VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

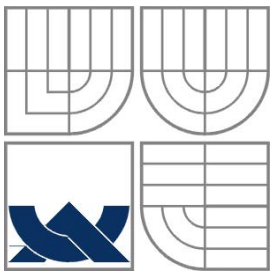
VALIDACE PARAMETRŮ SÍTĚ ZALOŽENÁ NA SLEDOVÁNÍ SÍŤOVÉHO PROVOZU

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

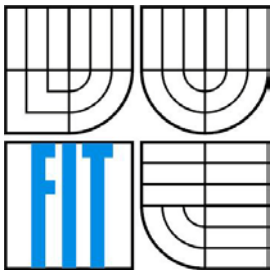
AUTOR PRÁCE
AUTHOR

BC. RADIM MARTÍNEK

BRNO 2011



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

VALIDACE PARAMETRŮ SÍTĚ ZALOŽENÁ NA SLEDOVÁNÍ SÍŤOVÉHO PROVOZU

VALIDATION OF NETWORK PARAMETERS BASED ON NETWORK MONITORING

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. RADIM MARTÍNEK

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. MARTIN ŽÁDNÍK

BRNO 2011

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačových systémů Akademický rok 2010/2011

Zadání diplomové práce

Řešitel: **Martínek Radim, Bc.**
Obor: Počítačové sítě a komunikace
Téma: **Validace parametrů sítě založená na sledování síťového provozu**
Validation of Network Parameters Based on Network Monitoring
Kategorie: Počítačové sítě

Pokyny:

1. Vyberte parametry sítě, které lze validovat sledováním provozu a nastudujte způsob definování vybraných parametrů.
2. Seznamte se s nástroji simulujícími chování sítě.
3. Zhodnoďte, které parametry sítě je vhodné validovat simulací a které sledováním skutečného provozu.
4. Navrhněte systém umožňující validovat parametry sítě pomocí sledování skutečného provozu.
5. Navržený systém implementujte.
6. Implementaci ověřte nasazením na experimentální síti, kterou výtvoříte v laboratoři.
7. Zhodnoťte dosažené výsledky.

Literatura:

- Dle pokynů vedoucího.

Při obhajobě semestrální části diplomového projektu je požadováno:

- Splnění bodů 1-4 zadání.

Podrobné závazné pokyny pro vypracování diplomové práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

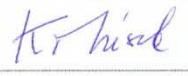
Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci ročníkového a semestrálního projektu (30 až 40% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Žádník Martin, Ing.**, UPSY FIT VUT
Datum zadání: 20. září 2010
Datum odevzdání: 25. května 2011

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačových systémů a sítí
612 66 Brno, Božetěchova 2

L.S.



doc. Ing. Zdeněk Kotásek, CSc.
vedoucí ústavu

Abstrakt

Diplomová práce představuje teoretický úvod, seznámení se s problematikou, návrh řešení a implementaci nástroje pro validaci parametrů sítě založeném na sledování síťového provozu. Nejprve je rozebrán současný stav návrhu počítačových sítí a jeho omezení. V důsledku toho je představen nový přístup pro realizaci a ověření požadovaného nastavení sítě využívající technik verifikace, simulace a validace. Po tomto úvodním zasazení práce do kontextu jsou dále zkoumány konkrétně validační techniky. Hlavním přínosem této práce je určení vhodných parametrů, které lze k validování použít a implementace nástroje zajišťující proces validace. Síťový provoz charakterizující chování sítě je sbírán pomocí NetFlow technologie, která vytváří síťové toky a ty jsou následně vytvořeným nástrojem využity k validování požadovaných parametrů sítě, čímž se celkově ověří, zda byly hlavní požadavky kladené na danou počítačovou síť splněny nebo ne.

Abstract

The Master's Thesis presents a theoretical introduction, familiarization with the issue and a implementation for a solution of a "network parameter validation" tool, which is founded on principle of network traffic monitoring. Firstly, the current development of computer network setup is analyzed with its limitations. This is an initial point for an introduction of a new approach for implementation and verification of required network setting, which uses techniques of verification, simulation and validation. After the introduction into the context, validation techniques are specifically examined. The Thesis main contribution lies in the capacity to determine appropriate parameters, which can be used for validation and also for implementation of the tool, which ensures validation process. The network traffic, which characterizes the behavior of the network, is collected by NetFlow technology, which generates network flows. These flows are consequently used by the designed tool used for validation of required network parameters. This process overall verifies whether the main computer network requirements have been met or not.

Klíčová slova

Validace, verifikace, NetFlow, Cisco Packet Tracer, OMNeT++, NFDUMP

Keywords

Validation, verification, NetFlow, Cisco Packet Tracer, OMNeT++, NFDUMP

Citace

Martínek Radim: Validace parametrů sítě založená na sledování síťového provozu, diplomová práce, Brno, FIT VUT v Brně, 2011

Validace parametrů sítě založená na sledování síťového provozu

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Martina Žádníka. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Radim Martínek
20. května 2011

Poděkování

Děkuji zejména mému vedoucímu práce Ing. Martinovi Žádníkovi, za poskytnutí četných konzultací, rad a připomínek a dále bych rád poděkoval všem, kteří mě byli při tvorbě této práce nápomocni.

Obsah

Úvod

Současné moderní počítačové sítě jsou výrazně komplexnější než jejich předchůdci. Již nestačí poskytovat pouze propojení mezi uzly sítě. Dnes se požaduje celá řada různých služeb, včetně záruk, zabezpečujících splnění předem určené kvality daných služeb. Jedná se o služby jako internet telefonie (VoIP), internet televize (IPTV), internet banking, virtuální privátní sítě (VPN), sdílení souborů a tiskáren, připojení vzdáleného diskového prostoru (iSCSI), systém centrálního zálohování. Většina zmíněných služeb má vysoké nároky na zabezpečení zasílaných dat (internet banking, iSCSI, sdílení, zálohování, VoIP, VPN). Některé služby potřebují pro svůj chod garanci dostatečně rychlé odezvy (VoIP, IPTV).

Návrh takto komplexních sítí se dost dobře nedá udělat pouhým odhadem a fyzickou realizací s následným ověřením dostupnosti uzlů pomocí nástrojů typu traceroute. Tímto přístupem nelze ověřit a zaručit požadované aspekty chování sítě, jako maximální zpoždění paketů, ztrátovost paketů, nebo dosažitelnost všech uzlů sítě za určitých krizových předpokladů. Vzniká potřeba nalézt inovativní prostředky a přístupy k vývoji nových, případně k ověření již existujících počítačových sítí a jejich služeb.

Diplomová práce je motivována návrhem pro implementaci technik verifikace, validace, a monitorování sítě. Návrh byl zpracován akademickými pracovníky UITS a UPSY (autoři: Jan Kořenek, Petr Matoušek, Ondřej Ryšavý, Martin Žádník). Jejich cílem byla snaha o verifikaci zrealizovaných počítačových sítí v oblasti bezpečnosti a spolehlivosti (pomocí formální analýzy a simulačních technik) a následně validaci zadaných požadavků pomocí monitorování reálného provozu v síti. Diplomová práce vychází ze zmíněného konceptu a pokusí se zrealizovat druhou zmíněnou oblast, tj. validaci parametrů sítě na základě sledování reálného provozu v síti. Hlavním cílem této práce je výběr vhodných parametrů sítě určených k validaci, navrhnout nástroj, který bude pomocí monitorování reálného provozu zvolené parametry validovat a následně tento nástroj implementovat. Pro vývoj a ověření nástroje bude využito zázemí sítové laboratoře školy, kde se pro testovací účely vytvoří experimentální síť.

První kapitola uvádí v širším kontextu základní informace, po jejichž přečtení lze získat představu o zasazení práce do kontextu problematiky návrhu počítačových sítí. Konkrétně bude představen celkový koncept struktury a jednotlivých částí technik verifikace, validace a monitorování sítě (vytvořené akademickými pracovníky UITS a UPSY) tak, aby bylo patrné, kterou část z popsaného konceptu si klade za cíl tato diplomová práce implementovat. Následně budou vysvětleny základní pojmy tvořící nutný základ pro dobré pochopení navazujících kapitol.

Kapitola druhá se věnuje již konkrétním cílům této práce. Jsou zde specifikovány požadavky na výstupní aplikaci, následující analýzou problematiky týkající se možných validovatelných

parametrů, možnostmi uložení a zpracování síťových toků a výběrem programovacích jazyků a technologií. Poté je v rámci této kapitoly předložen návrh řešení projektové části.

Kapitola třetí podrobně rozebírá zvolené implementované parametry k validaci. Ke každému parametru je popsán jeho význam, příklad použití a konkrétní postup jakým validace probíhá. Ke konci kapitoly je nastíněn doporučený způsob definování požadavků k popisu celkového požadovaného chování počítačové sítě.

Čtvrtá kapitola se věnuje implementaci nástroje. Nejprve je představen základní pohled na aplikaci včetně zobrazení objektové struktury ve formě diagramu tříd. Poté jsou v samostatných podkapitolách prezentovány jednotlivé části vytvořeného nástroje.

Kapitola pátá se zabývá testováním aplikace, které proběhlo ve školní laboratoři. Je zde vyobrazena topologie experimentální sítě vytvořená pro tyto testovací účely a její základní specifikace. Z této specifikace jsou poté definovány konkrétní požadavky k validaci, které ověřují, zda je daná specifikace dodržena. Pro každý požadavek je rozebrán výsledek jeho validace s ohledem na reálně uskutečněnou komunikaci v síti. Dále je v této kapitole popsán zátěžový test pro vymezení časové náročnosti aplikace a shrnutí výsledků validace.

V závěru diplomové práce jsou shrnuty nejpodstatnější informace sepsaných kapitol. Je zmíněno hodnocení, vlastní přínos a možné směry rozšíření práce do budoucna.

1 Úvod do problematiky

Cílem této úvodní kapitoly je prezentace možných technik návrhu počítačových sítí a zasazení práce do kontextu těchto technik. Je zde nejprve popsán celkový koncept struktury a jednotlivých částí technik verifikace, validace a monitorování sítě (vytvořené akademickými pracovníky UITS a UPSY), na kterém diplomová práce staví. V jednotlivých podkapitolách jsou popsány zásadní pojmy verifikace, simulace a validace v kontextu počítačových sítí. Nakonec jsou vysvětleny základní pojmy úzce se týkající realizace validačních technik navržených a implementovaných v navazujících kapitolách.

1.1 Koncepce technik verifikace a validace

Verifikace si klade za cíl ověřit vybrané vlastnosti sítě ještě před samotnou realizací, následně validace průběžně kontroluje shodu zverifikovaných výsledků se získanými daty ze sítě. Tato podkapitola popisuje současný stav návrhů sítí s jeho nedostatky a omezeními. Následuje motivace k nalezení jiných přístupů a nakonec je popsána základní struktura, význam a přínos jednotlivých fází a částí technik pro verifikaci, validaci a monitorování počítačových sítí.

1.1.1 Současný stav návrhu počítačových sítí a jeho limity

V případě návrhů počítačových sítí poskytujících komplexní a moderní služby charakteristické pro dnešní dobu (VoIP, IPTV, VPN, iSCSI), se lze stále nejčastěji setkat s již nevyhovujícím postupem. Typicky se pomocí nějakého nástroje (mnohdy tužka a papír) nakreslí předpokládaná topologie sítě obsahující rozvržení kabeláže, umístění zásuvek, rozvodny a aktivních prvků. Zvolí se požadované rychlosti linek, demilitarizované zóny, filtrovací pravidla, případně QoS. Následně se předpokládaná topologie fyzicky zrealizuje a síť se dle požadavků (filtrovací pravidla, QoS) vhodně nakonfiguruje. Poté následuje otestování funkčnosti sítě, což se provede otestováním dostupnosti všech spuštěných služeb a uzlů v nově zrealizované síti. Současné obecně známé a rozšířené metody testování dostupnosti míst v síti jsou skrze použití nástrojů typu „ping“ a „traceroute“. Článek [1] se zabývá zajištěním dostupnosti sítě skrze statickou analýzu a zároveň poukazuje na jisté omezení hojně používaných nástrojů „ping“ a „traceroute“. Některé z nich jsou zde uvedeny.

Existuje velké množství překážek bránící paketu v dosažení svého cíle (paketové filtry, přetížení sítě, chyby v konfiguraci sítě, chyby v návrhu sítě, selhání uzlu sítě a mnoho dalších). Běžnou praxí pro určení zda paket dosáhne svého cíle, jsou nástroje ping a traceroute, které pomocí zpráv protokolu ICMP vyšlou na cílovou IP adresu zprávu „Echo Request“ a očekávají jako odpověď zprávu „Echo Reply“. Pokud odpověď nepřijde, je uzel považován za nedostupný (program ping).

Program traceroute pomocí zmíněného principu, postupně testuje a vypisuje všechny uzly na cestě k cíli, čímž lze určit přibližné místo, kde se vyskytuje nějaký problém.

Bohužel aplikace tohoto typu ve skutečnosti určují dostupnost uzlů pouze pro protokol ICMP, ostatní protokoly mohou být filtrovány a zahozeny. V tom případě se bude uzel sítě jevit jako nedostupný. Dalším problémem je zvolená cesta ICMP paketu, která je závislá na aktuálních směrovacích tabulkách jednotlivých směrovačů. V dynamicky směrovaných sítích může být cesta pokaždé jiná. Navíc nelze ověřit komplexně všechny možné dostupné cesty k cíli (včetně záložních linek). Zvolené přístupy jsou použitelné pouze v provozuschopných sítích. Nelze je použit při návrhu sítové topologie a konfigurace. Už vůbec nám tyto aplikace neumožní rozumným způsobem ověřit, zda všechny uzly plánované jako nedostupné, jsou opravdu nedostupné a naopak. Nejenom tyto důvody podněcují snahu o vytvoření lepších přístupů k ověření různých vlastností a parametrů sítě.

1.1.2 Motivace vedoucí k hledání nových přístupů

Oblast návrhu počítačových sítí je značně komplikovaná a přináší sebou řadu výzev. Současné moderní sítě poskytují velké množství různých služeb, kde hrozí jejich vzájemné ovlivnění, což může vést k výskytu neočekávaných chyb a posléze k selhání dané služby. Je obtížné docílit kvalitně navržené sítě splňující požadavky spolehlivosti a bezpečnosti s ohledem na její komplikovanost a variabilitu jejich služeb. Zároveň je nutné brát v úvahu dynamicky se měnící charakter sítě zapříčiněný například poruchou aktivních prvků, výpadky linek, změnou směrovacích tabulek, použitím záložních linek a dalšími nestandardními událostmi. V takovém případě i návrh relativně malé, na první pohled jednoduché sítě může být značně komplikovaný a vybízí k použití automatizovaných nástrojů, kterých je ovšem v současné době nedostatek.

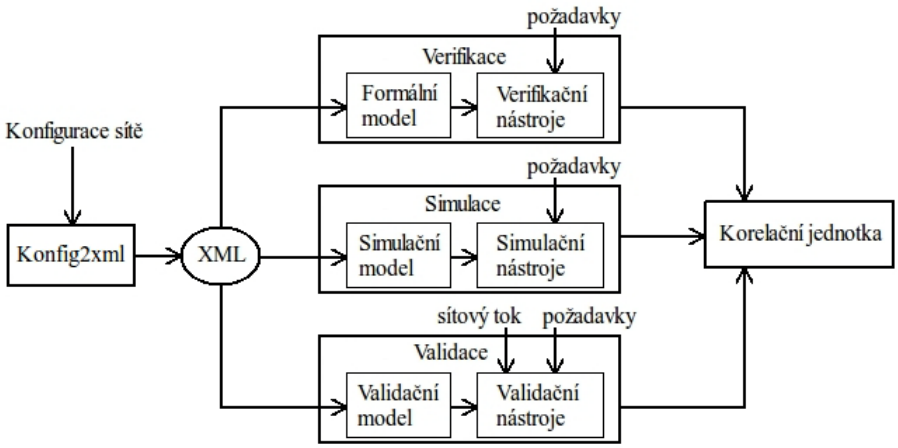
Z těchto důvodů je snaha vytvořit sadu nástrojů specializujících se na verifikaci, simulaci a validaci návrhu počítačové sítě. Zásadní motivací je identifikovat potencionální problémová místa a chyby v návrhu co nejdříve, jak je to jen možné. Taková aplikace by automaticky načetla konfiguraci všech aktivních prvků sítě, formálně verifikovala spolehlivost a bezpečnost dané konfigurace. Následně by nad touto konfigurací spustila několik různých typů simulací, které by otestovaly chování navržené sítě v různých nestandardních situacích. Pokud by dosavadní testování neobsahovalo žádné nedostatky, síť by byla fyzicky sestavena a pomocí monitorování reálného provozu by validovala požadované síťové parametry.

Významným inovativním aspektem zmíněného přístupu je uplatnění kombinace formálních metod a sledování skutečného provozu. Více o struktuře a použití technik verifikace, simulace a validace bude popsáno v následujících podkapitolách.

1.1.3 Struktura technik verifikace a validace

Z pohledu postupného provádění lze techniky verifikace a validace rozdělit do analytické, produkční a validační fáze. **Analytická fáze** zahrnuje vytvoření formálního modelu na základě konfigurace aktivních prvků (router, switch). Verifikaci a simulaci požadavků, které má síť ze své specifikace splnit. Případnou úpravu konfiguračních dat a specifikace (pokud verifikační a simulační nástroje prokázaly nesplnění zadaných požadavků) a opětovné spuštění analytické fáze. V případě splnění všech požadavků sítě dojde k **produkční fázi**, která obsahuje fyzické vytvoření sítě a rozmístění monitorovacích zařízení sloužících k zachycení reálného síťového toku dat. V poslední **validační fázi** monitorovací sondy posílají centrální jednotce datové toky, která tyto informace analyzuje a porovnává s požadavky, které má síť splňovat. Pokud je sítový provoz (chování sítě) ve shodě s definovanými požadavky, lze považovat síť za validní. V případně výskytu rozdílného chování sítě od požadovaného je třeba upravit model sítě a všechny tři fáze znovu projít.

Všechny části technik pro verifikaci, simulaci a validaci lze plánovat jako modulární s tím, že přidáním dalších částí, se jednoduše rozšíří funkcionalita. Celkem se popisovaný koncept skládá z těchto hlavních částí: Konf2xml, verifikační nástroje, simulační nástroje, monitorující zařízení a korelační nástroje. Struktura je zobrazena na Obr. 1.1. **Konf2xml** provádí překlad jednotlivých konfigurací všech aktivních prvků sítě do jednoho konfiguračního xml souboru, který slouží jako základní zdroj vstupních hodnot pro ostatní části aplikace. Tím je docíleno, že veškeré informace o uzlech, linkách, směrovacích pravidlech a paketových filtrech celé sítě budou obsaženy v jednom souboru. V bakalářské práci [2] lze nalézt zmíněný popis překladu konfiguračních údajů týkajících se zařízení firmy Cisco. **Verifikační nástroje** přečtou z xml souboru konfigurační údaje a převedou je na formální model, v rámci kterého se testují požadavky sítě. Podrobnější popis možností verifikace je rozepsán v kapitole 1.2. **Simulační část** bude využívat simulátor OMNet++ [3], který bude rozšířen o implementaci filtrovacích pravidel (ACL) a protokolů RIP a OSPF. Detailněji je simulační část rozebrána v kapitole 1.3. **Monitorující zařízení** budou představovat hardwarově akcelerované jednotky [4] sbírající charakteristiky posílaných paketů v síti a statistická data. Sesbírané data poslouží zejména pro validaci síťových parametrů, která je hlavní náplní této práce a bude postupně rozebrána. Poslední část **korelační jednotka** bude získávat výsledky verifikace, simulace, a validace. Na základě výsledků, konfigurace a požadavků sítě bude zhotovena zpráva pojednávající o kvalitách návrhu a chování sítě. V případě problémů v návrhu bude předložena opravující konfigurace a vysvětleno, proč a kde se vyskytl daný problém.



Obr. 1.1 Struktura technik pro verifikaci, validaci a monitorování počítačové sítě

1.2 Verifikační část

Podkapitola nejprve vysvětlí pojem verifikace v obecné rovině a následně jeho použití v kontextu počítačových sítí. Dále je popsáno, jaké přístupy verifikace využívá, co k tomu potřebuje a jaké vlastnosti návrhu sítě ověřuje.

Verifikace je proces, rozhodující zda produkt v dané fázi vývojového cyklu naplňuje požadavky zavedené během předchozí fáze vývojového cyklu [5]. Verifikace potvrzuje pravdivost shody mezi produktem a jeho specifikací [6]. Případně lze verifikaci chápat jako proces, kontrolující zda systém splňuje dané vlastnosti označující jako verifikační podmínky. Pokud podmínky splněny nejsou, provede se diagnóza problémů branících splnění verifikačních podmínek, vykonání z toho plynoucích korekcí a následně je verifikační proces zopakován za účelem ověření, že vykonané korekce nemají žádné nežádoucí následky. Verifikaci lze klasifikovat podle toho, zda dochází k testování systému na statickou (statická analýza, thorem proving, model checking) a dynamickou (deterministické, náhodné, statistické testování) [7].

Články [8], [9] popisují, jakým způsobem lze v kontextu počítačových sítí použitím metody „model checking“ verifikovat určité vlastnosti sítě (spolehlivost, dosažitelnost, robustnost, bezpečnost) v každém stavu, ve kterém se síť může nacházet. Je vytvořen formální model skládající se z uzlů, linek, směrovacích pravidel a paketových filtrů. Pokud vezmeme v úvahu výše zmíněnou definici verifikace (tj. verifikace potvrzuje pravdivost shody mezi produktem a jeho specifikací), tak produkt je v tomto případě počítačová sít reprezentována danými vlastnostmi a specifikace je vytvořený formální model. Následně jsou zkoumané vlastností sítě vyjádřeny ve formě logických formulí. A nakonec formální analýza ověří, zda určené vlastnosti daný model splňuje, či nikoliv (pokud je daná formule dokázána, vlastnost je splněna). Příklady vlastností, které lze v rámci verifikační části pomocí formální analýzy zkoumat, jsou v tabulce Tab. 1.1. Nevýhodou tohoto

postupu je nutnost předpočítat všechny možné směrovací tabulky pro všechny stavy topologie, což muže být ve velkých sítích neefektivní a tedy nepoužitelné.

Podobný přístup vytvářející formální model sítě popsaný v [10], [11] je efektivnější a nevyžaduje předpočítávání směrovacích tabulek. Hlavní myšlenkou verifikace vlastností v modelu sítě je sestavení a analýza univerzálních a kritických bodů. Kritické body představují uzly a linky sítě, které jsou obsaženy v každé teoreticky možné cestě splňující zkoumanou vlastnost. Při výpadku kritického bodu dojde vždy k selhání sledované služby. Oproti tomu univerzální body jsou uzly a linky sítě nevyskytující se v žádné z takových cest, jejich selhání nijak neovlivní dostupnost dané služby. Obě zmíněné množiny bodů jsou nezávislé na použitých směrovacích protokolech. Pomocí formálního modelu a analýzy těchto množin bodů lze detekovat potencionální kritičnost míst v síti, které jsou využity v rámci simulační části (o té je pojednáno v další kapitole). Kritičnost označuje míru použití daného místa v rámci prioritních cest, případně procentuální vyjádření cest, které využívají dané místo ze všech možných cest. V případě výskytu velké kritičnosti se jako vhodné opatření může jevit zavedení záložních linek.

Verifikace formální analýzou zkoumá kvalitativní parametry sítě, tzn. stav uzlů a linek sítě, naproti tomu simulací lze testovat kvantitativní parametry jako zpoždění a ztrátovost paketů, případně konvergenční čas použitých protokolu. Vice informací o formální analýze lze získat v [8], [9], [10], [11], [12].

Tab. 1.1 Příklady vlastností, které lze verifikovat a jejich logické formule **[8]**

dokazovaná vlastnost	**logická formule dané vlastnosti**
Jakékoliv externí uzly sítě „A“ mají přístup na emailový server.	SMTP ∈ NetReach φ (A), φ = p.proto = Tcp ∧ p.dstPort = 25
Webový server bude dostupný z uzlu „A“, i když dojde k výpadku linky „L“.	WWW ∈ NetReach ψ (A) =⇒ [↓ L]WWW ∈ NetReach ψ (A), ψ = p.proto = Tcp ∧ p.dstPort = 80
Payroll server není přístupný z žádného z externích uzlů sítě „A“.	Payroll ∈ NetReach φ (A), φ = ¬(p.srcIp = local ∧ p.proto = IP)

1.3 Simulační část

Simulace úzce navazuje na předchozí verifikační část. Kapitola zmiňuje, v čem je hlavní přínos simulace, jaké parametry sítě lze simulovat a které nástroje lze pro tuto činnost použít.

Simulací lze efektivně zkoumat chování sítě za určitých nastavených podmínek, kde chováním sítě se zejména myslí časové charakteristiky a statistické údaje. Například je snaha vytvořit co nejefektivnější topologii s ohledem na požadovanou dostupnost a stabilitu. V reálné síti by bylo experimentování s různými topologiemi, protokoly a konfiguracemi neúnosné. Pro představu testování náhodných výpadků uzlů a měření charakteristik, jak se s tím sít vyrovná, je v reálné

fyzické síti s dobrou vypovídající hodnotou nemožné. Naproti tomu v simulačních nástrojích lze automaticky testovat různé topologie s různými nastaveními a zaměřit se na konkrétní parametry vypovídající o kvalitách návrhu aktuální topologie. Zde je dobré si uvědomit vazbu na verifikační část. Ta při analýze odhalí kritická místa, která jsou dále důkladně zkoumána pomocí simulačních technik. Zároveň tím formální analýza přispěje k snížení počtu nutných simulačních scénářů (simulují se jen ty zajímavé s ohledem na nalezená kritická místa v síti).

Každá simulace má scénář skládající se ze sekvence událostí (např. výpadek linky, výpadek uzlu, ztráta paketu) měnící stav sítě. Události mohou mít pevně daný i náhodný čas výskytu. V probíhající simulaci se zaznamenávají časové charakteristiky v závislosti na počtu a typu vzniklých událostí. Lze tedy například ověřit, zda při selhání konkrétní linky sítě, bude dodrženo maximální povolené zpoždění paketu. Mimo zmíněné zpoždění lze sledovat také ztrátu paketu, čas potřebný k dosažení stabilizace dynamických protokolů (konvergenční čas protokolu), různé časovače a další kvantitativní parametry [10].

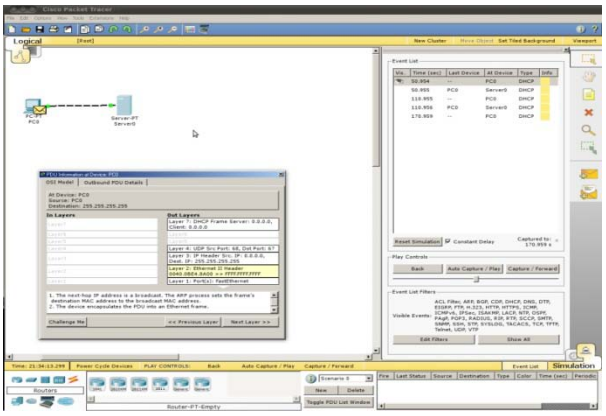
1.3.1 Simulační nástroje

Užitečnost simulace ovlivňuje zejména použitý simulační nástroj. Na tomto místě budou stručně popsány nástroje OMNeT++[3] a Cisco Packet Tracer[13]. Podrobnější porovnání i s konkrétním příkladem použití lze nalézt v práci [14], ze které bylo také částečně čerpáno.

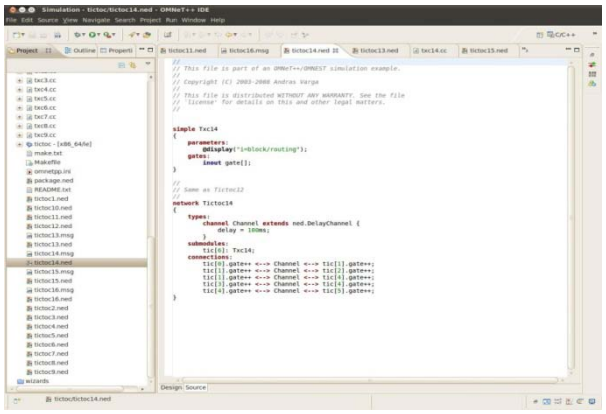
Packet Tracer je uživatelsky přívětivý simulační nástroj s grafickým rozhraním. Je vytvořen firmou Cisco a umožňuje modelování síťových návrhů skládajících se pouze z aktivních prvků této firmy. Často se využívá pří podpoře výuky CCNA kurzů. Studenti tak mohou i bez přístupu k fyzickým zařízením vyzkoušet jejich konfiguraci a odsimulovat podporované typy protokolů, kterých je v současné době několik desítek. Packet Tracer umožňuje modelování v rámci logického a fyzického pracovního prostředí. V logickém prostředí zobrazeném na Obr. 1.2 se vytváří síťová topologie (nastavení aktivních prvků a jejich propojení). Fyzické pracovní prostředí slouží k zohlednění konkrétního místa, ve kterém se topologie nachází (délka kabeláže, umístění v rámci budov při použití bezdrátové technologie). Simulovat lze v simulačním a reálném režimu. Simulační režim nabízí možnost zpomalit, zastavit, případně vrátit čas, čímž lze detailně sledovat veškerý provoz včetně obsahu posílaných PDU. Reálný režim znamená okamžitou odezvu na vykonané události tak jako v opravdových sítích.

Výhodou Packet Traceru je jeho jednoduchost, přímočarost a zároveň schopnost zprostředkovat reálnou práci se síťovými technologiemi (například konfiguraci jednotlivých zařízení lze provést stejně jako v praxi přes terminál pomocí příkazů operačního systému IOS). Nevýhodou je uzavřenost nástroje, nelze dodělat vlastní rozšíření, a tedy není vhodný pro specifickou analýzu síťového návrhu. Užití je spíše naučné, pro pochopení jak použitý protokol v praxi funguje a co

všechno lze nastavit. Pro požadavky navrhovaných technik verifikace, simulace a validace je Packet Tracer nedostačující.



Obr. 1.2 Cisco Packet Tracer – logic. prostředí, sim. Režim



Obr. 1.3 Vývojové prostředí OMNeT++

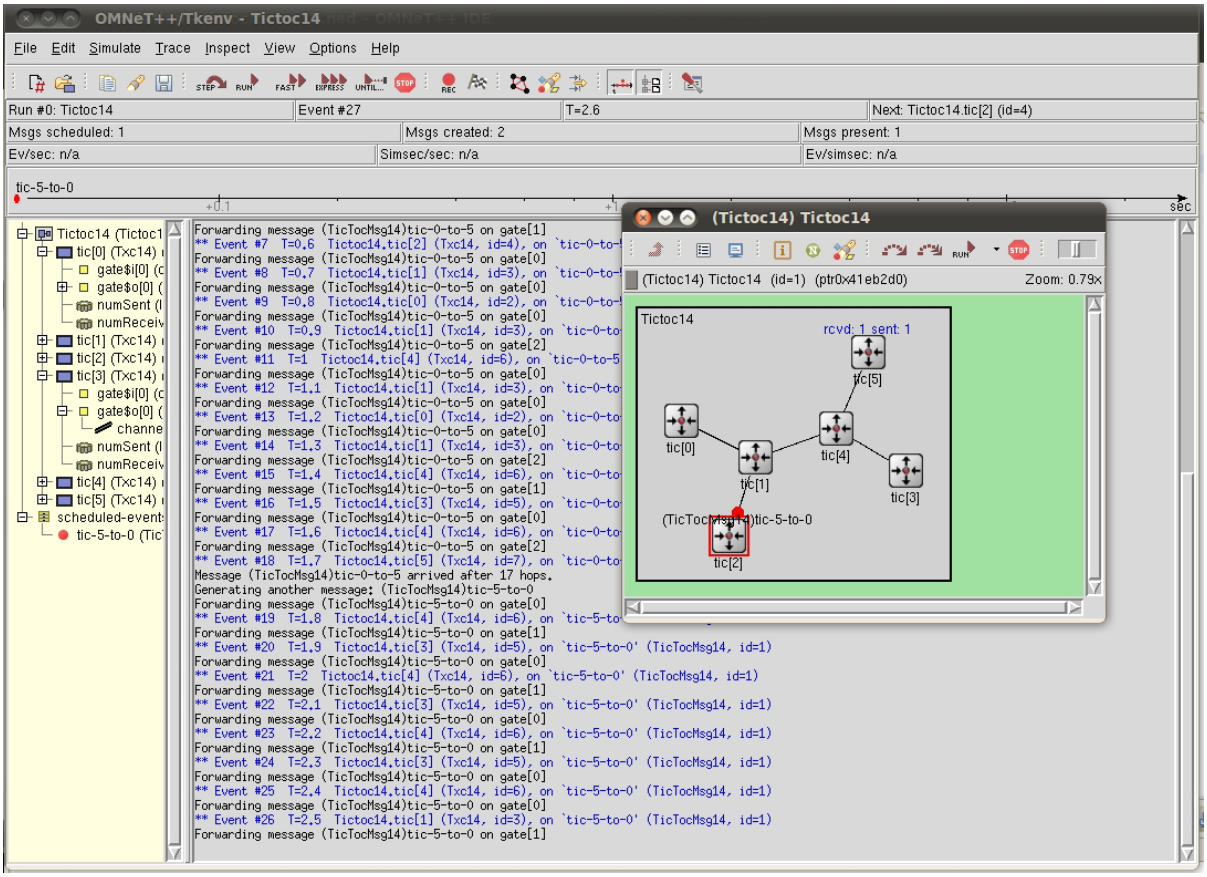
OMNeT++ je pro akademické účely volně dostupný open-source modulární objektově orientovaný simulační nástroj. Pro vytváření a konfiguraci simulačních modelů nabízí vývojové prostředí (IDE založené na Eclipse - Obr. 1.3). Samostatnou simulaci lze spustit v grafickém prostředí „Tkenv“ (Obr. 1.4) nebo v terminálu. Navíc kromě simulačního jádra OMNeT++ obsahuje překladač pro NED jazyk, makefile, dokumentaci, tutoriál včetně praktických příkladů a různé další utility (generátor náhodných čísel, sběr statistických dat).

Vytvořený simulační model se skládá z komponent, které mohou být hierarchicky zanořovány. Komponenty na nejnižší úrovni se programují v C++. Jejich sloučení se provede pomocí vlastního vysokoúrovňového jazyka NED. OMNeT++ přináší základní mechanismy a nástroje pro vytvoření simulace, nicméně neposkytuje žádné komponenty určené pro simulaci počítačových sítí. K tomuto účelu slouží různě simulační frameworky jako například INET [15].

Pro vytvoření simulačního modulu je nutné vytvořit soubor s koncovkou **.cc** (popis jednoduchých komponent), soubor s koncovkou **.ned** (navržená topologie sítě, propojení komponent), soubor s koncovkou **.ini** (pro automatické nastavení parametrů simulace), případně soubor s koncovkou **.msg** (popis posílaných zpráv v rámci komponent). Vývojové prostředí se snaží práci co nejvíce ulehčit, tam kde to jde, je možnost vyplnit formulář a následně se provede automatické vygenerování konfiguračního souboru (případ souborů s koncovkou .ini). Simulace spuštěná v Tkenv zobrazuje graficky zasílané zprávy v čase a statistická data.

Jedny z nejdůležitějších požadavků pro plnohodnotnou simulaci sítě jsou sledování cesty jednotlivých paketů v čase, přesné vytvoření modelů, simulace pádu linky, ztráty paketu a filtrování paketu. OMNeT++ je svou komplexností a poskytovanými možnostmi vybrán jako vhodný kandidát pro splnění těchto požadavků. Vývojová skupina ANSA[16], která se zabývá automatizovanou verifikací sítí na základě jejich konfigurace, vytvořila v prostředí OMNeT++ komponentu

„ANSA Router“. Díky této komponentě lze simulovat chování protokolů RIP, OSPF, EIGRP, případně nastavit ACL. Více informaci ohledně OMNeT++ lze nalézt v [3],[14].



Obr. 1.4 Simulační prostředí Tkenv

1.4 Validační část

Po verifikování návrhu a simulací různých scenérií se dostávají na řadu validační techniky. Tato podkapitola vysvětlí pojem validace. Popíše obecný koncept použití validace a její zásadní přínos v oblasti ověření správného chování počítačové sítě. Zmíní parametry sítě, jaké lze validovat pomocí sledování reálného provozu a čím se tato technika odlišuje od již zmíněných metod.

Validace je proces hodnocení produktu během nebo na konci vývoje pro zaručení shody s jeho požadavky [5]. Prokazuje vhodnost produktu pro jeho provozní poslání (zamýšlené použití). Validace nás ujišťuje, zda vyrábíme správný produkt. Naproti tomu verifikace nás ujišťuje, zda vyrábíme daný produkt správně [6].

V teoretické rovině, pokud by byly verifikační a simulační nástroje dokonalé a zároveň by naprosto přesně emulovaly reálný provoz, vystačily by samy o sobě. Jenomže v důsledku používání uzavřených protokolů, četných chyb v softwaru i hardwaru a výrazné složitosti možného provozu v síti, je nutné návrh ověřený v „laboratorních podmínkách“ verifikační a simulační částí validovat a tím potvrdit jeho požadované chování i v reálném prostředí, tj. ve funkční nakonfigurované fyzické

síti. Validace pomáhá odhalit situace, se kterými nebylo počítáno, nebo které nebylo možné popsat formálním a simulačním modelem. V reálném provozu se tedy zkoumá, zda se síť chová dle předem specifikovaných požadavků, ale umělé problémy jako úmyslné odpojování zařízení a linek se obvykle nevyvolává (Pokud ano, tak jenom v omezené míře. Komplexní testování v ostrém provozu na reálných zařízeních by bylo velice náročné, až nemožné). Cílem validace za pomocí monitorování reálného provozu je prokázat, že všechny stanovené podmínky kladené na danou síť, byly splněny. Podmínky si lze představit tak, že síť splňující tyto podmínky je ve shodě se svou specifikací, což znamená vhodnost pro své provozní poslání. Pro zaručení objektivních pozorování je nutné, aby tato validační část byla nezávislá na použitých verifikačních a simulačních metodách. Zároveň by však měla produkovat s těmito metodami porovnatelné výsledky.

Popisovaná technika validace je založena na sběru sítových toků, jejich následné analýze a rozhodnutí, zda daný tok splňuje zadané požadavky či nikoliv. K tomu je třeba mít v síti vhodně rozmístěné monitorovací sondy, znát síťovou topologii, počáteční konfiguraci a nakonec mít dobře definované požadavky (podmínky) sítě určené k validaci. Proces monitorování a validace je rozdělen do offline a online fáze. Během offline fáze konfigurační proces při zpracování vstupních dat vygeneruje nastavení pro monitorující sondy a validační proces. Následně v online fázi monitorující sondy sbírají síťové toky, které jsou po čase přeposílány centrální jednotce. Data z této jednotky jsou validačním procesem důkladně analyzována a na základě výsledků analýzy je zhotovena zpráva pojednávající o míře splnění zadaných podmínek sítě.

Požadavky na splnění podmínek sítě lze vyjádřit ve formě různých omezení jako ACL (access control list), nebo ve formě dohody jako SLA (service level agreement). Je třeba si uvědomit omezení vzniklé v důsledku použití validační techniky založené na sběru síťového toku. V tomto případě nelze jednoduše získat použitelné informace o časových charakteristikách jako například zpoždění paketu, nebo konvergenční čas použitého protokolu, které lze naopak dobře ověřovat v rámci simulační části. Pro validaci těchto problémových časových parametrů je třeba využít navíc optimalizovaných přístupů využívající další specifické sondy v síti s přesnou časovou synchronizací. Více o možnostech validace (konkrétně návrhu a implementaci nástroje za účelem validování síťových požadavků) bude popsáno v následujících kapitolách této diplomové práce.

1.5 Základní pojmy

Na tomto místě jsou vysvětleny zásadní pojmy, principy a technologie, kterých je využito při vytváření této diplomové práce. Jedná se zejména o technologii NetFlow a nástroj NFDUMP.

1.5.1 NetFlow

NetFlow [17] je Cisco technologie umožňující sledování charakteristických informací posílaných dat v síti neboli síťového provozu. Na rozdíl od IDS/IPS systémů nezkoumá obsah paketu, ale pouze informace vyskytující se v hlavičce datových paketů. Základní myšlenkou je rozdělení síťového provozu na síťové toky. V síti existuje zařízení nazývající se exportér (sonda, router), jehož cílem je sledování procházejících paketů, na základě kterých dojde k vytvoření nového nebo aktualizaci existujícího sítového toku. Zaznamenávají se obvykle statistické údaje jako počet přenesených bajtů a čas trvání toku. Po nějaké době dojde k přenesení záznamu o sítovém toku na kolektor, což je zařízení pro dlouhodobé uložení a analýzu sítových toků. Nad kolektorem pak obvykle pracuje nějaký nástroj, který informace o tocích zpracovává. NetFlow v9 je nejnovější verzí, oproti svým předchůdcům se liší zejména použitím šablon, které umožňují flexibilnější přístup k definování a sledování sítových toků.

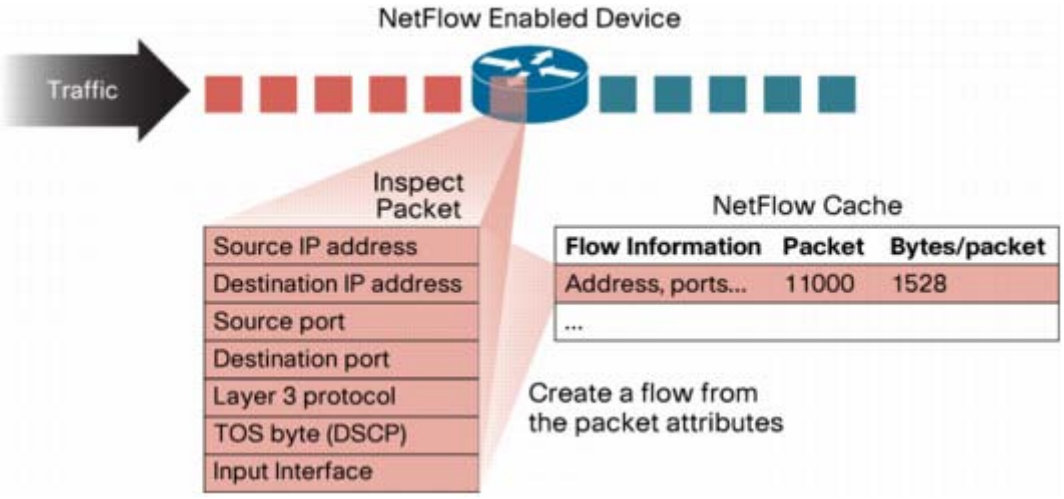
Uplatnění znalosti síťových toků lze najít při dodržování sítové politiky (limity příchozího a odchozího provozu, detekce skrytých serverů), účtování za poskytnuté služby, zobrazení spektra síťového provozu, odhalení podezřelých aktivit (skenování IP adres a portů, DoS) nebo také měření kvality služeb - QoS[18]. Nyní bude po úvodních informacích hlouběji vysvětlen síťový tok, FlowSet, NetFlow záznam, NetFlow paket, exportér, kolektor a celková architektura NetFlow v9.

Síťový tok, někdy označován jako IP tok, je tvořen množinou paketů s určitými stejnými vlastnostmi procházející daným místem (jedno nebo více sítových rozhraní) po určitou časovou dobu. Stejné vlastnosti, které musí pakety patřící pod jeden síťový tok splňovat, jsou zdrojová IP adresa, cílová IP adresa, zdrojové číslo portu, cílové číslo portu, číslo protokolu, někdy je vhodné navíc použít ToS a rozhranní směrovače/přepínače[19].

NetFlow záznam obsahuje informace týkající se síťového toku a může nabývat těchto typu: datový záznam, šablona datového záznamu, konfigurační záznam a šablona konfiguračního záznamu. NetFlow v9 používá pro své záznamy šablonovací systém, to znamená, že struktura datového záznamu (datové typy a jejich interpretace) je definována šablonou datového záznamu a struktura konfiguračního záznamu je definována šablonou konfiguračního záznamu. Datový záznam obsahuje informace vypovídající o jednotlivém síťovém toku jako počet paketů, počet bajtů, celkový čas měření toku, maska podsítě, TCP příznaky. Oproti tomu konfigurační záznam obsahuje informace o nastavení měřícího procesu na rozhraní exportéru, udává například míru vzorkování (pravděpodobnost, s jakou bude paket monitorován), typ použitého vzorkovacího algoritmu nebo celkový počet naměřených toků. NetFlow záznamy o stejné struktuře lze seskupovat dohromady a v takovém případě se nazývají **FlowSet (**existuje tedy datový FlowSet, šablonový FlowSet a šablonový konfigurační FlowSet**)**, toho se využívá pro optimalizaci posílání zpráv na kolektor. **NetFlow paket** je zpráva protokolu NetFlow, zaslaná exportérem směrem na kolektor. Tato zpráva obsahuje hlavičku, následovanou jedním nebo více typů FlowSet. Hlavička poskytuje informaci

o jakou verzi NetFlow protokolu se jedná, počet NetFlow záznamů obsažených v těle zprávy a sekvenční číslo paketu.

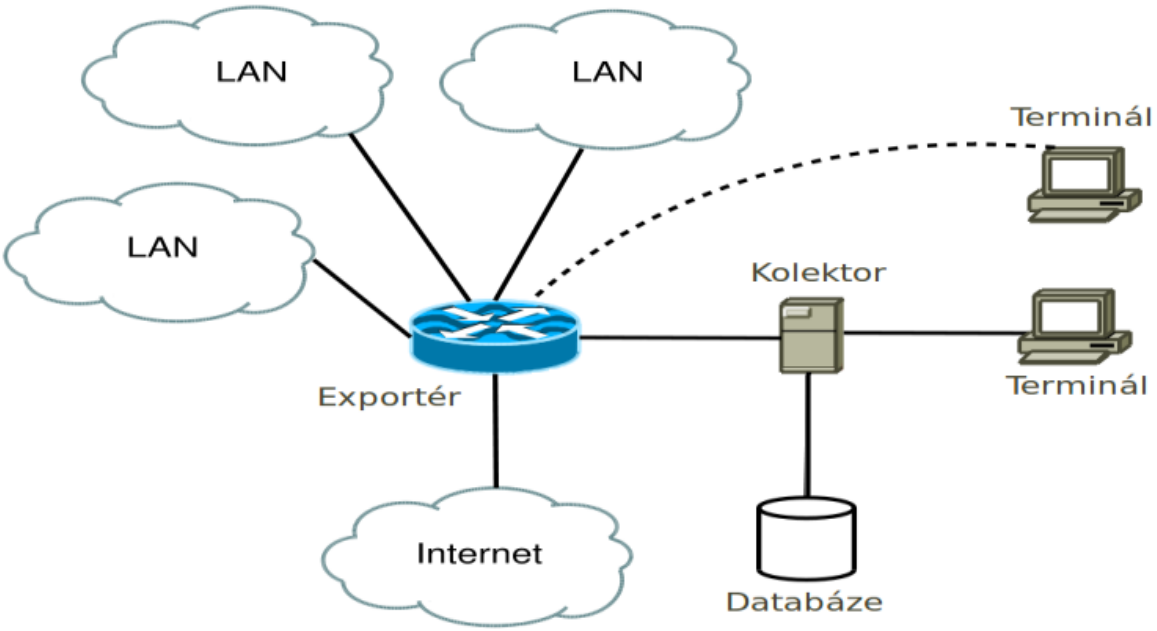
Exportér je zařízení připojené do sítě, na kterém lze monitorovat procházející komunikaci, z ní vytvářet síťové toky a později exportovat ve formě NetFlow záznamu. Exportér může představovat router, switch nebo specializované zařízení. V případě routerů může dojít k snížení výkonu nutného pro směrování a v takovém případě je třeba využít vzorkování. Naproti tomu specializované zařízení (sonda) je optimalizována pouze pro tyto účely a lze předpokládat schopnost monitorovat veškerý provoz na síti procházející skrze exportér. Po přijetí paketu exportérem jsou zaznamenány informace z jeho hlavičky, na základě kterých dojde k vyhledání příslušného datového záznamu síťového toku v paměti exportéru. Pokud není žádný nalezen, dojde k vytvoření záznamu nového (čímž vznikne nový síťový tok). V případě nalezení příslušného záznamu se jeho údaje dle procházejícího paketu aktualizují. Obr. 1.5 ilustruje činnost vedoucí k vytvoření záznamu síťového toku na exportéru, konkrétně zachycení paketu, zkoumaní dat z hlavičky paketu a následné vytvoření příslušného záznamu toku. Po určité době v závislosti na prostředcích použitého exportéru dojde k přesunutí NetFlow záznamu na kolektor. Ideální je přesunout záznam až po ukončení síťového toku, který reprezentuje. Konec lze detekovat pomocí TCP příznaku FIN a RESET, případně podle času po který nebyl záznam aktualizován (neaktivní timeout) nebo podle maximální délky trvání síťového toku (aktivní timeout).



Obr. 1.5 Proces vytvoření záznamu síťového toku a jeho uložení **[19]**

Kolektor je zařízení přijímající NetFlow pakety pocházející z jednoho nebo více exportérů. Tyto pakety analyzuje a vytvoří z nich NetFlow záznamy, které jsou uloženy a zpřístupněny pro administraci. Navíc obvykle bývají součástí kolektoru nástroje pro zpracování a prezentaci NetFlow záznamů. Výsledkem je zobrazení statistických dat, vývojových trendů, spektra síťového provozu, síťových anomálií nebo podezřelých aktivit. NetFlow záznamy mohou být uloženy do databáze (s výhodou lze použít dotazování) nebo do souboru (rychlejší oproti databázi).

Na Obr. 1.6 je vyobrazena **NetFlow architektura** skládající se z exportéru umístěném v rámci směrovače a kolektoru využívající pro dlouhodobější uložení záznamů služeb databáze. Administraci a zobrazení NetFlow záznamů exportéru i kolektoru lze provádět skrze terminál. Je vhodné mít celou architekturu pro sbírání toků v jedné lokální síti, neboť NetFlow v9 nepodporuje bezpečný přenos dat, což je bohužel při monitorování rozsáhlých sítí obtížně dosažitelné. Pro zasílání zpráv je podporován protokol UDP a SCTP. Prvně jmenovaný protokol je vybrán pro nízkou režii, nicméně při jeho použití je vhodné mít mezi exportérem a kolektorem dedikovanou linku (viz Obr. 1.6), v opačném případě může hrozit zahlcení sítě NetFlow pakety. V těchto situacích je lépe použít protokol SCTP, který se s přetíženou linkou umí vyrovnat.



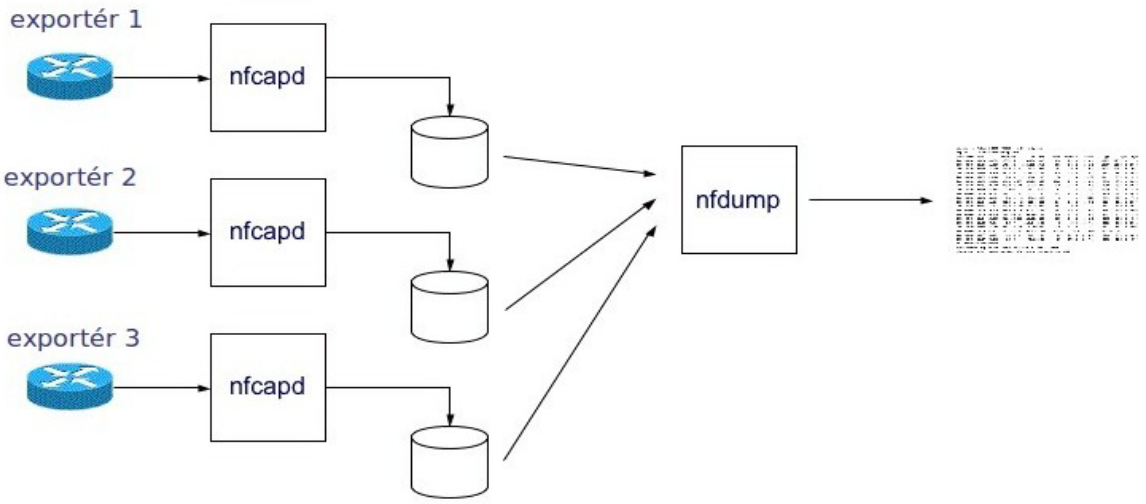
Obr. 1.6 Architektura NetFlow **[20]**

1.5.2 NFDUMP

NFDUMP [21], označuje množinu nástrojů pro sběr, zpracování a analýzu NetFlow dat. Jsou napsané v jazyku C, pracují v příkazové řádce a jsou tedy dostatečně rychlé. Umožňují agregaci toků, filtrování i tvorbu různých statistik. Konkrétně se jedná o nástroje nfcapd, nfdump, nfprofile, nfreplay, nfclean.pl, ft2nfdump. **Nfcapd (**NetFlow capture daemon**)** čte periodicky NetFlow data ze sítě a ukládá je do souboru na disku, typicky každých pět minut. Pro každý NetFlow stream pocházející ze sítě je nutné mít spuštěný jeden nfcapd proces. **Nfdump** (NetFlow dump) čte na vstupu data ze souboru uloženém pomocí nfcapd a zobrazuje je ve formě různých top statistik řazených dle zvolených vlastností. **Nfprofile** (NetFlow profiler) čte data ze souboru uloženého pomocí nfcapd, na tato data uplatňuje sadu filtrů a uloží výsledek opět do souboru. **Nfreplay** (NetFlow replay) data uložené pomocí nfcapd přeposílá po síti dalšímu uzlu. **Nfclean.pl** je skript pro odstranění starých

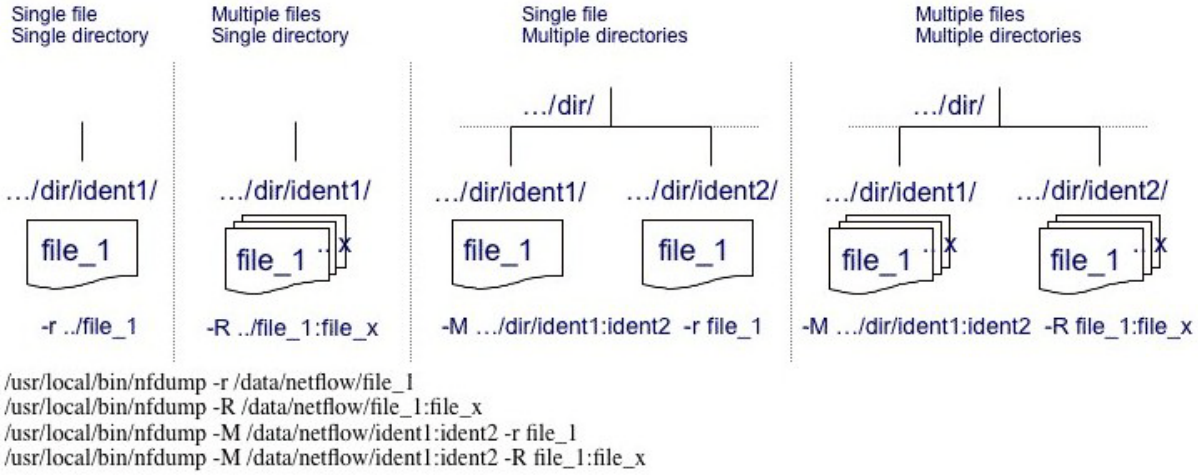
nepotřebných NetFlow dat. **Ft2nfdump** ze standardního vstupu nebo souboru čte příkazy nástroje flow-tools a převádí je pro zpracování v nástroji nfdump.

NFDUMP je schopen analyzovat NetFlow data pocházející z minulosti i současnosti. Omezení je dáno pouze maximální kapacitou úložného prostoru. Nástroje jsou optimalizovány na rychlost filtrování dat. Použité filtry mají podobnou syntaxi jako v nástroji tcpdump. Před analýzou jsou veškerá data uložena na disk, čímž je oddělen proces ukládání od zpracování. Obr. 1.7 představuje trasu NetFlow dat od jejich vzniku na exportéru, vysláním NetFlow paketu, jeho zachycení procesem nfcapd a uložením na disk. Poté jsou tato data přečtena procesem nfdump a následně jsou zobrazena výsledná statistická data.



Obr. 1.7 Cesta NetFlow dat při zpracování nástrojem NFDUMP **[21]**

Proces nfcapd periodicky (typicky každých 5 minut) ukládá soubor pojmenovaný dle času vzniku ve formátu nfcapd.YYYYMMddhhmm. Velkou výhodu tohoto přístupu je možnost načíst současně data (vytvořená nástrojem nfcapd) z různých adresářů a dokonce různých časových intervalů viz Obr. 1.8 ilustrující možné kombinace načítání souborů včetně použitých příkazů.



Obr. 1.8 Možné kombinace načítaní NetFlow souborů a použité příkazy [21]

Zpracované NetFlow data nástrojem nfdump mohou být vypsána na standardní výstup, nebo binárně do souboru, který může být opět přečten a znovu zpracován. Parametrem -o lze zadat způsob formátování výstupních dat, která mohou být raw, line, long, extended a custom. **Raw formát** (-o raw) zobrazí každou položku jednoho NetFlow záznamu na jeden řádek viz Obr. 1.9e, přičemž vypíše všechny možné dostupné položky. **Line formát (**-o line**)** je výchozí formátování, zobrazuje každý NetFlow záznam na jeden řádek viz Obr. 1.9a. **Long formát** (-o long) je stejný jako line, až na to, že přidává ve výpisu další položky jako TCP příznaky a ToS viz Obr. 1.9b. **Extended formát** (-o extended) přidává oproti long formátu položky jako pps (pakety za sekundu), bps (bity za sekundu) a bpp (bajtů na paket) viz Obr. 1.9c. **Custom formát** (například `-o fmt:"fmt:%ts %td %pr %sap -> %dap %pkt %byt %fl"`) je nejvíce flexibilní, uživatel sám specifikuje pomocí tagů zobrazených na Obr. 1.9d jak má formát vypadat a co všechno má obsahovat.

a) Line formát

```
Date flow start         Duration Proto    Src IP Addr:Port         Dst IP Addr:Port   Packets    Bytes Flows
2005-08-30 06:59:52.338    0.001 UDP    36.249.80.226:3040  ->   92.98.219.116:1434        1      404     1
```

b) Long formát

```
Date flow start         Duration Proto    Src IP Addr:Port         Dst IP Addr:Port   Flags Tos  Packets    Bytes Flows
2005-08-30 06:53:53.370   63.545 TCP    113.138.32.152:25    ->   222.33.70.124:3575  .AP.SF   0       62     3512     1
2005-08-30 06:53:53.370   63.545 TCP    222.33.70.124:3575   ->  113.138.32.152:25    .AP.SF   0       58     3300     1
```

c) Extended formát

```
Date flow start         Duration Proto    Src IP Addr:Port         Dst IP Addr:Port   Flags Tos  Packets    Bytes      pps      bps    Bpp Flows
2005-08-30 06:53:53.370   63.545 TCP    113.138.32.152:25    ->   222.33.70.124:3575  .AP.SF   0       62     3512        0      442     56     1
2005-08-30 06:53:53.370   63.545 TCP    222.33.70.124:3575   ->  113.138.32.152:25    .AP.SF   0       58     3300        0      415     56     1
Time window: Aug 30 2005 06:53:53 - Aug 30 2005 06:54:56
```

d) Tabulka předdefinovaných tagů

Tag	Description	Tag	Description
%ts	Start Time - first seen	%in	Input Interface num
%te	End Time - last seen	%out	Output Interface num
%td	Duration	%pkt	Packets
%pr	Protocol	%byt	Bytes
%sa	Source Address	%fl	Flows
%da	Destination Address	%pkt	Packets
%sap	Source Address:Port	%flg	TCP Flags
%dap	Destination Address:Port	%tos	Tos
%sp	Source Port	%bps	bps - bits per second
%dp	Destination Port	%pps	pps - packets per second
%sas	Source AS	%bpp	bps - Bytes per package
%das	Destination AS		

e) Raw formát

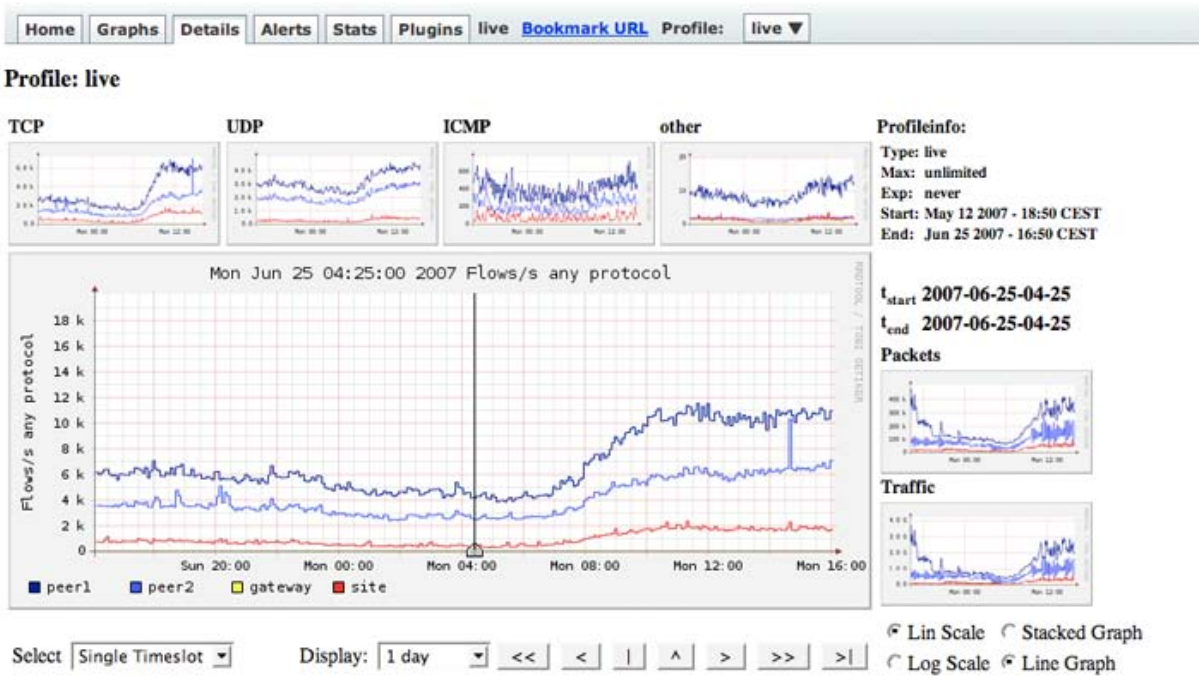
```
Flow Record:
  Flags       =      0x00000000
  size        =              52
  mark        =               0
  srcaddr     =    36.249.80.226
  dstaddr     =    92.98.219.116
  First       =      1125377992 [2005-08-30 06:59:52]
  Last        =      1125377992 [2005-08-30 06:59:52]
  msec_first  =             338
  msec_last   =             338
  dir         =               0
  tcp_flags   =               0
  prot        =              17
  tos         =               0
  input       =               5
  output      =               3
  srcas       =            1299
  dstas       =               0
  srcport     =            3040
  dstport     =            1434
  dPkts       =               1
  dOctets     =             404
```

Obr. 1.9 Možnosti formátování výstupu nástroje nfdump **[21]**

Použitím parametru -a dojde ke sloučení síťových toků se stejnými hodnotami položek: protokolu, zdrojové a cílové IP adresy, zdrojového a cílového portu. Pokud je třeba nastavit speciální parametry pro slučování toků, lze použít přepínač -A následovaný jakoukoliv kombinací klíčových slov srcip, dstip, srcport a dstport oddělené čárkou. Například `-A srcip, dstport`. Nfdump má vyvinutý

a optimalizovaný mechanismus pro filtrování záznamů. Ještě před zpracováním toků je lze efektivně filtrovat použitím filtrovacího pravidla v apostrofech jako posledního parametru příkazu, nebo načtením příslušného filtru ze souboru. Syntaxe filtrovacích pravidel je podobna aplikaci tcpdump. Každý filtr může obsahovat jeden nebo více výrazů. Pro lepší představu následuje příklad spuštění nástroje nfdump s parametry pro sloučení a filtraci sítových toků: `nfdump -r /and/dir/nfcapd.200407110845 -A srcip,dstport 'in if 5 and net 10.0.0.0/24 and not host 10.0.0.1 and bps > 10k and duration < 100 and dst port 1433'`. Další možností použití nástroje nfdump je vytváření statistik a řazení, což se docílí použitím parametru -s typ_statistiky/řazení obvykle doplněný parametrem -n určující počet zobrazených záznamů. Typ statistiky může nabývat hodnot: record, srcip, dstip, ip, srcport, dstport, port, srcas, dstas, as, inif, outif, if, proto. Znamená zobrazení nejčastěji se vyskytující hodnoty vybrané statistiky ve vstupních NetFlow záznamech. Například pro statistiku -s srcip -n 10 bude zobrazeno deset nejčastěji se vyskytujících zdrojových IP adres v dodaných datech. Zobrazené záznamy lze řadit podle počtu toků, paketů, bajtů, pps, bps nebo bpp. Výchozí hodnotou pro řazení je počet toků.

NFDUMP může produkovat i grafické zobrazení výsledku, což je docíleno za předpokladu použití jeho grafické nadstavby s webovým rozhraním jmenující se NFSEN [22]. Jedná se o oddělenou část a pro jeho fungování je třeba NFSEN (NetFlow sensor) doinstalovat. Ukázka výstupu nástroje NFSEN je na Obr. 1.10.



Obr. 1.10 Aplikace NFSEN **[22]**

2 Analýza a návrh řešení

Druhá kapitola nejprve konkrétně představí cíle diplomové práce. Dále budou specifikovány požadavky na výstupní projektovou část, čímž budou zodpovězeny základní typické otázky, jako například: Pro koho je výsledný produkt určen? Jaké výsledky lze očekávat? Jaký formát výstupu a ovládání je vhodné vybrat? Jaké je předpokládané použití s výhledem na další rozvoj? Následně bude popsána analýza problematiky rozebírající možné parametry validovatelné sledováním skutečného provozu, jaké objemy dat lze očekávat, jakým způsobem data uložit, jaké technologie a programovací jazyky lze pro tento účel využít, a další možné problémy, s kterýma je třeba počítat. Nakonec je předložen návrh řešení nástroje pro validaci parametrů sítě, jehož implementace bude později reálně provedena.

2.1 Cíle práce, specifikace požadavků

Hlavním cílem diplomové práce je nalezení vhodných parametrů, které by co nejpřesněji popisovaly typicky požadované chování počítačové sítě. Následně vytvořit nástroj umožňující (za pomocí sledování síťového provozu) zvolenou množinu parametrů validovat. Čímž lze poté rozhodnout, zda se sledovaná síť chová správně (splňuje dané požadavky) nebo nikoliv, případně konkrétně určit, který síťový provoz nesplňuje konkrétní požadavky. Práce vychází z konceptu pro verifikaci validaci a monitorování popsaném v první kapitole a je snaha pokud možno tuto práci do zmíněného konceptu začlenit. Pojem validace a možnost sledování síťového provozu pomocí NetFlow technologie již bylo v první kapitole popsáno. V této kapitole budou z větší míry již představené znalosti využity pro navržení požadovaného nástroje.

Výsledný nástroj je určen pro testování správného návrhu sítě v praxi, je určen pro uživatele, kteří počítačové sítě spravují a lze je považovat za odborníky. Předpokládá se grafické rozhraní s intuitivním ovládáním, schopné zobrazit výstižně a přesně výsledky validace. Hlavně je třeba případné nalezené problémy v síti dobře zdokumentovat, vybrat vhodný způsob zobrazení chybového výstupu, aby vývojáři snadno pochopili zdroj a příčinu problémů a mohli následně opravit konfiguraci sítě a spustit proces validování sítě znovu. Postačí jednouživatelský přístup k aplikaci, z logiky věci není potřeba, aby nástroj ukládal informace o uživatelích. Pro sledování a sběr síťového provozu se očekává využití technologie NetFlow. Uložení filtrování a zpracování NetFlow dat bude probíhat pomocí sady nástrojů NFDUMP. Vlastnosti sítě (parametry) určené pro validaci by měly být logicky strukturovány do skupin a vytvořený nástroj s nimi musí umět pracovat, to znamená načíst je nebo nadefinovat a vyexportovat pro další použití. Obecně by měl být umožněn import a export nastavení aplikace do konfiguračního souboru, zejména tedy popis jednotlivých zařízení v síti, požadavky na síťové parametry k validaci a nastavení zdrojových dat síťových toků. Běh aplikace

bude podporovat online i offline zpracování. Online zpracování umožní validovat parametry sítě i na základě aktuálně probíhající komunikace v síti, kdežto offline zpracování nabídne tuto funkcionalitu pro komunikaci nasbíranou z minulosti. Celkově se počítá s vytvořením základní verze aplikace, kterou bude možné v navazujícím výzkumu rozšiřovat o další funkcionalitu, zejména například zvětšováním podpory parametrů určených k validaci.

2.2 Analýza problematiky

V této části práce je rozebrána problematika validace vybraných síťových parametrů. Tyto parametry popisující požadované chování sítě budou rozděleny do skupin a bude naznačen možný způsob jejich validace. Následně budou analyzovány objemy síťových toků a jejich možnosti uložení. Nakonec budou zmíněny navržené programovací jazyky a technologie použité pro splnění cílů této práce.

2.2.1 Síťové parametry vhodné k validaci

Na první pohled zásadní věc pro splnění cílů práce je vhodně určit parametry k validaci. Při použití NetFlow toků je nutné počítat při vybírání vhodných parametrů s jistými omezeními. Záznamy síťových toků neobsahují informace o času, kdy pakety procházejí jednotlivými místy sítě. Proto není příliš snadné pomocí této techniky validovat časové charakteristiky, jako například zpoždění paketů (QoS) nebo konvergenční čas protokolů, které jsou naopak dobře analyzovatelné v simulačních nástrojích. Z tohoto důvodu je třeba počítat v pokročilejším stádiu této problematiky s rozšířením NetFlow technologie o přesnou časovou synchronizaci exportérů s přesnou lokalizací a použití dalších síťových sond specificky nastavených pro sbírání určených paketů k validaci síťových parametrů, které by pouze s technologii NetFlow validovat nešly. Jedná se zejména o zmíněné časové parametry.

Naproti tomu pro ověření různých síťových omezení, například filtrovacích pravidel, v praxi obvykle zadaných ve formě ACL, lze očekávat NetFlow data jako postačující, stejně jako pro ověření splnění maximální a minimální propustnosti linky nebo limitů příchozího a odchozího provozu. ACL (access kontrol list) představuje seznam pravidel řídících přístup k nějakému objektu. V oblasti počítačových sítí jsou nejčastěji ACL používány na aktivních prvcích pro omezení (filtrování) síťového provozu. Pro představu může existovat omezení (tedy parametr k zvalidování) přikazující v dané oblasti sítě používat pouze protokol SSH, nebo povolující komunikaci pouze určitým IP adresám v síti.

Další možnou validovatelnou vlastností je sledování dostupnosti určitých oblastí (uzlů, podsítí, sítí) v rámci zkoumané sítě. Lze si představit seznam všech oblastí, u nichž se požaduje definovaná míra dostupnosti, která by mohla být dále rozvinuta specifikováním dalších podmínek. Například je zadán požadavek, že uzly A, B, C musí být dosažitelné při použití konkrétních portů i za

podmínky, že došlo k výpadku uzlů D, E, F. Mírou dostupnosti se rozumí čas, po který musí být daná oblast dostupná, ale také i jako míru úspěšné komunikace z celkového počtu pokusů o komunikaci. Například požadavek může znít: „Z 1000 pokusů o komunikaci s webovým serverem je požadováno, aby 950 bylo úspěšných“.

Zmíněné síťové parametry lze rozdělit do tří hlavních skupin na **omezující parametry** (filtrovací pravidla, minimální nebo maximální propustnost, limity příchozího a odchozího provozu, maximální celkové zatížení sítě), **parametry popisující dosažitelnost uzlů** v síti a **časové parametry** (zpoždění paketů, odezva).

2.2.2 Nástin možností validace zvolených parametrů

Po základní představě parametrů sítě následuje rozbor možností jejich validace za pomocí síťových toků. Pro každý implementovaný parametr sítě je nutné vytvořit vlastní validační postup a uvědomit si jaké vstupní informace jsou potřebné a jaká je možná forma výstupu. Následující text obsahuje pro konkrétní parametry jejich možný způsob validace.

Omezující parametry

Ze skupiny omezujících parametrů lze vybrat filtrovací pravidla a limity odchozího a příchozího provozu. Požadavek na parametr sítě popisující limit odchozího/příchozího provozu musí obsahovat zdrojovou a cílovou oblast. Například mějme počítač A (147.229.206.191/23), který může odeslat směrem do wan rozhraní pouze 4GB dat za 4 dny. Takto definovaný požadavek lze validovat sesbíráním všech toků, které mají zdrojovou IP adresu 147.229.206.191 a zároveň cílová IP adresa nepochází ze sítě 147.229.206.0/23. Pro tyto toky vzniklé v intervalu do 4 dnů se sečte počet přenesených bajtů, přičemž pokud bude hodnota přenesených dat mít do 4GB, lze tvrdit, že daný požadavek byl splněn. V opačném případě dojde k porušení požadavku a je třeba z této skutečnosti vyvodit důsledky.

Při validaci filtrovacích pravidel zadaných ve formě ACL nastává několik problémů. Jednotlivá ACL jsou aktivována na určitém rozhraní aktivního prvku sítě, což omezuje oblast, ve které tyto požadavky platí. Dá se říci, že konkrétní ACL je platné pouze pro síťové toky, které prochází daným rozhraním. Z toho vyplývá, že musíme mít mechanizmus, jak oddělit toky procházející daným rozhraním od toků ostatních. Řešením by bylo, mít na každém rozhraní s aktivním ACL seznamem exportér a validovat pouze toky pocházející z daného exportéru. Vedlejším efektem tohoto přístupu je ale značné zvýšení zatížení směrovačů a může dojít i k nežádoucímu snížení propustnosti sítě. V případě použití speciální sondy jako exportéru je toto řešení nepoužitelné. Druhou možností je sloučení všech ACL do jednoho, které by platilo pro celou zkoumanou síť. V takovém případě není potřeba vědět, ze kterého exportéru toky pocházejí, protože validace může probíhat pro všechny dostupné síťové toky. Kritickým bodem tohoto řešení je převod všech ACL do jednoho (popisující chování všech v rámci celé sítě) bez změny významu. Nabízí se

také řešení využít ACL jako referenční metodu popisu filtrování paketů, pomocí které by se vytvořil požadavek na filtrování paketů v rámci celé sítě. To znamená, nepřeváděly by se jednotlivé ACL ze všech rozhraní, ale rovnou by se vytvořilo něco jako jedno nové ACL popisující chování filtrování paketů celé sítě.

Dalším problémem, se kterým je třeba se vyrovnat, je samostatný proces validující ACL. Takový seznam se skládá z filtrovacích pravidel seřazených dle priority, kde každé pravidlo zakazuje nebo povoluje určitý provoz v síti. V případě, že procházející paket splňuje zadaný filtr nějakého pravidla, bude dané pravidlo provedeno (zahození nebo poslání paketu dále). Následující pravidla v seznamu se pak již nevykonávají. Pokud chceme ověřit, že daný provoz skutečně odpovídá tomu, který ACL povoluje, je třeba vyjádřit všechna filtrovací pravidla pomocí mechanizmu, který by zaručil správné vyhodnocení s ohledem na prioritní postavení pravidel v rámci jednoho ACL. Článek [23] dokazuje, že filtrovací pravidla obsažená v ACL lze reprezentovat jako stromy vhodně zanořených logických formulí. Další práce [9] pak tuto informaci využívá a ukazuje převod ACL do formule predikátové logiky. Tohoto principu lze pří validování síťových toků dobře využít, kde pomocí negací vzniklého predikátu a jeho přepsání jako filtru použitém v nástroji NFDUMP, dojde k zobrazení množiny síťových toků, které definované ACL nesplňují (pro účely validace jsou téměř vždy důležité toky nesplňující specifikaci sítě, pokud žádné takové nejsou, lze prohlásit konkrétní požadavek za úspěšně zvalidovaný). Tabulka Tab. 2.1 ukazuje příklad ACL, jeho převod do predikátové logiky a následnou úpravu a použití jako filtru v aplikaci NFDUMP. Více o tomto postupu a použití je v další kapitole popisující implementaci práce.

Tab. 2.1 Příklad ACL, jeho převod do predikátové logiky a následně vytvoření filtru pro NFDUMP

1	ACL pravidla	Logické formule ACL pravidel
	permit tcp 192.168.1.0 0.0.0.255 any 80	$f_1(h) = (h.protocol \in TCP) \wedge (h.srcIP \in 192.168.1.0/24) \wedge (h.dstPort = 80)$
	deny ip 192.168.1.0 0.0.0.255 any	$f_2(h) = (h.protocol \in IP) \wedge (h.srcIP \in 192.168.1.0/24)$
	permit ip any any	$f_3(h) = (h.protocol \in IP)$
	Význam ACL: Komunikace se zdrojovou ip z rozsahu 192.168.1.0/24 je až na cílový port 80 zakázána. Ostatní ip adresy mohou komunikovat bez omezení.	
2	Výsledný filtrovací predikát zohledňující priority pravidel a jejich akci (deny | permit)	
	$\alpha = f_1(h) \vee (\neg f_2(h) \wedge f_3(h))$	
3	Úprava predikátu a vytvoření filtru použitelného v nástroji NFDUMP	
	not (src net 192.168.1.0/24 and proto tcp and dst port 80 or (not (src net 192.168.1.0/24)))	
	Význam filtru: Jsou zobrazeny toky, u kterých je filtr pravdivý, tj. ty, které mají zdrojovou ip z rozsahu 192.168.1.0/24 a jejich protokol není tcp nebo cílový port není 80.	

Parametry popisující dosažitelnost uzlů v síti

Při validaci požadavku na dosažitelnost určité oblasti v síti lze vycházet z přístupu popisující článek [24], který se zabývá sledováním dosažitelnosti vzdálených autonomních systémů, sítí a uzlů z vnitřní zkoumané sítě. Jejich metoda je založena na sběru toků z oblasti hraničních směrovačů a jejich následné analýze. Každých pět minut jsou nově vzniklé toky zpracovány a je snaha k tokům směřující ven ze sítě nalézt odpovídající tok směřující dovnitř. Po určité optimalizaci jako ignorování toků, které nepřináší vhodné údaje (průchozí komunikace, pouze příchozí toky, multicast), jsou dále zkoumány ty toky, které směřují ven ze sítě, a zároveň k nim není nalezena odpověď ve formě toku směřujícího nazpět. Z těchto informací je možno vyčíst vzdálené uzly, které v daném časovém intervalu nebyly ze zkoumané sítě dosažitelné. Navíc je možno sestavit důležitost nastalé situace podle počtu různých toků směřujících na stejné cílové umístění bez odpovědi.

V našem případě se snažíme definovat a posléze validovat chování v rámci jedné sítě. Požaduje se tedy ověření dosažitelnosti předem známých konkrétních zařízení v síti. Jako možné řešení se nabízí sledování všech toků směřující na definovaná místa v síti, u kterých nás zajímá jejich míra dosažitelnosti a následné hledání existence síťového toků obsahující jejich odpověď. Pro rozhodnutí, zda je daný uzel nedostupný, je nutné počítat i s analýzou zdrojové adresy snažící se dosáhnout sledovaný uzel. Čím více je toků s různými zdrojovými adresami a stejnou cílovou adresou, které nemají svou odpověď, tím větší je pravděpodobnost, že uzel daný cílovou adresou je nedostupný.

Časové parametry

Jak již bylo naznačeno v úvodu, časové parametry lze pomocí standardní NetFlow technologie jen těžce ověřovat. Jistým řešením by bylo použití specializovaných flexibilních sond umístěných na určených trasách, které by zkoumaly čas průchodu prvního paketu nově vzniklého toku a případně čas prvního paketu jeho odpovědi. Z takto nasbíraných dat by teoreticky šlo ověřovat časovou specifikaci, jako například maximální zpoždění paketu na určité cestě, nicméně je dobré si uvědomit, jaké další komplikace to přináší. Jedná se zejména o vyřešení přesné časové synchronizace sond sbírající síťové toky a zaznamenávající čas průchodu. Dále mít vhodně rozmístěné exportéry v síti, což také znamená mít k dispozici přesnou topologii zkoumané sítě. V závislosti na topologii je třeba se umět vyrovnat s případem, kdy odpověď může jit po jiné cestě než dotaz. Z těchto důvodů je v tuto chvíli validace části specifikace sítě týkající se časových parametrů ponechána na simulačních nástrojích, které jsou pro tento účel lépe vybaveny. Nicméně očekává se, že v pokročilejším stádiu této problematiky dojde i k validaci časových parametrů.

2.2.3 Objemy dat síťových toků a možnosti jejich uložení

Množství dat představující NetFlow záznamy, které kolektor uchovává, se odvíjejí od předpokládané velikosti sítě, účelu sítě, množství uživatelů, počtu a typů spuštěných služeb a v neposlední řadě od schopností použitého exportéru. Nicméně pro představu lze očekávat pro 100Mbit síť okolo 300MB NetFlow dat za hodinu a pro 1Gbit síť 600 MB, což může být po delší době problematicky uložitelná velikost. V takovém případě se vyplatí použít mechanizmy pro mazání nepotřebných dat. Například pro starší data lze snižovat míru detailu uložených toků. Pokud už byly uložené síťové toky analyzovány bez nalezení neshody s očekávanými požadavky, lze je přepsat novými čerstvými záznamy. Taktéž se vyplatí použít některé komprimovací metody použitelné pro síťové toky. Další možností omezení množství dat je nastavení vzorkování paketů na exportéru (pouze část procházejících paketů je zpracováno pro tvorbu síťových toků), čímž se sníží počet NetFlow záznamů, ale zároveň také klesá hodnověrnost validace síťových toků. V případě použití směrovačů jako exportérů je při vysoké zátěži v síti nutno vzorkování nastavit bez ohledu na použitý kolektor, neboť by v opačném případě směrovače nestíhaly plnit svůj hlavní účel, kterým je směrování paketů.

Co se týče ukládání síťových toků, nabízí se dva nejčastější způsoby, databáze a uložení do binárního souboru. Výhodou databáze je již vyřešené dotazování na uložená data pomocí SQL dotazů. Většinu práce při zpracovávaní dat tak udělá sám databázový systém. Nevýhodou je ovšem relativně pomalý příjem síťových toků do databáze, což je v případě častého zasílání dat značně omezující a může docházet k náhodným ztrátám síťových paketů. Naproti tomu ukládání dat do souboru je daleko rychlejší, ale v oblasti zpracování musí tvůrce dané aplikace doprogramovat dotazování nad síťovými toky. V rámci této práce bude pro praktickou část využita školní laboratoř, funkce exportéru budou plnit Cisco směrovače a jako kolektor bude využit nástroj NFDUMP, který ukládá síťové toky do souboru.

2.2.4 Použité technologie a programovací jazyky

Mnoho nástrojů zpracovávajících a zejména zobrazujících síťové toky jsou serverové aplikace s přístupem přes webové rozhraní. Výhodou je možnost přístupu prakticky odkudkoliv, kde je připojení k internetu, přičemž většina těchto aplikací nabízí zabezpečený víceuživatelský přístup s různými stupni oprávnění.

Nástroj navržený touto prací má poněkud odlišné požadavky, i když se z velké míry jedná o zpracování síťových toků. V tomto případě se vybrané toky vhodným způsobem použijí tak, aby bylo zřejmé, zda zkoumaná vlastnost sítě je danými toky ověřena nebo nikoliv. Uživatele pak zajímají hlavně tyto závěrečné výsledky. Použité toky se budou uživateli dále zobrazovat až v případě nalezení problémů při validačním procesu. Pro implementaci je třeba zvolit technologii umožňující tvorbu grafického rozhraní a pokud možno snadné využití služeb nástrojů NFDUMP, NFSEN, případně dalších nástrojů pro zpracování sítového provozu. Z popsané analýzy vyplývá při zohlednění

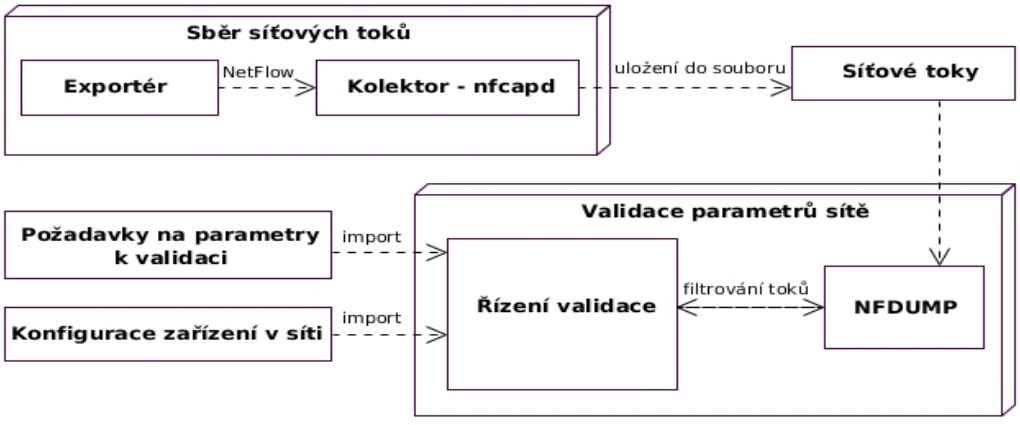
dosavadních zkušeností použití programovacího jazyka JAVA s vývojovým prostředím NetBeans. Při vývoji se předpokládá spolupráce s akademickými pracovníky a možnost použití jejich dosavadních výstupů. Například pro získání počáteční konfigurace celé sítě se nabízí použití vytvářeného převaděče konfigurace [2], který provádí překlad jednotlivých konfigurací všech aktivních prvků sítě do jednoho konfiguračního xml souboru.

2.3 Návrh řešení

V této kapitole je rozebrán základní návrh řešení aplikace pro validování požadavků definovaných parametrů sítě. Budou zde popsány vstupy a výstupy této aplikace včetně dalších nástrojů, jejichž služeb bude využito a předpokládané procesy uvnitř navrhovaného nástroje.

Nutná vstupní data pro proces validace jsou různá v závislosti na typech podporovaných parametrů sítě. Obecně je třeba mít vždy k dispozici definované požadavky na parametry sítě a síťové toky. Jako doplňující informace lze využít konfigurační údaje sítě. **Požadavky na síťové parametry** souhrnně definují očekávané chování zkoumané sítě. Jedná se tedy o konkrétní vlastnosti sítě, které jsou určeny k validaci. Výsledek celé validace má rozhodnout, zda byly tyto požadavky splněny nebo nikoliv. **Síťové toky**, jejichž popisu se věnuje první kapitola, představují zásadní vstupní data validačního procesu. Právě tyto toky budou prostřednictvím služeb nástroje NFDUMP filtrovány a zpracovávány, přičemž se budou hledat zejména ty toky, které nesplňují definované požadavky na vybrané parametry zkoumané sítě. **Konfigurační údaje sítě** obsahují informace o jednotlivých zařízeních v síti, jako název zařízení, počet rozhraní a jejich aktuální konfigurace. Očekává se, že částečně lze tuto konfiguraci získat z xml souboru, který je vytvořen pomocí nástroje převaděč konfigurace popsaném v první kapitole.

Výstupem navrhované aplikace je výsledek validace podrobně informující uživatele o splnění, či nesplnění konkrétních zadaných požadavků. Obr. 2.1 ukazuje zmíněné vstupy a výstupy validace, včetně dalších nástrojů, jejichž služeb se bude využívat.



Obr. 2.1 Nástroje podílející se na validaci, jejich vstupy a výstupy

Navrhovaný nástroj pro validaci podporuje tyto hlavní uživatelské činnosti: N**ačíst** požadavky na parametry k validaci, konfiguraci síťových zařízení, umístění síťových toků. **Nastavit** požadavky na parametry k validaci, umístění sítových toků, konfiguraci síťových zařízení. **Exportovat** požadavky na parametry k validaci, konfiguraci síťových zařízení, umístění síťových toků. **Zahájit** validační proces. **Ukončit** validační proces. **Zobrazit** výsledky validace.

Po spuštění procesu validace budou pomocí služeb nástroje NFDUMP prozkoumávány dostupné síťové toky a proběhne analýza, zda tyto toky splňují zadané požadavky na parametry sítě. Pro každý zadaný požadavek se vytvoří charakteristická konfigurace, která bude pro každý načtený síťový tok zkoumána. Podle toho, zda tuto charakteristiku tok obsahuje nebo ne, bude možné určit, zda je tok v souladu se zadaným požadavkem. Proces se opakuje, dokud nebudou zkontrolovány všechny dostupné toky nebo nedojde k ukončení procesu uživatelem. Poté bude následovat podrobné zobrazení zprávy o výsledcích validace.

2.3.1 Role NFDUMP

Nastroj NFDUMP je využit zejména pro filtrování síťových toků dle potřeb validace. Uživatel nastaví požadavky na síť a umístění sítových toků. Pro každý validovaný požadavek se vytvoří množina příkazů volání nástroje NFDUMP, jehož výsledkem je zobrazení síťových toků. Ty budou dále zpracovány ve formě skriptů nebo v aplikaci samotné. Při volání NFDUMP budou využity možnosti čtení zdrojových dat z více složek, nastavení výstupního formátu dat (vhodné pro rozlišení více výstupů v rámci validace jednoho požadavku) a filtrování toků, zejména dle zdrojových a cílových IP adres, zdrojových a cílových adres sítě s možností CIDR notace, typu protokolů a zdrojových a cílových portů. Pro ilustraci: pokud by existoval požadavek zakazující podsíti 192.168.1.3/24 jakoukoliv komunikaci pomocí protokolu telnet (číslo portu 23), vypadalo by volání nástroje NFDUMP pro tento požadavek následovně:

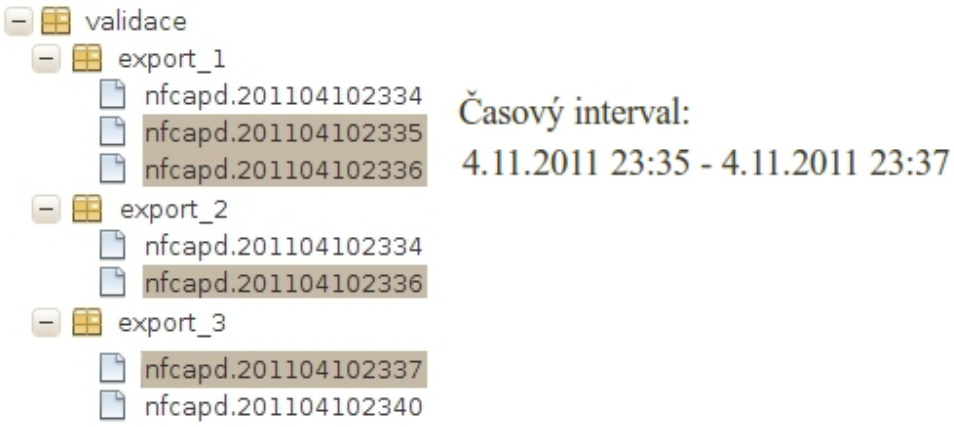
nfdump vstupni_data format_vystupu ´net 192.168.1.1/24 and port 23´

Výstupem by byla množina síťových toků, kde zdrojová nebo cílová adresa je z rozsahu 192.168.1.0 – 192.168.1.255 a zároveň číslo zdrojového nebo cílového portu je 23. V praxi se lze setkat s mnohem komplikovanějšími filtry, obzvláště při validaci požadavku ve formě ACL s více pravidly.

2.3.2 Sběr a umístění síťových toků

Proces získání síťových toků, tj. nastavení konkrétních exportérů a kolektorů v síti, je nezávislý na nastavení validačního nástroje. Nicméně vyvíjený nástroj očekává zdrojové síťové toky ve formátu vytvořeném nástrojem nfcapd. V tomto ohledu je třeba postupovat v dvou krocích. Nejprve vybrat vhodné místo v síti pro polohu exportérů a nakonfigurovat je. Současně nastavit patřičné kolektory, které budou ukládat síťové toky do souborů na disk. Vhodné je zvolit jeden kořenový adresář,

ve kterém bude pro každý exportér vytvořena složka obsahující příslušné vyexportované síťové toky. V druhém kroku se již v rámci validačního nástroje jako vstupní data nastaví určený kořenový adresář a zvolí se časový interval determinující síťové toky, které budou zkoumány. Tedy pouze síťové toky vzniklé v době určené časovým intervalem budou analyzovány. Tohoto chování je možné dosáhnout díky tomu, že název každého souboru sítového toku, vzniklého pomocí kolektoru, nfcapd obsahuje časovou značku doby vytvoření souboru a má formát nfcapd.YYYYMMddhhmm. Například nfcapd.201104112330 obsahuje všechny toky přijaté kolektorem od doby vytvoření předchozího souboru do doby 11.4.2011 23:30. Kromě zvolení časového intervalu by měl nástroj podporovat možnost zpracovávat na vstupu i toky nově vznikající, což představuje tzv. online validaci. Následující *Obr. 2.2* ukazuje možnou adresářovou strukturu uložení síťových toků s výběrem dle časového intervalu.



Obr. 2.2 Příklad vhodné struktury uložení síťových toků a jejich selekce dle časového intervalu

2.3.3 Import/export nastavení

Importovat a exportovat lze důležité nastavení a data obsahující definované požadavky na síťové parametry, konfiguraci síťových zařízení, umístění síťových toků a nastavení časového intervalu definující výběr síťových toků, které se použijí pro validaci zadaných požadavků.

3 Zvolené parametry k validaci

Kapitola obsahuje konkrétní popis vybraných parametrů, které jsou implementovány nástrojem vyvíjeným v rámci praktické části této práce. Pro každý parametr je uveden použitý specifický postup, který ověřuje, zda je požadavek na parametr splněn, včetně ilustrujícího příkladu.

3.1 Blokování komunikace $A \rightarrow|\leftarrow B$

Parametr umožňuje přidat do požadovaného chování sítě dvě množiny zařízení, které spolu nesmí komunikovat. Adresaci zařízení je možné provést zadáním rozsahu sítě, do které dané IP adresy patří nebo výběrem konkrétních zařízení v síti. Druhá popsaná možnost je povolena, pokud jsou daná zařízení obsažená v konfiguraci síťových zařízení. Krom adres zařízení je volitelně možno definovat také protokol a číslo portu. Jako množinu zařízení lze zvolit i síť 0.0.0.0/0 představující všechna zařízení. Toho lze využít například při zakázání použití nějakého protokolu v rozsahu dané sítě.

Pro ověření parametru je sestaven filtr pro NFDUMP představující toky nesplňující daný požadavek. To znamená, budou se hledat toky, jejichž zdrojová adresa je z rozsahu určeném první množinou (A) a cílová z rozsahu druhé množiny (B) nebo naopak. Přičemž pokud je zadán protokol a port jsou tyto údaje přidány do filtru také. Formát filtru představuje následující logický výraz:

((src A and dst B) or (src B and dst A)) and proto PROTO and port PORT

Po zavolání NFDUMP s vytvořeným filtrem je zpět načten výstup. Pokud výstupní data obsahují nějaká data, jedná se o síťovou komunikaci porušující zadaný požadavek. V opačném případě je parametr úspěšně validován. Tab. 3.1 obsahuje tři příklady požadavků na parametr blokování komunikace a k nim sestavený filtr. První požadavek zakazuje veškerou komunikaci mezi uzly s IP adresou 147.229.206.191 a 147.229.206.1. Druhý požadavek zakazuje komunikaci mezi sítí danou rozsahem 147.229.206.1/23 a uzlem s adresou 77.75.72.57 prostřednictvím protokolu telnet (číslo portu 23). Třetí požadavek zakazuje veškerou komunikaci prostřednictvím protokolu telnet v síti danou rozsahem 147.229.206.1/23.

Tab. 3.1 Příklady požadavků na parametr blokování komunikace a k ním sestavený filtr

Požadavek na blokování komunikace	Filtr pro NFDUMP
147.229.206.191 $\rightarrow\|\leftarrow$ 147.229.206.1	(((src ip 147.229.206.191) and (dst ip 147.229.206.1)) or ((src ip 147.229.206.1) and (dst ip 147.229.206.191)))
147.229.206.1/23 $\rightarrow\|\leftarrow$ 77.75.72.57 port = 23	(((src net 147.229.206.1/23) and (dst ip 77.75.72.57)) or ((src ip 77.75.72.57) and (dst net 147.229.206.1/23))) and port 23
147.229.206.1/23 $\rightarrow\|\leftarrow$ 0.0.0.0/0 port = 23	(((src net 147.229.206.1/23) and (dst net 0.0.0.0/0)) or ((src net 0.0.0.0/0) and (dst net 147.229.206.1/23))) and port 23

3.2 Jednosměrné omezení $A!\rightarrow\leftarrow B$

Zařízení definovaná množinou na levé straně od označení parametru mají povoleno komunikovat pouze se zařízeními definovanými druhou množinou vyznačenou na pravé straně. Je dobré si uvědomit skutečnost, že omezení se týká pouze levé množiny (pro pravou množinu z tohoto pravidla žádné omezení nevyplývá), proto je u ní znak „!“. Adresaci zařízení je možné provést zadáním rozsahu sítě, do které dané IP adresy patří nebo výběrem konkrétních zařízení v síti (pokud jsou konkrétní zařízení obsažená v konfiguraci síťových zařízení). Opět je volitelně možné dodefinovat protokol a číslo portu a v takovém případě se omezení vztahuje pouze na komunikaci při zvoleném protokolu nebo portu. Vhodné použití se nabízí, pokud existuje v síti zařízení nebo služba, na kterou má mít přístup pouze předem určená množina zařízení.

Validace probíhá sestavením filtru, který nalezne toky sítě nesplňující definovaný požadavek. To znamená, budou se hledat toky, jejichž komunikace je prostřednictvím zadaného protokolu a portu (pokud jsou definovány), zároveň zdrojová/cílová adresa je z rozsahu určeném množinou zařízení na levé straně (A) a zároveň není pravda, že cílová/zdrojová adresa je z množiny zařízení definované pravou stranou (B). Formát filtru představuje následující logický výraz:

(src A and not (dst B or not (proto PROTO and port PORT))) or směr toků $A \rightarrow B$

(not (src B or not (proto PROTO and port PORT)) and dst A) směr toků $B \rightarrow A$

Poté je proveden příkaz spuštění NFDUMP s příslušným filtrem a čeká se na výsledek. Pokud výsledek obsahuje nějaké toky, jedná se o síťovou komunikaci porušující zadaný požadavek. Tab. 3.2 obsahuje dva příklady možných požadavků na parametr jednosměrného omezení a jejich filtr pro NFDUMP. První z nich říká, že rozhraní směrovače s IP adresou 192.168.1.1 muže konfigurovat přes protokol ssh (číslo portu 22) pouze admin sítě identifikovaný IP adresou 192.168.1.10. Druhý požadavek kontroluje, aby na serveru s IP adresou 192.168.1.200 na portu 5555 byla spuštěna služba určená pouze pro síť 192.168.2.1/24, jinak řečeno na server s rozhraním 192.168.1.200 na portu 5555 se nesmí připojit žádné jiné zařízení než ze sítě 192.168.2.1/24.

Tab. 3.2 Příklady možných požadavků na parametr jednosměrného omezení a jejich filtr pro NFDUMP

Požadavek na jednosměrné omezení	Filtr pro NFDUMP
192.168.1.1 $!\rightarrow\leftarrow$ 192.168.1.10 port = 22	((src ip 192.168.1.1) and not ((dst ip 192.168.1.10) or not (port 22))) or (not ((src ip 192.168.1.10) or not (port 22)) and (dst ip 192.168.1.1))
192.168.1.200 $!\rightarrow\leftarrow$ 192.168.2.1/24 port = 5555	((src ip 192.168.1.200) and not ((dst net 192.168.2.1/24) or not (port 5555))) or (not ((src net 192.168.2.1/24) or not (port 5555)) and (dst ip 192.168.1.200))

3.3 Obousměrné omezení $A!\rightarrow\leftarrow!B$

Obousměrné omezení umožňuje specifikovat dvě množiny zařízení, u kterých je povolena pouze vzájemná komunikace. Neboli zařízení definovaná množinou na levé straně mají povoleno komunikovat pouze se zařízeními z druhé množiny a zároveň zařízení z druhé množiny mají povoleno komunikovat pouze se zařízeními z množiny první. S použitím předchozího parametru by definice vypadala následovně: $A!\rightarrow\leftarrow B \wedge B!\rightarrow\leftarrow A$. Pro zkrácení je použit zápis: $A!\rightarrow\leftarrow!B$. Dále je umožněno omezení specifikovat pro určitý typ protokolu nebo číslo portu. Možnosti adresace zařízení jsou stejné jako u předchozích parametrů. Uplatnění lze nalézt v síti, kde existují dvě množiny zařízení nebo služeb výhradně komunikujících mezi sebou.

Validace probíhá podobným způsobem jako jednosměrné omezení s tím rozdílem, že se navíc přidá filtrovací pravidlo pro opačný směr komunikace. Formát filtru zobrazuje logický výraz:

((src A and not (dst B or not (proto PROTO and port PORT))) or směr toků $A \rightarrow B\ (!\rightarrow\leftarrow)$

(not (src B or not (proto PROTO and port PORT)) and dst A)) *or* směr toků $B \rightarrow A\ (!\rightarrow\leftarrow)$

((src B and not (dst A or not (proto PROTO and port PORT))) or směr toků $B \rightarrow A\ (\rightarrow\leftarrow!)$

(not (src A or not (proto PROTO and port PORT)) and dst B)) směr toků $A \rightarrow B\ (\rightarrow\leftarrow!)$

Tab. 3.3 ilustruje příklad požadavku na obousměrné omezení komunikace dvou zařízení s IP adresami 192.168.3.10 a 192.168.4.10 na portu 9999. Pro lepší představu je naznačeno rozdělení filtru dle směru omezení, které implementuje.

Tab. 3.3 Příklad požadavku na parametr obousměrného omezení a jeho filtr pro NFDUMP

<table>
<tr><th>Požadavek na obousměrné omezení</th><th colspan="2">Filtr pro NFDUMP</th></tr>
<tr><td rowspan="2">192.168.3.10 !→←! 192.168.4.10
Port = 9999</td><td>směr !→←</td><td rowspan="2">(((src ip 192.168.3.10) and not ((dst ip 192.168.4.10) or not (port 9999))) or
(not ((src ip 192.168.4.10) or not (port 9999)) and (dst ip 192.168.3.10)))
or
(((src ip 192.168.4.10) and not ((dst ip 192.168.3.10) or not (port 9999))) or
(not ((src ip 192.168.3.10) or not (port 9999)) and (dst ip 192.168.4.10)))</td></tr>
<tr><td>směr →←!</td></tr>
</table>

3.4 Jednosměrně inicializovaná komunikace $A \rightarrow B$

Jednosměrně inicializovaná komunikace ($A \rightarrow B$) nastavuje kontrolu, aby vytvořené spojení mezi zařízeními z množiny A a B bylo vždy inicializováno množinou A. Jinak řečeno žádné zařízení z množiny B nesmí započít komunikaci se zařízením s množinou A. Toto omezení lze dále specifikovat pro určitý typ protokolu nebo číslo portu. Možnosti adresace zařízení jsou stejné jako

u předchozích parametrů. Vhodné použití lze nalézt v lokální síti představující množinu zařízení A, kde jsou pouze koncové stanice bez serverů a tedy se neočekává přístup do sítě inicializovaný směrem z ostatních sítí představující množinu B. S výhodou je možné adresovat množinu B pomocí konstrukce 0.0.0.0/0, která představuje všechny sítě. Stojí za zmínku upozornit, že v takto specifikovaném případě není povolena žádna komunikace požadující odpověď od zařízení z množiny A i v rámci komunikace mezi uzly z množiny A.

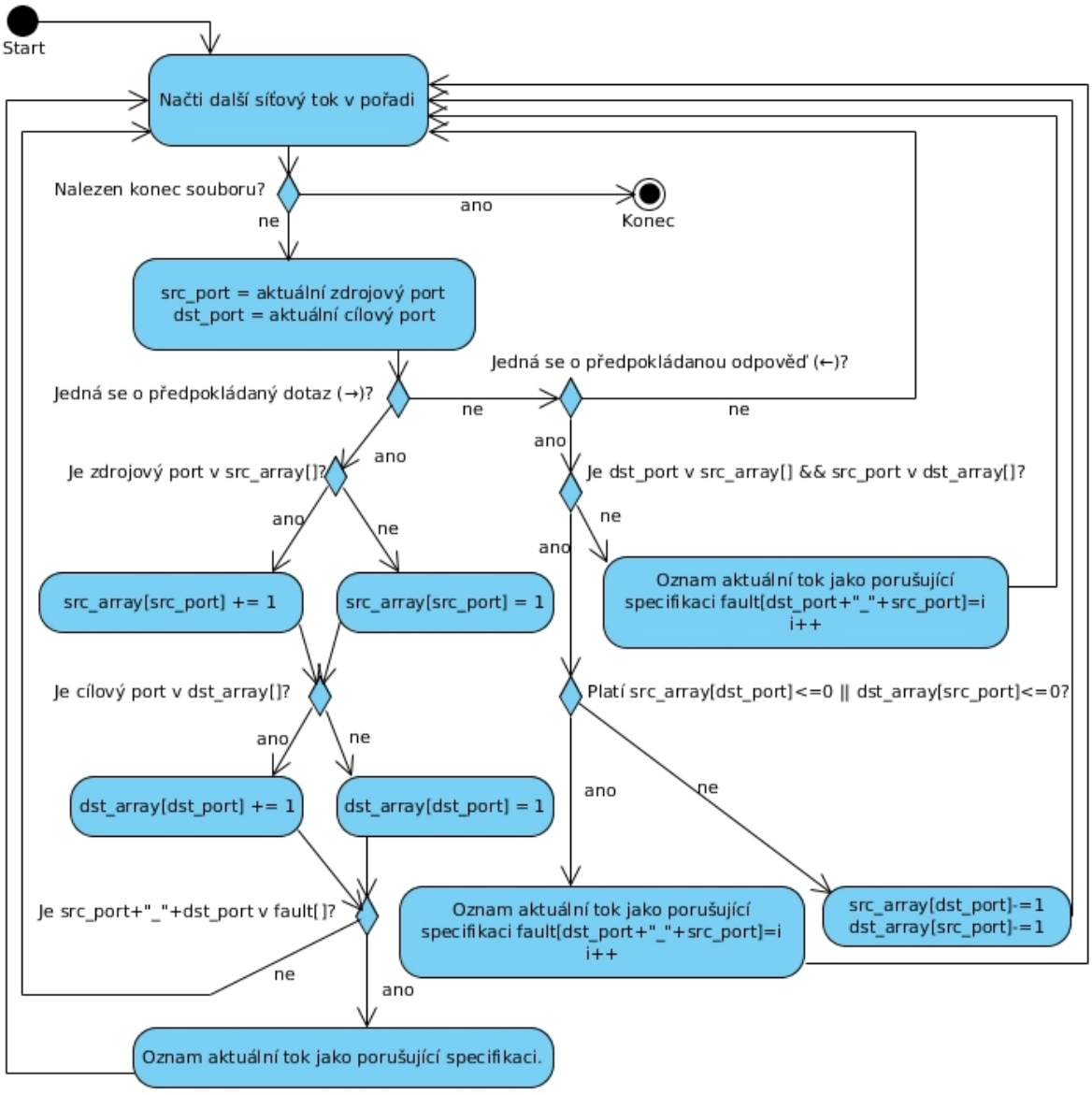
Validace tohoto parametru ($A \rightarrow B$) vychází z předpokladu, že veškeré toky směřující ze sítě A do sítě B představují dotaz a zároveň veškeré toky směřující ze sítě B do sítě A představují odpověď. Pokud se prokáže zmíněný předpoklad jako nepravdivý (což znamená, že některé toky směřující ze sítě B do sítě A představují dotaz), je nalezeno nežádoucí chování sítě. Konkrétně probíhá validace v několika krocích. Nejdříve jsou vytvořeny dva filtry pro NFDUMP. První filtr hledá toky, jejichž zdrojová adresa je z množiny A a cílová z množiny B. Druhý filtr hledá toky, jejichž zdrojová adresa je z množiny B a cílová z množiny A. Formáty obou filtrů zobrazují logické výrazy:

src A and dst B and proto PROTO and port PORT první filtr (předpokládaný dotaz)

src B and dst A and proto PROTO and port PORT druhý filtr (předpokládaná odpověď)

Poté je zavolán NFDUMP s prvním filtrem, přičemž při výstupu je k informacím o toku přidána formátovací značka „→“. A dále je zavolán NFDUMP s druhým filtrem s výstupem s formátovací značkou „←“. Tím je docíleno předpokládané oddělení toků představujících dotaz (→) od toků představujících odpověď (←). Tyto toky jsou seřazeny dle času vzniku a následuje analýza těchto dat za účelem ověření, zda dané toky opravdu patří do předpokládané skupiny. Zpracování probíhá po řádcích od začátku souboru, tzn., nejprve jsou zpracovány toky s nejstarší značkou času vzniku. Algoritmus zpracování předpokládaných toků dotazů a odpovědí je ve formě diagramu aktivity znázorněn na Obr. 3.1, přičemž zdrojový kód algoritmu je zobrazen v příloze číslo 1. Výstupem algoritmu je seznam dvojic toků (dotaz/odpověď) porušující daný požadavek. Z těch jsou odstraněny ty dvojice, které mají stejný čas vzniku (pokud existují) a poté jsou závěrečné výsledky uloženy do výsledné zprávy.

Validace tohoto typu parametru muže být v některých situacích problémová. Přesnost časové značky vzniku toku je v milisekundách, což muže být omezení v případě, kdy odchozí i příchozí tok má stejnou hodnotu času vzniku. Nelze pak přesně určit, který z toků nastal dříve, a takové toky jsou ignorovány. Těchto případů není mnoho, nicméně výhledově při dalším zrychlování odezvy komunikace by bylo řešením místo NetFlow použít IPFIX [25], který při použití specializovaných exportérů podporuje přesnost časových značek na nanosekundy. Dále je třeba se vypořádat se situací, kdy odpověď jde jinou cestou a je snímána jiným exportérem než dotaz. Pokud nejde této situaci zabránit vhodným umístěním exportérů, je třeba mít dobře vyřešenou časovou synchronizaci exportujících sond. Tab. 3.4 obsahuje ukázku požadavku specifikující, aby se nikdo nepokoušel dotazovat při http komunikaci (port 80) na zařízení dané IP adresou 147.229.206.191.



Obr. 3.1 Diagram aktivity validace parametru jednosměrně inicializované komunikace

Tab. 3.4 Příklad požadavku na parametr jednosměrně inicializované komunikace a jeho filtry pro NFDUMP

Definice požadavku	147.229.206.191 → 0.0.0.0/0 port = 80
Filtr pro předpokládané dotazy (→)	(src ip 147.229.206.191) and (dst net 0.0.0.0/0) and port 80
Filtr pro předpokládané odpovědi (←)	(src net 0.0.0.0/0) and (dst ip 147.229.206.191) and port 80

3.5 ACL

Možnosti řešení validace parametru sítě popsaném ve formě ACL (access control list) a problémy z toho plynoucí byly představeny v kapitole 2.2.2. Pro implementaci byl zvolen přístup využívající ACL jako referenční metodu popisu filtrování paketů procházející daným rozhraním v daném směru. Nicméně v tomto případě bude (na rozdíl od standardního použití) ACL popisovat požadované chování filtrování paketů v celé síti a ve všech směrech. To znamená, požadavek na parametr typu ACL bude představovat konkrétní ACL specifikující filtrování paketů v celé zkoumané síti.

Validující proces využívá skutečnost, že filtrovací pravidla obsažená v ACL lze reprezentovat jako stromy vhodně zanořených logických formulí, což lze dále využít při převodu ACL do formule predikátové logiky. Následně se vzniklý predikát neguje a výsledek se uplatní jako filtr pro NFDUMP, který po zavolání vrátí seznam síťových toků porušující definovaný požadavek na ACL. Příklad ACL, jeho převod do predikátové logiky a vytvoření filtru pro NFDUMP ukazuje Tab. 2.1.

Vhodné použití tohoto typu parametru se nabízí při specifikaci složitějších požadavků. Například chceme ověřit, že webový server (147.229.1.10) muže komunikovat na portu 80 pouze s vnitřní sítí A (147.229.1.0/24) a vnější sítí B (80.75.2.0/24). V takovém případě nelze použít parametr jednosměrného omezení ($!\rightarrow\leftarrow$), neboť ten podporuje pouze adresaci konkrétních ip adres nebo jednoho rozsahu sítě (v tomto případě by bylo potřeba adresovat dvě sítě na jedné straně parametru). Zároveň je třeba počítat s tím, že požadavek na parametr typu ACL se týká všech nasbíraných síťových toků. Vhodné je proto pomocí ACL nejprve zakázat nežádoucí komunikaci a ostatní komunikaci v méně prioritních pravidlech povolit. Zmíněný příklad požadavku na webový server a jeho filtr pro NFDUM je ukázán v Tab. 3.5.

Tab. 3.5 Příklady požadavku na parametr typu ACL a jeho filtr pro NFDUMP

Požadavek na parametr typu ACL	Filtr pro NFDUMP
permit 147.229.1.10 80 147.229.1.0/24 permit 147.229.1.10 80 80.75.2.0/24 deny 147.229.1.10 80 0.0.0.0/0 permit 0.0.0.0/0 0.0.0.0/0 → povolí zbytek	not ((src net 147.229.1.10/32 and dst net 147.229.1.0/24 and (src port = 80)) or ((src net 147.229.1.10/32 and dst net 80.75.2.0/24 and (src port = 80)) or (not (src net 147.229.1.10/32 and dst net 0.0.0.0/0) and ((src net 0.0.0.0/0 and dst net 0.0.0.0/0)))))

3.6 Shrnutí

Klíčem k výběru pravě těchto parametrů k implementaci byla snaha o pokrytí typického požadovaného chování sítě. Protože jednodušší parametry ($\rightarrow|\leftarrow$, $!\rightarrow\leftarrow$, $!\rightarrow\leftarrow!$) se ukázaly pro ověření některých případů požadovaného chování sítě nedostatečné, byly později implementovány parametry složitější a obecnější ($\rightarrow$, ACL). Doporučený postup při definování požadavku na síť je

nejdříve využít možností parametrů definovaných v kapitolách 3.1 ($\rightarrow|\leftarrow$), 3.2 ($!\rightarrow\leftarrow$), 3.3 ($!\rightarrow\leftarrow!$), 3.4 ($\rightarrow$). Až poté pro zbylé požadované chování využít parametr ACL z kapitoly 3.5, který má větší a obecnější popisující sílu, ale zároveň je složitější.

Z analýzy problematiky lze usoudit, že tyto parametry spadají do kategorie omezujících, přičemž ostatní kategorie parametrů jsou nechány pro navazující práce. Možný další směr vývoje je naznačen v kapitole 2.2 zabývající se analýzou problematiky, zejména lze doporučit v budoucnu validovat parametry popisující dosažitelnost uzlů v síti.

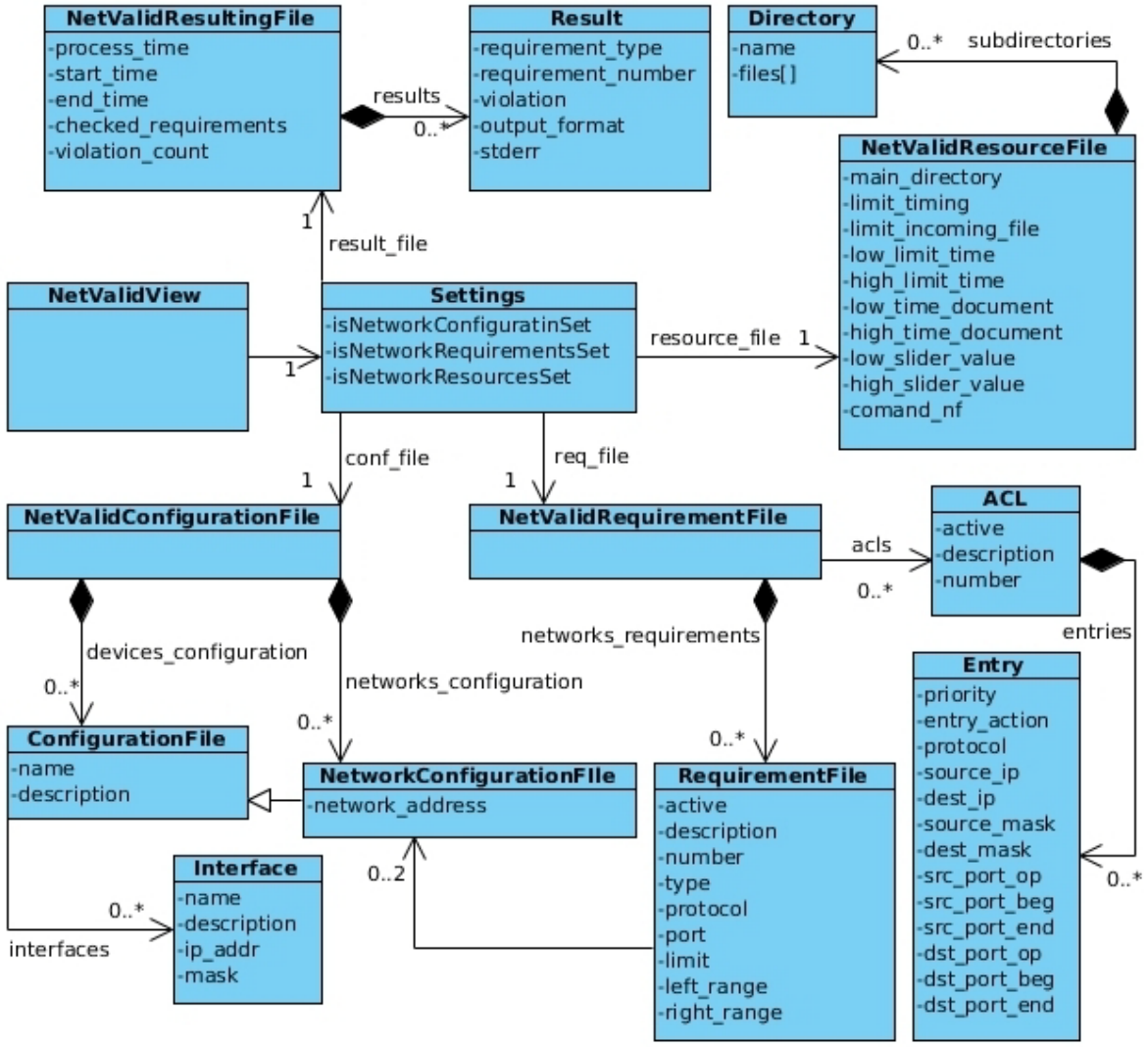
4 Implementace nástroje

Kapitola ukazuje implementační detaily praktické části této diplomové práce. Nejprve je předložen základní popis aplikace včetně zobrazení a vysvětlení objektové struktury ve formě diagramu tříd. Následně jsou v samostatných podkapitolách popsány jednotlivé části aplikace týkající se konfigurace sítových zařízení, definování požadavků k validaci, nastavení vstupních sítových toků, možnostmi zobrazení výsledků a importem/exportem nastavení.

4.1 Základní popis aplikace

Aplikace má grafické uživatelské rozhrání. Je programována v jazyku JAVA s využitím vývojového prostředí NetBeans. Základní aplikační třídou je `NetValidApp`. Ta dědí z předpřipravené třídy `SingleFrameApplication`, která poskytuje metody pro práci s jednoduchou grafickou aplikací založenou na Swingu (aplikační framework součásti jazyku JAVA) s jedním primárním JFrame. Konkrétně zabezpečuje správnou počáteční inicializaci, ukončení aplikace, uložení a obnovu stavu aplikace. Pomocí metody `show` třída `NetValidApp` inicializuje a zobrazí objekt třídy `NetValidView`, ta dědí z předpřipravené třídy `FrameView`, která se používá pro sestrojení hlavního okna aplikace a již v sobě obsahuje typicky používané komponenty, jako menu bar, tool bar, component, a status bar. Tím tedy dojde k zobrazení hlavního okna aplikace, jejíž běh lze dále rozdělit do několika samostatných částí, kterým budou věnovány následující podkapitoly.

Každé okno aplikace implementuje samostatná třída zajišťující zejména sestavení jednotlivých komponent okna a obsluhu událostí. Zároveň je pro každou třídu reprezentující okno aplikace vytvořena třída implementující datový model, čímž je minimalizována závislost zpracování dat na konkrétním okně a tím lze bez větších komplikací dělat změny v grafickém uživatelském rozhraní. Obr. 4.1 ukazuje objektový model aplikace ve formě diagramu tříd. Vzhledem k prostorové náročnosti diagramu jsou v něm znázorněny zejména třídy pro zpracování dat a jejich atributy bez použitých metod. Jak je na obrázku vidět, třída `NetValidView` představující hlavní okno aplikace obsahuje referenci na objekt třídy `Settings`, který zapouzdřuje veškerá potřebná data pro běh aplikace. Konkrétně obsahuje reference na objekty tříd `NetValidConfigurationFile` (nastavení síťových zařízení), `NetValidRequirementFile` (definice požadavku na síťové parametry k validaci), `NetValidResourceFile` (nastavení zdrojového adresáře síťových toků a časového intervalu, ze kterého se vyberou příslušné toky) a `NetValidResultingFile` (data potřebná pro zobrazení výsledné zprávy). Podrobnější pohled na objektovou strukturu bude částečně ukázán v následujících kapitolách, nebo lze případně zhlédnout přímo zdrojové kódy aplikace, které jsou součásti diplomové práce v přiloženém cd.



Obr. 4.1 Základní objektový model aplikace

4.2 Nastavení síťových zařízení

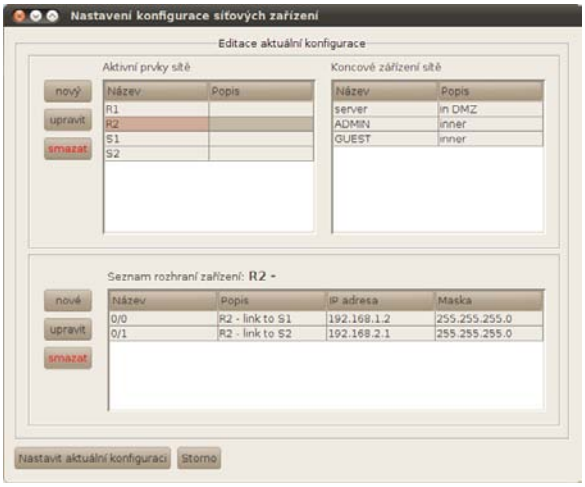
Prvotní představa o účelu znalosti konfigurace síťových zařízení byla taková, že uživatel nejdříve nastaví všechna existující zařízení v síti a poté mu bude nabídnut jednoduchý způsob, jak na tato zařízení vznést požadavky k validaci. Tento koncept se časem ukázal jako příliš svazující a proto byl mírně pozměněn. V současném stavu je nastavení konfigurace síťových zařízení volitelná, ale doporučená, neboť později ulehčí adresaci prvků a sítí při definování požadavků pro validaci.

Konkrétní nastavení síťového zařízení představuje označení, zda se jedná o aktivní prvek sítě (směrovač, přepínač) nebo koncové zařízení (notebook, stolní PC, server), určí se název zařízení, jeho popis a nakonec se nastaví jednotlivá rozhraní daného zařízení. Rozhraní, kterých může mít každé zařízení více, obsahuje název rozhraní, popis rozhraní, ip adresu a síťovou masku. Tohoto nastavení je využito při definování požadavků i samotném validačním procesu. Rozdělení síťových zařízení na aktivní prvky a koncové zařízení je v tuto chvíli nevyužité, nicméně se předpokládá možné použití při

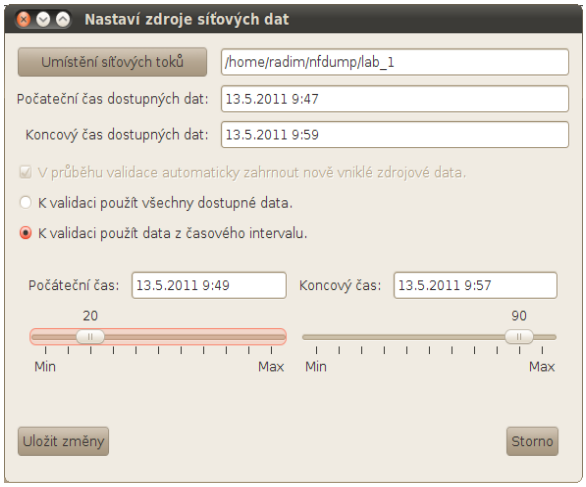
validaci specifičtějších požadavků, jako například: „všechny aktivní prvky lze konfigurovat pouze skrze protokol ssh“, kde adresace aktivních prvků by se provedla pouze zatržením příslušného přepínače.

Objektový model pro uložení a zpracování konfigurace síťových zařízení představuje třída `NetValidConfigurationFile`. Ta v sobě definuje třídy `ConfigurationFile` (představující konfiguraci jednoho síťového zařízení) a `NetworkConfiguratinFile` (obsahující konfiguraci sítě vzniklé zadáním alespoň jednoho síťového zařízení). Po nastavení síťových zařízení dojde automaticky k vytvoření síťových konfigurací. Ty vzniknou postupným prohledáváním síťových zařízení, kde se analyzuje, do jaké adresy sítě dané zařízení patří. Pokud tato síť nebyla ještě vytvořena, dojde k tomu nyní a dané zařízení se do něj vloží. Smyslem znalosti síťových konfigurací je nabídnout (při definici požadavků k validaci) uživateli možnost adresace dle síťové adresy s výběrem zařízení, které daná síť obsahuje.

I bez nastavení zmíněných konfiguračních údajů lze definovat požadavky na parametry k validaci, nicméně v tom případě nelze později využít adresace pomocí automaticky vytvořených sítí obsahujících jednotlivé ip adresy zadaných zařízení. V takovém případě musí uživatel provést adresaci manuálně pomocí zadání rozsahu, do kterého požadované zařízení patří. Obr. 4.2 ukazuje okno aplikace pro nastavení konfigurace síťových zařízení.



Obr. 4.2 Okno aplikace pro konfiguraci síťových zařízení



Obr. 4.3 Okno aplikace pro nastavení zdroje síťových toků

4.3 Definování požadavků k validaci

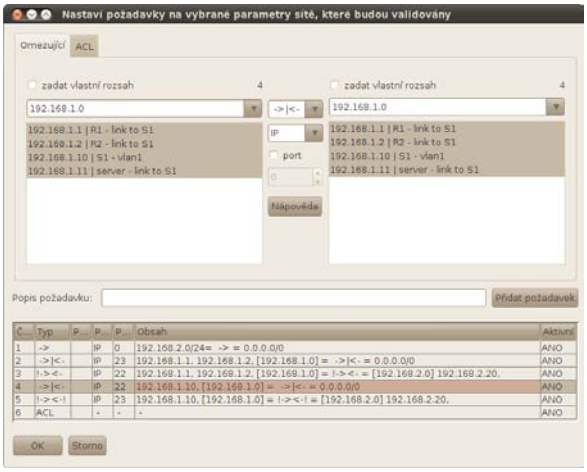
Po kliknutí v hlavním okně na tlačítko „Požadavky k validaci“ umožní běh aplikace definovat požadavky na síťové parametry. Okno (viz Obr. 4.4) je rozděleno na horní a dolní část. Horní část je strukturovaná do dvou záložek (s popisem Omezující a ACL) a slouží pro samostatnou definici požadavků. Dolní část představuje tabulku s již vytvořenými požadavky, kde se při dvojkliku na řádek otevře další okno zobrazující detailní definici požadavku, který je spojen s příslušným

řádkem tabulky. V tomto okně lze vytvořený požadavek zakázat, povolit, odstranit a v případě požadavku na parametr typu ACL je navíc možnost měnit prioritu položek ACL, vytvářet položky a mazat položky.

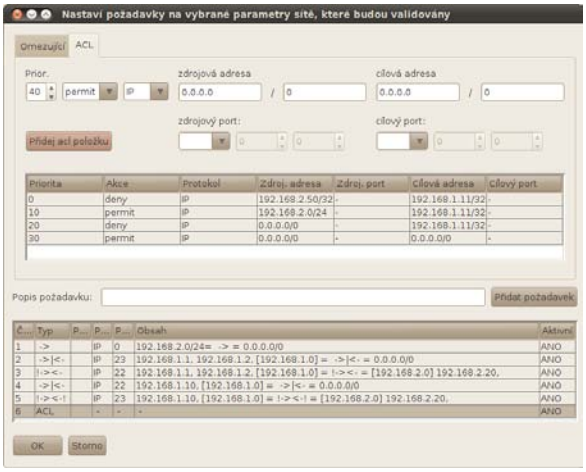
Záložka s popisem „Omezující“ představuje prostředí pro definici parametrů typu: blokování komunikace ($\rightarrow|\leftarrow$), jednosměrné omezení ($!\rightarrow\leftarrow$), obousměrné omezení ($!\rightarrow\leftarrow!$), jednosměrně inicializovaná komunikace ($\rightarrow$). Konkrétní význam a popis validace těchto parametrů byl popsán v kapitole 3. Po výběru typu parametru je třeba zadat adresaci zařízení vyskytujících se na levé a pravé straně označení parametru. To je možné provést výběrem konkrétní sítě z padací lišty a následným označením zařízení, které do této sítě patří nebo ručním zadáním rozsahu sítě. První možnost je poskytnuta pouze v případě, že byly nastaveny konfigurační údaje týkajících se síťových zařízení. Druhá možnost se povolí pří aktivaci volby „zadat vlastní rozsah“. Dále je možno nastavit pro daný požadavek protokol (IP, TCP, UDP) a číslo portu.

Druhá záložka s popisem „ACL“ umožňuje zadat požadavek na parametr typu ACL ve významu popsaném v kapitole 3.5. Pro každou položku ACL se nastaví priorita, akce, typ protokolu, zdrojová adresa, cílová adresa a rozsah zdrojových a cílových portů. Následně se vytvářená položka kliknutím na tlačítko „Přidej acl položku“ přidá do seznamu a nakonec po definování všech potřebných položek stačí kliknout na tlačítko „Přidat požadavek“, které daný požadavek zobrazí v tabulce umístěné v dolní části okna a uloží do datového modelu.

Strukturu pro uložení všech zadaných požadavků implementuje třída `NetValidRequirementFile`, která obsahuje pole objektů třídy `RequirementFile` sloužící pro uložení všech požadavků definovaných v první záložce okna a dále obsahuje pole objektů třídy `NetValidACL` představující datovou strukturu pro uložení požadavků na parametr typu ACL.



Obr. 4.4 a Okno pro definování omezujících požadavků



Obr. 4.4b Okno pro definování požadavků typu ACL

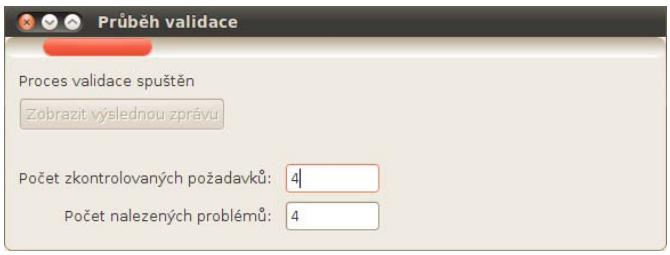
4.4 Načtení síťových toků

Pro nastavení zdroje síťových toků slouží okno zobrazené po kliknutí na tlačítko „Zdroj síťového provozu“ v úvodním okně aplikace. Základní informace o požadavcích na umístění a typ síťových toků jsou rozebrány v kapitole 2.3.2 (jen pro připomenutí, aplikace očekává síťové toky ve formátu vytvořeném nástrojem nfcapd).

Po kliknutí na tlačítko „Umístění síťových toků“ v okně pro nastavení zdroje síťových toků (viz Obr. 4.3) se objeví dialogové okno pro výběr kořenového adresáře obsahujícího uložené síťové toky. Po načtení dojde k zobrazení počátečního a koncového času dostupných toků, přičemž uživatel má možnost zúžit tento časový interval dle své potřeby. Dále je možné při zatržení příslušného přepínače zahrnout do procesu validace nově vzniklá data ve zvoleném adresáři. Třída implementující nastavení zdrojů síťových toků se nazývá `NetValidResourceFile`. Ta obsahuje kromě atributů základních typů také pole objektů třídy `Directory`, které uchovává pro každý podadresář zadaného kořenového adresáře seznam souborů obsahující síťové toky.

4.5 Průběh validace

Po kliknutí v úvodním okně na tlačítko „Spustit“, dojde k vytvoření nového vlákna aplikace a v něm k zahájení validačního procesu (nové vlákno je vytvořeno, aby validační proces neblokoval vlastní běh aplikace). V cyklu se postupně zpracovávají a tím validují jednotlivé požadavky. Platí, že každý typ parametru má své vlastní zpracování, které již bylo popsáno v kapitole 3. Po spuštění validace se zobrazí nové okno typu „progress bar“ informující uživatele o aktuálním počtu zkontrolovaných požadavků a počtu nesplněných požadavků (ty, u nichž bylo při validaci nalezeno nežádoucí chování sítě). Zmíněné okno je vidět na Obr. 4.5. Po zpracování každého požadavku je výsledek uložen do objektu třídy `NetValidResultingFile`. Po ukončení procesu validace lze kliknout na nabízené tlačítko „Zobrazit výslednou zprávu“, které zobrazí závěrečnou zprávu validace. Třída implementující samostatný proces validace se nazývá `ThreadStartValidateProcess`, jako hlavní atributy obsahuje referenci na objekt třídy `NetValidSettings` a objekt třídy `NetValidProgressBarForm`, představující popisovaný progress bar.



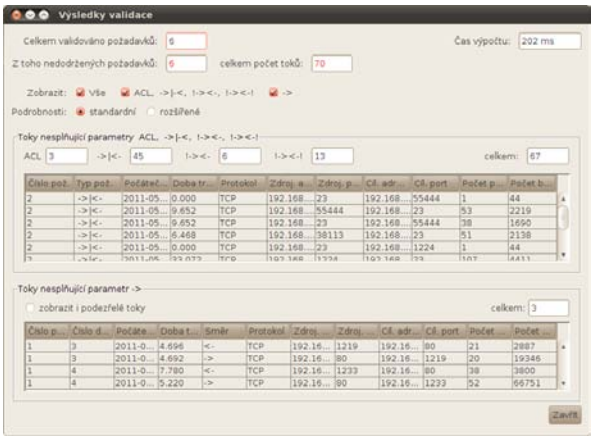
Obr. 4.5 Okno progress baru zobrazující průběh procesu validace

4.6 Zobrazení výsledků validace

Zobrazit výsledky validace lze po ukončení procesu validace kliknutím na příslušné tlačítko přímo v progress baru nebo z hlavního okna aplikace při kliknutí na tlačítko „Zobrazit výsledky“. Následně se objeví okno představující výslednou zprávu o průběhu validace (viz Obr. 4.6). V horní části okna jsou informace o času validace, počtu validovaných požadavků, počtu nesplněných požadavků a pro každý typ parametru je zobrazen počet toků, které jej v některém z definovaných požadavků nesplňují. Okno dále zahrnuje dvě tabulky, první z nich obsahuje v jednotlivých řádcích toky nesplňující požadavky na parametry typu $\rightarrow|\leftarrow$, $!\rightarrow\leftarrow$, $!\rightarrow\leftarrow!$, ACL a druhá tabulka zobrazuje problémové toky nevyhovující požadavkům na parametr typu $\rightarrow$. U této tabulky je možnost si nechat zobrazit i toky, které jeví známky podezřelého chování, ale za některých okolností mohou být validní, například při použití NAT nebo při vzniku nepřesnosti v hodnotě počátečního času toku.

Výsledek validace je ve formě zobrazení toků, které nesplňují některý z definovaných požadavků. Pokud je výsledek příznivý a všechny požadavky jsou splněny, nebude výsledná zpráva obsahovat žádné informace. Pro přehlednost má uživatel možnost volby mezi standardním a rozšířeným zobrazením detailů výsledků. Taktéž je možno skrýt vybranou tabulku.

Po dvojkliku na vybraný řádek první tabulky (reprezentující jeden síťový tok), dojde k otevření nového okna zobrazujícího detailní informace o daném síťovém toku. Mezi tyto informace patří počáteční čas, doba trvání, zdrojová a cílová adresa, zdrojový a cílový port, počet paketů, počet bajtů a další. Dále je možné v tomto okně přejit k zobrazení definice požadavku, který tento vybraný tok nesplňuje. Při dvojkliku na řádek druhé tabulky je otevřeno okno zobrazující detail toku reprezentující dotaz a navíc detail toku reprezentující odpověď. Takové zobrazení je vybráno, aby bylo jasně vidět, proč byl daný tok shledán jako nesplňující požadavek na parametr typu jednosměrně inicializované komunikace. Dále je opět možnost při kliku na tlačítko „Zobrazit definici požadavku“ nechat si zobrazit požadavek, který daný tok nesplňuje.



Obr. 4.6 Okno zobrazující výsledek validačního procesu

Obr. 4.7 Okno pro import/export nastavení aplikace

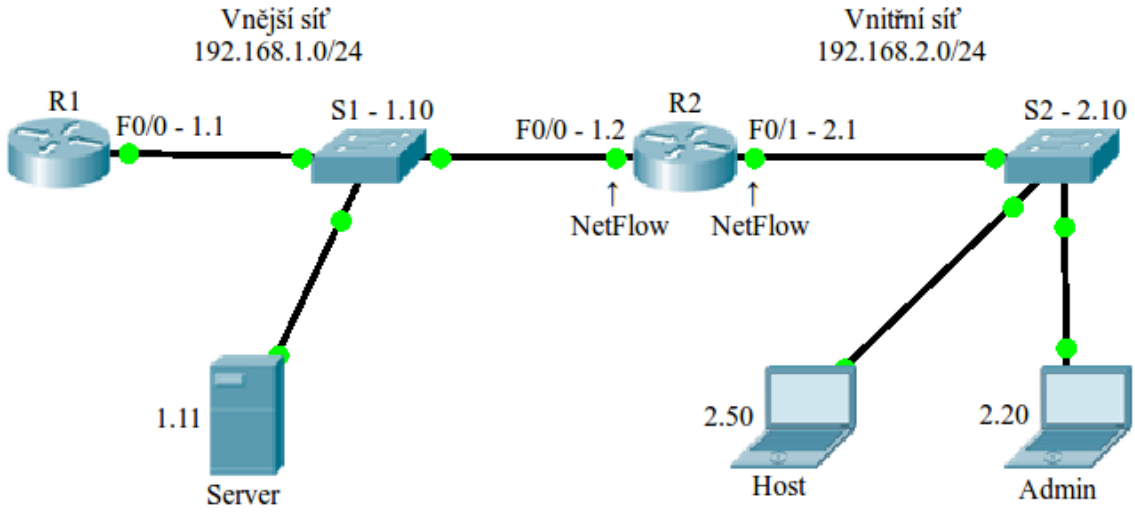
4.7 Import/export nastavení

V Menu hlavního okna se nachází položka Import/Export nastavení, která po kliknutí způsobí zobrazení okna pro načtení a uložení nastavení aplikace (viz Obr. 4.7). Exportované nastavení je ve formě xml souboru. Do exportovaných dat je možné zahrnout nastavení síťových zařízení, požadavky na síť a nastavení zdroje síťových toků. Při importu dojde k načtení vybraného souboru a uživateli se zobrazí, které konkrétní části z možného nastavení byly naimportovány.

5 Testování aplikace

K testování aplikace byla vytvořená ve školní laboratoři experimentální síť s požadavky zaměřenými na otestování validace všech typů implementovaných síťových parametrů. Topologie sítě je zobrazena na Obr. 5.1. Jsou vytvořeny dvě podsítě, vnější síť s adresou 192.168.1.0/24 představuje DMZ a vnitřní síť s adresou 192.168.2.0/24 představuje sít pro připojení klientských stanic.

Následuje popis požadované specifikace sítě, ze které vyplynou konkrétní požadavky na jednotlivé parametry sítě. Tyto požadavky budou v praxi pomocí vytvořeného nástroje validovány. Pro kontrolu zda validace proběhla správně a odhalila veškeré nežádoucí chování, bude také popsána uskutečněná komunikace v síti. Ke konci kapitoly bude popsán zátěžový test aplikace, jehož hlavním účelem je změřit časovou náročnost aplikace v závislosti na rozsahu zadaných požadavků a počtu zdrojových toků.



Obr. 5.1 Topologie experimentální sítě

5.1 Specifikace požadavků sítě

Pro vytvořenou síť platí následující specifikace: Žádné zařízení nemá přístup do vnitřní sítě identifikované adresou 192.168.2.0/24. Pro konfiguraci směrovačů R1, R2 platí zákaz použití protokolu telnet, přičemž tuto konfiguraci lze provádět pomocí protokolu ssh pouze ze stanice reprezentující administrátora, tj. z IP adresy 192.168.2.20. Pro konfiguraci přepínače S1 platí naopak zákaz použití protokolu ssh a je nutné tuto konfiguraci provést pomocí protokolu telnet opět pouze ze stanice administrátora. Zároveň platí, že administrátor smí použít telnet pouze v případě konfigurace přepínače S1. Přístup na server v DMZ mají všechna zařízení z vnitřní sítě 192.168.2.0/24 kromě uzlu Host identifikovaném IP adresou 192.168.2.50. Všechny ostatní uzly mají na server v DMZ přístup zakázán.

Zmíněnou specifikaci určenou pro experimentální síť lze přepsat pomocí definování požadavků na parametry sítě. Tab. 5.1 ukazuje, které části specifikace sítě jsou obsaženy konkrétními definicemi požadavků na parametry sítě. Následující podkapitoly budou s ohledem na reálně uskutečněnou komunikaci popisovat pro jednotlivé typy požadavků výsledky validace.

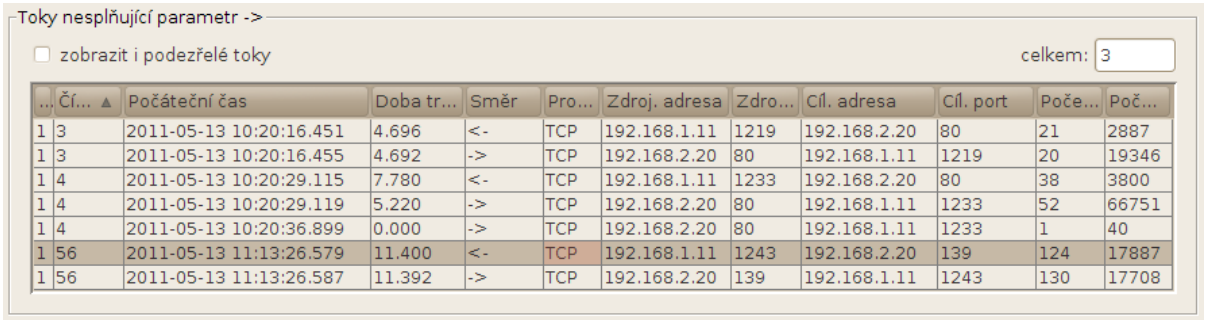
Tab. 5.1 Definice požadavků a jejich a jejich význam v odpovídající specifikaci.

Číslo	**Definice požadavku**	**Odpovídá specifikaci**
1	192.168.2.0/24 → 0.0.0.0/24	Žádné zařízení nemá přístup do vnitřní sítě identifikované adresou 192.168.2.0/24.
2	192.168.1.1, 192.168.1.2 →|← 0.0.0.0/0, port = 23	Pro konfiguraci směrovačů R1, R2 platí zákaz použití protokolu telnet…
3	192.168.1.1, 192.168.1.2 !→← 192.168.2.20, port = 22	…přičemž tuto konfiguraci lze provádět pomocí protokolu ssh pouze ze stanice reprezentující administrátora, tj. z IP adresy 192.168.2.20.
4	192.168.1.10 →|← 0.0.0.0/0, port = 22	Pro konfiguraci přepínače S1 platí naopak zákaz použití protokolu ssh…
5	192.168.1.10 !→←! 192.168.2.20, port = 23	…a je nutné tuto konfiguraci provést pomocí protokolu telnet opět pouze ze stanice administrátora. Zároveň platí, že administrátor smí použít telnet pouze v případě konfigurace přepínače S1.
6	0 deny IP 192.168.2.50 192.168.1.11 10 permit IP 192.168.2.0/24 192.168.1.11 20 deny IP 0.0.0.0./0 192.168.1.11 30 permit IP 0.0.0.0/0 0.0.0.0/0	Přístup na server v DMZ mají všechna zařízení z vnitřní sítě 192.168.2.0/24 kromě uzlu Host identifikovaném IP adresou 192.168.2.50. Všechny ostatní uzly mají na server v DMZ přístup zakázán.

5.2 Validace požadavku číslo 1

V pořadí první definovaný požadavek kontroluje, aby vytvořené spojení mezi zařízeními s IP adresou z rozsahu vnitřní sítě (192.168.2.0/24) a jakýmkoliv jiným uzlem sítě bylo vždy inicializováno z vnitřní sítě. V praxi byla uskutečněna kromě komunikace směrem ven z vnitřní sítě i nežádoucí komunikace směrem dovnitř, konkrétně se jednalo o přístup z PC 192.168.1.11 na web server (běžící na 192.168.2.20 – PC Admin). A dále byla na PC – Admin nasdílena data, na která bylo přistupováno z PC s IP 192.168.1.11 (vnější síť). Validace tohoto požadavku nalezla oba typy nežádoucích toků

(viz Obr. 5.2), kde toky s portem 80 označují zakázaný přístup na webový server a toky, u nichž se vyskytuje port číslo 139, představují nežádoucí přístup na nasdílená data.


Toky nesplňující parametr ->

zobrazit i podezřelé toky

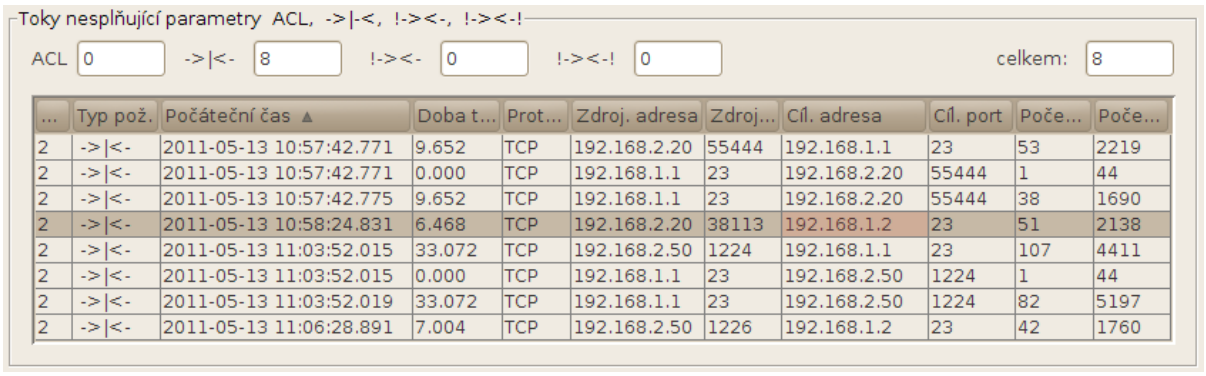
celkem: 3

...	Čí... ▲	Počáteční čas	Doba tr...	Směr	Pro...	Zdroj. adresa	Zdro...	Cíl. adresa	Cíl. port	Poče...	Poč...
1	3	2011-05-13 10:20:16.451	4.696	<-	TCP	192.168.1.11	1219	192.168.2.20	80	21	2887
1	3	2011-05-13 10:20:16.455	4.692	->	TCP	192.168.2.20	80	192.168.1.11	1219	20	19346
1	4	2011-05-13 10:20:29.115	7.780	<-	TCP	192.168.1.11	1233	192.168.2.20	80	38	3800
1	4	2011-05-13 10:20:29.119	5.220	->	TCP	192.168.2.20	80	192.168.1.11	1233	52	66751
1	4	2011-05-13 10:20:36.899	0.000	->	TCP	192.168.2.20	80	192.168.1.11	1233	1	40
1	56	2011-05-13 11:13:26.579	11.400	<-	TCP	192.168.1.11	1243	192.168.2.20	139	124	17887
1	56	2011-05-13 11:13:26.587	11.392	->	TCP	192.168.2.20	139	192.168.1.11	1243	130	17708

Obr. 5.2 Výsledek validace požadavku číslo 1 (192.168.2.0/24 → 0.0.0.0/24).

5.3 Validace požadavku číslo 2

Druhý požadavek zakazuje použití protokolu telnet při komunikaci se směrovači R1 a R2. Při testování bylo na směrovače R1 a R2 přistupováno z uzlu Admin (192.168.2.20) a uzlu Host (192.168.2.50) při použití protokolů ssh (port 22) i telnet (port 23), čímž byl částečně zadaný požadavek porušen. Po validačním procesu byly prohlášeny za nežádoucí ty toky, kdy Admin a Host přistupoval na směrovač R1 a R2 při použití telnetu, tedy byla odhalena veškerá komunikace porušující definovaný požadavek 192.168.1.1, 192.168.1.2 →|← 0.0.0.0/0, port = 23. Okno zobrazující výsledky validace tohoto požadavku je níže na Obr. 5.3.


Toky nesplňující parametry ACL, ->|-<, !-><-, !-><-!

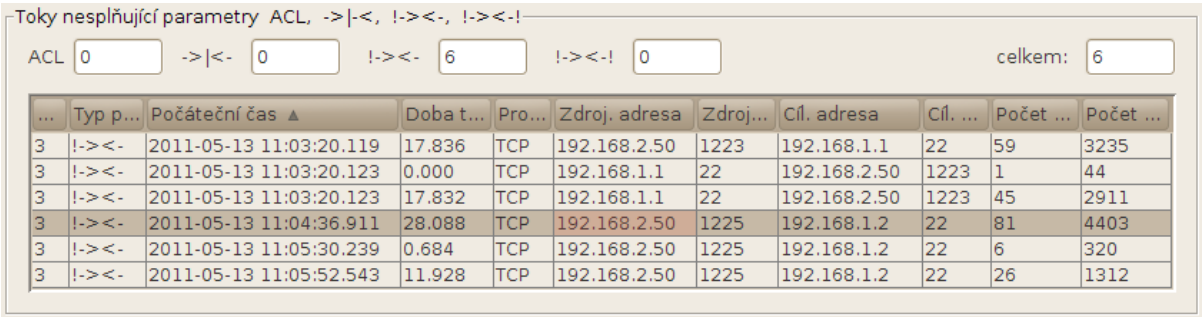
ACL 0 ->|<- 8 !-><- 0 !-><-! 0 celkem: 8

...	Typ pož.	Počáteční čas ▲	Doba t...	Prot...	Zdroj. adresa	Zdroj...	Cíl. adresa	Cíl. port	Poče...	Poče...
2	->|<-	2011-05-13 10:57:42.771	9.652	TCP	192.168.2.20	55444	192.168.1.1	23	53	2219
2	->|<-	2011-05-13 10:57:42.771	0.000	TCP	192.168.1.1	23	192.168.2.20	55444	1	44
2	->|<-	2011-05-13 10:57:42.775	9.652	TCP	192.168.1.1	23	192.168.2.20	55444	38	1690
2	->|<-	2011-05-13 10:58:24.831	6.468	TCP	192.168.2.20	38113	192.168.1.2	23	51	2138
2	->|<-	2011-05-13 11:03:52.015	33.072	TCP	192.168.2.50	1224	192.168.1.1	23	107	4411
2	->|<-	2011-05-13 11:03:52.015	0.000	TCP	192.168.1.1	23	192.168.2.50	1224	1	44
2	->|<-	2011-05-13 11:03:52.019	33.072	TCP	192.168.1.1	23	192.168.2.50	1224	82	5197
2	->|<-	2011-05-13 11:06:28.891	7.004	TCP	192.168.2.50	1226	192.168.1.2	23	42	1760

Obr. 5.3 Výsledky validace požadavku číslo 2 (192.168.1.1, 192.168.1.2 →|← 0.0.0.0/0, port = 23)

5.4 Validace požadavku číslo 3

Třetí požadavek povoluje možnost konfigurace směrovačů R1 a R2 při použití protokolu ssh pouze ze stanice PC Admin. Jak již bylo napsáno u předchozího požadavku, na směrovače R1 a R2 přistupoval při použití protokolu ssh i uzel Host. A přesně tyto toky, reprezentující komunikaci mezi PC Host (192.168.2.50) a směrovači R1 a R2 při použití protokolu ssh byly procesem validace shledány jako nežádoucí (viz Obr. 5.4).



Toky nesplňující parametry ACL, ->|-<, !->-<, !->-<!

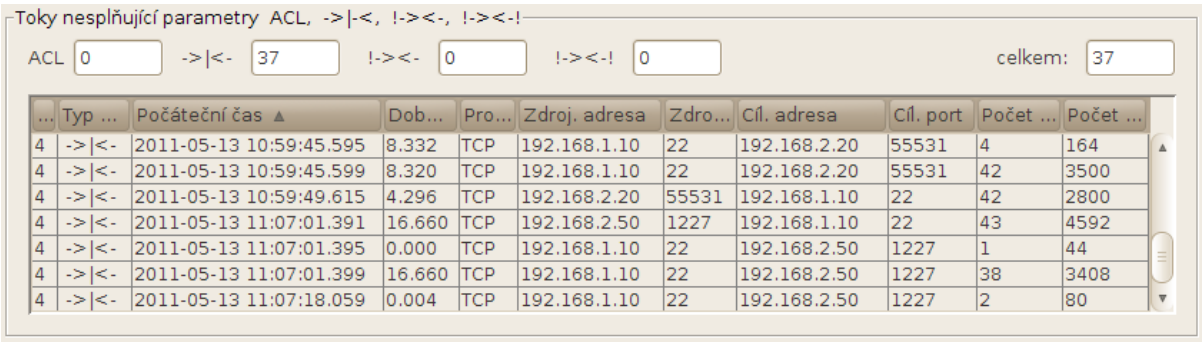
ACL 0 ->|<- 0 !->- 6 !->-<! 0 celkem: 6

...	Typ p...	Počáteční čas ▲	Doba t...	Pro...	Zdroj. adresa	Zdroj...	Cíl. adresa	Cíl. ...	Počet ...	Počet ...
3	!->-	2011-05-13 11:03:20.119	17.836	TCP	192.168.2.50	1223	192.168.1.1	22	59	3235
3	!->-	2011-05-13 11:03:20.123	0.000	TCP	192.168.1.1	22	192.168.2.50	1223	1	44
3	!->-	2011-05-13 11:03:20.123	17.832	TCP	192.168.1.1	22	192.168.2.50	1223	45	2911
3	!->-	2011-05-13 11:04:36.911	28.088	TCP	192.168.2.50	1225	192.168.1.2	22	81	4403
3	!->-	2011-05-13 11:05:30.239	0.684	TCP	192.168.2.50	1225	192.168.1.2	22	6	320
3	!->-	2011-05-13 11:05:52.543	11.928	TCP	192.168.2.50	1225	192.168.1.2	22	26	1312

Obr. 5.4 Výsledky validace požadavku číslo 3 (192.168.1.1, 192.168.1.2 !→← 192.168.2.20, port = 22)

5.5 Validace požadavku číslo 4

Validace čtvrtého požadavku je podobná validaci požadavku druhého. V tomto případě je zakázáno konfigurovat přepínač S1 pomocí protokolu ssh. Nepřímo je tedy přikázáno pro vzdálenou konfiguraci použit telnet. V praxi byla vzdálená konfigurace přepínače provedena pomocí obou zmíněných protokolů. Jako nežádoucí toky byly správně odhaleny ty, které použily protokol ssh. Viz Obr. 5.5 ukazující výsledek validace.



Toky nesplňující parametry ACL, ->|-<, !->-<, !->-<!

ACL 0 ->|<- 37 !->- 0 !->-<! 0 celkem: 37

...	Typ ...	Počáteční čas ▲	Dob...	Pro...	Zdroj. adresa	Zdro...	Cíl. adresa	Cíl. port	Počet ...	Počet ...
4	->|<-	2011-05-13 10:59:45.595	8.332	TCP	192.168.1.10	22	192.168.2.20	55531	4	164
4	->|<-	2011-05-13 10:59:45.599	8.320	TCP	192.168.1.10	22	192.168.2.20	55531	42	3500
4	->|<-	2011-05-13 10:59:49.615	4.296	TCP	192.168.2.20	55531	192.168.1.10	22	42	2800
4	->|<-	2011-05-13 11:07:01.391	16.660	TCP	192.168.2.50	1227	192.168.1.10	22	43	4592
4	->|<-	2011-05-13 11:07:01.395	0.000	TCP	192.168.1.10	22	192.168.2.50	1227	1	44
4	->|<-	2011-05-13 11:07:01.399	16.660	TCP	192.168.1.10	22	192.168.2.50	1227	38	3408
4	->|<-	2011-05-13 11:07:18.059	0.004	TCP	192.168.1.10	22	192.168.2.50	1227	2	80

Obr. 5.5 Výsledky validace požadavku číslo 4 (192.168.1.10 →|← 0.0.0.0/0, port = 23)

5.6 Validace požadavku číslo 5

Požadavek číslo 5 říká, že konfiguraci přepínače S1 při použití telnetu smí provádět pouze PC Admin a zároveň je to jediný případ, kdy je dovoleno stanici PC Admin použití protokolu telnet. Po validaci tohoto požadavku se ukázaly jako nežádoucí ty toky, kdy Admin komunikoval pomocí telnetu se směrovači R1, R2 a dále toky, reprezentující komunikaci přepínače S1 se stanicí PC Host při použití telnetu. Tedy validní toky při telnetu (pokud se do komunikace zapojila stanice PC Admin nebo přepínač S1), zůstaly pouze ty mezi PC Admin a přepínačem S1. Následující Obr. 5.6 zobrazuje popsané nežádoucí toky při validaci tohoto požadavku.

Toky nesplňující parametry ACL, ->|-<, !-><-, !-><-!

ACL 0 ->|<- 0 !-><- 0 !-><-! 13 celkem: 13

...	Typ p...	Počáteční čas	Doba...	Pro...	Zdroj. adresa	Zdro...	Cíl. adresa	Cíl. port	Počet...	Počet...
5	!-><-!	2011-05-13 10:36:36.379	2.108	TCP	192.168.2.20	35898	192.168.2.1	23	7	339
5	!-><-!	2011-05-13 10:37:05.231	8.060	TCP	192.168.2.20	35899	192.168.2.1	23	39	1646
5	!-><-!	2011-05-13 10:37:36.395	4.044	TCP	192.168.2.20	35899	192.168.2.1	23	21	851
5	!-><-!	2011-05-13 10:37:42.235	9.232	TCP	192.168.2.20	35901	192.168.2.1	23	41	1731
5	!-><-!	2011-05-13 10:37:52.607	1.452	TCP	192.168.2.20	35902	192.168.2.1	23	19	835
5	!-><-!	2011-05-13 10:38:21.119	5.392	TCP	192.168.2.20	35902	192.168.2.1	23	23	938
5	!-><-!	2011-05-13 10:57:42.771	0.000	TCP	192.168.1.1	23	192.168.2.20	55444	1	44
5	!-><-!	2011-05-13 10:57:42.771	9.652	TCP	192.168.2.20	55444	192.168.1.1	23	53	2219
5	!-><-!	2011-05-13 10:57:42.775	9.652	TCP	192.168.1.1	23	192.168.2.20	55444	38	1690
5	!-><-!	2011-05-13 10:58:24.831	6.468	TCP	192.168.2.20	38113	192.168.1.2	23	51	2138
5	!-><-!	2011-05-13 11:07:26.287	0.000	TCP	192.168.1.10	23	192.168.2.50	1228	1	44
5	!-><-!	2011-05-13 11:07:26.287	11.896	TCP	192.168.2.50	1228	192.168.1.10	23	66	2743
5	!-><-!	2011-05-13 11:07:26.299	11.888	TCP	192.168.1.10	23	192.168.2.50	1228	44	1945

Obr. 5.6 Výsledky validace požadavku číslo 5 (192.168.1.10 !→←! 192.168.2.20, port = 23)

5.7 Validace požadavku číslo 6

Poslední testovaný požadavek je typu ACL. Definuje, která zařízení mohou přistoupit na server v DMZ s IP 192.168.1.11. Konkrétně povoluje komunikaci všem uzlům z vnitřní sítě 192.168.2.0/24 kromě uzlu PC Host (192.168.2.50). Všechny ostatní uzly mají na server v DMZ přístup zakázán. Pro testování byla simulována komunikace ze stanic PC Admin (povolená komunikace) a PC Host (nežádoucí komunikace) pomocí nástroje ping (použití protokolu ICMP). Jak je vidět na Obr. 5.7, výsledkem validace je nalezení těch toků, které směřují ze stanice PC Host směrem na server v DMZ.

Toky nesplňující parametry ACL, ->|-<, !-><-, !-><-!

ACL 3 ->|<- 0 !-><- 0 !-><-! 0 celkem: 3

Číslo pož.	Typ pož.	Počáteč...	Doba trv...	Protokol	Zdroj. adresa	Zdroj. ...	Cíl. adresa	Cíl. ...	Počet ...	Počet ...
6	ACL	2011-05-...	54.448	ICMP	192.168.2.50	0	192.168.1.11	0.0	42	23016
6	ACL	2011-05-...	58.240	ICMP	192.168.2.50	0	192.168.1.11	8.0	43	64500
6	ACL	2011-05-...	0.988	ICMP	192.168.2.50	0	192.168.1.11	0.0	2	1096

Obr. 5.7 Výsledky validace požadavku číslo 6 (požadavek na parametr typu ACL)

5.8 Zátěžový test

Cílem zátěžového testu není získat relevantní výsledky validovaných toků, ale změřit čas, po který proces validace běží. Test prokázal, že čas výpočtu nejvýznamněji ovlivňuje rozsah zadaných definic požadavků na parametry sítě. Nejhorší případ představuje zadání požadavku 0.0.0.0/0 →|← 0.0.0.0/0, který zakazuje veškeré síťové toky. V tomto umělém případě jsou shledány všechny toky jako nežádoucí a čas výpočtu se pohybuje kolem 1s na každé 2MB sítových toků. Po zadání definice požadavku, kde počet nežádoucích toků byl 0.72% z celkového počtu zdrojových toků, se čas výpočtu pohyboval kolem 1s na každých 200MB zdrojových síťových toků. Testy proběhly na PC s CPU intel Core2 Duo E8500, OS Ubuntu 10.04, verze jádra 2.6.32-31-generic.

5.9 Shrnutí dosažených výsledků

Ve školní laboratoři byla otestována základní funkcionalita nástroje pro validaci požadavků na vybrané síťové parametry. Uskutečněná komunikace v síti byla zrealizována prostřednictvím protokolů telnet, ssh, http, smb a icmp. Ukázalo se, že veškerá uměle vyvolaná nežádoucí komunikace byla při testování odhalena, čímž lze považovat hlavní cíl této diplomové práce za splněný.

Pro opakování experimentu jsou na přiloženém CD nahrány zdrojové síťové toky, vyexportované nastavení validačního nástroje a návod pro instalaci aplikace a spuštění testu. Pro rekonstrukci síťové topologie jsou na zmíněném CD taktéž uloženy konfigurace použitých aktivních síťových prvků.

Závěr

Diplomová práce obsahuje teoretický úvod, seznámení se s problematikou, návrh řešení a implementaci nástroje pro validaci vybraných parametrů sítě založeném na sledování síťového provozu. Hlavním přínosem této práce je analýza a následné určení vhodných parametrů, které lze k validování použít a implementace nástroje zajišťující proces validace.

Značnou částí přípravy pro mne bylo porozumění verifikačním a simulačním procesům pro ověření návrhu počítačové sítě, které je úzce spojeno s validační technikou, jenž je cílem této práce. Mimo to jsem se věnoval porozumění a praktickému odzkoušení simulačních nástrojů OMNeT++ a Cisco Packet Tracer. Také jsem si vyzkoušel práci s nástroji tcpreplay, tcpdump, nfdump a fprobe. Velkým přínosem pro mě bylo při časté analýze síťových toků pochopení, jakým způsobem v praxi opravdu komunikace po síti probíhá.

S průběhem a výsledky práce jsem spokojen. Veškeré cíle práce, včetně úspěšného otestování nástroje ve školní laboratoři, byly splněny. Jako vhodné rozšíření této práce se nabízí zvýšení počtu podporovaných parametrů k validaci, například o parametry popisující dosažitelnost uzlů v síti. Navazující vývoj si lze také představit ve vytvoření nástroje pro automatizované sestavení požadavků k validaci ze zadané specifikace sítě, případně z informací o aktuálním nastavení sítě, například z exportované konfigurace aktivních prvků v síti.

Následující text obsahuje shrnutí a základní poznatky jednotlivých kapitol diplomové práce. První kapitola nejprve představila pro uvedení do problematiky této práce koncept verifikace, validace a monitorování sítě, který byl vytvořen akademickými pracovníky UITS a UPSY fakulty informačních technologii VUTBR. Nově zmiňované techniky lze rozdělit do tří základních částí (verifikace, simulace a validace). Verifikace pracuje s formálním modelem. Ověřuje vlastnosti sítě jako spolehlivost, bezpečnost a dosažitelnost za pomoci sledování kvalitativních parametrů sítě (stav uzlů a linek v síti). Simulace využívá simulační nástroje k otestování chování sítě za určitých nastavených podmínek. Zkoumá zejména kvantitativní parametry, neboli časové charakteristiky a statistické údaje (ztrátovost a zpoždění paketů, konvergenční čas protokolů a různé časovače). Validace probíhá na reálné fyzicky vytvořené síti. Ověřuje, zda síť splňuje svou specifikaci na základě požadavků ve formě ACL a SLA. Shrnuto: verifikace odhalí kritická místa, simulace otestuje chování sítě s ohledem na nalezená kritická místa a nakonec validace ověří, zda reálně nasazená síť splňuje základní požadavky. V druhé části první kapitoly byly popsány základní technologie, principy a nástroje, kterých je využito při realizaci následujících částí práce. Konkrétně byla vysvětlena technologie NetFlow v9 a nástroj NFDUMP. NetFlow [17] je Cisco technologie, která za pomoci rozdělení provozu na síťové toky umožňuje sledování charakteristických informací posílaných dat v síti. K tomu potřebuje dva typy zařízení nazývající se exportér a kolektor. Exportér sleduje procházející pakety a na jejich základě vytváří a aktualizuje informace o sítových tocích.

Kolektor časem dostává informace od exportérů, ukládá je a případně analyzuje. Uplatnění této znalosti se používá například pro dodržování síťové politiky, účtování, zobrazení spektra síťového provozu nebo odhalování různých podezřelých aktivit v síti. NFDUMP [21] označuje množinu nástrojů pro sběr, zpracování a analýzu NetFlow dat, pracuje v příkazové řádce a je optimalizován pro rychlé zpracování. Konkrétně se jedná o nástroje nfcapd (příjem dat z exportéru a uložení pro pozdější zpracování), nfdump (zpracování a zobrazení NetFlow záznamů), nfprofile (filtruje data a ukládá pro další použití do souboru), nfreplay (přeposílá data po síti), nfclean.pl (odstranění nepotřebných dat), ft2nfdump (převod příkazů nástroje flow-tools do NFDUMP).

Druhá kapitola představila konkrétní cíl této práce, kterým je vytvoření nástroje pro validaci určených parametrů sítě pomocí sledování síťového provozu. Dále byly specifikovány požadavky na výstupní projektovou část, mezi které patří například grafické rozhraní, využití sítových toků a nástroje NFDUMP. Poté byla provedena analýza možných validovatelných parametrů, jejichž výsledkem je rozdělení daných parametrů do tří hlavních skupin na omezující parametry (filtrovací pravidla, minimální nebo maximální propustnost, limity příchozího a odchozího provozu, maximální celkové zatížení sítě), parametry popisující dosažitelnost uzlů v síti a časové parametry (zpoždění paketů, odezva). Analýza se poté zabývala možnostmi uložení a zpracování síťových toků a výběrem programovacích jazyků a technologií (JAVA, NetBeans). Nakonec byl představen návrh řešení projektové části, který obsahuje popis vstupů, výstupů a jednotlivých procesů, které budou implementovány.

Třetí kapitola podrobně rozebrala implementované parametry k validaci, mezi které patří blokování komunikace ($\rightarrow|\leftarrow$), jednosměrné omezení ($!\rightarrow\leftarrow$), obousměrné omezení ($!\rightarrow\leftarrow!$), jednosměrně inicializovaná komunikace ($\rightarrow$) a ACL. Ke každému parametru byl popsán jeho význam, příklad použití a postup jeho validace. Následně byl nastíněn doporučený způsob definování požadavků za účelem popsání celkového požadovaného chování sítě.

Čtvrtá kapitola se věnovala implementaci nástroje. Nejprve byl představen základní pohled na aplikaci (JAVA s využitím vývojového prostředí NetBeans), včetně zobrazení objektové struktury ve formě diagramu tříd. Poté byly v samostatných podkapitolách prezentovány jednotlivé části vytvořeného nástroje týkající se konfigurace sítových zařízení, definování požadavků k validaci, nastavení vstupních sítových toků, možnostmi zobrazení výsledků validace a importem/exportem nastavení.

Kapitola pátá popsala průběh testování aplikace, který proběhl ve školní laboratoři. K tomuto účelu byla vytvořena experimentální počítačová síť s požadavky zaměřenými na otestování validace všech typů implementovaných síťových parametrů. Pro každý požadavek byl rozebrán výsledek jeho validace s ohledem na reálně uskutečněnou komunikaci v síti. Následně byl popsán zátěžový test určující časovou náročnost aplikace a celkové shrnutí výsledků validace.

Literatura

[1] **Xie, G, a další.** On static reachability analysis of ip networks. *INFOCOM*. 2005. stránky 2170–2183.

[2] **Scherfel, P.** *Simulace chování sítě na základě analýzy konfiguračních souborů aktivních síťových zařízení.* Brno : FIT VUT, 2009. bakalářská práce.

[3] Welcome to the OMNeT++ Community! *OMNeT++*. [Online] [Citace: 25. 10 2010.] http://www.omnetpp.org/.

[4] **Žádník, M, a další.** Network Probe for Flexible Flow Monitoring. *Design and Diagnostics of Electronic Circuits and Systems, 2008. DDECS 2008. 11th IEEE Workshop on.* 2008. ISBN: 978-1-4244-2276-0.

[5] **Boehm, B W.** Verifying and Validating Software Requirements and Design Specifications. *Software, IEEE.* Leden 1984, Sv. 1, 1, stránky 75 - 88.

[6] **Boehm, B W.** Guidelines for Verifying and Validating Software Requirements and Design Specifications. *Euro IFIP 79.* 1979, stránky 711-719.

[7] **Avižienis, A, a další.** Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing.* 2004, Sv. 1, 1, stránky 11 - 33.

[8] **Čejka, R, a další.** *A Formal Approach to Network Security Analysis.* Brno : FIT VUT, 2008. str. 20.

[9] **Matoušek, P, a další.** A Formal Model for Network-wide Security Analysis. *Proceeding of the 15 IEEE International Symposium and Workshop on the Engineering of Computer-based Systems.* 2008. stránky 171-181. ISBN 0-7695-3141-5.

[10] **Matoušek, P, a další.** Combination of Simulation and Formal Methods to Analyse Network Survivability. *Proceedings of the IEEE 3rd International ICST Conference on Simulation Tools and Techniques.* 2010. str. 6. ISBN 978-963-9799-87-5.

[11] **Silva, G.** Using Formal Analysis Approach on Networks with Dynamic Behaviors. *5th EEICT student conference.* 2009. 4, stránky 390 - 394.

[12] **Švéda, M, a další.** An Approach for Automated Network-Wide Security Analysis. *Proceedings of the Ninth International Conference on Networks ICN 2010.* 2010. stránky 294-299. ISBN 978-0-7695-3979-9.

[13] Cisco Packet Tracer. [Online] [Citace: 15. 10 2010.] http://www.cisco.com/web/learning/netacad/course_catalog/PacketTracer.html.

[14] **Rybová, V.** *Modelování a simulace návrhových vzorů směrování v počítačových sítích.* Brno : FIT VUT, 2009. bakalářská práce.

[15] Welcome to the INET Framework! . *INET FRAMEWORK.* [Online] [Citace: 9. 11 2010.] http://inet.omnetpp.org/.

[16] Welcome to project ANSA homepage. *ANSA.* [Online] [Citace: 8. 11 2010.] https://nes.fit.vutbr.cz/ansa/pmwiki.php.

[17] RFC 3954 - Cisco Systems NetFlow Services Export Version 9. *Internet FAQ Archives.* [Online] [Citace: 5. 10 2010.] http://www.faqs.org/rfcs/rfc3954.html.

[18] **Žádník, M.** Monitorování sítí na základě IP toků. *CONNECT!* 2008, 2, stránky 14-15.

[19] Introduction to Cisco IOS NetFlow - A Technical Overview. *CISCO.* [Online] [Citace: 9. 11 2010.] http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6555/ps6601/prod_white_paper0900aecd80406232.html.

[20] What Is NetFlow? *altus security.* [Online] [Citace: 27. 12 2010.] http://www.netflowanalyser.co.uk/netflow/index.php.

[21] NFDUMP. *SOURCEFORGE.NET.* [Online] [Citace: 29. 12 2010.] http://nfdump.sourceforge.net/.

[22] NfSen - Netflow Sensor. *SOURCEFORGE.NET.* [Online] [Citace: 30. 12 2010.] http://nfsen.sourceforge.net/.

[23] **Christiansen, Mikkel a Fleury, Emmanuel.** An Interval Decision Diagram Based Firewall. *3rd IEEE International Conference on Networking (ICN '04).* 2004.

[24] **Schatzmann, Dominik, a další.** FACT: Flow-based Approach for Connectivity Tracking. *Passive and Active Measurement conference (PAM 2011).* 2011.

[25] IP Flow Information Export (ipfix). *datatracker.ietf.org* . [Online] [Citace: 17. 05 2011.] http://datatracker.ietf.org/wg/ipfix/charter/.

Seznam příloh

Přílohy

Příloha 1. Kód skriptu pro analýzu parametru jednosměrně inicializované komunikace ($A \rightarrow B$).

```
awk 'BEGIN {i=1;} {
 source_port=$7
 dest_port=$9
 if ($4 == "->") {
  if (source_port in source_port_array) {
   source_port_array[source_port] = source_port_array[source_port] + 1;
  } else {
   source_port_array[source_port] = 1;
  }
  if (dest_port in dest_port_array) {
   dest_port_array[dest_port] = dest_port_array[dest_port] + 1;
  } else {
   dest_port_array[dest_port] = 1;
  }
  if (source_port"_"dest_port in fault) {
   print fault[source_port"_"dest_port], " ", $1, $2, " ", $3, " ", $4, " ", $5, " ", $6…;
  }
 } else if ($4 == "<-") {
  if (dest_port in source_port_array && source_port in dest_port_array) {
   if (source_port_array[dest_port] <= 0 || dest_port_array[source_port] <= 0) {
    print i, " ", $1, $2, " ", $3, " ", $4, " ", $5, " ", $6…;
    fault[dest_port"_"source_port]=i;
    i++;
   } else {
    source_port_array[dest_port] = source_port_array[dest_port] - 1;
    dest_port_array[source_port] = dest_port_array[source_port] - 1;
   }
  } else {
   print i, " ", $1, $2, " ", $3, " ", $4, " ", $5, " ", $6…;
   fault[dest_port"_"source_port]=i;
   i++;
  }
 }
}' tmp_1 | sort -n > tmp_2
```

Příloha 2. CD

Přiložený nosič obsahuje potřebná data pro přeložení, instalaci a otestování aplikace. Obsah CD má následující adresářovou strukturu:

\aplikace	Přeložená aplikace
\dokument	Text diplomové práce ve formátu pdf.
\navody	Návody pro instalaci překlad a otestování aplikace.
\test	Data a nastavení pro testování aplikace popsaném v kapitole 5.
\test\fotogalerie	Fotografie pořízené pří sestavení experimentální sítě ve školní laboratoři.
\test\nastaveni_netvalid	Exportované nastavení validačního nástroje.
\test\sit_konfigurace	Použitá konfigurace aktivních prvků sítě.
\test\sitove_toky	Nasbírané síťové toky - vstupní data k validaci.
\test\sit_pt	Topologie a nastavení sítě pro testování v Cisco Packet Tracer.
\zdrojove_kody	Zdrojové kódy vytvořené aplikace.
\zdrojove_kody\java	Zdrojové kódy jazyku JAVA.
\zdrojove_kody\scrip	Skript pro volání nástroje NFDUMP.