na surové dáta uložené v štruktúre `struct packet_info`. Pomocou tejto štruktúry sa zásuvným modulom odovzdá nový paket pri zachytení na sieťovom rozhraní.

### *Rozhranie pre pridávanie zásuvných modulov*

Modul `plugin` poskytuje rozhranie pre pridávanie zásuvných modulov. Definuje štruktúru, prostredníctvom ktorej zásuvný modul odovzdá ukazovatele na svoje funkcie spätného volania.

### *Logovanie*

Modul `util` poskytuje funkcie pre logovanie na rôznych úrovniach .

## 9.3 Zásuvný modul StateTrans

Vyvíjaný zásuvný modul pre nástroj Ucollect implementujúci navrhnutý model bol nazvaný *StateTrans* (z angl. ***STAte TRANSition***). Pre lepšie pochopenie budú v tejto časti uvedené aj anglické názvy, ktoré v zdrojovom kóde používajú vo veľkej miere.

### 9.3.1 *Hlavné komponenty zásuvného modulu*

Obrázok 17 zobrazuje hlavné komponenty zásuvného modulu *StateTrans* v rámci nástroja *uCollect*. Šipky naznačujú hlavný tok údajov. Základné komponenty zásuvného modulu sú:

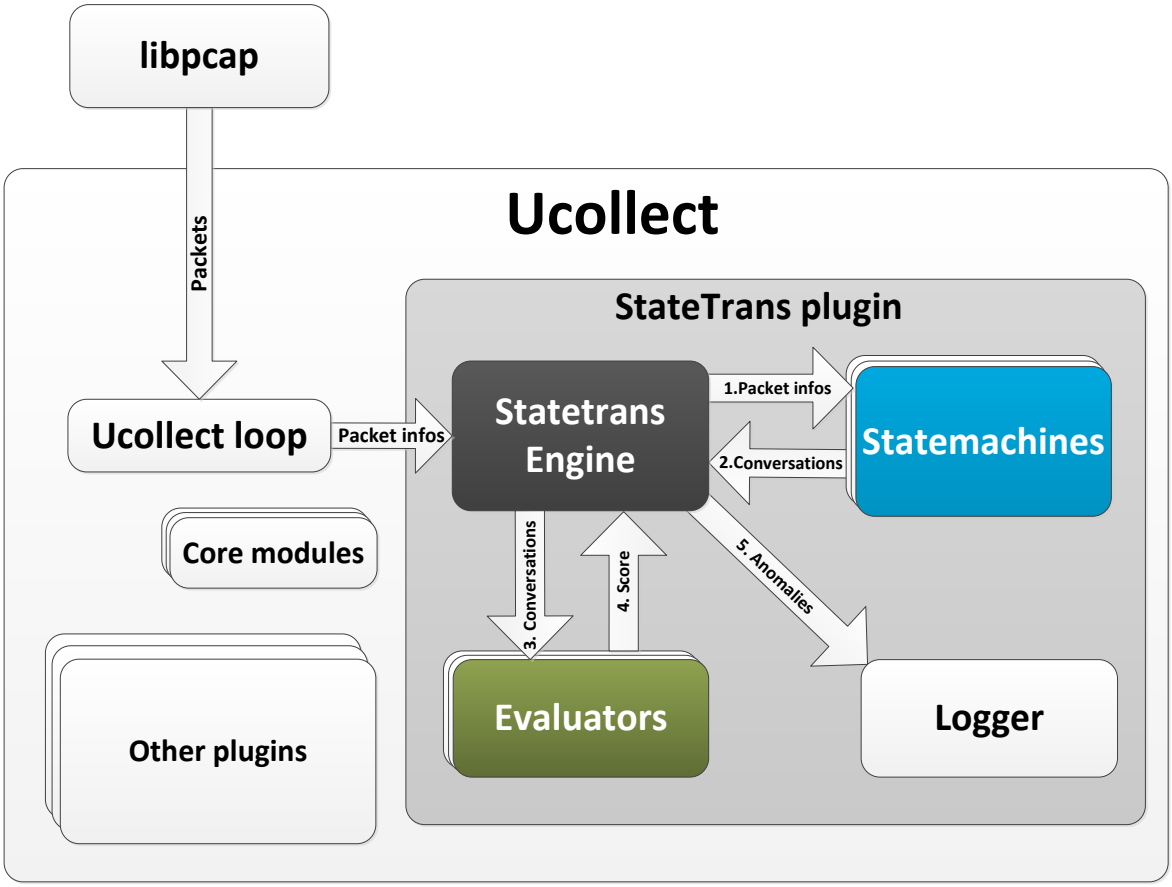
- Hlavná výkonná jednotka (***engine***)
- Stavové automaty (***statemachines***)
- Vyhodnocovače (***evaluators***)
- Logger

Na sieťovom rozhraní sa zachytávajú pakety. Knižnica *libpcap* ich odovzdáva zachytené pakete hlavnej slučke (***Ucollect loop***) nástroja *Ucollect*. V rámci hlavnej slučky sa zo zachyteného paketu extrahujú vlastnosti paketu a spolu s ukazovateľom na surové dáta a uložia do štruktúry `struct packet_info`. Táto štruktúra sa potom odovzdá prostredníctvom príslušnej funkcie spätného volania zásuvného modulu *StateTrans* (a taktiež ostatným zásuvným modulom, ktoré majú túto funkciu spätného volania nastavenú).

Zásuvný modul *StateTrans* môže obsahovať jeden alebo viac **stavových automatov** (***Statemachines***), ktoré sledujú pre rôzne protokoly stav aktívnych konverzácií a zbierajú informácie o vykonaných prechodoch. Po ukončení konverzácie stavový automat odovzdá informácie o danej konverzácii (vrátane počtu jednotlivých prechodov) **hlavnej výkonnej jednotke**(***Statetrans Engine***).

Zásuvný modul môže mať aktívny jeden alebo viac **vyhodnocovačov** (***Evaluators***). Hlavná výkonná jednotka zásuvného modulu po prijatí ukončenej konverzácie odovzdá túto konverzáciu všetkým aktívnym vyhodnocovačom. Tie v režime učenia vytvárajú na základe videných konverzácií profily pre jednotlivých hostiteľov na lokálnej sieti. V režime detekcie pre každú prijatú konverzáciu vypočítajú **skóre odchýlky** od bežného správania popísaného profilom.

Hlavná výkonná jednota vypočíta celkové skóre odchýlky od bežného správania a v prípade prekročenia hraničnej hodnoty sa daná udalosť zaznamená prostredníctvom komponentu pre logovanie (***Logger***).




*Obrázok 17: Diagram komponentov zásuvného modulu StateTrans*

### 9.3.2 *Hlavná výkonná jednotka rozširujúceho modulu*

Hlavná výkonná jednotka (***engine***) je srdcom celého zásuvného modulu. Stará sa o inicializáciu stavových automatov a vyhodnocovačov. Pre stavové automaty a vyhodnocovače je vytvorené rozhranie pre zjednodušené pridávanie nových modelov. V oboch prípadoch je toto rozhranie

podobné rozhraniu pre pridávanie zásuvných modulov do nástroja uCollect. Funkcia so stanoveným názvom vráti štruktúru obsahujúci ukazovatele na funkcie spätného volania pre podporované udalosti.

Hlavná výkonná jednotka koordinuje preposielanie údajov medzi stavovými automatmi a vyhodnocovačmi. Pri volaní funkcií spätného volania hlavná výkonná jednotka navyše pripraví pre každý stavový automat a každý vyhodnocovať jeho individuálny kontext. Kontext okrem iného obsahuje aj počet a zoznam dĺžok časových intervalov (***timeslots***), v ktorých sa majú počítať prechody.

Hlavná výkonná jednotka spravuje stromy profilov, ktoré predstavujú jednotné úložisko pre profily hostiteľov. Pre fázu učenia a fázu detekcie sú oddelené stromy profilov. Pre každý vyhodnocovač je vyhradená istá časť profilu s takou veľkosťou o akú vyhodnocovač požiadal. Cieľom tohto centralizovaného úložného priestoru pre profily hostiteľov je zmenšiť réžiu vyhľadávania profilu. Pri získaní ukončenej konverzácie od stavového automatu hlavná výkonná jednotka vyhľadá adekvátny profil hostiteľa len raz a túto akciu nemusí vykonávať každý vyhodnocovač individuálne. Pre vytvorenie databázy profilov bol použitý modul jadra s názvom `trie`.

### 9.3.3 *Stavové automaty*

Pre stavové automaty je vytvorené všeobecné rozhranie vo forme hlavičkového súboru `statemachine.h`. Štruktúra `struct statemachine` predstavuje popis stavového automatu, v ktorom sa okrem iného odovzdajú aj funkcie spätného volania. Implementovaný je stavový automat pre protokol TCP podľa časti návrh.

Hlavná výkonná jednotka zabezpečí, že pri prijatí nového paketu pre každý stavový automat zavolá funkciu `packet_callback()`, v ktorej by mal stavový automat priradiť získaný paket do správnej konverzácie, sledovať zmenu stavu konverzácie v dôsledku pridania daného paketu a zaznamenať prípadný prechod (ak nejaký nastal) v to v rámci rôznych časových intervalov.

Následne opakovane volá pre daný stavový automat funkciu `get_next_finished_conv_callback()`, ktorej úlohou je odovzdať ďalšiu ukončenú konverzáciu. Pokiaľ stavový automat interne už neeviduje ďalšiu ukončenú konverzáciu odovzdá hodnotu `NULL`.

#### *Sledovanie prechodov pre jednu konverzáciu*

Pre každú konverzáciu sa v rámci stavového automatu sleduje počet prechodov a to v rôznych časových intervaloch. Zoznam časových intervaloch stavový automat získa prostredníctvom kontextu, ktorý je argumentov volania funkcie spätného volania. Po ukončení spojenia sa pre každý časový interval vypočíta priemerný počet daných prechodov.

Pre každú konverzáciu sa v každom časovom intervale a pre každý možný prechod zbierajú nasledujúce údaje:

- `value` - počet prechodov v aktuálnom časovom intevale
- `aggr_value` – agregovaný počet prechodov (za všetky doterajšie prechody)
- `aggr_cnt` – počet časových intervalov, ktoré boli započítané do `aggr_value`

Po ukončení konverzácie je možné jednoducho vypočítať pre konkrétny časový interval a konkrétny prechod priemerný počet týchto prechodov za daný interval:

```
aggr_value / aggr_cnt.
```

Pre každý časový interval sa taktiež ukladá časová značka paketu, ktorý daný interval odštartoval. Pomocou tohto údaju sa pre nový paket na základe jeho časovej značky dá určiť, či pre niektorý časový interval ešte spadá do aktuálneho časového intervalu alebo je treba otvoriť nový časový interval. V prípade, že je potrebné otvoriť nový časový interval, starý interval sa započíta do agregovanej časti:

```
aggr_value += value
aggr_cnt++
```

Pre nový časový interval sa následne len vynuluje hodnota `value`.

Konverzácia môže byť ukončená či už prirodzeným dosiahnutím ukončeného stavu, ale aj vypršaním časového limitu od poslednej posledného videného paketu. Po zistení ukončeného stavu sa konverzácia odovzdá hlavnej výkonnej jednotke v rámci volania funkcie `get_next_finished_conv_callback()`.

#### *Úložisko aktívnych konverzácií*

Stavový automat potrebuje rýchle vyhľadávanie konverzácií podľa kľúča (IP adresy a porty). Na tento účel je vhodné použiť **prefixový strom** `trie`, ktorý poskytuje jadro. Keďže implementovaný prefixový strom nepodporuje operáciu mazania je potrebné tento problém vyriešiť. Po komunikácii s tvorcami nástroja *Ucollect* bol zvolený prístup, ktorý na podobný problém už používa zásuvný modul *refused*.

Základná myšlienka spočíva v tom, že do každej položky v prefixovom strome sa pridá atribút určujúci sa má položky vymazať. Raz za stanovený interval (časom alebo položiek označených na zmazanie) sa potom spustí implementovaná operácia nazývaná **konsolidácia** (***consolidation***). Konsolidácia spočíva vo vytvorený nového prefixového stromu v novom pamäťovom bloku a následnom skopírovaní tých položiek, ktoré nie sú označené na zmazanie. Po skopírovaní sa starý prefixový strom s aktívnymi spojeniami nahradí novým a pamäťový blok starého prefixového stromu sa resetuje (t.j. uvoľní sa).

Keďže je potrebné zároveň overovať aj to, či medzi časom pre niektorú konverzáciu nevypršal časový limit od posledného videného paketu, položky v prefixovom strome zároveň tvoria aj **zreťazený zoznam**. Použitý je zreťazený zoznam `link_list` z jadra nástroja Ucollect. V tomto zozname platí, že konverzácia, pre ktorú bol videný paket naposledy, sa vždy presunie na koniec zoznamu. Takto sa dosiahne, aby boli na začiatku zoznamu konverzácie, pre ktoré bol posledný paket registrovaný najdávnejšie a sú teda s väčšou pravdepodobnosťou po uplynutí časového limitu (***timed out***).

### 9.3.4 *Vyhodnocovače*

Pre vyhodnocovače je vytvorené všeobecné rozhranie vo forme hlavičkového súboru `evaluator.h`. Štruktúra `struct evaluator` predstavuje popis stavového automatu, v ktorom sa okrem iného odovzdajú aj funkcie spätného volania.

Vo fáze učenia hlavná výkonná jednotka odovzdá každú novú ukončenú konverzáciu každému vyhodnocovaču volaním jeho funkcie `learn_callback()`. Jedným z parametrov volania je aj ukazovateľ na časť profilu učenia hostiteľa, ktorá prislúcha danému vyhodnocovaču. Vo fáze detekcie sa obdobne volá pre každý vyhodnocovač funkcia `detect_callback()`, s tým rozdielom, že miesto profilu učenia sa odovzdá detekčný profil hostiteľa a táto funkcia vracia návratovú hodnotu, ktorá určuje **mieru odchýlky od bežného správania** (***anomaly score***).

Medzi režimom učenia a režimom detekcie sa v rámci režimu vytvárania profilov pre každého hostiteľa zavolá a každý vyhodnocovač zavolá funkcia `create_profile()`, ktorej úlohou je na základe dát uložených v profile učenia vytvoriť výsledný detekčný profil.

Implementovaný je vyhodnocovač, ktorý do profilu detekcie pre každého hostiteľa a každý časový interval vypočíta na základe videných **priemer** $\mu$ a **varianciu** $\sigma^2$ zaznamenaných priemerných počtov jednotlivých prechodov v rámci jednej konverzácie. Vo režime detekcie sa na určenie pravdepodobnosti anomálie použije **Čebyševova nerovnosť**.

### 9.3.5 *Komponent pre logovanie*

Komponent Logger slúži ako abstraktná vrstva pre logovanie výstupu detegovaných anomálii. Predvoleným výstupom je logovací súbor.

## 9.4 Možnosti centralizovaného vyhodnocovania

Vytvorené riešenie je možné vďaka rozhraniu nástroja Ucollect pre komunikáciu zásuvných modulov so serverovou časťou relatívne ľahko rozšíriť o odosielanie agregovaných dát na server. Mohli by sa napríklad odosielať zoznamy vzdialených IP adries, ktoré sú spojené s konverzáciami, ktoré boli s vysokou pravdepodobnosťou označené ako anomálne.

Na serverovej časti by sa vytvoril agregovaný pohľad na zozbierané údaje, podľa ktorého by bolo možné vygenerovať zoznam IP adries, ktoré často figurujú v spojeniach s vysokou pravdepodobnosťou anomálie. Tento zoznam by sa následne mohol spätne prešíriť do smerovačov vo forme pravidiel pre bezpečnostnú bránu.

# 10 Overenie riešenia

Pre overenie funkčnosti riešenia bol spustený nástroj *ucollect* so zapnutým rozširujúcim modulom *StateTrans*. Pre vytvorenie profilu boli vykonané akcie, ktoré simulujú bežné surfovanie na Internete. Po prepnutí do režimu detekcie bolo z rovnakého zariadenia vykonané skenovanie portov na verejnú IP adresu `1.1.1.1`.

## 10.1 Simulovanie bežnej aktivity v režime učenia

V režime bola generovaná HTTP premávka podobná surfovaniu po internete. Pre urýchlenie a automatizovanie bol použitý nástroj *curl*[6]. Pomocou tohto nástroja boli vykonané bežné dopyty, ktoré z technického hľadiska môžu zodpovedať dopytom vykonaným pomocou webového prehliadača.

Pre generovanie týchto dopytov bol vytvorený jednoduchý skript:

```
#!/bin/bash
for i in `seq 1 1000`
do
    curl www.google.com > /dev/null;
done
```

## 10.2 Generovanie anomálnej aktivity v režime detekcie

Po prechode do režimu detekcie bola vygenerovaná anomálne premávka. Anomálna premávka bola generovaná použitím nástroja *nmap*[7], ktorý slúži na skenovanie počítačových sietí, čo je možné považovať za potencionálne nebezpečnú aktivitu.

Príkaz použitý na generovanie TCP SYN skenovania:

```
nmap -PN -n -sS -p 1000-1100 1.1.1.1
```

Po vypršaní časového limitu polootvorených spojení rozširovací modul zaznamenal do logovacieho súboru nasledujúci výstup:

```
2016-05-08 02:07:09 [ANOMALY]: score(0.85) 10.0.2.15:48565 -> 1.1.1.1:1003
2016-05-08 02:07:09 [ANOMALY]: score(0.85) 10.0.2.15:48565 -> 1.1.1.1:1008
2016-05-08 02:07:09 [ANOMALY]: score(0.85) 10.0.2.15:48565 -> 1.1.1.1:1000
2016-05-08 02:07:09 [ANOMALY]: score(0.85) 10.0.2.15:48565 -> 1.1.1.1:1006
2016-05-08 02:07:09 [ANOMALY]: score(0.85) 10.0.2.15:48565 -> 1.1.1.1:1004
2016-05-08 02:07:09 [ANOMALY]: score(0.85) 10.0.2.15:48565 -> 1.1.1.1:1007
2016-05-08 02:07:09 [ANOMALY]: score(0.85) 10.0.2.15:48565 -> 1.1.1.1:1009
2016-05-08 02:07:09 [ANOMALY]: score(0.85) 10.0.2.15:48565 -> 1.1.1.1:1002
2016-05-08 02:07:09 [ANOMALY]: score(0.85) 10.0.2.15:48565 -> 1.1.1.1:1005
2016-05-08 02:07:09 [ANOMALY]: score(0.85) 10.0.2.15:48565 -> 1.1.1.1:1001
```

Anomálna aktivita bola úspešne detegovaná.

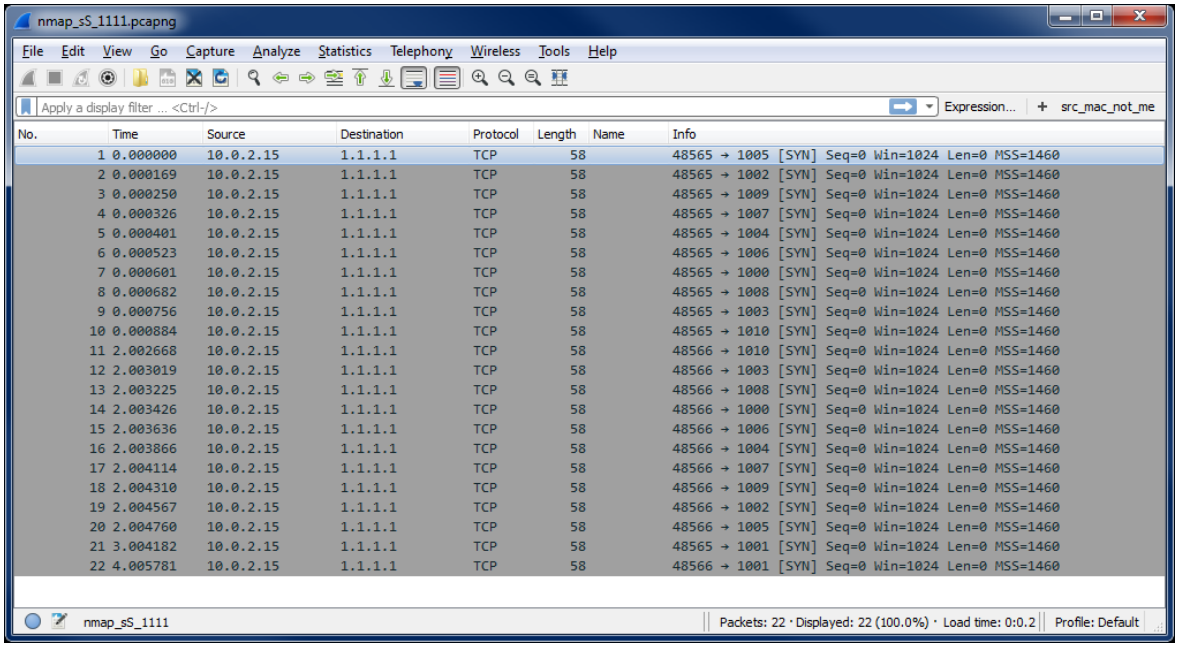
Daný výstup korešponduje s premávkou zachytenou pomocou nástroja Wireshark[8] na obrázku 18. Na premávke je vidieť že všetky spojenia obsahovali iba príznak SYN. To znamená, že sa

[6] https://curl.haxx.se/

[7] https://nmap.org/

[8] https://www.wireshark.org/

v stavovom automate pre protokol TCP vykonal len jeden prechod. Toto správanie sa zásadne líši od riadne vytvoreného a ukončeného spojenia. Preto bolo možno túto aktivitu označiť za anomálnu.




*Obrázok 18: Skenovanie portov metódou TCP SYN*

# 11 Zhodnotenie

V diplomovej práci sa podarilo zadefinovať pojem počítačová bezpečnosť a jej vlastnosti. Taktiež bol opísaný rozdiel medzi hrozbou a útokom. Bolo uvedené základné rozdelenie hrozieb a útokov. Ďalej bola analyzovaná bezpečnosť domácich počítačových sietí a pozícia domáceho smerovača v týchto sieťach. Bol opísaný model systému pre detekciu prienikov a zaviedla sa kategorizácia týchto systémov. Podrobne boli opísané metódy analýzy stavových protokolov a bol predstavený možný spôsob kombinácie tohto prístupu s klasickou detekciou anomálií.

V práci sa špecifikovali požiadavky na sieťový systém pre detekciu prienikov (NIDS) určený pre domácu sieť. Bol navrhnutý model založený na analýze stavových protokolov na sieťovej a transportnej vrstve v kombinácii s detekciou anomálií. Hlavnou myšlienkou modelu je sledovanie distribúcie prechodov v stavovom automate pre zvolené protokoly. Distribúcie prechodov sa sledujú vo viacerých časových intervaloch (napr. 1ms – 1h) pre pokrytie rôznych druhov komunikácie. V návrhu bol taktiež opísaný stavový automat pre protokol TCP a zaviedli sa umelé stavy pre nestavové protokoly (IP, UDP).

Pre vytváranie profilov a detekciu anomálií bola navrhnutá štatistická metóda. Tá spočíva v určení priemeru a rozptylu každej sledovanej hodnoty vo fáze učenia. Vo fáze detekcie sa použije Čebyševova nerovnosť, ktorá určí pravdepodobnosť anomálie. Pre pokročilejšie vyhodnocovanie by bolo možné nasadiť metódy výpočtovej inteligencie.

Ďalej boli navrhnuté dve alternatívy umiestnenia NIDS v domácej sieti. Prvé, preferované, riešenie integruje NIDS do smerovača. Druhé, alternatívne, riešenie separuje NIDS na dedikované zariadenie a je určené pre smerovače, ktorých výkon nepostačuje na navrhovanú detekciu prienikov v reálnom čase.

Pre účely experimentálneho overenia bolo rozšírené existujúce rozšírenie *Heuristate* pre nástroj *Snort,* ktoré implementuje podobný model. Experimentom bola demonštrovaná súvislosť medzi prítomnosťou anomálie a zmenou distribúcie prechodov v stavovom automate protokolu TCP. Pri experimentálnom overení boli vykonané pokusy s použitím strojového učenia pre vytváranie profilov. Testovaná metóda SVM detekcie noviniek (novelty detection) však nepriniesla postačujúce výsledky. Túto metódu vytvárania profilov by bolo potrebné ešte odladiť prípadne vyskúšať iné alternatívy.

Navrhnutý model bol implementovaný ako rozširujúci model pre nástroj *ucollect*, ktorý je súčasťou projektu *Turris*. Pri vývoji sa dbalo na modularitu a ľahkú rozšíriteľnosť modulu. Preto bolo vytvorené generické rozhranie pre pridávanie nových stavových automatov, a taktiež rozhranie pre pridávanie ďalších komponentov pre vyhodnocovanie anomálií. Implementovaný bol stavový automat pre protokol TCP a štatistický komponent pre vyhodnocovanie anomálií založený na Čebyševovej nerovnosti. Detegované anomálie sa zaznamenávajú do logovacieho súboru.

Implementovaný rozširovací modul by mohol byť v budúcnosti doplnený o ďalšie stavové automaty opisujúce ďalšie protokoly. Taktiež by sa dal ľahko rozšíriť o ďalšie komponenty pre vyhodnocovanie anomálií. Pre túto úlohu by sa mohli použiť metódy výpočtovej inteligencie.

Ďalšou oblasťou v ktorej by bolo možné vytvorený systém vylepšiť je využitie architektúry nástroja *ucollect*, ktorý umožňuje komunikáciu s centrálnym serverom. Server by mohol vyhodnocovať detegované anomálie a následne by sa prešírili pravidlá pre bezpečnostné brány na domácich

smerovačoch. Tieto pravidlá by blokovali komunikáciu s verejnými IP adresami, ktoré globálne figurujú vo veľkom množstve detegovaných anomálií.

# Použitá literatúra


1. *HomeNet Router* [online]. 2. Apríl. 2013 [cit. 2014-November]. Dostupné z: http://homenetrouter.com/blog/homenetrouter/us-average-of-internet-connected-devises-on-the-rise-national-average-is-5-7-devices-per-household

2. Dazeinfo. *Internet Around The World: 25% People Access Internet Through Mobile Devices!* [online]. 3. Apríl. 2014 [cit. 2014-November]. Dostupné z: http://www.dazeinfo.com/2014/04/03/internet-around-world-25-people-access-internet-mobile-devices/

3. *Project:Turris* [online]. 2014 [cit. November]. Dostupné z: https://www.turris.cz

4. STRAND, L. K. *Adaptive distributed firewall using intrusion detection*. University of Oslo, Department of Informatics, University of Oslo, Department of Informatics, 2004. Dostupné také z: http://urn.nb.no/URN:NBN:no-10126

5. MAIER, G. A. S. F. A. F. A. NAT Usage in Residential Broadband Networks. In: *Passive and Active Measurement, Lecture Notes in Computer Science Volume 6579*. Berlin, Heidelberg: Springer-Verlag, 2011, s. 32-41. ISBN 978-3-642-19259-3. Dostupné také z: http://dl.acm.org/citation.cfm?id=1987510.1987514

6. METWALLY, A. a M. PADUANO. Estimating the number of users behind IP addresses for combating abusive traffic. In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining …*. New York (NY, USA): ACM, s. 249-57. Dostupné také z: http://doi.acm.org/10.1145/2020408.2020452

7. GOLLMANN, D. *Computer Security*. Third edition. JohnWiley and Sons Ltd, 2011. ISBN 978-0-470-74115-3.

8. DEPARTMENT OF DEFENSE. In: *DOD Dictionary of Military and Associated Terms* [online]. 15. November. 2015. Dostupné také z: http://www.dtic.mil/doctrine/new_pubs/jp1_02.pdf

9. *Request For Comments*. Request For Comments, 2010. Dostupné také z: http://www.ietf.org/rfc/rfc2828.txt

10. KISSEL, R. L. In: *Glossary of Key Information Security Terms* [online]. Dostupné také z: http://dx.doi.org/10.6028/NIST.IR.7298r2

11. WILLIAM R. CHESWICK, S. M. B. A. A. D. R. *Repelling the Wily Hacker , Second Edition*. Second edition. Addison-Wesley, 2003. ISBN:978-0201634662.

12. BISHOP, M. *Introduction to Computer Security*. Addison-Wesley, 2004. ISBN: 0-321-24744-2.

13. Information Security Handbook. *Confidentiality, Integrity & Availability* [online]. [cit. 2015-Apríl-20]. Dostupné z: http://ishandbook.bsewall.com/risk/Methodology/CIA.html

14. *Uncover Security Design Flaws Using The STRIDE Approach* [online]. 2006 [cit. 2015-Máj-01]. Dostupné z: https://msdn.microsoft.com/en-us/magazine/cc163519.aspx

15. KAHN, D. *The Codebreakers*. Scribner, 1996. ISBN 978-0684831305.

16. BISHOP, M. *Computer Security: Art and Science.* Addison-Wesley, 2002. ISBN 0-201-44099-7.

17. INTERNATIONAL TELECOMMUNICATION UNION (ITU). Security Architecture For Open Systems Interconnection. In: *X.800 Standard X.800, The International Telegraph and Telephone Consultative*. 1991. Dostupné také z: http://www.itu.int/rec/T-REC-X.800-199103-I/e

18. *nonrepudiation definition* [online]. 2008 [cit. 2015-Máj-01]. Dostupné z: http://searchsecurity.techtarget.com/definition/nonrepudiation

19. TANENBAUM, A. S. a M. VAN STEEN. *Distributed Systems: Principles and Paradigms (2nd Edition)*. Second edition. Pearson Education. Inc, 2006. ISBN:0-13-239227-5.

20. *The STRIDE Threat Model* [online]. 2002 [cit. 2015-Máj-01]. Dostupné z: https://msdn.microsoft.com/en-us/library/ee823878(v=cs.20).aspx

21. *Threat Risk Modeling* [online]. [cit. 2015-Máj-01]. Dostupné z: https://www.owasp.org/index.php/Threat_Risk_Modeling

22. KHALEEL AHMAD, S. V. N. K. J. S. Classification of Internet Security Attacks. In: *5th National Conference on Computing for Nation Development INDIACom-2011*. Delhi: 2011. Dostupné také z: http://www.bvicam.ac.in/news/INDIACom%202011/321.pdf

23. ROSSOW, C. In: *Amplification Hell: Revisiting Network Protocols for DDoS Abuse* [online]. 2014. Dostupné také z: http://dx.doi.org/10.14722/ndss.2014.23233

24. MORVAY, T. a R. SZABÓ. *Systém pre detekciu útokov v SQL dopytoch*. 2014. Bakalárska práca.

25. WILLIAM G.J. HALFOND, A. O. AMNESIA: Analysis and Monitoring for NEutralizing SQL-Injection Attacks. In: *Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering*. Long Beach, CA, USA: 2005, s. 174-183. ISBN:1-58113-993-4. Dostupné také z: http://doi.acm.org/10.1145/1101908.1101935

26. *Developing Network Security Strategies* [online]. 4. Október. 2010 [cit. 2015-Máj-01]. Dostupné z: http://www.ciscopress.com/articles/article.asp?p=1626588&seqNum=2

27. *Intel Technology Journal*. 15. November 2002, 37-48 s.. Dostupné také z: http://www.intel.com/content/dam/www/public/us/en/documents/research/2002-vol06-iss-4-intel-technology-journal.pdf

28. SCHOLTEN, H. A. D. H. V. Home Network Security. In: *Proceedings of the Seventh International Conference on Networking*. Washington, DC, USA: IEEE Computer Society,

2008, s. 249-55. ISBN 978-0-7695-3106-9. Dostupné také z: http://dx.doi.org/10.1109/ICN.2008.80

29. POMERLEAU, P. *Networking for English Majors*. 28: CreateSpace Independent Publishing Platform, 2008, Október s. [cit. 2014-November-20]. ISBN 978-1434891594. Dostupné z: http://books.google.sk/books?id=e47-BDa-pIkC&lpg=PP1&hl=sk&pg=PA147

30. VOICA, A. Imagination Blog. In: *Lantiq talks MIPS CPUs and the changing role of the broadband gateway* [online]. 24. jún. 2014. Dostupné také z: http://blog.imgtec.com/mips-processors/lantiq-talks-mips-cpus-and-the-changing-role-of-the-broadband-gateway

31. *OpenWrt* [online]. [cit. 2014-November]. Dostupné z: https://openwrt.org

32. *DD-WRT* [online]. [cit. 2014-November]. Dostupné z: http://www.dd-wrt.com

33. *The Open Source WRT54G Story* [online]. 8. November. 2005 [cit. 2014-November]. Dostupné z: http://www.wi-fiplanet.com/tutorials/article.php/3562391

34. *The Top 6 Alternative Firmwares For Your Router* [online]. 28. Február. 2011. Dostupné také z: http://www.makeuseof.com/tag/top-6-alternative-firmwares-router/

35. *Using the SDK* [online]. [cit. 2015-Máj-01]. Dostupné z: http://wiki.openwrt.org/doc/howto/obtain.firmware.sdk

36. *Creating packages* [online]. [cit. 2015-Máj-01]. Dostupné z: http://wiki.openwrt.org/doc/devel/packages

37. *Tomato Firmware* [online]. [cit. 2014-November]. Dostupné z: http://www.polarcloud.com/tomato

38. KARAMANOS, E. In: *Investigation of home router security* [online]. 2010. Diplomová práca. Dostupné také z: http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-91107

39. DENNING, D. E. In: *An intrusion-detection model* [online]. 1987. Dostupné také z: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.102.5127

40. SCARFONE, K. A. a P. M. MELL. *SP 800-94. Guide to Intrusion Detection and Prevention Systems (IDPS)*. Gaithersburg, MD, United States: National Institute of Standards & Technology, 2007. Dostupné také z: http://www.nist.gov/manuscript-publication-search.cfm?pub_id=50951

41. MARTIN ARVIDSON, M. C. In: *Intrusion Detection Systems – Technologies, Weaknesses and Trends* [online]. 2003. Dostupné také z: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.1.3354

42. HERVE DEBAR, M. D. A. W. *A Revised Taxonomy for Intrusion-Detection Systems*. Yorktown Heights: IBM. Zurich Research Laboratory, 1999. ISSN 0003-4347. Dostupné také z: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.124.3517

43. R. SEKAR, A. G.. J. F.. T. S.. A. T.. H. Y.. S. Z. Specification-based Anomaly Detection: A New Approach for Detecting Network Intrusions. In: *Proceedings of the 9th ACM Conference on Computer and Communications Security*. New York, NY, USA: 2002, s. 265-74. ISBN:1-58113-612-9. Dostupné také z: http://doi.acm.org/10.1145/586110.586146

44. Symantec. *An Introduction To Distributed Intrusion Detection Systems* [online]. [cit. 2015-03-03]. Dostupné z: http://www.symantec.com/connect/articles/introduction-distributed-intrusion-detection-systems

45. *Hardware* [online]. [cit. 2015-Máj-05]. Dostupné z: https://www.turris.cz/cs/hardware

46. *Software* [online]. [cit. 2015-Maj-05]. Dostupné z: https://www.turris.cz/cs/software

47. *Bezpečnosť* [online]. [cit. 2015-Máj-05]. Dostupné z: https://www.turris.cz/cs/security

48. *Co se sbírá* [online]. [cit. 2015-Máj-05]. Dostupné z: https://www.turris.cz/doc/collect/start

49. CHRISTOPHER KRUEGEL, G. V. *Anomaly Detection of Web-based Attacks*. 2003.

50. MARKO, P. *Heuristate - preprocesor pre nástroj Snort*. [Osobná komunikácia]. Október: 2015.

51. University of New Brunswick. *UNB ISCX Intrusion Detection Evaluation DataSet* [online]. 2012 [cit. 2015-November]. Dostupné z: http://www.unb.ca/research/iscx/dataset/iscx-dataset.html

52. *Snort* [online]. [cit. 2015-September]. Dostupné z: https://www.snort.org/

53. ROESCH, , et al. The Snort Project. *SNORT Users Manual* [online]. [cit. 2015-Október-20]. Dostupné z: http://manual.snort.org/

54. TEAM, T. S. *Snort++ User Manual* [online]. [cit. 2015-November-20]. Dostupné z: https://s3.amazonaws.com/snort-org-site/production/release_files/files/000/002/753/original/snort_manual.html?AWSAccessKeyId=AKIAIXACIED2SPMSC7GA&Expires=1449674337&Signature=oTaOBsaY5McD6P%2B0lbR6fCXFdV4%3D

55. INSTITUTE, I. S. In: *RFC 793: Transmission Control Protocol* [online]. 1981. Dostupné také z: https://tools.ietf.org/html/rfc793

56. SHIRAVI, A. A. S. H. A. T. M. A. G. A. A. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. In: *Computers & Security*. 2012, s. 357-74. ISSN 0167-4048, 10.1016/j.cose.2011.12.012. Dostupné také z: http://www.sciencedirect.com/science/article/pii/S0167404811001672

57. KARATZOGLOU, A. D. MEYER a K. HORNIK. Support Vector Machines in R. In: *Journal of Statistical Software, Volume 15, Issue 9*. 2006. Dostupné také z: http://www.jstatsoft.org/article/view/v015i09/v15i09.pdf

58. ucollect. *CZ.NIC Labs* [online]. [cit. 2016-Január]. Dostupné z: https://gitlab.labs.nic.cz/turris/ucollect

59. libatsha204. *CZ.NIC Labs* [online]. [cit. 2016-Január]. Dostupné z: https://gitlab.labs.nic.cz/turris/libatsha204

# Príloha A: Technická dokumentácia

## Prehľad zdrojových kódov riešenia

Vstupná funkcia implementovaného modulu vracajúca štruktúru s popisom zásuvného modulu.

```
#ifdef STATIC
struct plugin *plugin_info_tukabel(void) {
#else
struct plugin *plugin_info(void) {
#endif
    ulog(LLOG_WARN, "TUKABEL plugin_info\n");
    //static struct pluglib_import *imports[] = {
    //    &hello_world_import,
    //    NULL
    //};
    static struct plugin plugin = {
          .name = "Statetrans",
          .version = 1,
          //.imports = imports,
          .init_callback = init,
          .finish_callback = destroy,
          .packet_callback = packet_handle,
          .uplink_data_callback = communicate,
          .config_check_callback = config_check,
          .config_finish_callback = config_finish
    };
    return &plugin;
}
```

## Inštalačná príručka

Vytvorený rozširujúci modul *StateTrans* je potrebné skompilovať spolu s nástrojom *Ucollect*. Tento nástroj by mal byť kompatibilný s bežnými distribúciami operačného systému *Linux*. Vývojári nástroja *Ucollect* v dokumentácii uvádzajú ako použitý operačný systém *Debian 7*. Pri vývoji modulu bol použitý operačný systém *Ubuntu 14.04.4 LTS*.

### *1. Inštalácia klienta Ucollect*

Inštalácia vývojárskych nástrojov

```
apt-get install gcc g++ git qt4-qmake pkg-config cmake
```

Inštalácia závislostí nástroja Ucollect

```
apt-get install zlib1g-dev libpcap-dev python-dev socat
```

Ďalej je potrebné nainštalovať balíček `libuci`, ktorý je súčasťou operačného systému *OpenWrt* a slúži na manipuláciu s UCI konfiguračnými súbormi:

```
git clone git://nbd.name/uci.git
cd uci
git checkout v0.8.0
cmake -D BUILD_LUA:BOOL=OFF .
make
make install
```

`Libatsha204` je knižnica pre komunikáciu so šifrovacím čipom *Atmel ATSHA204*, ktorý obsahujú smerovače z projektu *Turris*. Túto knižnicu použijeme v režime softvérovej emulácie. Pre inštaláciu knižnice je potrebné najprv nainštalovať jej závislosti a následne samotnú knižnicu:

```
apt-get install libunbound-dev libssl-dev
git clone https://gitlab.labs.nic.cz/turris/libatsha204.git
cd libatsha204
git submodule init
git submodule update
make USE_LAYER=USE_LAYER_EMULATION NO_DOC=1
    CONFIG_PATH="/etc/atsha204.sw"
make install
```

Ďalej je potrebné skopírovať konfiguračný súbor pre túto knižnicu na očakávané umiestnenie

```
cp atsha204.sw /etc/.
```

Poznámka: Konfiguračný súbor opisuje obsah vnútornej pamäte čipu ATSHA204. Nie všetky pamäťové sloty sú potrebné alebo korektné.

Inštalácia nástroja Ucollect zo zdrojových súborov:

```
cd ucollect
git submodule init
git submodule update
make NO_DOC=1 NO_MASTER=1
```

Súbor `Makefile` pre zostavenie (kompiláciu) nástroja neobsahuje cieľ `install`. Nástroj ucollect je možné spustiť nasledujúcim príkazom:

```
./bin/ucollect
```

### *2. Inštalácia serverovej časti Ucollect*

Inštalácia vývojárskych nástrojov a závislostí:

```
apt-get install gcc g++ git qt4-qmake pkg-config
apt-get install zlib1g-dev libqt4-dev
apt-get install python-dev
apt-get install python-twisted python-dateutil python-psycopg2
```

Zostavenie zvyšných závislostí zo zdrojového kódu:

```
cd src/python
python setup.py build
python setup.py install
```

Zostavenie nástroja soxy:

```
cd src/master/soxy
qmake
make
```

### *3. Nastaveni komunikácia medzi klientom a serverom*

Generovanie SSL certifikátu pre šifrovanie spojenia medzi klientom a serverom:

```
openssl req -new -x509 -keyout /etc/ucollect-dev.pem
     -out /etc/ucollect-dev.pem -days 1825 -nodes
     -subj "/C=CZ/ST=Prague/L=Prague/O=Ucollect"
```

Parametre C, ST, L a O sú samozrejme konfigurovateľné.

Konfigurácia server – inštalácia databázového systému PostgreSQL:

```
apt-get install postgresql
```

Vytvorenie používateľov v databáze:

```
root@ucollect-server:~# su postgres
postgres@ucollect-server:/root$ cd
postgres@ucollect-server:~$ createdb ucollect
postgres@ucollect-server:~$ createuser ucollect
postgres@ucollect-server:~$ createuser authenticator
postgres@ucollect-server:~$ createuser updater
postgres@ucollect-server:~$ createuser jenkins
postgres@ucollect-server:~$ createuser cleaner
postgres@ucollect-server:~$ createuser archivist
```

Pre niektorých používateľov je potrebné nastaviť heslo:

- `updater` (súbor `src/master/collect-master.conf`)
- `authenticator` (súbor `src/master/authenticator/authenticator.conf`)

Inicializácia databázy:

```
./initdb debug
```

Zmena nastavení servera (certifikát a kľúč) – súbor src/master/collect-master.conf:

```
; The SSL certificate
cert = /etc/ucollect-dev.pem
key = /etc/ucollect-dev.pem
```

Spustenie autentifikátora:

```
cd src/master/authenticator
./authenticator.py authenticator.conf
```

Spustenie ucollect servera:

```
cd src/master
./collect-master.py collect-master.conf
```

Klienska strana - získanie koreňového bodu dôvery (***root trust anchor***) pre ubound/libubound:

```
apt-get install unbound-anchor
mkdir /etc/unbound
unbound-anchor -u http://data.iana.org
     -x /root-anchors/root-anchors.xml –v
```

Vytvorenie konfiguračného súboru pre klienta ucollect:

```
mkdir /etc/config
vim /etc/config/ucollect
```

Príklad konfigurácie klienta:

```
config interface
    option ifname 'eth0'

config uplink
    option name localhost
    option service 5679
    option cert "/etc/ucollect-dev.pem"

config plugin
    option libname 'libplugin_statetrans.so'
```

Spustenie klienta:

```
./bin/ucollect
```

Každý klient je identifikovaný indetifikátorom `ID`. `ID` je postupnosť čísel uložených v 2 OTP slotoch v čipe ATSHA204. Toto `ID` je možné získať príkazom

```
atsha204cmd serial-number
```

`ID` musí byť na strane servera v tabuľke `clients`. Pole `mechanism` je pri vývoji v tomto prípade dôležité. Hodnota `Y` znamená, že sa nebudú sa overovať práva tohto klienta a tento klient bude povolený.

## Príloha B: Príspevok na vedeckú konferenciu IIT.SRC 2016

# Home Network Security

Tomáš MORVAY*

*Slovak University of Technology in Bratislava*
*Faculty of Informatics and Information Technologies*
*Ilkovičova 2, 842 16 Bratislava, Slovakia*
*and*
*Eset Research Center*
tomasmorvay@gmail.com

**Abstract.** This paper deals with home network security and detection of intrusions in home network. The aim of this work is to design and implement a model for network intrusion detection system (NIDS), which can be easily deployed in home network. The model is based on stateful protocol analysis on network and transport layer. Profile is created for each device connected to network. Traffic is evaluated using these profiles in order to detect anomalies. Proposed NIDS is integrated into open source tool Ucollect. Furthermore, distributed approach to creation of firewall rules and employment of computational intelligence are discussed.

# 1. Introduction

Most of home end users connected to the Internet are part of their own network which typically consists of one combined Wi-Fi router with NAT functionality and multiple terminal devices. Recently, there has been a growing tendency for the typical number of terminal devices in household. Europe has 10 internet connected devices per household[9], while the national average for homes in United States is 5.7 devices per household[10]. Terminal devices are generally desktop computers, network drives, multimedia devices, smartphones, laptops and other devices/appliances capable of accessing the Internet.
Router is a centralized access point to the Internet in such a network. It typically provides NAT functionality, local DNS server, DHCP server and additionally it offers firewall functionality. Hardware resources and software functionalities are also improved with the growing number of terminal devices connected to the network.

These home routers represent an ideal place for monitoring the network traffic of end users, creating statistics and application of various rights and restrictions for terminal devices [1][2]. Network traffic monitoring from the perspective of the home router has several advantages compared to monitoring on internet service provider's (ISP) devices. The first advantage is that terminal devices are hidden

* Master degree study programme in field: Software Engineering
Supervisor: Assoc. Professor Ladislav Hudec, Institute of Applied Informatics, Faculty of Informatics and Information Technologies STU in Bratislava

[9]http://www.dazeinfo.com/2014/04/03/internet-around-world-25-people-access-internet-mobile-devices/

[10]http://homenetrouter.com/blog/homenetrouter/us-average-of-internet-connected-devises-on-the-rise-national-average-is-5-7-devices-per-household

from the perspective of the ISP due to employment of the NAT functionality [3]. However, information about the uniqueness of the source and type of communication (smartphone/PC) may be important in profiling and evaluation of characteristics of communication of individual devices [4]. The second advantage is that with the growing number of devices connected to home networks the monitoring of unique devices from the viewpoint of the ISP is relatively demanding on computer resources.

# 2. Detection model

Proposed detection model is based on stateful protocol analysis in combination with anomaly detection. State machines for stateful protocol analysis are constructed according to specifications of monitored protocols (e.g. RFC documents) while keeping in mind the reuse of functionalities of existing solutions and integration into them.

## 2.1 TCP protocol state machine

State machine for TCP protocol is based on state machine specification created by Sekar and his co-workers in [5]. In general this state machine follows the *RFC 793* [6], while there is an emphasis placed on abstraction. Abstraction simplifies the design and implementation of such a state machine. However, the main advantage of this approach is the prevention of problems which could arise when standards are strictly followed. Normal traffic could be marked at anomalous due to insignificant difference in interpretation of protocols. Moreover, the probability of this happening raises thanks to the fact that various implementation of TCP protocol are used. Figure 1 shows state machine created by Sekar and his co-workers. Each transition is numbered.

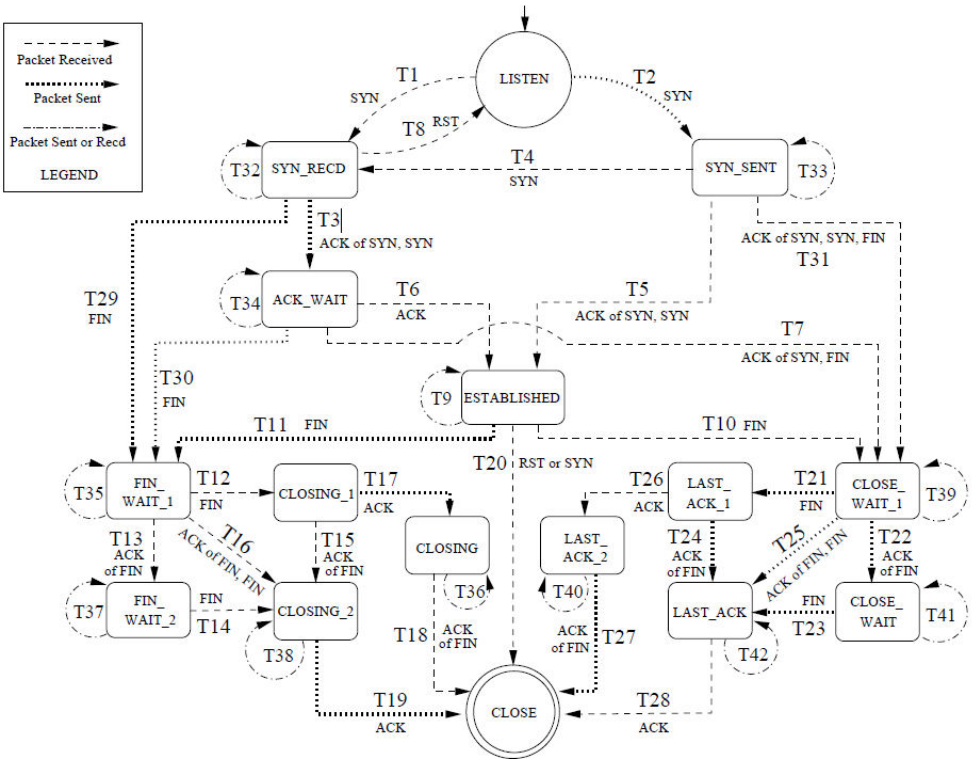



*Figure 1. TCP protocol state machine.*

To reduce resource requirements (and complexity) of the final solution transitions leading back to the same state (loops) are ignored (e.g. T32). Transitions which are not part of this state machine are automatically treated as violations of standard sequence of transitions. Such transitions are evaluated as anomalies.

## 2.2 State machine for stateless protocols

For stateless protocols which don't provide natural way of following the state of "connection" artificial states are created. For example for UDP protocol which is connectionless the following four states can be defined:

- `NONE` – no packet have seen for the communication
- `CLIENT_SEEN` – packets seen only in one direction (from client to server)
- `ESTABLISHED` – packets seen in both directions (from client to server and vice-versa)
- `TIMEDOUT` – no packet seen for predefined time (in any direction)

## 2.3 Anomaly detection

Basic method for evaluation of anomalies is creation of profile for each IP address. This profile describes distribution of transitions in different time intervals (10ms to 1 000 000ms). Statistical approaches are used for evaluation. First threshold is set. Each communication which varies from the profile more than the threshold allows is marked as anomalous.

Advanced method for evaluation of anomalies employs machine learning techniques. This approach can follow more complex patterns of end user behavior. However, it requires more tuning [8].

# 3. Integration into home network

The solution is based on tool Ucollect [9]. This tool is part of project Turris[, but can be run separately. .can be run separately. The main aim of this project is to create package that would run NIDS directly on the home router (see Figure 2). However this approach may require router with higher performance.

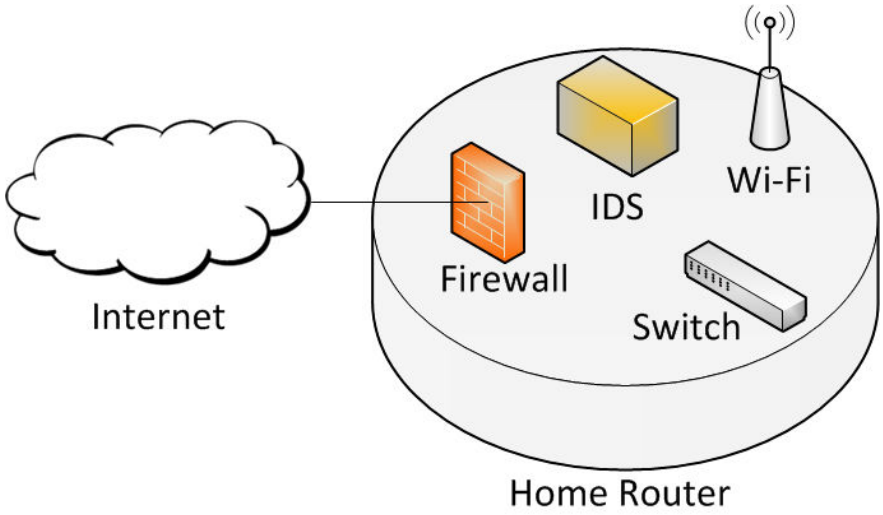



*Figure 2. Integration of NIDS directly into home router.*

## 3.1 Dedicated device for NIDS

Figure 3 shows deployment of the NIDS on separate dedicated device. This approach is an alternative for networks where the home router does not provide enough performance to feed the anomaly detection process in real time.

For correct functionality of anomaly detection, all the traffic passing through the router must be forwarded to machine dedicated for NIDS. This is achieved by setting one of the router ports to mode, in which all the traffic passing through the router is also forwarded to this port. This approach is called port mirroring and the port is called SPAN port.

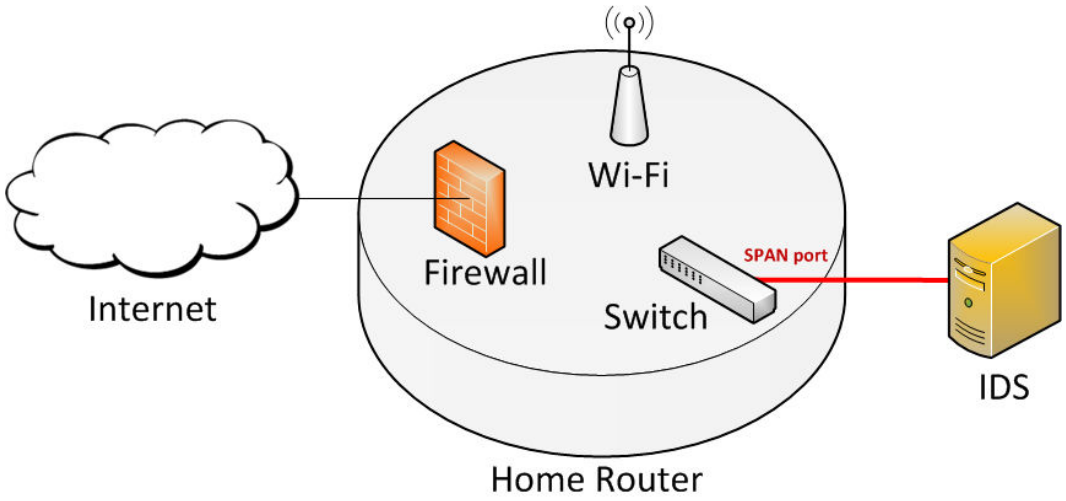



*Figure 3. Integration of NIDS into network using dedicated device.*

# 4. Experimental results

Dataset UNB ISCX IDS[11] (University of New Brunswick, Information Security Centre of Excellence, Intrusion Detection Evaluation Dataset) was used to experimentally verify the proposed approach. This dataset contains realistic traffic in .pcap format and XML files labelling each connection as normal or attack [7].
From this dataset one day only with normal traffic and one day with DDoS attack was selected. These days were analysed using proposed state machine model. Because the attack was aimed at the IP of web server, only this network device (identified by its IP address) is discussed in this section. Average state transition distributions for TCP protocol for different time intervals were calculated (from 1ms to 1 000 000ms).

Figure 4 shows the results in form of histogram for 1ms interval. Each bar represents the percentage of given transition from all monitored transitions. Grey bars show distribution of transitions in normal traffic and shaded red bar represents traffic containing attacks. It is obvious that this difference can be detected using simple statistics.

Further experiments were performed using machine learning. One-class SVM and k-means clustering were evaluated. This approach still need some tuning. However if properly selected machine learning technique would be fine-tuned the resulting detection should be able to detect more sophisticated attack patterns and also unseen/unknown attacks.

Centralized evaluation from one central point would also provide a global view to all attacks and this could be used to create a distributed firewall rules preventing global attacks.

[11] http://www.unb.ca/research/iscx/dataset/iscx-dataset.html

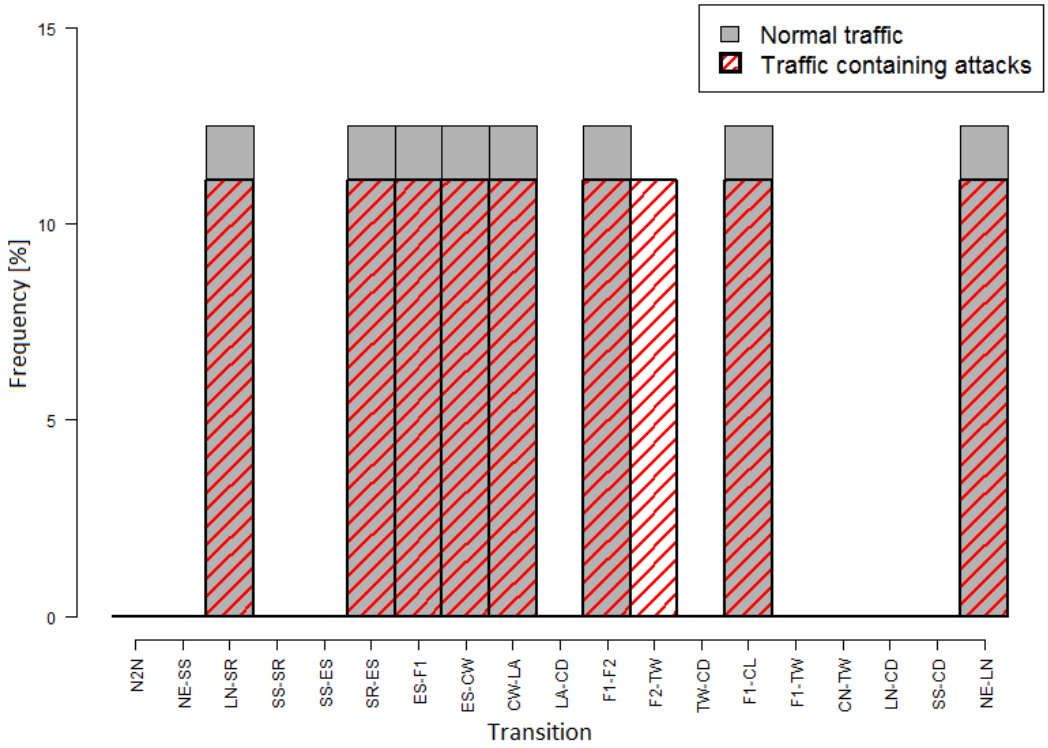



*Figure 4.Experimental results.*

# 5. Related work

Project Turris is the project with a very similar aim. It is non-for-profit research project of CZ.NIC, the registry of Czech national top level domain .CZ. This project is a service helping to protect its user's home network with the help of a special router. It has its own open source software based on OpenWrt and the hardware design is also open.

Besides ordinary functionalities the router analyses the traffic between internet and the home network, and identifies suspicious data flows. It alerts the Turris central of a possible attacks. At the system headquarters, data from many connected Turris routers can be compared and it is possible to assess the security status of detected traffic. In case an attack is detected, corresponding updates are prepared and distributed to the whole Turris network. This is called distributed adaptive firewall. Distributed adaptive firewall can be divided into four parts [2]:

1. *Collection* of input data (most importantly monitoring of end devices)
2. *Analysis* of obtained data on central server
3. *Preparation* of firewall rules based on data analysis
4. *Update* of end devices

Tool called Ucollect, which is part of this project, is used as base for our solution.

# 6. Conclusion

In this paper we presented a novel approach of intrusion detection in home. We have introduced detection model based on stateful protocol analysis in combination with anomaly detection. Implemented model can be deployed directly on home router supporting OpenWrt if it offers the required performance. For routers with less computational resources another approach was shown separating the NIDS onto dedicated device. Proposed approach was experimentally evaluated. Statistical evaluation shows that the solution is capable of detecting differences between normal and anomalous network traffic. Furthermore, distributed approach to creation of firewall rules and employment of computational intelligence were discussed.

*Acknowledgement:* This work was partly sponsored by the Eset Research Centre and Scientific Grant Agency of the Slovak Republic, grant No. VEGA 1/0836/16 "Methods and algorithms for improving efficiency and multimedia content delivery in IP networks".

## References

[1] Strand, L. K. *Adaptive distributed firewall using intrusion detection*. Master thesis, University of Oslo, (2004). Available at: http://urn.nb.no/URN:NBN:no-10126

[2] CZ.NIC: *Project Turris* [Online; accessed February 10, 2016]. Available at: https://www.turris.cz

[3] Maier, G., Schneider, F., Feldmann & A.. NAT Usage in Residential Broadband Networks. In: *Proceeding PAM'11 Proceedings of the 12th international conference on Passive and active measurement.* Springer-Verlag, Berlin Heidelberg, (2011), pp. 32-41. ISBN 978-3-642-19259-3. Available at: http://dl.acm.org/citation.cfm?id=1987510.1987514

[4] Metwally, A., Paduano, M. Estimating the number of users behind IP addresses for combating abusive traffic. In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, New York, (2011), pp. 249-57. Available at: http://doi.acm.org/10.1145/2020408.2020452

[5] Sekar, R., Gupta, A., Frullo, J., Shanbhag, T., Tiwari, A., Yang, H. & Zhou, S. Specification-based Anomaly Detection: A New Approach for Detecting Network Intrusions. In: *Proceedings of the 9th ACM Conference on Computer and Communications Security*. ACM, New York, (2002), pp. 265-74. ISBN:1-58113-612-9. Available at: http://doi.acm.org/10.1145/586110.586146

[6] J. Postel, *RFC 793: Transmission Control Protocol*, (1981). Available at: https://tools.ietf.org/html/rfc793

[7] Ali Shiravi, Hadi Shiravi, Mahbod Tavallaee, Ali A. Ghorbani, Toward developing a systematic approach to generate benchmark datasets for intrusion detection. In: *Computers & Security, Volume 31, Issue 3*. (2012), pp. 357-374, ISSN 0167-4048. Available at: http://www.sciencedirect.com/science/article/pii/S0167404811001672

[8] Kumar, G., Kumar, K., Sachdeva, M.: The use of artificial intelligence based techniques for intrusion detection: a review. In: *Artif Intell Rev, vol. 34, issue 4*. Kluwer Academic Publishers Norwell, (2010), pp. 369–387. Available at: http://dx.doi.org/10.1007/s10462-010-9179-5

[9] CZ.NIC: *Turris - uživatelská dokumentace - Co se sbírá* [Online; accessed April 10, 2016]. Available at: https://www.turris.cz/doc/collect/start

# Príloha C: Obsah elektronického média

- `/doc`
  Adresár obsahujúci elektronickú verzia tohto dokumentu
- `/references`
  Adresár obsahujúci použitú literatúru v elektronickej forme
- `/src/`
  Zdrojové kódy nástroja *ucollect*
- `/src/ucollect/src/plugins/statetrans/`
  Zdrojové kódy implementovaného rozširujúceho modulu *StateTrans*
- `/experimenta/src/`
  Zdrojové kódy použité pri experimentálnom overení
- `/experimental/graphs/`
  Grafy výsledkom