



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

DETEKCE ANOMÁLIÍ V SÍŤOVÉM PROVOZU

NETWORK ANOMALY DETECTION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. VÁCLAV BARTOŠ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. MARTIN ŽÁDNÍK

BRNO 2011

Abstrakt

Tato práce se zabývá systémy a metodami pro detekci anomálií v provozu na počítačových sítích. Nejdříve je uvedeno rozdělení systémů pro zajištění síťové bezpečnosti a je stručně popsáno množství nejrůznějších metod používaných v systémech detekce anomálií. Hlavní náplní této práce je však optimalizace metody detekce anomálií, kterou navrhli Lakhina et al. a která je založená na detekci změn distribuce hodnot z hlaviček paketů. Tato metoda je v práci podrobně popsána a jsou navrženy dvě její optimalizace – první se zaměřuje na rychlost a paměťovou náročnost, druhá zlepšuje její detekční schopnosti. Dále je popsán program vytvořený pro testování těchto optimalizací a jsou prezentovány výsledky experimentů na reálných datech s uměle generovanými i skutečnými anomáliemi.

Abstract

This work studies systems and methods for anomaly detection in computer networks. At first, basic categories of network security systems and number of methods used for anomaly detection are briefly described. The core of the work is an optimization of the method based on detection of changes in distributions of packet features originally proposed by Lakhina et al. This method is described in detail and two optimizations of it are proposed – first is focused to speed and memory efficiency, second improves its detection capabilities. Next, a software created to test these optimizations is briefly described and results of experiments on real data with artificially generated and also real anomalies are presented.

Klíčová slova

Detekce anomálií, síťová bezpečnost, entropie, optimalizace, hash funkce

Keywords

Anomaly detection, network security, entropy, optimization, hash function

Citace

Václav Bartoš: Detekce anomálií v síťovém provozu, diplomová práce, Brno, FIT VUT v Brně, 2011

Detekce anomálií v síťovém 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.

.....

Václav Bartoš
24. května 2011

Poděkování

Především bych chtěl poděkovat svému vedoucímu diplomové práce, Ing. Martinu Žádníkovi za užitečné rady, čas věnovaný konzultacím a za zprostředkování přístupu k potřebným datům.

© Václav Bartoš, 2011.

Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.

Obsah

1 Úvod	3
2 Přehled metod pro detekci anomálií	5
2.1 Statistické metody	5
2.2 Metody založené na strojovém učení	7
2.2.1 Bayesovské sítě	7
2.2.2 Analýza hlavních komponent	7
2.2.3 Ostatní metody strojového učení	8
2.3 Metody založené na dolování dat	9
2.3.1 Metody založené na klasifikaci	9
2.3.2 Shluková analýza a detekce outliers	10
2.3.3 Asociační pravidla	11
3 NetFlow	13
3.1 Architektura	13
3.2 Protokol	14
4 Popis vybrané metody	16
5 Optimalizace metody	20
5.1 Zrychlení algoritmu	20
5.2 Zvýšení detekční citlivosti	21
6 Testovací software	24
6.1 Formát vstupních dat	24
6.2 Parametry a výstup programu	25
6.3 Příklady použití	26
7 Výsledky	27
7.1 Míra rozlišitelnosti	27
7.2 Optimalizace pomocí hashovací funkce	27
7.3 Hledání jiných dimenzí a metrik	31
7.3.1 Popis grafů a značení	32
7.3.2 Rozlišitelnost jednotlivých anomálií	32
7.3.3 Příklad anomálie v časových průbězích	34
7.3.4 Shrnutí	36
8 Závěr	37

A	Obsah CD	41
B	Parametry programu	42
	B.1 Specifikace dimenzí	43
	B.2 Příklady	44
C	Grafy rozlišitelnosti anomálií	45

Kapitola 1

Úvod

Počítačové sítě jsou dnes běžnou součástí našeho života. Na jejich spolehlivost a bezpečnost spoléhá obrovské množství lidí i služeb. Avšak zajistit bezpečnost sítě a ochránit ji před stále stoupajícím počtem útočníků či škodlivého software není jednoduchá úloha. Dnes běžnou součástí zabezpečení každé sítě (a často i jednotlivých počítačů) je firewall. Ačkoli je firewall pro zabezpečení sítě nezbytný, poskytuje jen základní ochranu a na všechny typy útoků zdaleka nestačí. Proto se vyvíjejí různé počítačové systémy, které umožňují sledovat provoz na síti a automaticky něm vyhledávat potenciální hrozby. Ty jsou pak hlášeny operátorovi sítě, který může aplikovat příslušná opatření, případně mohou být aplikována automaticky.

Se stále rostoucími rychlostmi sítí již navíc nestačí zabývat se jen otázkou účinnosti systému pro detekci útoků, ale důležitá je i jeho výpočetní náročnost. Vzhledem k tomu, že přenosové rychlosti sítí rostou mnohem rychleji, než výkon počítačů, bude u takových systémů rychlost zpracování do budoucna čím dál důležitější, především na páteřních linkách internetu a přípojných bodech velkých společností či univerzit, kde se rychlosti mohou pohybovat v řádu jednotek až desítek, během několika let možná i stovek gigabitů za sekundu.

Systémy pro detekci útoků (*intrusion detection systems* – IDS) se dělí do dvou kategorií podle základního použitého principu – detekce signatur (vzorů) a detekce anomálií.

Systémy založené na detekci signatur závisí na znalosti specifických vzorů, jimiž se různé útoky projevují. Detekce pak spočívá ve vyhledávání těchto vzorů v datech síťového provozu a při nalezení shody je hlášen útok. Jinak řečeno, rozhodnutí je učiněno na základě znalostí získaných z modelu útoku a podle pozorované stopy zanechané útokem v systému. Takové metody tedy v datech hledají stopy útoku bez ohledu na vlastnosti normálního provozu [1].

Systémy založené na detekci anomálií místo modelování útoku vytvářejí model normálního chování. Každá významnější odchylka od tohoto modelu je pak označena jako anomálie a tedy pravděpodobně útok. Tato metoda tedy vezme trénovací množinu normálních dat a další, neoznačená data pro testování a jejím cílem je určit, jestli testovací data odpovídají normálnímu chování, či zda se jedná o anomálii [2].

Jednou z hlavních výhod první metody je schopnost spolehlivě detekovat známé útoky s velmi malou pravděpodobností falešných poplachů. Navíc, vzhledem k tomu, že detekce probíhá na základě vzorů specifických pro konkrétní útoky, je vždy snadné určit, o jaký typ útoku se jedná. Další dobrou vlastností těchto systémů je to, že dokáží síť chránit ihned po instalaci.

Nedostatkem těchto metod je ovšem to, že dokáží detekovat jen útoky, které jsou již známy a jejich signatura je uložena v databázi systému. Je tedy nutné zajistit co nejčastější aktualizaci této databáze. Dalším problémem je náročnost uchovávání stavové informace v případě, že kompletní signatura může být rozprostřena do více diskrétních událostí, na-

příklad když hledaný řetězec zasahuje do více navazujících paketů [1].

Systémy založené na tomto principu jsou běžně používány pro zabezpečení všech větších sítí. Jsou k dispozici jak komerční systémy, tak i open-source varianty, především systém SNORT¹.

Mezi výhody metod založených na detekci anomálií patří především schopnost rozpoznat i nové, dříve neznámé typy útoků. To proto, že útok je rozpoznán na základě toho, že se odchyluje od normálního provozu, ne proto, že by někdo systém nastavil právě na jeho detekci. Dále, vzhledem k tomu, že v modelu normálního chování mohou být zahrnuty i profily používání sítě jednotlivými uživateli nebo skupinami uživatelů, mohou tyto systémy zachytit i útoky z vnitřku sítě, kdy uživatel, který má do sítě přístup (nebo někdo používající ukradený účet), začne konat činnost, která se značně liší od jeho běžného využívání sítě. Navíc díky tomu, že je systém založen na profilech na konkrétní síti, je pro útočníka obtížné zjistit, co vše může udělat, aniž by byl detekován.

Mezi nevýhody detekce anomálií patří to, že při nasazení musí systém nejprve projít trénovací fází, při které si vytvoří model normálního chování, a až poté může začít detekovat anomálie a pomáhat tak chránit síť. Kromě toho, že samotné vytváření a udržování tohoto modelu je značně obtížné a vytvoření nepřesného modelu vede na velmi malou úspěšnost celého systému, hlavní nevýhodou většiny metod založených na detekci anomálií je vysoká míra falešných poplachů, především tzv. *false positives*, tedy označení nějaké události za útok, přestože se o útok nejedná [1]. Snižování míry falešných poplachů je v současnosti jedním z hlavních cílů výzkumu v této oblasti.

Je to také jeden z hlavních důvodů, proč tyto systémy stále nejsou v praxi běžně nasazovány, narozdíl od systémů založených na vyhledávání signatur, jejichž nasazení je dnes samozřejmou součástí zabezpečení každé větší sítě. Až v posledních několika letech se objevují první komerční IDS systémy založené na detekci anomálií.

Mnoho reálných systémů nepoužívá pouze jeden z uvedených přístupů, ale kombinuje obojí, detekci vzorů i detekci anomálií. Takové systémy se obvykle nazývají *hybridní systémy*.

Následující kapitola je stručným shrnutím různých metod a technik, které se používají, nebo se v historii používaly pro detekci anomálií v počítačových sítích. Po ní následuje kapitola 3, která krátce popisuje NetFlow protokol sloužící k reprezentaci IP toků, což jsou data, nad nimiž často metody detekce anomálií pracují.

Jádrum práce je pak optimalizace metody založené na detekci změn v distribucích hodnot vlastností síťového provozu pomocí výpočtů entropie, navržená Lakhinou et. al. v práci [3]. Tato metoda je podrobně popsána v kapitole 4 a v kapitole 5 jsou navrženy dva způsoby jejího vylepšení – první je zaměřeno na optimalizaci rychlosti a paměťové náročnosti algoritmu, druhým je návrh změn směřujících ke zvýšení detekční citlivosti.

Pro testování navržených optimalizací byl vytvořen program, který je popsán v kapitole 6. V kapitole 7 jsou pak popsány výsledky testování a je zhodnocen přínos navržených vylepšení. V poslední kapitole je uvedeno shrnutí celé práce.

¹<http://www.snort.org/>

Kapitola 2

Přehled metod pro detekci anomálií

Pro detekci anomálií v síťovém provozu existuje mnoho různých metod využívajících velmi odlišné přístupy. Všechny metody však mají společné to, že nejprve musí systém projít trénovací fází, po které následuje fáze testovací. V trénovací fázi systém pouze pasivně sleduje provoz na síti a vytváří si model normálního chování. V testovací fázi pak systém porovnává aktuální charakteristiku provozu na síti s vytvořeným modelem a nalezené odchylky označuje jako anomálie.

Formálně můžeme systém pro detekci anomálií definovat jako pár $S = (M, D)$, kde M je model normálního chování sledovaného systému a D je míra, která k dané události určí její odchylku od modelu M [2]. Pro aplikaci v počítačových sítích je navíc důležité, že charakteristika normálního provozu na síti se může časem měnit, a proto i její model se musí neustále přizpůsobovat a systém se tak vlastně stále učí.

Metody detekce anomálií můžeme rozdělit do několika základních kategorií podle principu, na němž jsou založeny.

Při psaní této kapitoly bylo čerpáno především z přehledu technik pro detekci anomálií od A. Patcha a J. Park [1].

2.1 Statistické metody

V roce 1987 publikoval D. E. Denning obecný model IDS systému založeného na statistických metodách [4], který tvoří základ i pro mnoho dnešních systémů. Podle tohoto modelu byl brzy na Stanford Research Institute vyvinut jeden z vůbec prvních IDS systémů – *Intrusion Detection Expert System* (IDES) [5, 6]. Později byla publikována jeho vylepšená verze *Next-generation Intrusion Detection Expert System* (NIDES) [7].

Tyto systémy jsou navrženy pro práci v reálném čase, neustále tedy sledují chování uživatelů a dokáží detekovat podezřelé události ihned, jakmile se objeví. Jejich hlavním prvkem je sice použití statistických metod pro detekci anomálií, obsahují ale i modul pro detekci známých útoků na základě signatur.

Detekce anomálií spočívá v uchování profilu normálního chování, založeného na množině několika vybraných proměnných. Aktuální aktivita uživatele (či systému, sítě) je pak porovnávána s očekávanými hodnotami těchto proměnných podle uloženého profilu a pokud je zaznamenána aktivita dostatečně daleko od očekávaných hodnot, je hlášena anomálie. Každá proměnná v uloženém profilu odráží míru, do jaké je daný typ chování podobný pro-

filu vytvořenému v normálních podmínkách. Toto je počítáno tak, že každá míra/proměnná je přiřazena nějaké náhodné proměnné. V průběhu času je počítána a aktualizována jejich frekvenční distribuce (histogram). Je počítána jako vážená suma s exponenciálním rozložením vah se středem 30 dní (tzn. záznamy staré 30 dní mají poloviční váhu oproti těm nejnovějším, 60 dní staré čtvrtinovou atd.) Frekvenční distribuce je uchovávána ve formě histogramu, kde je ke každému možnému rozsahu hodnot, přiřazena jeho pravděpodobnost. Pomocí ní je pak možné vypočítat hodnotu, určující, jak je proměnná z aktuálního záznamu daleko od „normální“ hodnoty. Kombinací hodnot všech proměnných a úvahou korelace mezi proměnnými pak lze vypočítat, jak je od normálu daleko celý aktuální záznam. Záznamy za jistou hranicí jsou označeny jako možné útoky.

Nevýhodou této techniky je, že definuje normální hodnoty každé proměnné nezávisle na ostatních, ve skutečnosti ale útoky často ovlivňují několik proměnných najednou.

Je vhodné poznamenat, že tyto systémy, stejně jako některé uvedené dále, sbírají data přímo na sledovaných počítačích (data o přihlašování k účtům, spotřebě CPU a paměti, přístupu k souborům, vytváření procesů, atp.), nikoliv na síti, a nejde tedy přímo o detekci anomálií v síťovém provozu. V této práci jsou však přesto uvedeny, protože patří mezi již klasické systémy, na jejichž principech staví mnoho novějších metod detekce anomálií.

Dalším z prvních systémů detekce anomálií založených na statistických metodách je systém *Haystack* [8], navržený v roce 1988 pro detekci útoku ve víceuživatelském počítačovém systému amerického letectva [9]. Jde o hybridní systém, obsahuje tedy moduly jak pro detekci anomálií tak i pro detekci signatur.

Při detekci anomálií jsou parametry systému modelovány jako nezávislé Gaussovské proměnné, přičemž je pro každou z nich je definován normální rozsah hodnot. Pokud během sezení nějaký parametr vybočí z normálu, je zvýšeno *anomaly score* (hodnota udávající míru abnormality) tohoto sezení. Když *anomaly score* překročí určitou mez, je spuštěn alarm. Statistické údaje jsou sledovány jak na úrovni jednotlivých uživatelů, tak různých předdefinovaných skupin. Velkou nevýhodou tohoto systému však je, že byl navržen pro práci offline. Na provádění statistické analýzy v reálném čase neměly tehdejší systémy dostatečný výkon.

Statistical Packet Anomaly Detection Engine (SPADE) [10] je plugin pro SNORT, který je dalším zástupcem statistických systémů pro detekci anomálií. Je zde použit jednoduchý frekvenční přístup, jak vypočítat *anomaly score* paketu – čím méně často je daný paket zaznamenán (s tím, že nedávno zaznamenané pakety mají větší váhu než starší), tím vyšší je jeho *anomaly score*. Když *anomaly score* překročí jistou mez, je hlášena anomálie. Tento systém je velmi jednoduchý, jeho velkou nevýhodou je však poměrně velká míra falešných poplachů, protože každý neznámý paket je automaticky klasifikován jako možný útok.

Mezi další práce zabývající se detekcí anomálií založenou na statistických metodách patří například práce od Ye et al. [11], ve které autoři navrhují použití statistického testu Hotelling's T^2 , díky čemuž systém dokáže zachytit i vzájemné relace mezi různými měřenými proměnnými. Anomálie způsobené útoky totiž obvykle zapříčiňují odchylky ve více parametrech najednou, jejich vzájemné relace jsou tedy důležitou informací, jejíž znalost pomáhá vylepšovat výsledky detekce.

Existují i analytické studie systémů pro detekci anomálií. Lee a Xiang [12] navrhují pro vytváření či testování vhodnosti modelů normálních dat, používat některé míry z teoretické informatiky, jako např. entropii, relativní entropii či informační zisk. Ukázali, že tyto míry dokáží charakterizovat pravidelnost zaznamenaných dat, a také jak toho lze využít při vytváření modelu.

2.2 Metody založené na strojovém učení

Strojové učení lze definovat jako schopnost programu či systému se postupem času učit a vylepšovat svou výkonnost pro danou úlohu. Narozdíl od statistických metod se tyto nesnaží pochopit podstatu procesů generujících data, ale místo toho vytvářejí systémy, které se neustále vylepšují na základě předchozích výsledků.

Tento přístup je obecně mladší než přístupy předchozí – většina z dále uvedených metod byla publikována kolem roku 2000.

2.2.1 Bayesovské sítě

Bayesovská síť je grafický model, zachycující závislosti mezi různými náhodnými proměnnými. Má pro detekci anomálií dvě výhodné vlastnosti, zaprvé, protože v Bayesovské síti jsou zachyceny vztahy mezi jednotlivými proměnnými, lze ji použít i tehdy, když nějaká část dat chybí. Zadruhé, dokáže zachytit i kauzální vztahy, takže pomocí ní lze předvídat následky nějaké akce. Díky těmto vlastnostem lze Bayesovské sítě dobře použít tam, kde je třeba kombinovat data a nějaké předchozí znalosti.

Bayesovské sítě byly pro detekci anomálií použity například v práci od Valdes et al. [13]. Zde je popsán systém využívající naivní Bayesovskou síť, který má schopnost zachytit i takové distribuované útoky, u kterých jednotlivá sezení samostatně nejsou dostatečně abnormální aby spustila alarm.

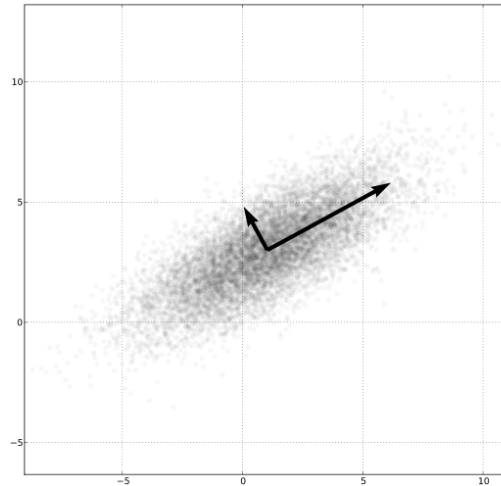
Bayesovské sítě se ale nepoužívají jen přímo pro hledání anomálií, často slouží také pro klasifikaci a potlačení falešných poplachů. Příkladem takové práce je článek od Kruegel et. al. [14], kde bylo pro agregaci výsledků různých metod detekce anomálií navrženo právě použití Bayesovských sítí (místo běžného porovnání součtu výsledků s nějakou mezí). Ani následující metoda neslouží přímo pro detekci anomálií, ale spíše k předzpracování dat.

2.2.2 Analýza hlavních komponent

Datové sady používané při detekci anomálií jsou obvykle velmi rozsáhlé a mají mnoho dimenzí. Při dnešních velkých rychlostech sítí je zpracování takových dat značně výpočetně náročné, a proto se v mnoha případech začala používat metoda pro snížení množství dimenzí dat, známá jako *Analýza hlavních komponent* (*Principal Component Analysis*, PCA). Je to matematická metoda, pomocí které lze množinu několika pozorování závislých proměnných převést na množinu hodnot nezávislých proměnných, zvaných hlavní komponenty, přičemž počet hlavních komponent je menší nebo roven počtu původních proměnných. Nové nezávislé proměnné jsou lineárními kombinacemi původních proměnných a reprezentují stejná data v jednodušší formě.

Transformace je definována tak, že první hlavní komponenta má největší možný rozptyl (je to tedy projekce dat takovým směrem, aby byl rozptyl projekce maximální), každá další komponenta pak má největší možný rozptyl při zachování podmínky, že je ortogonální ke všem komponentám předchozím. Příklad hlavních komponent Gaussovské proměnné v dvourozměrném prostoru je uveden na obrázku 2.1. V mnoha případech obsahuje již prvních několik hlavních komponent většinu celkového rozptylu dat. Zbytek pak lze s minimální ztrátou informace zanedbat, a tak výrazně snížit počet dimenzí zpracovávaných dat.

Tato metoda byla pro detekci anomálií využita například v práci od Shyu et. al. [16]. Zde se vychází z toho, že na anomálie lze nahlížet jako na tzv. *outliers* (osamocené hodnoty daleko od ostatních) a je vytvořen klasifikátor nepracující přímo s daty, ale s jejich hlavními komponentami. Autoři v této práci ukázali, že jejich klasifikátor založený na PCA



Obrázek 2.1: Příklad dvourozměrné Gaussovské distribuce. Šipky znázorňují její hlavní komponenty. [15]

poskytuje lepší výsledky než jiné běžně používané metody hledání outliers, jako například *Local Outlier Factor* nebo vyhledávání nejbližších sousedů.

2.2.3 Ostatní metody strojového učení

Autoři Mahoney a Chan [17, 18, 19] navrhli několik systémů pro detekci anomálií v síťovém provozu pomocí analýzy hlaviček paketů. Všechny mají společné to, že aplikují metody strojového učení k získání profilu normálního chování pro síťové protokoly na různých úrovních. Jsou to systémy *Packet Header Anomaly Detector* [17] (PHAD), *Learning Rules for Anomaly Detection* [18] (LERAD) a *Application Layer Anomaly Detector* [19] (ALAD).

Tyto metody sledují různé položky hlaviček paketů, což jsou vícedimenzionální data. Tyto metody pracují tak, že tento vícedimenzionální problém rozbijí na jednodimenzionální problémy, zpracují jednotlivé dimenze samostatně a výsledky (s různými vahami) sečtou. Tento způsob je sice výhodný z hlediska výpočetní složitosti, ztratí se tím ovšem vzájemné vazby mezi různými dimenzemi, což může znemožnit detekci některých útoků. Například typický *SYN flood* útok se projevuje větším počtem SYN paketů současně s nižším počtem ACK paketů. Oba tyto projevy samy o sobě jsou běžné, o útok se jedná jen když se objeví současně, což ovšem tyto metody nedokážou zachytit.

V oblasti detekce anomálií byly využívány i *Markovovy řetězce* a *skryté Markovovy modely* (*hidden Markov models*, HMM). Například Warrender et al. [20] porovnávali čtyři různé metody hledání anomálií v sekvencích systémových volání, z nichž jedna byla založena právě na HMM. Ukázali sice, že tento model dává lepší výsledky než ostatní tři porovnávané metody, ale za cenu mnohem větší výpočetní náročnosti. Byl totiž použit HMM s plně propojenými stavy (tzn. z každého stavu je možné přejít do jakéhokoliv jiného stavu), což pro systém s S stavy dává $2S^2$ hodnot v přechodové tabulce. Vzhledem k tomu, že počet stavů odpovídá počtu různých systémových volání generovaných aplikací, a těch je obvykle poměrně velké množství, je modelování normálních sekvencí systémových volání pomocí HMM v reálném čase výpočetně nezvládnutelné.

2.3 Metody založené na dolování dat

Stále častěji se pro detekci anomálií využívají také metody *dolování dat*.

Dolování dat, někdy nazývané také *získávání znalostí z databází*, je proces vyhledávání zajímavých a užitečných informací a vztahů ve velkém množství dat [21]. Jinak řečeno, je to schopnost vzít vstupní data a získat z nich informace, které nejsou patrné na první pohled.

Na základě metod dolování dat bylo navrženo mnoho systémů pro detekci anomálií používajících různé techniky.

2.3.1 Metody založené na klasifikaci

Klasifikací rozumíme metody, které rozdělují vstupní data do několika tříd na základě sady pravidel, vzorů nebo jiných podobných technik. V případě detekce anomálií se obvykle jedná o binární klasifikaci, tedy rozdělení pouze do dvou tříd – normální data a anomálie.

Algoritmy pro odvozování pravidel

Jsou to metody, které dokáží vyhledávat vztahy mezi proměnnými ve formě asociačních pravidel tvaru „*když se objevila událost X, pak se pravděpodobně objeví událost Y.*“ Jde tedy o hledání takových množin X a Y , že X „implikuje“ Y (alespoň s nějakou významnou pravděpodobností). Při klasifikaci se uvažuje Y konstantní a úkolem je najít takové množiny X , které dobře předpovídají dané Y . Výhodou používání pravidel je to, že jsou jednoduchá a intuitivní, nestrukturovaná a ne zcela přísná. Na druhou stranu je obtížné jejich udržování a v některých případech nejsou zcela vhodná pro reprezentaci všech informací.

Bylo navrženo několik metod pro automatické odvozování pravidel z dat. Některé z nich nejdříve vytváří *rozhodovací strom* a pomocí něj poté generují klasifikační pravidla.

Ostatní algoritmy (například RIPPER [22] nebo C4.5 [23]) odvozují pravidla přímo z dat pomocí přístupu rozděl a panuj. Po skončení učící fáze pak následuje ještě odstranění či prořezání některých pravidel pro zlepšení přesnosti algoritmů. Algoritmus RIPPER byl použit v několika systémech pro detekci anomálií ke klasifikaci příchozích dat. Například Lee et al. [24] ho použili k charakterizaci sekvencí objevujících se v normálních datech pomocí malé sady pravidel, která popisovala společné prvky těchto sekvencí. V testovací fázi pak byly sekvence, které neodpovídaly těmto pravidlům, označeny jako anomálie.

Fuzzy logika

Fuzzy logika je další z technik používaných v oblasti počítačové bezpečnosti a detekce anomálií. Využití fuzzy logiky při detekci útoků je vhodné hlavně ze dvou důvodů [25]: Zaprvé, při detekci útoků je sledováno mnoho kvantitativních proměnných (např. číslo portu, délka spojení nebo počet různých TCP/UDP služeb na jednom hostiteli), na které lze pohlížet jako na fuzzy proměnné. Zadruhé, koncept bezpečnosti sám o sobě je do jisté míry neurčitý (fuzzy).

Uvažujeme-li, že je definován nějaký interval, ve kterém je hodnota dané proměnné považována za normální, pak v běžném případě jsou všechny instance dat uvnitř/vně tohoto intervalu považovány za stejně normální/abnormální bez ohledu na jejich vzdálenost od hranic intervalu. Fuzzy logika pomáhá vyhledat takové přísné rozdělení na normální a abnormální chování, a to tak, že datům přiřazuje k dané třídě vždy nějakou míru příslušnosti.

Příkladem IDS systému používajícího fuzzy logiku je systém *Fuzzy Intrusion Recognition Engine* (FIRE) [26]. Pomocí jednoduchých metod dolování dat zpracovává vstupní data

a vygeneruje fuzzy množiny pro všechny pozorované vlastnosti. Na základě těchto množin jsou pak vytvořena fuzzy pravidla pro klasifikaci dat na normální provoz a anomálie. FIRE si neuchovává žádný model normálního chování systému, pravidla jsou vytvářena ručně pro detekci konkrétních útoků. Autoři ukázali, že tento přístup je zvláště účinný pro detekci různých způsobů skenování sítě. Hlavní nevýhodou systému je ale náročnost procesu vytváření pravidel, které se musí provádět ručně.

Genetické algoritmy

Tato technika, používaná ke hledání přibližných řešení a optimalizaci vyhledávacích problémů, byla také mnohokrát použita v oblasti detekce útoků k rozlišení anomálií od normálního provozu. Hlavní výhodou genetických algoritmů pro tuto oblast je to, že jsou to velmi flexibilní a robustní metody pro prohledávání stavového prostoru. Navíc toto prohledávání konverguje k řešení z více směrů a je založeno na pravděpodobnostních pravidlech namísto deterministických.

V oblasti detekce anomálií byly genetické algoritmy použity mnoha způsoby. Některé přístupy je použily přímo pro generování klasifikačních pravidel, jiné je použily pouze pro nalezení vhodné množiny vlastností ke sledování, či pro určení optimálních parametrů různých funkcí, přičemž pro samotné získání pravidel byly použity jiné metody.

2.3.2 Shluková analýza a detekce outliers

Shluková analýza (někdy také *klastrování*, angl. *clustering*) je souhrnný název pro metody vyhledávání shluků v nepopsaných datech o mnoha dimenzích. Hlavní výhodou těchto metod pro detekci anomálií je jejich schopnost učit se z dat a vyhledávat v nich útoky bez toho, aby někdo musel poskytnout popis různých typů útoků. Množství trénovacích dat, které je nutno systému dodat, je také menší, než u jiných metod.

Outliers jsou z pohledu shlukové analýzy objekty, které nenáležejí do žádného shluku, a v kontextu detekce anomálií mohou reprezentovat útoky.

Existují nejméně dva způsoby, jak shlukovou analýzu využít pro detekci anomálií. Prvním způsobem je, použít pro učení nepopsaná data obsahující jak normální provoz, tak útoky. Druhým pak je použít k učení jen normální data a vytvořit tak profil normálního provozu. První způsob je založen na předpokladu, že anomálie či útoky tvoří jen velmi malou část dat, díky čemuž mohou být data klasifikována podle velikosti shluků. Velké shluky odpovídají normálnímu provozu, malé shluky či samostatné body (*outliers*) pak znamenají útoky.

Klíčovou záležitostí ve shlukové analýze je použitá míra vzdálenosti. Často používaná je běžná Euklidovská míra. V ní se ale každá vlastnost/dimenze započítává se stejnou vahou, což je v mnoha aplikacích nežádoucí, především když mají různé dimenze velmi rozdílný rozptyl, nebo mají různá měřítka. Proto se někdy používá také Mahalanobisova vzdálenost¹, která dává dimenzím různou váhu podle jejich rozptylu.

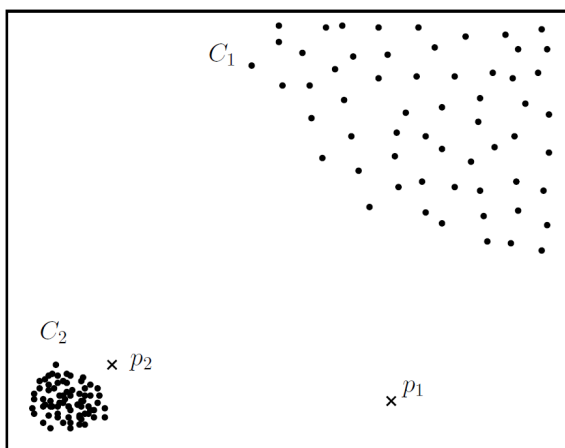
Jedním ze systémů pro detekci anomálií, který používá metody shlukové analýzy, je *Minnesota Intrusion Detection System* (MINDS) [27]. Ten přiřazuje každému spojení míru abnormalnosti na základě tzv. *Local Outlier Factor* (LOF). Metoda LOF nebere při ohodnocování bodu v prostoru v úvahu pouze vzdálenost od ostatních bodů, ale započítává také

¹Pokud máme shluk bodů a chceme-li vypočítat vzdálenost nějakého bodu x od středu μ tohoto shluku, je Mahalanobisova vzdálenost definována jako $D_M(x) = \sqrt{(x - \mu)^T S^{-1} (x - \mu)}$, kde S je kovarianční matice.

hustotu okolí. Pokud jsou body v okolí všechny velmi blízko u sebe, stačí pro označení bodu za anomálii menší vzdálenost k nejbližšímu jinému bodu, než když je okolí relativně řídké.

Pro každý bod je jeho hodnota LOF rovna poměru průměrné lokální hustoty jeho k nejbližších sousedů a lokální hustoty tohoto bodu samotného. Pro výpočet lokální hustoty bodu x je nejprve nalezena nejmenší hyperkoule se středem v x obsahující k jeho sousedů. Lokální hustota bodu je pak rovna hodnotě k vydělené objemem této hyperkoule. Pro body reprezentující normální data bude jejich lokální hustota podobná, jako hustota nejbližších sousedů, a jejich LOF tedy bude malý. Naopak lokální hustota bodu reprezentujícího anomálii (outlier) bude výrazně nižší, než lokální hustota jeho normálních sousedů, kteří tvoří nějaký shluk.

Výhoda použití metody LOF je znázorněna na příkladu na obrázku 2.2. Zde jsou dva shluky normálních hodnot, C_1 a C_2 , a dva samostatné body, p_1 a p_2 , které reprezentují anomálie. Uvažujme nejprve jednoduchý přístup, který by za anomálii označil každý bod, jehož vzdálenost ke k -tému nejbližšímu sousedovi je větší než nějaká mez. Tato mez by musela být nastavena na takovou hodnotu odpovídající vzdálenostem mezi body ve shluku C_1 . Bod p_1 by pak sice byl označen za anomálii správně, ale vzdálenost bodu p_2 od shluku C_2 je srovnatelná se vzdálenostmi mezi body v C_1 a proto bude bod p_1 označen za normální. Při použití LOF bude zohledněna různá hustota shluků C_1 a C_2 a bod p_2 bude označen správně [28].



Obrázek 2.2: Příklad situace, kdy je vhodné použít metodu Local Outlier Factor. [28]

2.3.3 Asociační pravidla

Vyhledávání asociačních pravidel je další z metod dolování dat, která se používá i v oblasti detekce anomálií v síťovém provozu. Je to jedna z mnoha technik popisujících, které události se často vyskytují společně.

Asociační pravidla můžeme definovat takto: Nechť I je množina binárních vlastností, D je množina transakcí T nazývaná *databáze*. Každá transakce má unikátní identifikátor a obsahuje množinu vlastností z I . Pravidlo je pak definováno jako implikace tvaru $X \Rightarrow Y$, kde $X, Y \subset I$ a $X \cap Y = \emptyset$.

Podpora (angl. *support*) $\text{supp}(X)$ množiny vlastností X udává poměr počtu transakcí obsahujících X ku všem transakcím v D . Pokud tedy např. 20% transakcí obsahuje všechny

položky v X , pak $\text{supp}(X) = 0, 2$. *Důvěra* (angl. *confidence*) pravidla $X \Rightarrow Y$ v databázi je definována jako $\text{supp}(X \cup Y) / \text{supp}(X)$. Vyjadřuje tedy, jak velká část transakcí obsahujících X obsahuje také Y .

Metoda asociačních pravidel byla několikrát použita pro vyhledání vzorů normálního chování při dolování dat pro detekci anomálií. Příkladem je systém *Audit Data Analysis and Mining* ADAM [29]. To je systém pro detekci anomálií pracující v reálném čase, který navíc obsahuje modul pro klasifikaci podezřelých událostí na falešné popluchy a skutečné útoky.

K objevení útoků používá ADAM kombinaci asociačních pravidel, dolování dat a klasifikace. Během trénovací fáze si ADAM vytváří databázi často se vyskytujícími množinami vlastností pomocí dat bez útoků. Potom se online sleduje provoz na síti a hledají se časté množiny vlastností v posledních n spojeních, které se porovnávají s těmi uloženými v databázi. Ty, které jsou považovány za podezřelé, jsou potom odeslány do klasifikátoru, který byl předtím naučen na klasifikování podezřelých událostí na známé útoky, neznámé útoky a falešné popluchy. Asociační pravidla jsou použita k získání potřebných znalostí o vstupních datech. Jakmile podpora nějaké množiny vlastností překročí určitou mez, je označena za podezřelou. Klasifikátor je učen pomocí trénovacích dat, které obsahují různé útoky, ale jejich poloha je předem známa.

Kapitola 3

NetFlow

Systémy pro detekci anomálií pracují s daty ze síťového provozu, což mohou být přímo hodnoty z hlaviček jednotlivých paketů, často se ale pracuje na úrovni tzv. *IP toků* (angl. *flow*). Tok je obvykle definován jako sekvence paketů se shodnou pěticí údajů: zdrojová a cílová adresa, zdrojový a cílový port a číslo protokolu. Ke každému toku jsou pak kromě těchto pěti údajů zaznamenávány další hodnoty, jako například čas zahájení komunikace, délka jejího trvání, počet přenesených paketů a bajtů a další.

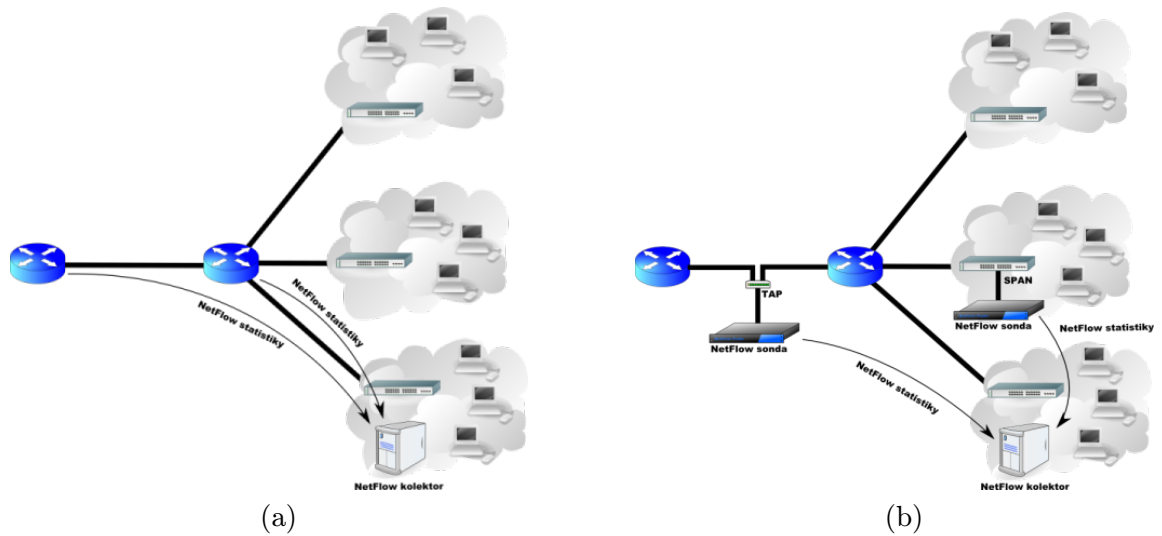
Sledování statistik na úrovni toků je pro detekci anomálií výhodné ze dvou důvodů. Zprv, získáváme navíc cenné informace o celých spojeních. Sice se zároveň některé informace ztrácejí (obsah paketů, některé hodnoty z hlaviček paketů), ty ale pro detekci anomálií obvykle nejsou tak relevantní, jako informace o toku jako celku. Navíc, pokud přesto některé hodnoty potřebujeme, lze do informací o toku snadno přidat nějaké jejich statistické shrnutí (např. průměrnou hodnotu a rozptyl). Druhým důvodem je pak výrazně menší množství paměti nutné k uchování informací o tocích, než by bylo potřeba k uchovávání všech paketů (i kdyby jsme ukládali jen hlavičky). Menší množství dat obvykle také znamená mnohem rychlejší zpracování.

NetFlow je otevřený protokol vyvinutý společností Cisco Systems, jehož hlavním účelem je právě sledování síťového provozu na základě IP toků. Tento protokol se stal standardem pro uchovávání informací o tocích a je používán prakticky ve všech monitorovacích systémech od různých společností.

3.1 Architektura

Architektura NetFlow se typicky skládá z několika exportérů a jednoho kolektoru. NetFlow exportér je zařízení, které je připojené k monitorované lince a analyzuje procházející pakety. Na základě zachycených toků generuje NetFlow statistiky, které odesílá kolektor. NetFlow kolektor je zařízení s velkou úložnou kapacitou, které sbírá statistiky z několika exportérů a ukládá je do databáze. Na těmito daty mohou běžet různé aplikace, které je umí efektivně vizualizovat a umožňují tak analyzovat provoz i běžnému uživateli, nebo je může využívat bezpečnostní systém založený na detekci anomálií.

Tradiční architektura NetFlow (obr. 3.1a) předpokládá, že roli exportérů zastupují směrovače, které kromě své hlavní funkce také počítají NetFlow statistiky. To je sice výhodné z toho hlediska, že není zapotřebí žádné další zařízení, ovšem směrovače, které toto podporují, bývají výrazně dražší. Navíc výpočet těchto statistik spotřebovává značnou část výkonu zařízení, proto se často na vstupu využívá vzorkování, tedy že se k výpočtu Net-



Obrázek 3.1: (a) Tradiční architektura NetFlow využívající jako exportéry směrovače. (b) Architektura NetFlow využívající specializované NetFlow sondy. [30]

Flow používá jen každý n -tý paket. Především pro bezpečnostní aplikace, které se snaží zachytit v ideálním případě všechny hrozby, může ale ztráta i jediného paketu představovat problém.

Proto se v poslední době často využívají i pasivní NetFlow sondy, což jsou zařízení specializovaná na monitorování a export NetFlow statistik (obr. 3.1b). Vzhledem k tomu, že tato zařízení nedělají nic jiného, nemusí používat vzorkování a přitom jsou poměrně levná. Navíc je lze připojit do libovolného bodu na síti a to zcela transparentně – sondy procházející data pouze sledují a nijak do nich nezasahují, pro odesílání statistik na kolektor používají dedikované linky. Navíc je na takovéto specializované sondě snazší provádět tzv. *Deep Packet Inspection*, tedy podrobnou analýzu nejen hlaviček paketů, ale i přenášených dat, což může poskytnout množství dalších užitečných informací.

Tato architektura NetFlow s pasivními sondami, je zvláště výhodná pro systémy detekce anomálií. Ty totiž obvykle musí v reálném čase zpracovávat obrovské množství dat, což je velmi výpočetně náročné a některé algoritmy jsou na vysokorychlostních sítích běžnými výpočetními prostředky nerealizovatelné. NetFlow sondy, což jsou často hardwarová zařízení s velkým výkonem na zpracovávání síťového provozu, by proto mohly být použity k nějakému druhu předzpracování dat pro algoritmus detekce anomálií, čímž by se výpočetní zátěž rozložila mezi více zařízení.

3.2 Protokol

Existuje několik verzí NetFlow protokolu, široce rozšířené jsou však jen dvě, verze 5 a 9. Na základě protokolu NetFlow v9 [31] v nedávné době vznikl standard IETF *Internet Protocol Flow Information eXport* [32], známý pod zkratkou IPFIX. Tento protokol se začíná hojně rozšiřovat a je pravděpodobné, že se v budoucnosti stane průmyslovým standardem [30].

NetFlow pouze definuje formát záznamů o tocích, neobsahuje žádný protokol pro konfiguraci spojení mezi exportéry a kolektorem. Záznamy jsou na kolektor odesílány protokoly UDP (User Datagram Protocol) nebo SCTP (Stream Control Transmission Protocol).

Každý NetFlow záznam odpovídá jednomu IP toku a obsahuje o něm různé statistické

údaje. Záznam ve verzi NetFlow v5 má pevně danou strukturu, obsahující následující položky [30]:

- Číslo verze
- Sekvenční číslo
- Číslo vstupního a výstupního rozhraní
- Čas začátku a konce toku
- Počet bajtů a paketů v toku.
- Údaje z IP hlavičky:
 - Zdrojové a cílové IP adresy
 - Zdrojové a cílové porty
 - IP protokol
 - Typ služby (Type of Service, ToS)
- U TCP toků obsahuje množinu všech TCP flagů, které se v toku vyskytly
- Směrovací informace

NetFlow v5 podporuje pouze protokol IPv4. NetFlow v9 má již flexibilní formát, který může obsahovat všechny položky z verze 5 a navíc ještě další volitelné položky, díky čemuž podporuje i jiné protokoly, například IPv6 nebo MPLS.

Kapitola 4

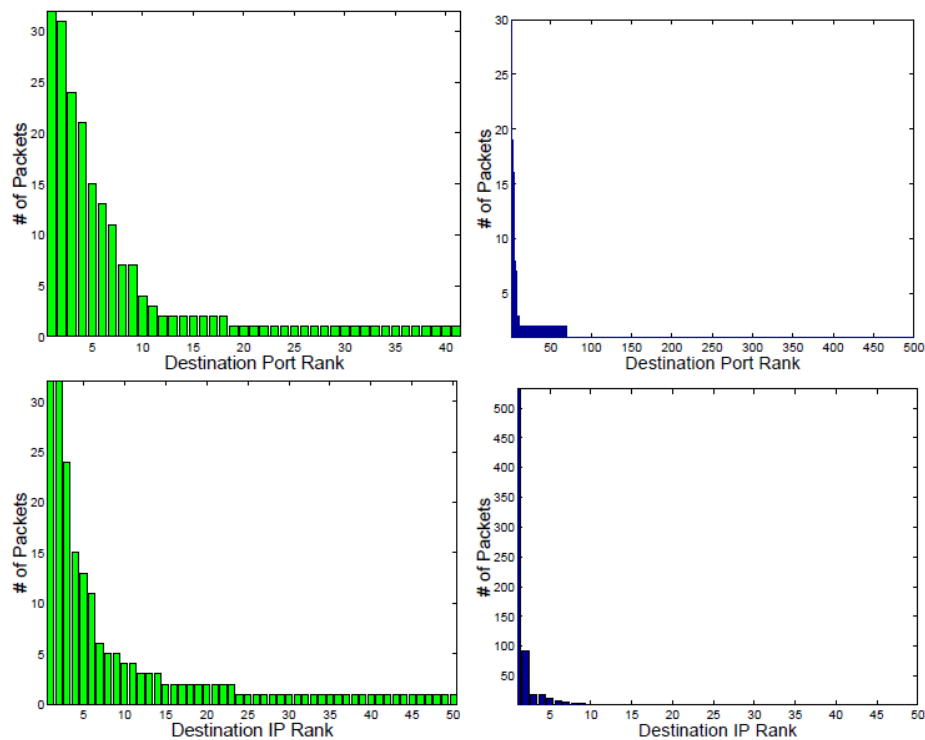
Popis vybrané metody

V dalším textu se budeme zabývat metodou detekce anomálií založenou na výpočtech entropie vlastností síťového provozu navrženou v práci od Lakhina et al. [3]. Jedná se o nepříliš starou metodu, která má poměrně dobré výsledky a byla už i použita v praxi například jako součást systému CAMNEP [33].

Tato metoda vychází z faktu, že přes celkovou rozdílnost různých druhů anomálií má většina jednu věc společnou: způsobují změny v distribuci hodnot z hlaviček paketů (zdrojové a cílové adresy a porty). Například typický DoS útok způsobí, že distribuce provozu podle cílové adresy bude více koncentrovaná na adresu oběti, naopak např. síťový sken hledající zranitelný port způsobí větší rozptýlení distribuce cílových adres a zároveň větší koncentraci cílových portů právě na skenovaný port. V tabulce 4.1 jsou uvedeny různé typy anomálií a položky, jejichž distribuce jsou těmito anomáliemi ovlivněny.

Typ anomálie	Popis	Hodnoty, jejichž distribuce je ovlivněna
Alfa tok	Tok mezi dvěma body s neobvykle velkým objemem	Zdrojová a cílová adresa (možno i porty)
DoS	Tzv. <i>Denial of Service</i> útok, tedy pokus o přetížení serveru velkým množstvím falešných požadavků	Zdrojová a cílová adresa
Flash Crowd	Neobvyklý nárůst provozu na jeden cíl z obvyklé množiny zdrojů. Způsobeno například velkým zájmem uživatelů o novou službu.	Cílová adresa, cílový port
Skenování portů	Pokusy o připojení na mnoho cílových portů na malém množství cílových adres	Cílová adresa, cílový port
Skenování sítě	Pokusy o připojení na několik cílových portů na mnoha cílových adresách	Cílová adresa, cílový port
Výpadky	Změny provozu způsobené poruchami nebo údržbou	Hlavně zdrojová a cílová adresa
Jeden na mnoho	Provoz z jednoho zdroje na mnoho cílů, např. distribuce obsahu	Zdrojová a cílová adresa
Síťoví červi	Skenování sítě pocházející od červů, hledajících zranitelné počítače	Cílová adresa, cílový port

Tabulka 4.1: Efekt různých anomálií na distribuce hodnot z hlaviček paketů. [3]



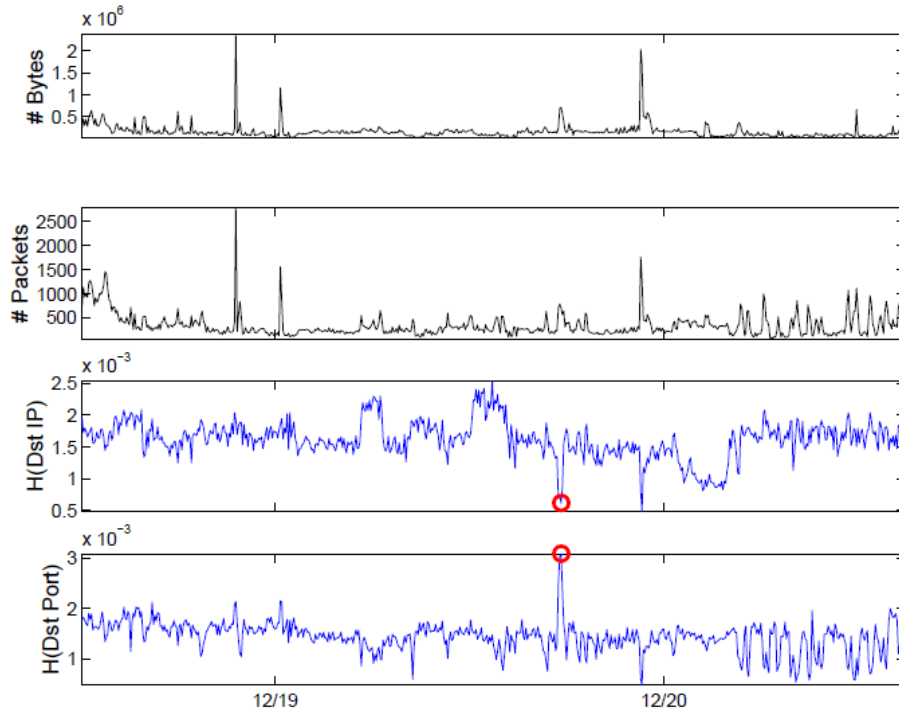
Obrázek 4.1: Změny v distribuci položek z hlaviček paketů. Vlevo: normální, vpravo: během anomálie [3]

V původní práci se pracuje jen se čtyřmi položkami z paketových hlaviček – zdrojovou a cílovou adresou a zdrojovým a cílovým portem, ale metoda je obecná a tuto čtveřici lze snadno rozšířit o další položky, pokud se ukáže, že je to z hlediska detekce útoků užitečné.

Na obrázku 4.1 je znázorněn příklad, jak se mění distribuce položek při skenování portů. V horní části je znázorněno rozložení cílových portů, v dolní části cílových adres. Každý graf ukazuje četnost výskytů daných hodnot během 5 minut, a to ve formě histogramu seřazeného od největší hodnoty k nejmenší. Grafy v levé části jsou vytvořeny z dat normálního provozu, v pravé části jsou grafy z doby, kdy na síti probíhalo skenování portů.

Z horních grafů lze vyčíst (pozor, grafy mají různá měřítka na ose x), že ačkoli četnost nejčastěji se vyskytujícího portu je v obou případech přibližně stejná, celkový počet zaznamenaných portů je během anomálie výrazně vyšší. To má za následek, že distribuce portů je během anomálie mnohem více rozptýlená, než za normálních podmínek. Opačný efekt nastává u cílových adres. Dolní grafy mají měřítka osy x stejné, to znamená, že množství různých adres je přibližně stejné, ale měřítka na ose y se liší. Je vidět, že nejvyšší četnost výskytu nějaké adresy je během anomálie asi 20x větší, než při normálním provozu, a distribuce je tedy při anomálii výrazně koncentrována k jedné hodnotě.

Z tohoto příkladu je tedy zřejmé, že pomocí analýzy distribuce vybraných položek z hlaviček paketů lze poměrně spolehlivě detekovat přinejmenším některé typy útoků. Bohužel, využití těchto vlastností pro konstrukci IDS systému není přímočaré. Distribuce položek je mnohorozměrný objekt a je tedy obtížné s ní pracovat přímo. Naštěstí však ve většině případů nepotřebujeme znát přesnou podobu distribuce, ale stačí nám nějaká míra jejího rozptýlení či koncentrace. Vy výše uvedeném příkladu nám k detekování i klasifikaci ano-



Obrázek 4.2: Časové průběhy počtu bajtů a paketů a entropie cílových adres a portů. Červeně je vyznačen okamžik, kdy probíhalo skenování portů. [3]

málie stačí vědět, že distribuce portů byla rozptýlena a zároveň distribuce adres byla koncentrována.

Ve své práci Lakhina et al. ukazují, že zvláště vhodná míra pro tento účel je *entropie vzorku (sample entropy)*. Nejdříve je nutné vytvořit histogram $X = \{n_i, i \in (1 \dots N)\}$, znamenající, že položka i se v daném vzorku dat objevila n_i -krát. Dále si označme celkové množství dat ve vzorku jako $S = \sum_{i=1}^N n_i$. Pak je entropie tohoto vzorku definována jako

$$H(X) = - \sum_{i=1}^N \frac{n_i}{S} \log_2 \frac{n_i}{S}.$$

Entropie vzorku leží v rozmezí od 0 do $\log_2 N$, přičemž nulové hodnoty nabývá v případě, že distribuce je maximálně koncentrována (tj. v celém vzorku se vyskytuje pouze jediná hodnota), maximální hodnoty nabývá v případě, že je distribuce maximálně rozptýlená (tj. všechny hodnoty se vyskytují stejně často). Tato entropie je tedy mírou, pomocí které se dají dobře zachytit změny v distribuci hodnot položek paketů.

Navíc, vzhledem k tomu, že maximální míra entropie závisí na N , tedy na množství různých hodnot ve vzorku, má entropie tendenci stoupat s velikostí vzorků, tedy s rostoucím provozem na síti. To má za následek, že anomálie projevující se výraznou změnou objemu provozu se projeví i změnou entropie a jsou tak touto metodou také zachytitelné.

Detekce anomálií pomocí této metody tedy probíhá tak, že se pro každou dimenzi samostatně (dimenzemi se zde rozumí zdrojové a cílové adresy a porty) vytvoří histogram jejich hodnot v daném vzorku dat (záznamy o IP tocích, exportované obvykle v pětiminutových intervalech). Z těchto histogramů se vypočítá entropie vzorku v každé dimenzi. Pomocí

statistických metod se pak na základě historie odhadují očekávané nové hodnoty entropie a pro každou dimenzi je určena odchylka od této očekávané hodnoty. Pokud jsou odchylky výrazné a především pokud tomu tak je ve více dimenzích současně, je hlášena anomálie.

Příklad využití entropie pro detekci anomálií je uveden na obrázku 4.2. Jsou zde uvedeny časové průběhy různých metrik síťového provozu v době, kdy došlo ke skenování portů, jehož histogramy jsou uvedeny výše. Horní dva grafy ukazují množství bajtů a paketů procházejících sítí. Spodní grafy pak ukazují entropii cílových adres a portů. Přesný čas anomálie je označen kroužky ve spodních grafech. Z tohoto obrázku je mimo jiné vidět výhoda této metody proti obyčejnému sledování objemu provozu. Pouze sledováním objemu by bylo velmi těžké takovouto anomálii najít, naproti tomu v údajích o entropii je anomálie dobře rozpoznatelná jako prudký pokles entropie cílových adres spolu s nárůstem entropie cílových portů.

Autoři v článku dále ukazují, že hodnoty entropie položek paketů lze kromě detekce anomálií využít také pro jejich klasifikaci do tříd podle typu útoku. Je využito toho, že každý typ útoku se projevuje v jiných dimenzích a jiným způsobem. Každou položku tedy budeme uvažovat jako jednu dimenzi n -rozměrného prostoru, do nějž vyneseme anomálie jako body, jejichž souřadnice budou odpovídat odchylkám entropie dané položky/dimenze od normálu. Na tato data pak lze aplikovat nějaký běžný shlukovací algoritmus, který anomálie roztřídí do skupin, z nichž každá odpovídá nějakému typu útoku (případně jiné anomálii, která není útokem).

Kapitola 5

Optimalizace metody

V následujících podkapitolách budou navrženy dvě změny výše uvedené metody. První z nich zajišťuje zvýšení rychlosti výpočtu a úsporu paměti použitím hashovací funkce, druhá je zaměřena na zlepšení detekčních schopností metody nalezením vhodnějších dimenzí a jejich kombinací a použitím i jiných metrik, než je entropie.

5.1 Zrychlení algoritmu

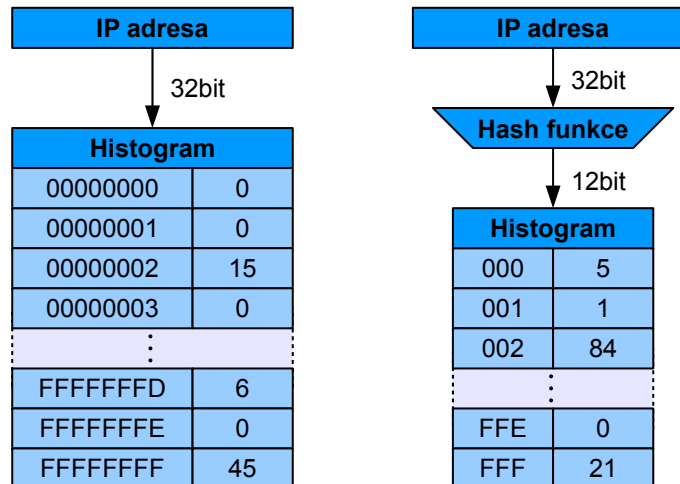
Doba výpočtu ani množství spotřebované paměti není u tohoto algoritmu na dnešních výkonných počítačích příliš problém. Přesto má význam se těmito vlastnostmi zabývat, a to jednak proto, že rychlost sítí, a tedy i množství dat jimi procházejících, dlouhodobě roste výrazně rychleji než výkon počítačů, a také pro případ, že by tento algoritmus měl pracovat v nějakém vestavěném zařízení s malým výkonem a omezenou pamětí. Také je nutno počítat s tím, že nemusí být zdaleka jediným výpočtem, který bude dané zařízení provádět.

Co se týče doby výpočtu, měření ukázala, že náročnější než samotný výpočet entropie je vytvoření histogramu, pomocí něž se entropie počítá. Složitost výpočtu entropie totiž roste jen s počtem sloupců v histogramu, tedy s počtem unikátních hodnot v dané dimenzi a intervalu, ovšem při vytváření histogramu se musí zpracovat všechny síťové toky, kterých mohou být na vysokorychlostní síti v několikaminutovém intervalu i miliony.

Hlavní problém je následující: Vytvořit histogram znamená pro každou vyskytující se hodnotu dané položky uložit počet jejích výskytů. Nabízí se použití obyčejného pole čísel indexovaného sledovanými hodnotami. To je ovšem možné jen v případě dimenzí s malým počtem možných hodnot (např. protokol či porty), u jiných dimenzí by to vedlo na obrovské množství spotřebované paměti (např. pro adresy IPv4 by bylo potřeba pole s 2^{32} položkami, pro IPv6 dokonce 2^{128} , naprostá většina položek by přitom zůstala nevyužita). Pro takové dimenze je tedy nutné použít jinou datovou strukturu – nějaký typ asociativního pole, které dokáže paměť využít mnohem efektivněji. Ovšem vyhledávání položek v asociativním poli bude vždy výrazně pomalejší.

Navrhují následující řešení. Pro uložení histogramu se použije obyčejné pole, nebude ale indexované přímo zaznamenávanou hodnotou, ale hodnotou získanou z ní pomocí nějaké hashovací funkce. Výstup hashovací funkce by měl mít mnohem méně bitů, než původní hodnota, díky čemuž stačí pole výrazně menší. Princip je znázorněn na obrázku 5.1.

Při použití hash funkce a zvláště při její malé vstupní šířce mohou pochopitelně vznikat kolize. Výsledky experimentů uvedené dále však ukazují, že pokud není počet bitů výstupu hash funkce až příliš malý, kolize nemají na výsledky výrazný vliv.



Obrázek 5.1: Vlevo: Použití obyčejného pole pro uložení četnosti výskytů jednotlivých hodnot. Vpravo: Navržená optimalizace, zmenšení velikosti pole využitím hash funkce pro převod hodnoty na nižší počet bitů.

Výhodou tohoto řešení je tedy poměrně malá a navíc předem známá a volitelná spotřeba paměti a za předpokladu, že použitá hashovací funkce nebude příliš výpočetně náročná, i velká rychlost.

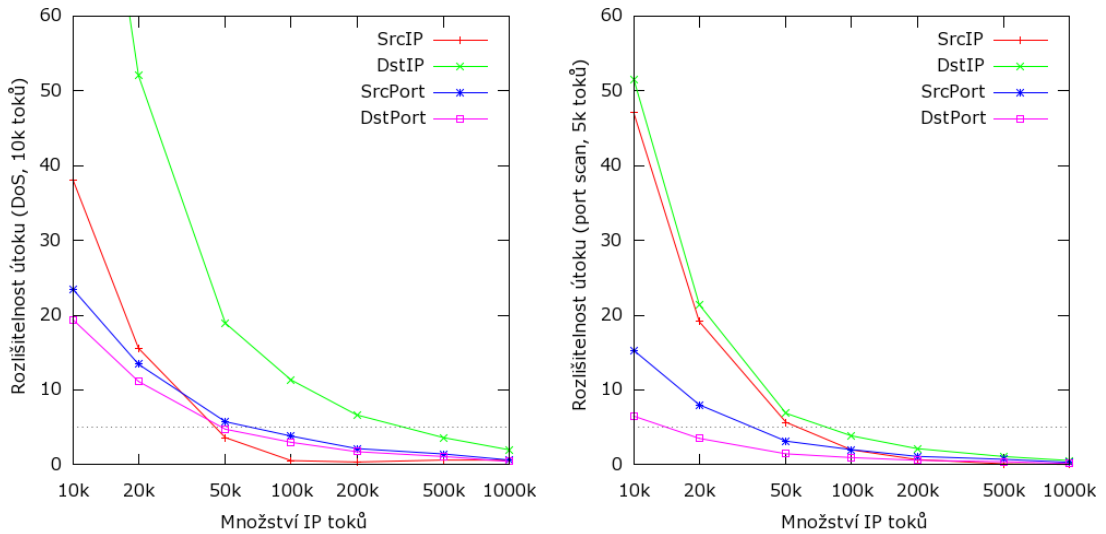
5.2 Zvýšení detekční citlivosti

Metoda, tak jak byla původně popsána v [3], tedy používající entropii zdrojových a cílových adres a zdrojových a cílových portů, detekuje útoky dobře pouze pokud je provoz na síti poměrně malý (resp. pokud útok činí relativně velkou část celkového provozu). Ovšem na moderních vysokorychlostních sítích, kde se objevují statisíce IP toků za minutu, je již relativní změna způsobená běžným útokem pro tuto metodu příliš malá.

Na obrázku 5.2 jsou uvedeny grafy ukazující závislost rozeznatelnosti útoku na množství provozu. Graf vlevo je pro DoS útok s 10 000 toků, vpravo pak skenování portů několika počítačů, celkem 5 000 toků. Na vodorovné ose je vyneseno množství toků normálního provozu. Na svislé ose je poměr změny entropie způsobené útokem ku jejímu běžnému kolísání v daném vzorku dat (tj. míra rozlišitelnosti útoku podrobněji definovaná dále v kapitole 7.1). Za dobře rozpoznatelný lze útok v dané dimenzi považovat tehdy, pokud tato míra překročí hodnotu alespoň 5.

Z grafů je patrné, že čím větší je normální provoz, tím hůře jsou anomálie detekovatelné. V uvedených příkladech křivky klesají pod hranici rozlišitelnosti přibližně v okamžiku, kdy se množství toků normálního provozu a anomálie liší o jeden řád. Jinými slovy, aby byla anomálie touto metodou detekovatelná, musí tvořit alespoň přibližně 10 % celkového provozu. To na vysokorychlostních sítích znamená, že metoda dokáže detekovat jen velmi masivní útoky. Je tedy třeba nějak zvýšit její citlivost.

Už v původním článku autoři naznačili, že je možné použít i jiné vlastnosti toků, než jen adresy a porty, byla to ovšem jen teoretická možnost a žádné výsledky s jinými dimenzemi nebyly prezentovány. Rozhodl jsem se proto vyzkoušet, které další vlastnosti by mohly být vhodné pro detekci různých útoků.



Obrázek 5.2: Závislost rozlišitelnosti útoku na množství provozu. Vlevo: DoS útok o 10 000 tocích. Vpravo: Skenování portů o 5 000 tocích.

Kromě toho existuje také možnost všechny dimenze různě kombinovat a vytvářet tak dimenze nové. Například sledování četnosti výskytů kombinací cílové adresy a cílového portu může poskytnout jiné informace než sledování každé z těchto vlastností zvlášť.

Další možností, jak potenciálně zvýšit citlivost metody, je použít pro vyjádření změn v distribucích jinou míru, než je entropie. Autoři sice v původní práci uvádějí, že vyzkoušeli více metrik a entropie se ukázala jako nejlepší, ale toto tvrzení není ničím podloženo, ani není uvedeno, které jiné metriky byly testovány. Proto jsem se rozhodl navrhnout a otestovat i několik jiných metrik a zjistit, které jsou nejlepší pro které dimenze a typy útoků.

Pokud při anomálii dochází ke koncentraci histogramu, znamená to, že se v něm začne objevovat jedna nebo několik málo výrazných špiček. K detekci takových špiček je zapotřebí míra, u které mají vysoké hodnoty na výsledek mnohem větší vliv než hodnoty menší. Takovou mírou je *suma čtverců*, neboli součet druhých mocnin všech hodnot. Aby tato metrika nebyla tolik závislá na množství provozu, ale jen na rozložení jeho hodnot, vydělíme ji ještě celkovým počtem toků. Sumu čtverců pak tedy definujeme jako

$$M_2(X) = \frac{1}{S} \sum_{i=1}^N n_i^2$$

kde, stejně jako u výše uvedené definice entropie vzorku, N je počet hodnot histogramu, n_i je jeho i -tá položka, tedy počet výskytů hodnoty i , a S je počet IP toků ve vzorku dat, či také $S = \sum_{i=1}^N n_i$.

Podobně pro detekci rozptýlení histogramu potřebujeme míru, u které budou mít velmi malé hodnoty větší vliv na výsledek, než hodnoty velké. Takovou vlastnost má například *suma převrácených hodnot odmocnin*. Když ji opět podělíme součtem všech hodnot histogramu, abychom se zbavili vlivu aktuálního množství provozu, bude definována jako

$$M_{-\frac{1}{2}}(X) = \frac{1}{S} \sum_{i=1}^N \frac{1}{\sqrt{n_i}} = \frac{1}{S} \sum_{i=1}^N n_i^{-\frac{1}{2}}$$

Dále je možné, že se ukáže, že je pro detekci vhodnější místo dělení sum celkovým počtem toků, S , dělit je počtem unikátních hodnot vyskytujících se ve vzorku dat, tedy počtem nenulových hodnot v histogramu. Označme tuto hodnotu jako N' a definujme ji jako

$$N' = \sum_{i=1}^N \text{nonzero}(n_i), \quad \text{nonzero}(x) = \begin{cases} 0 & \text{pokud } x = 0 \\ 1 & \text{jinak} \end{cases}$$

Pak můžeme definovat další varianty sumy čtverců a sumy převrácených hodnot odmocnin:

$$M'_2(X) = \frac{1}{N'} \sum_{i=1}^N n_i^2$$

$$M'_{-\frac{1}{2}}(X) = \frac{1}{N'} \sum_{i=1}^N n_i^{-\frac{1}{2}}$$

Užitečnou mírou pro detekci rozptýlení histogramu by však mohl být i samotný počet unikátních hodnot, definujeme tedy poslední míru:

$$M_0(X) = N'$$

Navrhli jsme tedy 5 různých metrik, které by mohly být vhodné pro detekci anomálií. Všechny tyto metriky jsou sice navrženy pouze pro detekci koncentrace, nebo pouze rozptýlení histogramu, přičemž původní entropie dokáže detekovat obojí, ale použití více metrik zároveň není zásadní problém, zvlášť pokud to umožní zvýšit citlivost metody.

Všechny navržené metriky spolu s entropií budou otestovány na různých dimenzích a s různými anomáliemi, aby bylo experimentálně zjištěno, jaké kombinace dimenzí a metrik jsou dobře použitelné pro detekci anomálií.

Kapitola 6

Testovací software

Uvedená metoda, včetně navržených optimalizací, byla implementována v jazyce C++. Program je určen především pro testování optimalizace pomocí hashovací funkce a pro průzkum, které dimenze a metriky jsou pro detekci útoků nejvhodnější. Není to tedy kompletní systém detekce anomálií určený pro reálné nasazení, přestože mnoho jeho částí by šlo při návrhu takového systému snadno použít.

Program ze standardního vstupu načítá záznamy o síťových tocích v jednoduchém textovém formátu. Pomocí parametrů mu jsou předány informace o tom, které dimenze chceme počítat, jakou metriku na nich použít, případně jestli v dané dimenzi chceme použít hashování a na kolik bitů. Program pak načítané záznamy rozděljuje podle časových značek do intervalů specifikované délky, v každém intervalu vytvoří z hodnot sledovaných dimenzí histogramy a z nich vypočítá požadovanou metriku. Výstupem je tabulka, ve které jeden řádek odpovídá jednomu intervalu a obsahuje čas začátku intervalu, počet toků v tomto intervalu a vypočítané hodnoty metrik ve všech dimenzích.

Programu je možné předat i název souboru s uloženými toky nějaké anomálie. V takovém případě jsou toky ze standardního vstupu považovány za normální provoz a po vypočítání metrik každého intervalu jsou do histogramů přidány ještě data anomálie a metriky jsou vypočítány znovu pro data s přidanou anomálií. Výstupem jsou pak dvě tabulky, jedna s daty ze standardního vstupu (normální provoz) a druhá s hodnotami po vložení toků anomálie do každého intervalu. Souborů s anomáliemi může být specifikováno i více, pak získáme jednu tabulku pro každou anomálii.

Dále je možné nechat vypsát základní statistické údaje pro všechny dimenze: průměr a směrodatnou odchylku normálního provozu a v případě přidání anomálie ještě průměr a směrodatnou odchylku dat s anomálií a také průměrnou, minimální a maximální odchylku od normálních dat.

6.1 Formát vstupních dat

Vstupní data jsou textová, každý řádek je záznam jednoho IP toku v tomto formátu:

datum_a_čas_zacátku_toku	číslo_protokolu	zdrojová_adresa	zdrojový_port	
cílová_adresa	cílový_port	počet_paketů_v_toku	počet_bytů_v_toku	tcp_flagy

Příklad vstupu s několika záznamy:

2011-03-31	11:07:28.674	6	147.229.2.85	443	85.162.65.122	50019	15	7198	.AP.S.
2011-03-31	11:07:28.776	6	147.229.145.43	52365	63.242.149.90	80	6	795	.AP.SF
2011-03-31	11:07:28.929	17	147.229.205.141	1513	86.22.141.59	1024	1	44

V případě uložení dat ve standardním formátu programu nfdump lze data získat v požadovaném formátu pomocí parametrů:

```
nfdump -o "fmt:%ts %pr %sa %sp %da %dp %pkt %byt %flg" -N -q
```

6.2 Parametry a výstup programu

Program je konzolová aplikace a veškeré ovládání se provádí pomocí spouštěcích parametrů (detaily viz příloha B. Ty výrazně ovlivňují jeho výstup.

V nejjednodušší variantě bez parametrů vypisuje program jeden řádek, na kterém je uveden čas začátku prvního toku, počet toků a vypočítané metriky všech definovaných dimenzí (standardně entropie zdrojové a cílové adresy a zdrojového a cílového portu).

```
09:11:01 270593 9.34903 12.58467 12.28253 7.23352
```

Záznamy IP toků je možno pomocí parametru `-i` *sekundy* rozdělit do časových intervalů podle jejich časových značek (data na vstupu musí být seřazena). V tom případě je pak pro každý interval zvlášť vypočítána entropie a vypsán jeden řádek.

Pomocí parametru `-d` lze definovat použití jiných dimenzí a metrik, případně použití hashovací funkce, ve výstupu je pak jeden sloupec hodnot pro každou dimenzi.

Pro větší přehlednost je možné parametrem `-t` zapnout vypsání popisků jednotlivých sloupců tabulky.

Time	Flows	sa#12/ent	da#16/ent	dp+lpkt/sos	flg/sos	sp/rsqrt	dp/rsqrt
10:07:07	1383750	10.54983	12.47009	1.4701e+04	4.7631e+05	1.8709e-02	1.8914e-02
10:12:07	1480777	10.53244	12.41065	1.5638e+04	5.0872e+05	1.6929e-02	1.7125e-02
10:17:07	1524623	10.53879	12.45395	1.6959e+04	5.1127e+05	1.6328e-02	1.6471e-02
10:22:07	1457885	10.53029	12.48242	1.5842e+04	4.9928e+05	1.7283e-02	1.7443e-02
10:27:07	28914	10.24759	11.23688	3.6323e+02	1.0738e+04	4.0472e-01	3.9472e-01

V případě, že mají být do dat vloženy anomálie ze zvláštního souboru, je takovýchto tabulek vypsáno více, první obsahuje hodnoty vypočtené z normálních dat, každá další tabulka pak hodnoty z dat s vloženou anomálií. Před každou tabulkou s daty anomálie je jeden prázdný řádek a řádek s názvem souboru anomálie.

Parametrem `-s` lze zapnout výpočet základních statistických údajů (shrnutí) přes všechny intervaly. V takovém případě za tabulkami následují pro každou dimenzi ještě hodnoty průměru a směrodatné odchylky normálních dat a všech anomálií. U anomálií je ještě uvedena průměrná a minimální změna, kterou daná anomálie způsobila, a poměr těchto změn ku směrodatné odchylce normálního provozu (tj. míra rozlišitelnosti útoku definovaná dále v kapitole 7.1).

První a poslední interval často obsahuje menší množství záznamů než ostatní, proto je možné je pomocí parametru `-g` vyloučit z výpočtu statistik.

Následuje příklad výstupu programu při rozdělení dat do dvouminutových intervalů, použití 5 dimenzí, vložení 2 anomálií a výpisu statistik.

Time	Flows	sa/ent	da/ent	sp/ent	dp/ent	dp+lpkt/sos
09:07:18	253480	9.37204	12.54021	12.28272	7.24390	2.2288e+04
09:12:18	270224	9.35340	12.58377	12.28028	7.24651	2.2878e+04
09:17:18	287595	9.33204	12.64900	12.25034	7.17394	2.4845e+04
09:22:18	9299	8.57050	11.03438	10.98487	5.98463	8.9287e+02
scan_10k_out.flow						
09:07:18	263480	9.24916	12.42302	12.46816	7.57614	2.1443e+04
09:12:18	280224	9.24177	12.47526	12.45877	7.56209	2.2063e+04
09:17:18	297595	9.23061	12.54760	12.42052	7.47674	2.4012e+04
09:22:18	19299	5.12865	8.03506	12.93060	9.04621	4.3569e+02
dos_out_100k_10.flow						
09:07:18	353480	8.51941	9.85194	13.48709	5.40708	4.8632e+04
09:12:18	370224	8.56587	10.02643	13.45604	5.49185	4.8476e+04
09:17:18	387595	8.60506	10.20926	13.39283	5.51536	4.9323e+04
09:22:18	109299	4.18827	1.35861	14.74543	0.66269	9.2172e+04
=== Summary ===						
Normal data:						
Average		9.15700	12.20184	11.94955	6.91224	1.7726e+04
StDev		0.33891	0.67515	0.55711	0.53635	9.7647e+03
Anomaly (scan_10k_out.flow):						
Average		8.21255	11.37024	12.56951	7.91529	1.6988e+04
StDev		1.78050	1.92608	0.20924	0.65404	9.6035e+03
AvgDiff		-0.94445	-0.83161	0.61996	1.00305	-7.3779e+02
MinDiff		-0.10143	-0.10140	0.17018	0.30280	-4.5718e+02
AvgDiff/NormStDev		-2.79	-1.23	1.11	1.87	-0.08
MinDiff/NormStDev		-0.30	-0.15	0.31	0.56	-0.05
Anomaly (dos_out_100k_10.flow):						
Average		7.46966	7.86156	13.77035	4.26925	5.9650e+04
StDev		1.89475	3.75660	0.56399	2.08263	1.8779e+04
AvgDiff		-1.68734	-4.34028	1.82080	-2.64300	4.1924e+04
MinDiff		-0.72698	-2.43974	1.14250	-1.65857	2.4477e+04
AvgDiff/NormStDev		-4.98	-6.43	3.27	-4.93	4.29
MinDiff/NormStDev		-2.15	-3.61	2.05	-3.09	2.51

Detailní popis parametrů programu včetně seznamu podporovaných dimenzí a metrik je uveden v příloze [B](#).

6.3 Příklady použití

Data jsou načítána ze standardního vstupu, předpokládá se tedy buď přesměrování souboru obsahujícího data ve vhodném formátu na standardní vstup, nebo spojení s programem `nfdump` pomocí roury. Následuje několik příkladů použití programu:

```
$ ./entropy < data.flow

$ ./entropy -i 120 -t -d "sa#12 da#16 sp dp da+dp#16/sos lpkt/sos" -a attack.flow
< data.flow

$ alias mynfdump='nfdump -o "fmt:%ts %pr %sa %sp %da %dp %pkt %byt" -N -q'
$ mynfdump -R nfcapd.201102170920 | ./entropy -t -s -D dims.txt -a attack.flow

$ mynfdump -R nfcapd.201102171200:nfcapd.201102171255 -m | ./entropy -t -s -i 300
-g 1,1 -D dims.txt -a anomaly1.flow -a anomaly2.flow
```

Kapitola 7

Výsledky

K testování byly použity záznamy o IP tocích z bodu připojení univerzitní sítě do internetu. Jedná se o linku s provozem až 0,5 Gb/s a s počtem toků pohybujícím se mezi 1 a 2 miliony v každém z pětiminutových intervalů, v nichž jsou data o tocích exportována. Tato data jsou považována za normální provoz, který neobsahuje anomálie, a anomálie jsou uměle generovány a do těchto dat vkládány.

Testování probíhá tak, že je v každém časovém intervalu a v každé dimenzi vypočítána nejdříve entropie (či jiná metrika) dat pouze normálního provozu a poté normálního provozu, do kterého byla přidána data nějakého útoku. Díky tomu víme pro každý interval, jak by se změnila entropie jednotlivých dimenzí, kdyby právě v tomto intervalu probíhal útok.

7.1 Míra rozlišitelnosti

Použitý program je zaměřen pouze na generování histogramů a počítání entropie, v reálné aplikaci by za tímto musela následovat ještě nějaká statistická metoda, která by ve vypočítaných datech hledala odchylky od normálu a tak detekovala anomálie. Ať už by tato metoda byla jakákoliv, čím větší odchylku entropie, v porovnání s jejím běžným kolísáním, útok způsobí, tím snáze bude tento útok detekován.

Pro účely porovnání různých variant metody tedy definujeme *míru rozlišitelnosti anomálie* σ jako poměr absolutní hodnoty změny entropie či jiné metriky způsobené anomálií ku směrodatné odchylce této metriky při normálním provozu. Matematicky zapsáno $\sigma = |\Delta|/\delta$, kde Δ je změna hodnoty metriky způsobená anomálií a δ je směrodatná odchylka normálního provozu. Čím větší hodnoty tato míra dosáhne, tím lépe. Konkrétní hodnotu, od které lze považovat anomálii za dobře detekovatelnou, nelze jednoznačně určit, řekněme však, že minimum je cca hodnota 3, lépe však alespoň 5.

7.2 Optimalizace pomocí hashovací funkce

V tabulce 7.1 jsou uvedeny výsledky původní verze a výsledky optimalizované verze s různou šířkou výstupu hash funkce. Vždy je zde pro každou dimenzi uvedena průměrná hodnota entropie (μ), její směrodatná odchylka (δ), průměrná změna entropie způsobená DoS útokem (Δ_{dos}) a poměr mezi touto změnou a směrodatnou odchylkou ($|\Delta_{dos}|/\delta$), tedy výše definovaná míra rozlišitelnosti útoku.

		SrcIP	DstIP	SrcPort	DstPort
Původní verze	μ	9,948	12,712	12,589	6,549
	δ	0,199	0,056	0,085	0,255
	Δ_{dos}	-0,781	-2,222	0,925	-1,388
	$ \Delta_{dos} /\delta$	3,917	34,207	10,859	5,444
16-bit hash	μ	9,711	12,460	10,961	5,848
	δ	0,183	0,060	0,073	0,203
	Δ_{dos}	-0,726	-2,162	0,716	-1,222
	$ \Delta_{dos} /\delta$	3,970	36,314	9,788	6,010
12-bit hash	μ	9,520	10,946	10,650	5,615
	δ	0,175	0,036	0,055	0,190
	Δ_{dos}	-0,687	-1,805	0,448	-1,166
	$ \Delta_{dos} /\delta$	3,929	50,102	8,216	6,161
8-bit hash	μ	7,440	7,858	7,634	4,260
	δ	0,102	0,011	0,016	0,126
	Δ_{dos}	-0,301	-1,093	0,127	-0,846
	$ \Delta_{dos} /\delta$	2,955	108,456	8,121	6,741
4-bit hash	μ	3,926	3,982	3,951	2,556
	δ	0,026	0,003	0,004	0,060
	Δ_{dos}	0,000	-0,357	0,020	-0,452
	$ \Delta_{dos} /\delta$	0,011	113,485	5,082	7,573

Tabulka 7.1: Rozlišitelnost útoku v závislosti na použití hash funkce a její šířce.

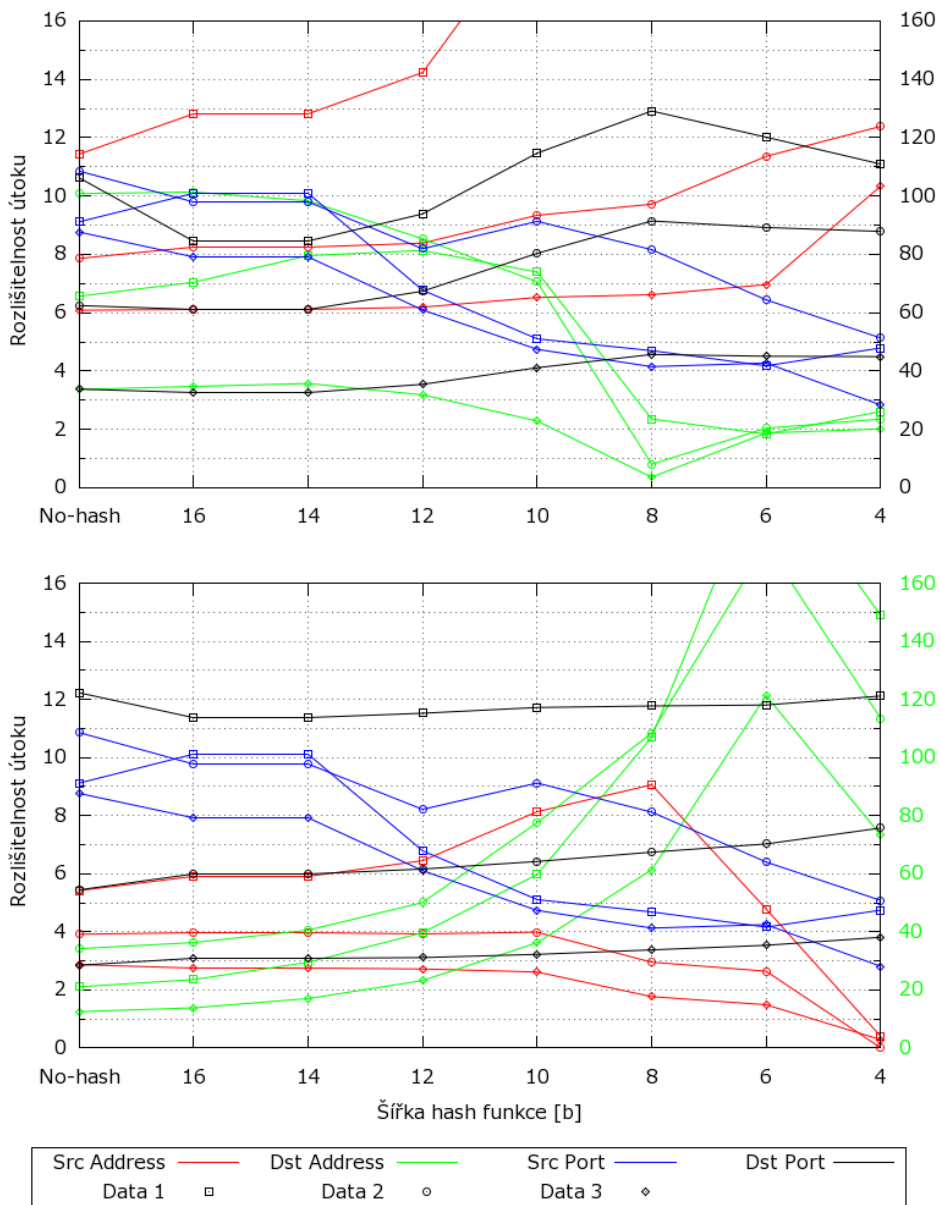
Více informací je uvedeno v grafech na obrázku 7.1, kde je vynesena rozlišitelnost útoku pro různou šířku hash funkce a tři různé vzorky dat. Jedná se o dvouhodinové vzorky reálných dat, do kterých byly vloženy dva vygenerované útoky – odchozí DoS útok (10 zdrojových adres, z každé je během pětiminutového intervalu odesláno 10 000 jednopaketových toků na jednu cílovou adresu a port oběti) a skenování portů (skenování prvních 1024 portů na cca 100 počítačích). V tabulce 7.1 jsou uvedeny konkrétní hodnoty pro skenování portů a vzorek dat označený jako data 2.

Při skenování portů dosahuje rozlišitelnost podle cílových adres výrazně větších hodnot než ostatní dimenze a má proto v grafu vlastní osu (zeleně vpravo).

Z tabulky a z grafů je patrné, že zavedení hashovací funkce nijak výrazně neovlivní výsledky, a to v mnoha případech ani při snižování počtu bitů na velmi malé hodnoty. U některých dimenzí se snižováním počtu bitů rozlišitelnost dokonce zvyšuje (obvykle u dimenzí, kde daný útok způsobuje výrazně častější výskyt jedné hodnoty). Přesto, pokud je k dispozici dostatek paměti, doporučuji použít větší počet bitů, a to jednak proto, že čím méně bitů, tím více výsledky závisí na náhodě, ale také proto, že pro detekci různých typů útoků je obvykle potřeba detekovat i rozptýlení histogramu, a v tom případě dochází při malém počtu bitů spíše ke snížení rozeznatelnosti.

V tabulce 7.2 je uvedena doba výpočtu¹ původní varianty algoritmu a optimalizované varianty s různými šířkami výstupu hash funkce. Jsou uvedeny časy pro tři různé velikosti vstupních dat. Z tabulky je patrné, že zavedením hash funkce lze dosáhnout velmi výrazného zrychlení, přičemž čím více dat se zpracovává, tím je dosažené zrychlení větší.

¹Je měřena skutečně jen doba vytváření histogramů a výpočet entropie, celková doba běhu programu je výrazně větší kvůli načítání vstupu v textovém formátu.



Obrázek 7.1: Grafy závislosti rozlišitelnosti útoku na použití hashovací funkce a šířce jejího výstupu. Nahoře: Skenování portů, Dole: DOS útok

Varianta	0,5 mil. toků		2 mil. toků		11 mil. toků	
	Doba výpočtu [s]	Zrychlení	Doba výpočtu [s]	Zrychlení	Doba výpočtu [s]	Zrychlení
Původní	0,66	—	3,07	—	21,11	—
16-bit hash	0,06	11,0	0,22	13,9	1,24	17,0
12-bit hash	0,05	13,2	0,19	16,1	1,12	18,8
8-bit hash	0,04	16,5	0,18	17,1	1,08	19,5
4-bit hash	0,04	16,5	0,16	19,2	1,00	21,1

Tabulka 7.2: Tabulka rychlosti výpočtu algoritmu v závislosti na použití hashovací funkce a na množství zpracovávaných dat.

Také nižší počet bitů výstupu hash funkce znamená kratší dobu výpočtu a tedy větší zrychlení, ovšem tento rozdíl není nijak zvlášť velký a snaha snižovat počet bitů tak má význam především z hlediska paměťové náročnosti.

Celkově lze tuto optimalizaci shrnout tak, že umožňuje zpracovávat data s miliony toků při použití pouze 1 MB paměti (4 dimenze, v každé 16-bit hash, tzn. 65k položek, každá položka 4 byty) a přitom cca 10–20× rychleji než původní algoritmus. Navíc pokud nevádí mírné zhoršení detekčních schopností algoritmu, lze paměťovou náročnost snížit ještě mnohem více. Tyto vlastnosti činí optimalizovanou metodu vhodnou i pro malá vestavěná zařízení s velmi omezenou pamětí a výpočetním výkonem.

7.3 Hledání jiných dimenzí a metrik

Pro otestování, které dimenze a jejich kombinace a které metriky jsou vhodné pro detekci anomálií, a případně jakých, bylo vybráno cca 50 dimenzí a nad každou bylo vypočítáno všech šest výše uvedených metrik.

K testování byly použity tři hodinové vzorky dat rozdělené do pětiminutových intervalů. Do těchto dat byly vkládány následující útoky/anomálie:

- Denial of Service (DoS) – Jednoduchý distribuovaný SYN flood útok. Pakety tečou z 10 zdrojových adres uvnitř sledované sítě na jednu cílovou adresu a port v internetu. Jedná se tedy o odchozí útok simulující situaci, kdy je několik počítačů uvnitř sítě součástí větší skupiny počítačů ovládaných útočníkem (tzv. botnet). Každý tok obsahuje právě jeden TCP paket s nastaveným SYN flagem. Byly simulovány různé rozsáhlé útoky, obsahující celkem 5, 10, 20, 50 a 100 tisíc paketů.
- DoS s podvrženými zdrojovými adresami – Stejně jako předchozí, ale zdrojové adresy jsou náhodně generovány.
- Skenování portů – Skenování prvních 1024 portů na několika počítačích. Všechny toky mají stejnou zdrojovou adresu a cílový port, je několik cílových adres a zcela náhodné zdrojové porty. Každý tok obsahuje právě jeden TCP paket s nastaveným SYN flagem. Různé mohutnosti útoku – 5, 10, 20, 50 a 100 tisíc paketů.
- Výpadek serveru – Z dat normálního provozu jsou odfiltrovány všechny toky s IP adresou některého z nejpoužívanějších serverů, čímž je simulován jeho výpadek. Celkem bylo odstraněno cca 1000 toků v každém intervalu pro server 1 a 13 tisíc toků pro server 2.
- Reálná anomálie – V reálných datech byla objevena asi hodinu trvající anomálie – přibližně 10 tisíc toků každých 5 minut z jedné adresy uvnitř sítě na jednu adresu v internetu na port 80. Všechny tyto toky vždy sestávají z 5 paketů a přenášejí přesně 545 bytů. Z těchto informací sice nelze s určitostí říci, zda se jedná o legitimní provoz či nikoliv, je ale možné, že jde o pokus o DoS útok na web server. Každopádně během trvání anomálie tvoří tento proud dat více než 7 % z celkového počtu toků na sledované lince a zdrojová IP adresa je tak vůbec neaktivnější ze všech, což samo o sobě znamená, že je vhodné takovou událost detekovat. Z reálných dat byly tedy extrahovány všechny toky z dané adresy a byly dále použity stejně jako ostatní anomálie.

V každém intervalu byly pro všechny dimenze vypočítány hodnoty všech šesti metrik (entropie a pět metrik navržených v kapitole 5.2) a to vždy pro normální data i pro data se všemi výše uvedenými anomáliemi (s každou zvlášť). Byla určena směrodatná odchylka normálních dat a průměrná změna způsobená anomálií a z toho pak míra rozlišitelnosti anomálie v dané dimenzi pomocí dané metriky.

Jako dimenze byl kromě zdrojových a cílových adres a portů použit ještě počet paketů v toku, počet bajtů v toku a TCP flagy (logický součet flagů všech paketů v toku)². Tyto základní dimenze byly dále kombinovány do dvojic či trojic a tak bylo vytvořeno množství

²Mohl by být zahrnut ještě protokol, ovšem ten je již dobře pokryt dimenzí TCP flagů, protože ty jsou při jiném než TCP protokolu (tedy pravděpodobně UDP, jiné se vyskytují minimálně) nastaveny na samé nuly.

dalších dimenzí. Kombinaci dimenzí zde lze chápat jako bitovou konkatenci hodnot do jedné širší hodnoty.

U dimenzí s počtem paketů či bajtů v toku nejsou ve skutečnosti používány přímo tyto hodnoty, ale jejich dvojkové logaritmy. Je tomu tak proto, že počty paketů či bajtů jsou spojité dimenze, u kterých rozdíl dvou malých hodnot nese více informace, než stejně velký rozdíl mezi dvěma velkými hodnotami (tzn. může být užitečné vědět, zda byly přeneseny 2 nebo 3 pakety, ale to, jestli bylo paketů přeneseno např. 1255 nebo 1256, je obvykle zanedbatelné). Právě použitím logaritmu a zaokrouhlením výsledku na celá čísla zachováme vysoké rozlišení na malých hodnotách, přičemž velké hodnoty se sloučí do větších skupin.

7.3.1 Popis grafů a značení

Metriky a dimenze jsou v grafech i v dalším textu označovány názvy použitými v testovacím software. Entropie je tedy označena zkratkou **ent**, sumě čtverců M_2 , resp. M_2' odpovídá **sos**, resp. **sos2** (z anglického *sum of squares*), sumě převrácených hodnot odmocnin $M_{-1/2}$, resp. $M_{-1/2}'$ odpovídá **rsqrt**, resp. **rsqrt2** (z anglického *reciprocal square roots*) a počet unikátních hodnot je značen **uniq**.

Označení základních dimenzí je následující: **sa**, **da** je zdrojová a cílová adresa, **sp**, **dp** značí zdrojový a cílový port, **lpkt** je logaritmus počtu paketů, **lbyt** logaritmus počtu bajtů a **flg** označuje TCP flagy. Při značení kombinací se tyto dimenze spojují znakem '+'. Detailnější popis značení lze dohledat v příloze **B.1**.

Pro každou anomálii byla vypočítána její rozlišitelnost ve všech dimenzích a se všemi metrikami, a tyto rozlišitelnosti byly zaneseny do grafů. Vybrané grafy jsou uvedeny v příloze **C**, zbývající jsou pak pouze v elektronické podobě na přiloženém CD.

Jde o sloupcové grafy, kde každá skupina sloupců reprezentuje jednu dimenzi a každý sloupec ve skupině určuje rozlišitelnost při použití jedné metriky. Sloupce ve skupině jsou pro větší čitelnost grafu barevně rozlišeny, každé metrice tak odpovídá jedna barva. Na svislé ose je vynesena rozlišitelnost dané anomálie, přičemž hodnota 5, hranice rozlišitelnosti, je v grafu zvýrazněna vodorovnou přerušovanou čarou.

V těch dimenzích a metrikách, kde sloupec nepřekročí hodnotu 5, anomálie nezpůsobuje žádnou odchylku, nebo je tato odchylka příliš malá na to, aby byla spolehlivě odlišitelná od běžného kolísání dat, a anomálii zde považujeme za nedetekovatelnou. Tam, kde sloupec překročí hodnotu 5, považujeme anomálii za detekovatelnou, přičemž čím větší je hodnota rozlišitelnosti, tím lépe.

Výsledky původního algoritmu jsou v grafech reprezentovány červenými sloupci (entropie) v prvních čtyřech dimenzích (zdrojové a cílové adresy a porty).

7.3.2 Rozlišitelnost jednotlivých anomálií

Rozlišitelnost každé anomálie byla testována se třemi různými vzorky normálních dat, proto ke každé anomálii existují tři grafy. Protože testované DoS útoky i skenování portů pracují s TCP protokolem, omezíme celou detekci anomálií jen na TCP protokol, a to tak, že všechny toky s jiným protokolem z normálních dat odfiltrujeme. Taková technika může být použitelná i v praxi, protože se tím značně zmenší celkové množství toků, a anomálie jsou pak v datech o něco lépe viditelné. Paralelně přitom můžeme detekovat útoky i nad jinými protokoly.

Datové vzorky normálních dat po filtrování obsahují 200–350 tisíc toků v každém pětiminutovém intervalu.

DoS útok

Podívejme se nejdříve na grafy pro DoS útok s deseti tisíci toků na vzorcích dat číslo 1 a 2 (grafy C.1 a C.2). Z červených sloupců zcela vlevo vidíme, že při použití entropie a základních dimenzí je útok v datech č. 1 sotva rozlišitelný (podle cílové adresy) a v datech č. 2 není dokonce rozlišitelný vůbec. O něco vyšší hodnoty rozlišitelnosti při použití entropie dávají kombinace cílové adresy s logaritmem počtu paketů či bajtů nebo s TCP flagy. V těchto dimenzích se už hodnoty pohybují kolem hranice rozlišitelnosti i v datech č. 2.

Pokud se ale jako metrika místo entropie použije suma čtverců, zvýší se rozlišitelnost několikanásobně a útok se stane snadno detekovatelným v jakékoliv dimenzi obsahující cílovou adresu. Přitom příliš nezáleží na tom, jestli se použije `sos` nebo `sos2`, resp. M_2 nebo M'_2 , hodnoty pro obě metriky jsou velmi podobné.

Datový vzorek č. 3 obsahuje o něco více toků normálního provozu a tento útok tam není detekovatelný pomocí žádné dimenze a metriky. Ovšem rozsáhlejší útok s 50 tisíci paketů je již velmi dobře detekovatelný i v těchto datech, a přestože takto masivní útok se projeví i v jiných dimenzích a metrikách, nejvyšší rozlišitelnosti dosahuje opět suma čtverců nad dimenzemi s cílovou adresou.

DoS útok s podvrženými adresami

Podívejme se nyní na grafy DoS útoku s podvrženými zdrojovými adresami (obrázky C.7–C.12). Zdrojové adresy jsou zde generovány zcela náhodně, což by mělo způsobit značné rozptýlení histogramu zdrojových adres. V ostatních dimenzích se útok neliší od předchozího a tak je útok dobře rozlišitelný podle sumy čtverců v dimenzích s cílovou adresou, stejně jako v předchozím případě.

Ovšem ještě mnohem lépe se u tohoto útoku uplatnily metriky navržené pro detekci rozptýlení histogramu, tedy ty založené na převrácených hodnotách odmocnin a také prostý počet unikátních hodnot, a to právě na dimenzích obsahujících zdrojovou adresu. Všechny tři metriky `rsqrt`, `rsqrt2` a `uniq` dosahují poměrně vysokých hodnot na dimenzích kombinujících zdrojovou adresu s počtem paketů, bajtů, či s TCP flagy, ovšem jako nejlepší dimenze se ukazuje samotná zdrojová adresa. Která z metrik je nejlepší již tak jednoznačné není, neboť to se v různých datových vzorcích liší, každá ovšem výrazně přesahuje hranici rozlišitelnosti ve všech sledovaných případech.

Skenování portů

Dalším testovaným útokem bylo skenování portů, pro něž jsou grafy rozlišitelnosti uvedeny na obrázcích C.13–C.18. Tento útok sestává z jednopaketových toků směřujících z jedné zdrojové adresy na malou množinu cílových adres, přičemž zdrojové i cílové porty mají velmi různé hodnoty.

Když se podíváme na graf C.16, vidíme, že tento útok je zachytitelný téměř ve všech dimenzích a často hned pomocí několika metrik. Uplatňují se zde jak metriky zaměřené na koncentraci histogramu (`sos`, `sos2`), tak na rozptýlení histogramu (`rsqrt`, `rsqrt2`, `uniq`), ovšem celkově nejlepších hodnot zde dosahuje entropie.

Když se podíváme na grafy dalších vzorků dat (C.17, C.18), vidíme, že zde už je útok o něco hůře rozlišitelný, ovšem některé dimenze a metriky si zachovávají stále vysoké hodnoty. Je to především kombinace zdrojové adresy a zdrojového portu (případně ještě v kombinaci s počtem paketů, bajtů nebo s TCP flagy) s metrikou `rsqrt2`. Dále je pro detekci

vhodná entropie kombinace zdrojových a cílových adres a sumy čtverců (`sos` i `sos2`) zdrojové adresy kombinované s počtem paketů, bajtů či s TCP flagy.

Výpadek serveru

Další simulovanou anomálií není útok, ale výpadek serveru, či celé podsítě. Jak se taková anomálie projeví je uvedeno v grafech C.19 až C.24.

Na prvních třech grafech jsou uvedena data z výpadku serveru s provozem cca 1000 toků během každého pětiminutového intervalu. Je vidět, že tato změna se na statistikách z celé sítě téměř nijak neprojevila. Naproti tomu výpadek jiného serveru, na kterém je provoz průměrně 13 tisíc toků během pěti minut, je již poměrně dobře rozlišitelný, a to především na dimenzích zdrojového portu a jeho kombinacích s dalšími dimenzemi a na kombinacích cílového portu s počtem bajtů či s TCP flagy. Na všech těchto dimenzích je výpadek rozlišitelný s použitím sumy čtverců. Dále je rozlišitelný použitím sumy převrácených hodnot odmocnin na kombinaci zdrojové adresy a cílového portu.

Ačkoliv se dá očekávat, že konkrétní změny se budou pro různé typy serverů lišit, výsledky ukazují, že metoda dokáže detekovat nejen útoky, ale i některé jiné změny v provozu na síti, pokud jsou dostatečně velké. Přitom se ale každý typ anomálie projevuje v jiných dimenzích a na jiných metrikách, takže je lze navzájem bez problémů rozlišit.

Reálná anomálie

Jak bylo řečeno výše, byla testována i rozlišitelnost anomálie nalezené v reálných datech. Její grafy jsou na obrázcích C.25 až C.27. Jelikož tato anomálie vypadá jako DoS útok na úrovni HTTP protokolu, jsou její projevy v distribucích hodnot do značné míry podobné dříve uvedenému, uměle vygenerovanému DoS útoku – vysokých hodnot rozlišitelnosti dosahuje opět suma čtverců na všech dimenzích obsahujících cílovou adresu.

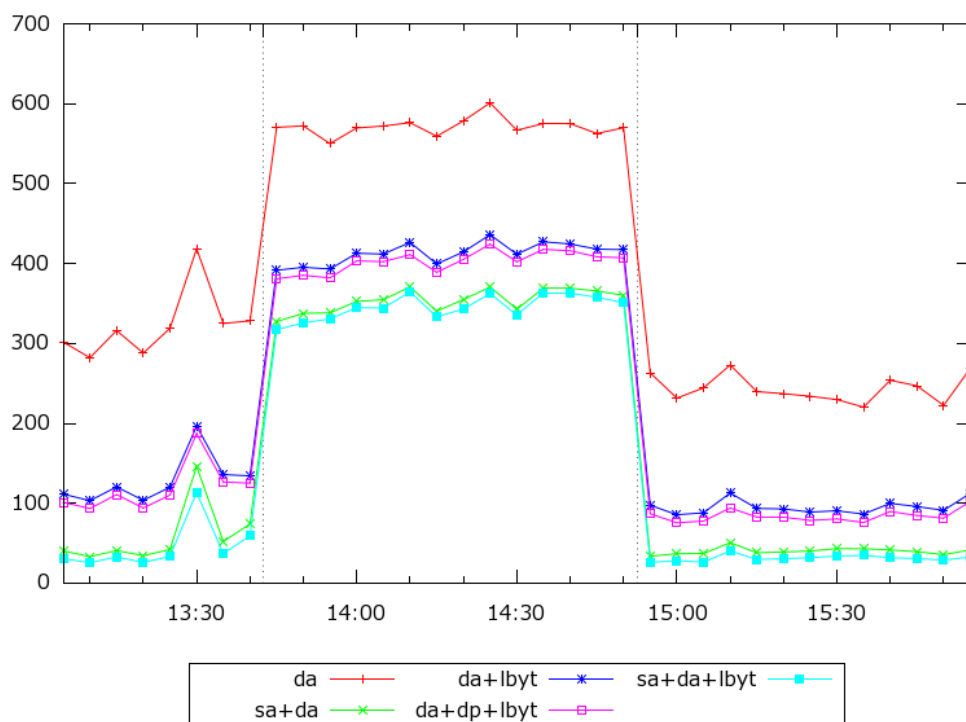
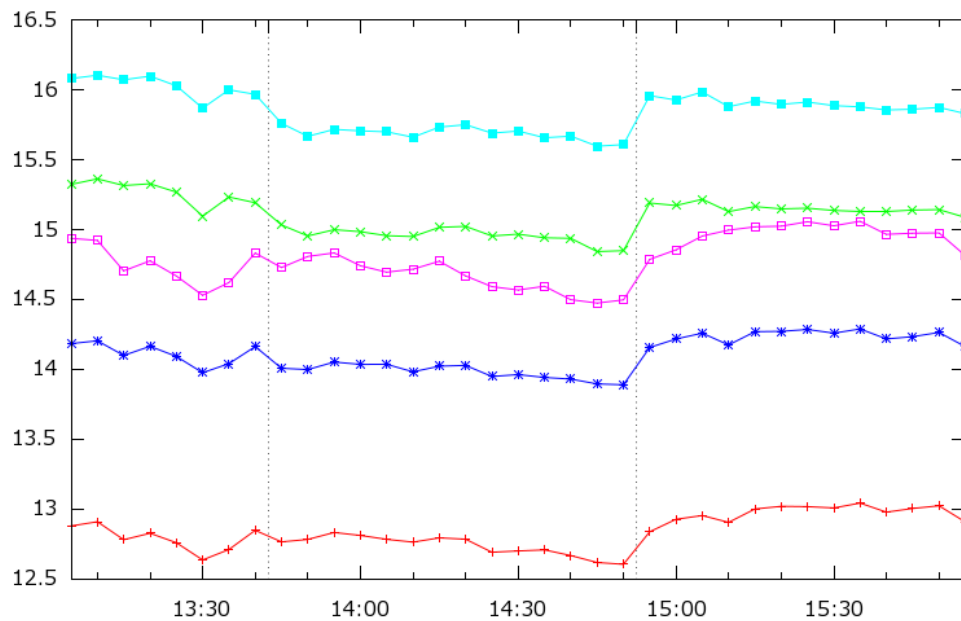
Narozdíl od umělého útoku, který byl mírně distribuovaný (10 zdrojových adres), zde jdou všechny toky z jedné adresy. Proto je tato anomálie dobře rozlišitelná navíc i v některých kombinacích obsahujících zdrojovou adresu.

7.3.3 Příklad anomálie v časových průbězích

Všechny výsledky jsme dosud hodnotili jen na základě míry rozlišitelnosti anomálie. Uvěďme tedy ještě ukázkou, jak vypadá anomálie, vyneseme-li do grafu hodnoty metrik v závislosti na čase. Na obrázku 7.2 jsou časové průběhy entropie a sumy čtverců na několika vybraných dimenzích, v okolí časového intervalu, kdy anomálie probíhala. Jedná se o výše zmíněnou anomálii nalezenou v reálných datech a identifikovanou jako pravděpodobný DoS útok, proto byly vybrány dimenze, které vykazovaly velké hodnoty rozlišitelnosti pro DoS útoky.

Na horním grafu jsou uvedeny hodnoty entropie. Je vidět, že alespoň v některých dimenzích jsou v časech začátku a konce anomálie patrné jisté změny hodnot, avšak tyto změny jsou velmi malé a jen těžko rozeznatelné od náhodného kolísání. To odpovídá situaci v grafech rozlišitelnosti reálné anomálie C.26 a C.27, kde se entropie v daných dimenzích pohybuje kolem hranice rozlišitelnosti.

Na dolním grafu jsou uvedeny hodnoty sumy čtverců (`sos`, tedy M_2) ve stejném časovém okně a pro stejné dimenze. V těchto datech je již anomálie mnohem lépe rozpoznatelná, neboť odchylky způsobené anomálií u těchto metrik značně přesahují velikost náhodného kolísání.



Obrázek 7.2: Grafy časových průběhů entropie (nahore) a sumy čtverců (dole) v několika dimenzích. Časový interval, kdy probíhala anomálie, je znázorněn svislými přerušovanými čarami.

7.3.4 Shrnutí

Z množství uvedených grafů rozlišitelnosti několika různých anomálií tedy vyplývá, že jak zavedení nových dimenzí a především jejich kombinací, tak i použití jiných metrik než entropie, může velmi výrazně pomoci detekovat i anomálie, které by byly při daném množství provozu pro původní algoritmus příliš slabé a neprojevíly by se dostatečně velkou odchylkou v entropii.

Navíc, protože každá anomálie se projevuje v jiných dimenzích, může velké množství dimenzí pomoci lépe klasifikovat různé druhy anomálií.

Výpočet histogramů pro velké množství dimenzí a poté ještě několika metrik nad každou z nich sice výrazně zvyšuje výpočetní náročnost algoritmu, ovšem použití optimalizace pomocí hashovací funkce, uvedené v kapitole 5.1, to může do značné míry kompenzovat. Navíc není nutné implementovat všechny zde testované dimenze a metriky. Například metriky `sos` a `sos2` se ve všech případech chovaly velmi podobně, takže může stačit používat pouze jednu z nich. Podobně některé kombinace dimenzí se chovají velmi podobně a pokud je třeba šetřit výpočetním výkonem a pamětí, zdaleka nemusí být počítány všechny.

Kolik dimenzí a metrik je vhodné použít tedy záleží především tom, kolik máme k dispozici výpočetního výkonu a kolik dimenzí jsme schopni dále efektivně zpracovat (detekce odchylek, korelace odchylek v různých dimenzích, klasifikace anomálií).

Kapitola 8

Závěr

V úvodu této práce byly popsány základní principy detekce anomálií a bylo uvedeno množství různých metod, které se v této oblasti používaly nebo používají. Dále vybrána vybrána metoda založená na výpočtech distribuce hodnot vlastností síťového provozu a její entropie. Tato metoda byla podrobně popsána a byly uvedeny její nedostatky.

V další části byly navrženy dva způsoby jejího vylepšení. První spočívá v použití hash funkce k výraznému zrychlení algoritmu a snížení spotřeby paměti. Experimenty s daty z vysokorychlostní sítě ukázaly, že i při takovém nastavení parametrů, kdy je spotřeba paměti menší než 1 MB, nejsou nijak výrazně zhoršeny detekční schopnosti algoritmu. Navíc výpočet probíhá $10-20 \times$ rychleji než u původní verze.

Druhým navrhovaným vylepšením je použití více dimenzí, než jen zdrojových a cílových adres a portů, a především jejich kombinací, a také využití jiných metrik, než jen entropie. Provedené experimenty potvrdily, že jak kombinace více dimenzí, tak nově navržené metriky mohou výrazně zlepšit detekční schopnosti metody.

Literatura

- [1] Patcha, A.; Park, J.-M.: An Overview of Anomaly Detection Techniques: Existing Solutions and Latest Technological Trends. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, ročník 51, srpen 2007: s. 3448–3470, ISSN 1389-1286.
- [2] Zhang, W.; Yang, Q.; Geng, Y.: An Overview of Anomaly Detection Techniques: Existing Solutions and Latest Technological Trends. *International Symposium on Computer Network and Multimedia Technology (CNMT), 2009.*, leden 2009: s. 1–3.
- [3] Lakhina, A.; Crovella, M.; Diot, C.: Mining Anomalies Using Traffic Feature Distributions. *ACM SIGCOMM Computer Communication Review*, ročník 35, č. 4, říjen 2005.
- [4] Denning, D. E.: An Intrusion-detection Model. *IEEE Transactions on Software Engineering*, ročník 13, č. 2, 1987: s. 222–232, ISSN 0098-5589.
- [5] Lunt, T.; Jagannathan, R.: A Prototype Real-time Intrusion-detection Expert System. In *Proceedings of the IEEE Symposium on Security and Privacy*, 1988, s. 59–66.
- [6] Lunta, T. F.; Tamaru, A.; Gilham, F.: A Real-time Intrusion-Detection Expert System (IDES). Technická zpráva, Computer Science Laboratory, SRI International, Menlo Park, CA, USA, únor 1992.
- [7] Anderson, D.; Frivold, T.; Valdes, A.: *Next-generation Intrusion Detection Expert System (NIDES): A Summary*. Computer Science Laboratory, SRI International, 1995.
- [8] Smaha, S.: Haystack: An Intrusion Detection System. *Aerospace Computer Security Applications Conference, 1988, Fourth*, prosinec 1988: s. 37–44.
- [9] Axelsson, S.: Intrusion Detection Systems: A Survey and Taxonomy. Technická zpráva, Chalmers University of Technology, Göteborg, Sweden, 2000.
- [10] Staniford, S.; Hoagland, J. A.; McAlerney, J. M.: Practical Automated Detection of Stealthy Portscans. *Journal of Computer Security*, ročník 10, 2002: s. 105–136, ISSN 0926-227X.
- [11] Ye, N.; Emran, S.; Chen, Q.; aj.: Multivariate Statistical Analysis of Audit Trails for Host-based Intrusion Detection. *IEEE Transactions on Computers*, ročník 51, 2002: s. 810–820, ISSN 0018-9340.

- [12] Lee, W.; Xiang, D.: Information-Theoretic Measures for Anomaly Detection. In *Proceedings of the 2001 IEEE Symposium on Security and Privacy*, 2001, s. 130–143.
- [13] Valdes, A.; Skinner, K.: Adaptive Model-based Monitoring for Cyber Attack Detection. In *Recent Advances in Intrusion Detection*, Toulouse, France, říjen 2000, s. 80–93.
- [14] Kruegel, C.; Mutz, D.; Robertson, W.; aj.: Bayesian Event Classification for Intrusion Detection. In *19th Annual Computer Security Applications Conference (ACSAC '03)*, Las Vegas, Nevada, USA, prosinec 2003, ISBN 0-7692-2041-3, s. 14–23.
- [15] Principal Component Analysis. Wikipedia [online], 2011 [cit. 2011-01-05]. URL http://en.wikipedia.org/wiki/Principal_component_analysis
- [16] Shyu, M.-L.; Chen, S.-C.; Sarinnapakorn, K.; aj.: A Novel Anomaly Detection Scheme Based on Principal Component Classifier. In *Proceedings of the IEEE Foundations and New Directions of Data Mining Workshop*, 2003, s. 172–179.
- [17] Mahoney, M. V.; Chan, P. K.: PHAD: Packet Header Anomaly Detection for Identifying Hostile Network Traffic. Technická zpráva, Florida Institute of Technology, 2001.
- [18] Mahoney, M. V.; Chan, P. K.: Learning Models of Network Traffic for Detecting Novel Attacks. Technická zpráva, Florida Institute of Technology, 2002.
- [19] Mahoney, M. V.; Chan, P. K.: Learning Nonstationary Models of Normal Network Traffic for Detecting Novel Attacks. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002, ISBN 1-58113-567-X, s. 376–385.
- [20] Warrender, C.; Forrest, S.; Pearlmuter, B.: Detecting Intrusions Using System Calls: Alternative Data Models. In *Proceedings of the 1999 IEEE Symposium on Security and Privacy*, 1999, ISBN 0-7695-0176-1, s. 133–145.
- [21] Clifton, C.: Data mining. Encyclopedia Britannica [online], 2011 [cit. 2011-01-04]. URL <http://www.britannica.com/EBchecked/topic/1056150/data-mining>
- [22] Cohen, W. W.: Fast Effective Rule Induction. In *Proceedings of the Twelfth International Conference on Machine Learning*, 1995, s. 115–123.
- [23] Quinlan, J. R.: *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, 1993.
- [24] Lee, W.; Stolfo, S. J.: Data Mining Approaches for Intrusion Detection. In *Proceedings of the 7th conference on USENIX Security Symposium - Volume 7*, Berkeley, CA, USA: USENIX Association, 1998, s. 79–94.
- [25] Bridges, S. M.; Vaughn, R. B.: Fuzzy Data Mining And Genetic Algorithms Applied To Intrusion Detection. In *Proceedings of the National Information Systems Security Conference (NISSC)*, Baltimore, MB, USA, říjen 2000.
- [26] Dickerson, J.; Dickerson, J.: Fuzzy Network Profiling for Intrusion Detection. In *Proceedings of the 19th International Conference of the North American Fuzzy Information Processing Society (NAFIPS)*, Atlanta, GA, USA, 2000, s. 301–306.

- [27] Ertöz, L.; Eilertson, E.; Lazarevic, A.; aj.: MINDS - Minnesota Intrusion Detection System. In *Data Mining: Next Generation Challenges and Future Directions*, MIT Press, říjen 2004, ISBN 0-262-61203-8, s. 199–218.
- [28] Chandola, V.; Banerjee, A.; Kumar, V.: Anomaly Detection: A Survey. *ACM Computing Surveys*, ročník 41, červenec 2009: s. 1–72.
- [29] Barbará, D.; Couto, J.; Jajodia, S.; aj.: ADAM: Detecting Intrusions by Data Mining. In *Proceedings of the IEEE Workshop on Information Assurance and Security*, 2001, s. 11–16.
- [30] Netflow. Wikipedia [online], 2011 [cit. 2011-01-09].
URL <http://cs.wikipedia.org/wiki/Netflow>
- [31] Claise, B.: RFC 3954: Cisco Systems NetFlow Services Export Version 9. říjen 2004.
URL <http://www.ietf.org/rfc/rfc3954.txt>
- [32] Claise, B.: RFC 5101: Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information. leden 2008.
URL <http://www.ietf.org/rfc/rfc5101.txt>
- [33] Reháč, M.; Pěchouček, M.; Bartoš, K.; aj.: CAMNEP: An Intrusion Detection System for High-speed Networks. *Progress in Informatics*, ročník 2008, č. 5, 2008: s. 65–74, ISSN 1349-8614.

Příloha A

Obsah CD

K této práci je přiloženo CD se doplňujícími zdroji v elektronické podobě. Jeho obsahem je:

- Elektronická verze textu diplomové práce ve formátu PDF
- Zdrojový kód programu pro výpočty entropie a dalších metrik (tj. software popisovaný v kapitole 6)
- Skript v jazyce Python pro generování různých druhů útoků (anomálií)
- Grafy rozlišitelnosti anomálií v elektronické podobě. Jsou zde i grafy pro více různých mohutností útoků, než ty, které jsou uvedeny v příloze C.

Příloha B

Parametry programu

V této příloze je uveden seznam spouštěcích parametrů programu představeného v kapitole 6 a popis jejich použití. Dále je uveden seznam podporovaných dimenzí a metrik a popsán formát, v jakém se jejich specifikace programu předává. Podobný text, jako je v této příloze, je uložen i v programu a zobrazí se jako nápověda při spuštění s parametrem `-h`.

`-a soubor`

(append/anomaly/attack) Po výpočtu hodnot z normálních dat budou přidána data z uvedeného souboru a hodnoty vypočítány znovu. Výstupem budou dvě tabulky, jedna pro normální data, druhá pro data s přidanou anomálií/útokem. Tento parametr může být zadán i vícekrát, v tom případě bude vypočítána tabulka pro každou anomálii.

`-d specifikace_dimenzí`

(dimensions) Specifikuje, které dimenze a metry se mají počítat. Formát této specifikace je uveden dále, v podkapitole B.1. Pokud parametr není zadán, je použito "sa da sp dp", tedy entropie zdrojových a cílových adres a portů.

`-D soubor`

(dimensions file) Načte specifikaci dimenzí ze souboru.

`-g x,y`

(ignore) Do statistik nebude započítáno x prvních a y posledních řádků. Má význam pouze společně s parametrem `-i`.

`-h`

(help) Vypíše nápovědu.

`-i sekundy`

(interval) Rozdělí data podle časových značek toků do intervalů o zadané délce (v sekundách). Data musí být seřazena.

-m *metrika*

(metric) Pro ty dimenze, u kterých není žádná metrika zvolena explicitně ve specifikaci dimenzí, bude použita *metrika*. Výchozí hodnotou je entropie.

-s

(summary/statistic) Na konci vypíše shrnutí – základní statistické údaje přes všechny intervaly. Má význam pouze společně s parametrem -i.

-S

(summary/statistic only) Bude vypsáno pouze shrnutí.

-t

(title) Na prvním řádku vypíše názvy sloupců tabulky.

B.1 Specifikace dimenzí

Specifikace dimenzí sestává z jednoho nebo více specifikátorů jedné dimenze, oddělených libovolnými bílými znaky. Formálně zapsáno:

```
<dimensions> = <dimspec> [<dimensions>]
```

kde *dimspec* je specifikace jedné dimenze. Ta má následující formát:

```
<dimspec> = <dim>[+<dim>[+<dim>]][#<hash_bits>][/<metric>]  
  
<dim> = sa | da | sp | dp | pkt | byt | lpkt | lbyt | flg  
  
<hash_bits> = kladné celé číslo  
  
<metric> = ent | sos | sos2 | rsqrt | rsqrt2 | uniq
```

Lze tedy zvolit jednu základní dimenzi, či pomocí znaku '+' kombinaci dvou nebo tří základních dimenzí. Těmi jsou:

- **sa** – Zdrojová adresa
- **da** – Cílová adresa
- **sp** – Zdrojový port
- **dp** – Cílový port
- **pkt** – Počet paketů v toku
- **byt** – Počet bytů v toku
- **lpkt** – Logaritmus počtu paketů v toku
- **lbyt** – Logaritmus počtu bytů v toku
- **flg** – Logický součet polí s TCP flagy všech paketů v toku

Pokud chceme zapnout optimalizaci pomocí hashovací funkce, následuje znak '#' a kladné celé číslo, udávající výstupní šířku hash funkce v bitech.

Dále může následovat lomítko a specifikace metriky, kterou chceme pro v danou dimenzi použít, jinak je použita výchozí metrika, což je standardně entropie (lze změnit pomocí parametru `-m`). Lze použít tyto metriky:

- `ent` – Entropie, definovaná v kap. 4 a značená jako $H(X)$.
- `sos` – Suma čtverců, vydělená celkovým počtem toků. V kap. 5.2 značeno jako $M_2(X)$.
- `sos2` – Suma čtverců, vydělená počtem unikátních hodnot v dané dimenzi. V kap. 5.2 značeno jako $M'_2(X)$.
- `rsqrt` – Suma převrácených hodnot odmocnin, vydělená celkovým počtem toků. V kap. 5.2 značeno jako $M_{-\frac{1}{2}}(X)$.
- `rsqrt2` – Suma převrácených hodnot odmocnin, vydělená počtem unikátních hodnot v dané dimenzi. V kap. 5.2 značeno jako $M'_{-\frac{1}{2}}(X)$.
- `uniq` – Počet unikátních hodnot v dané dimenzi. V kap. 5.2 značeno jako $M_0(X)$.

B.2 Příklady

Následuje několik příkladů specifikace dimenzí:

- `sa da sp dp` – Čtyři dimenze z původního algoritmu, zdrojové a cílové adresy a porty a jako metrika entropie.
- `sa#16 da#16 sp#10 dp#10` – Stejně jako předchozí, ale s optimalizací pomocí hashovací funkce. Šířka výstupu hash funkce je pro adresy 16 bitů, pro porty 10 bitů.
- `da+dp/sos` – Kombinace cílové adresy a cílového portu. Jako metrika se použije suma čtverců.
- `sa+lpkt+flg#16/rsqrt2` – Kombinace zdrojové adresy, logaritmu počtu paketů v toku a TCP flagů. Jako metrika se použije suma převrácených hodnot odmocnin.

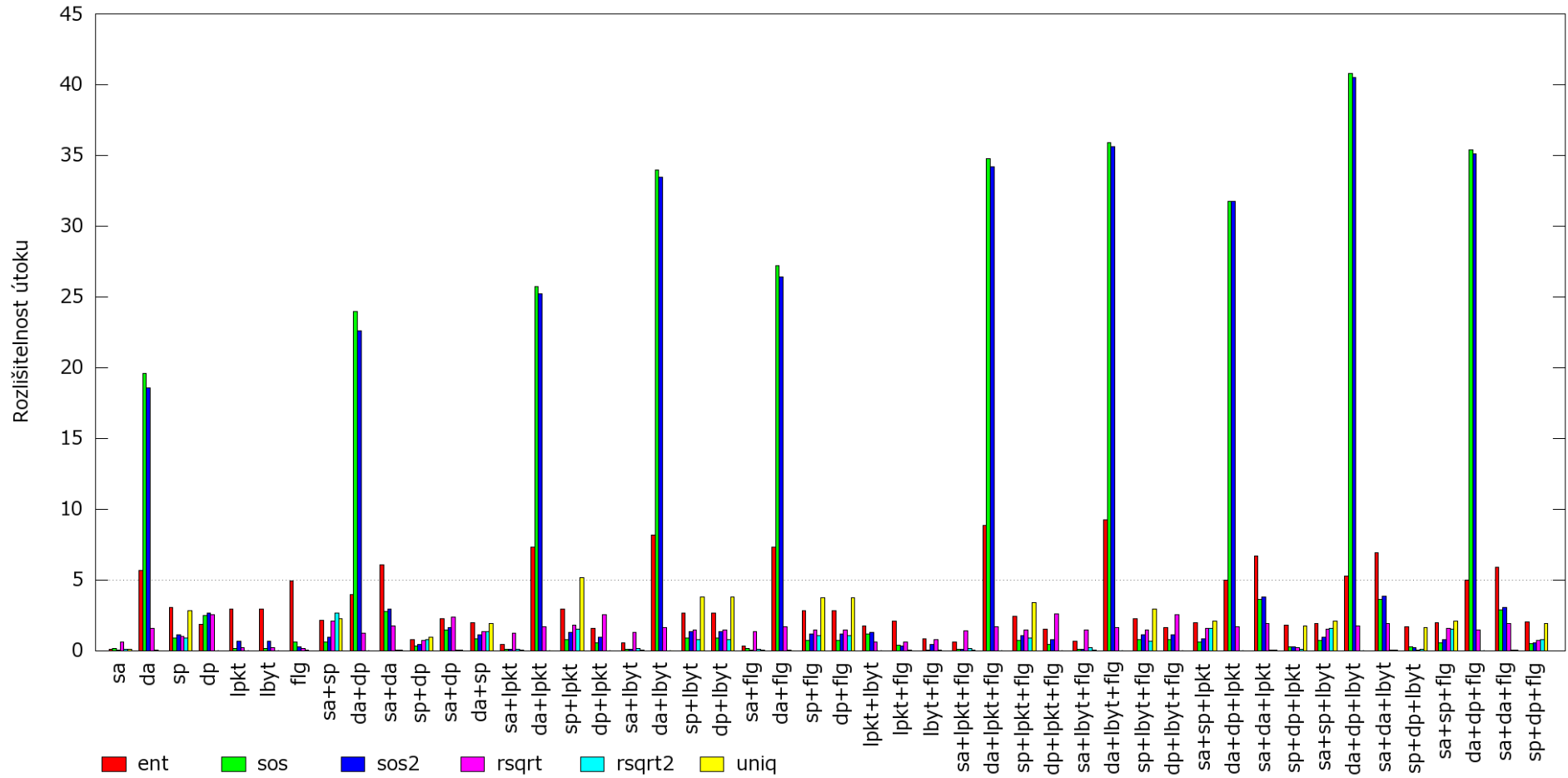
Příloha C

Grafy rozlišitelnosti anomálií

Na následujících stranách jsou uvedeny grafy rozlišitelnosti anomálií podle různých metrik v různých dimenzích. Pro každou anomálii a každý ze tří vzorků normálních dat je uveden jeden graf.

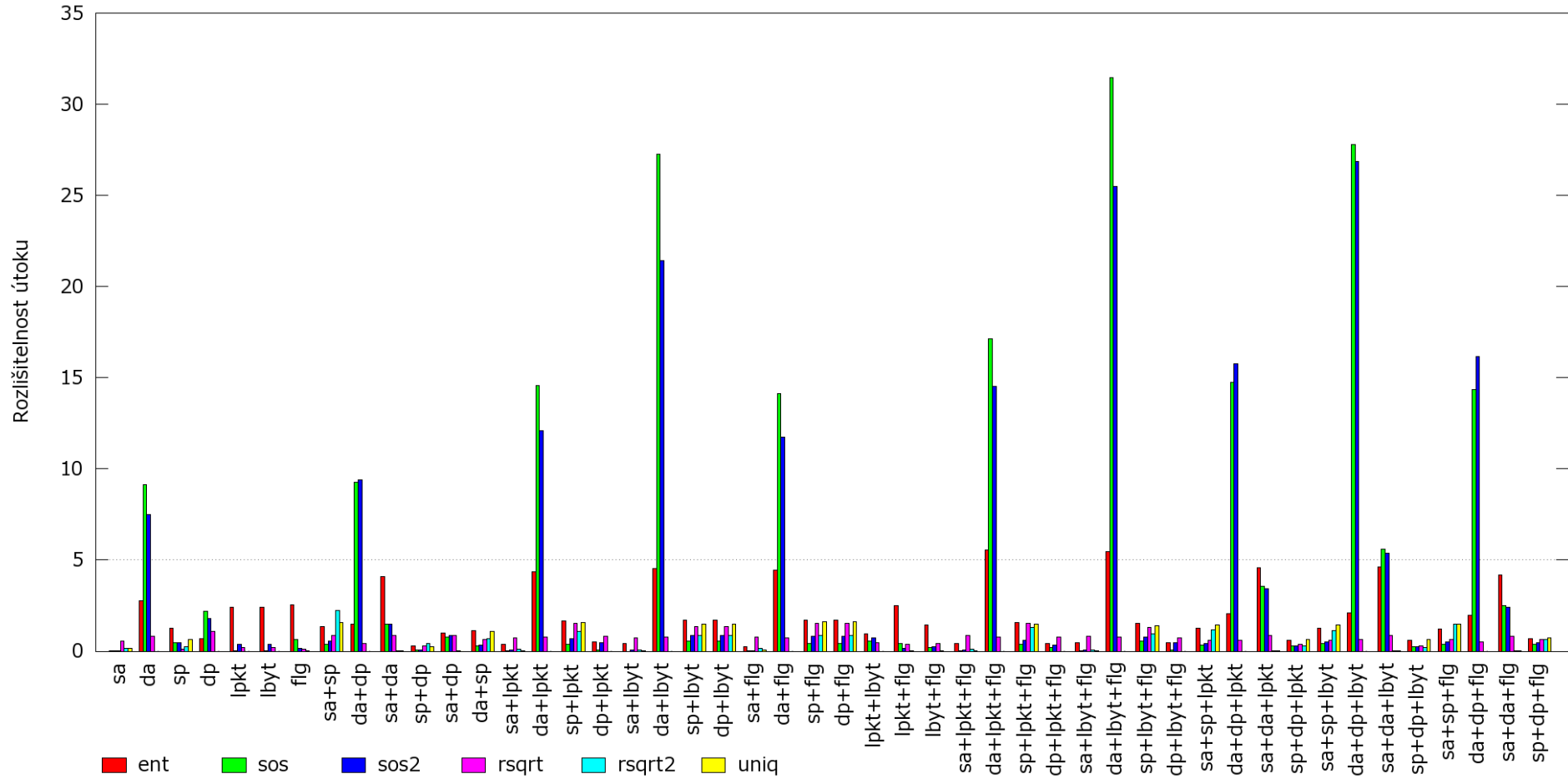
Grafy pro jiné mohutnosti útoků, než zde uvedené, lze najít v elektronické podobě na přiloženém CD.

DoS útok, 10k, data 1



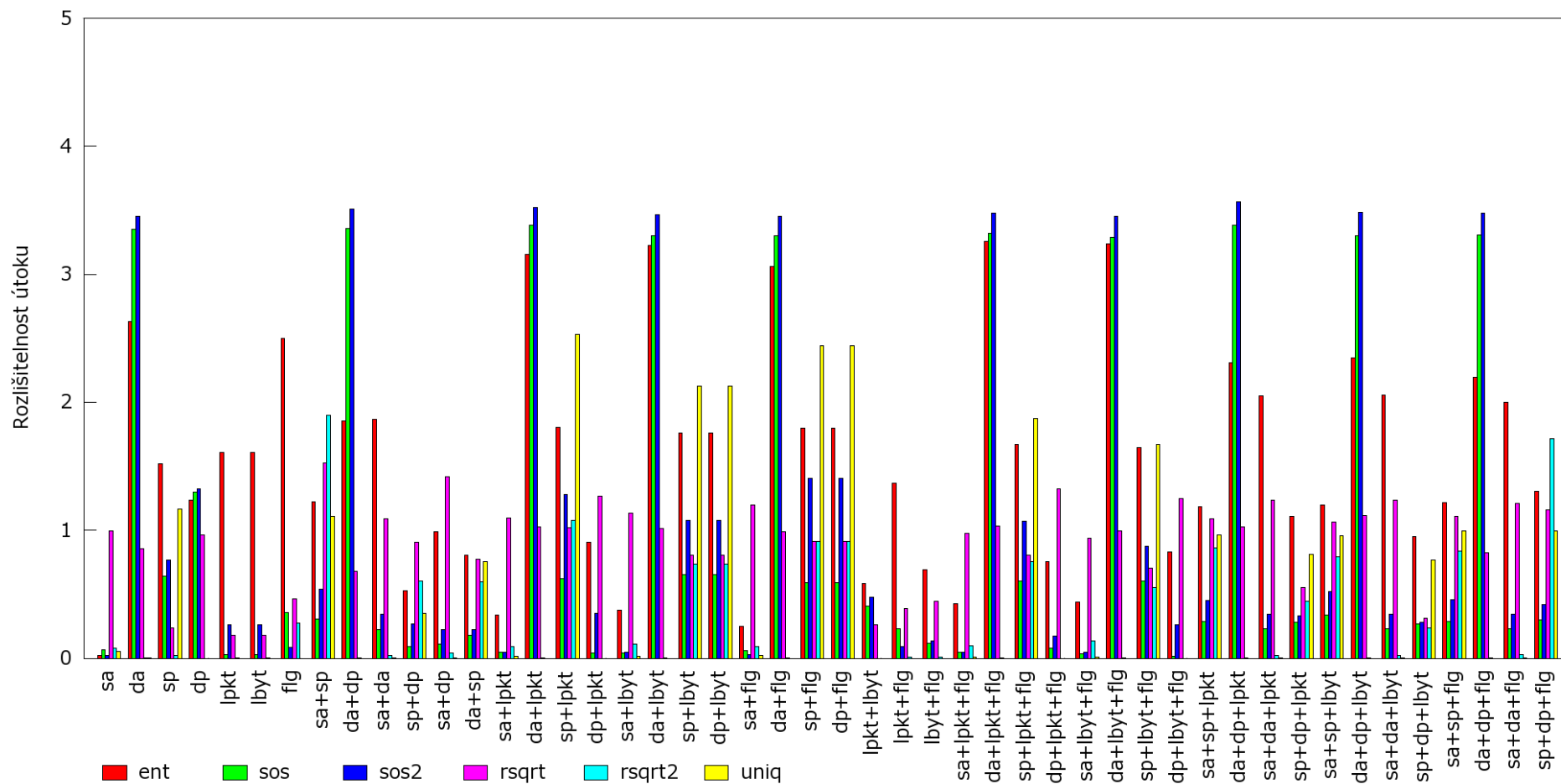
Obrázek C.1: DoS útok, 10 tisíc toků, vzorek dat č. 1

DoS útok, 10k, data 2



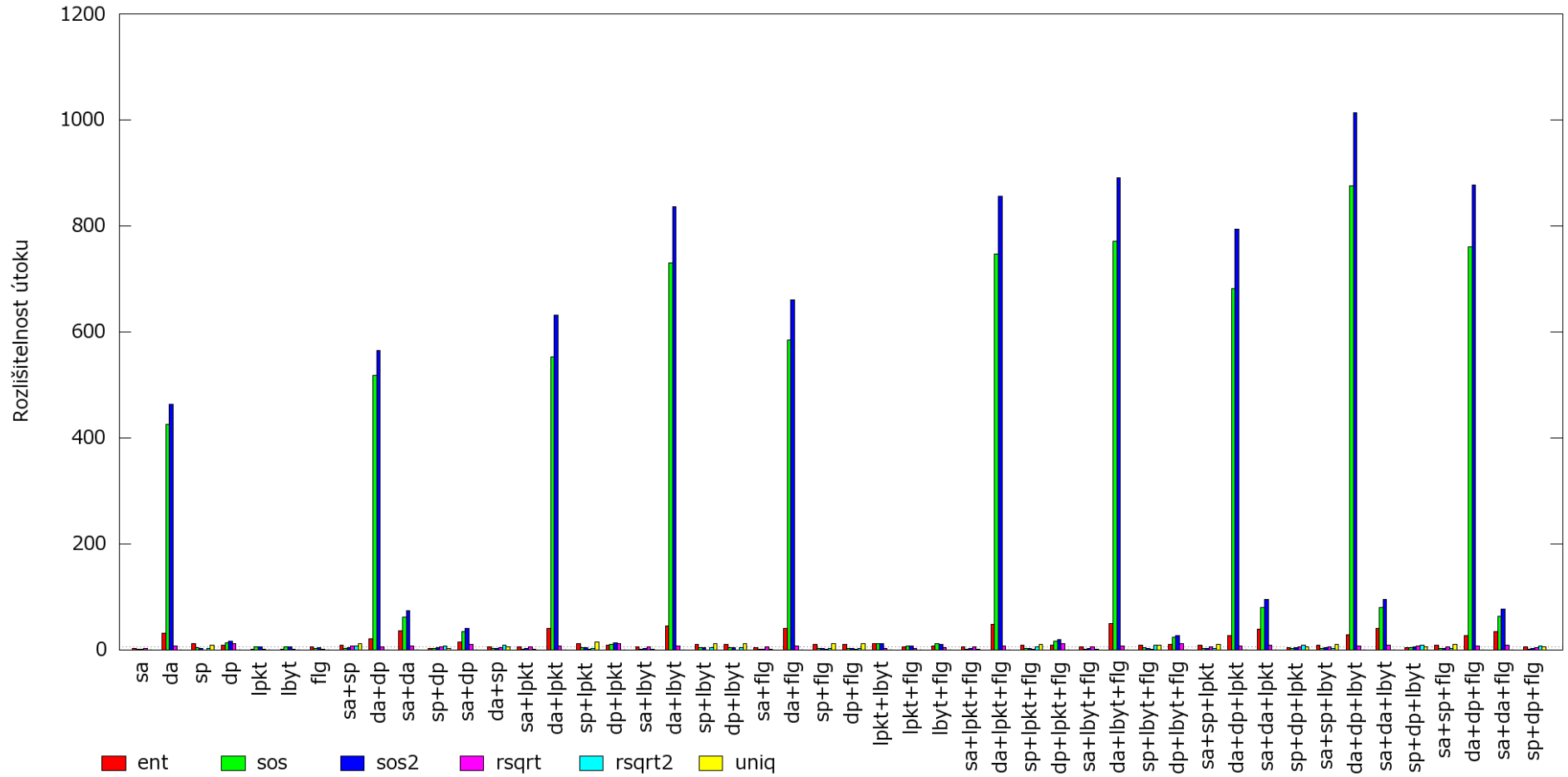
Obrázek C.2: DoS útok, 10 tisíc toků, vzorek dat č. 2

DoS útok, 10k, data 3



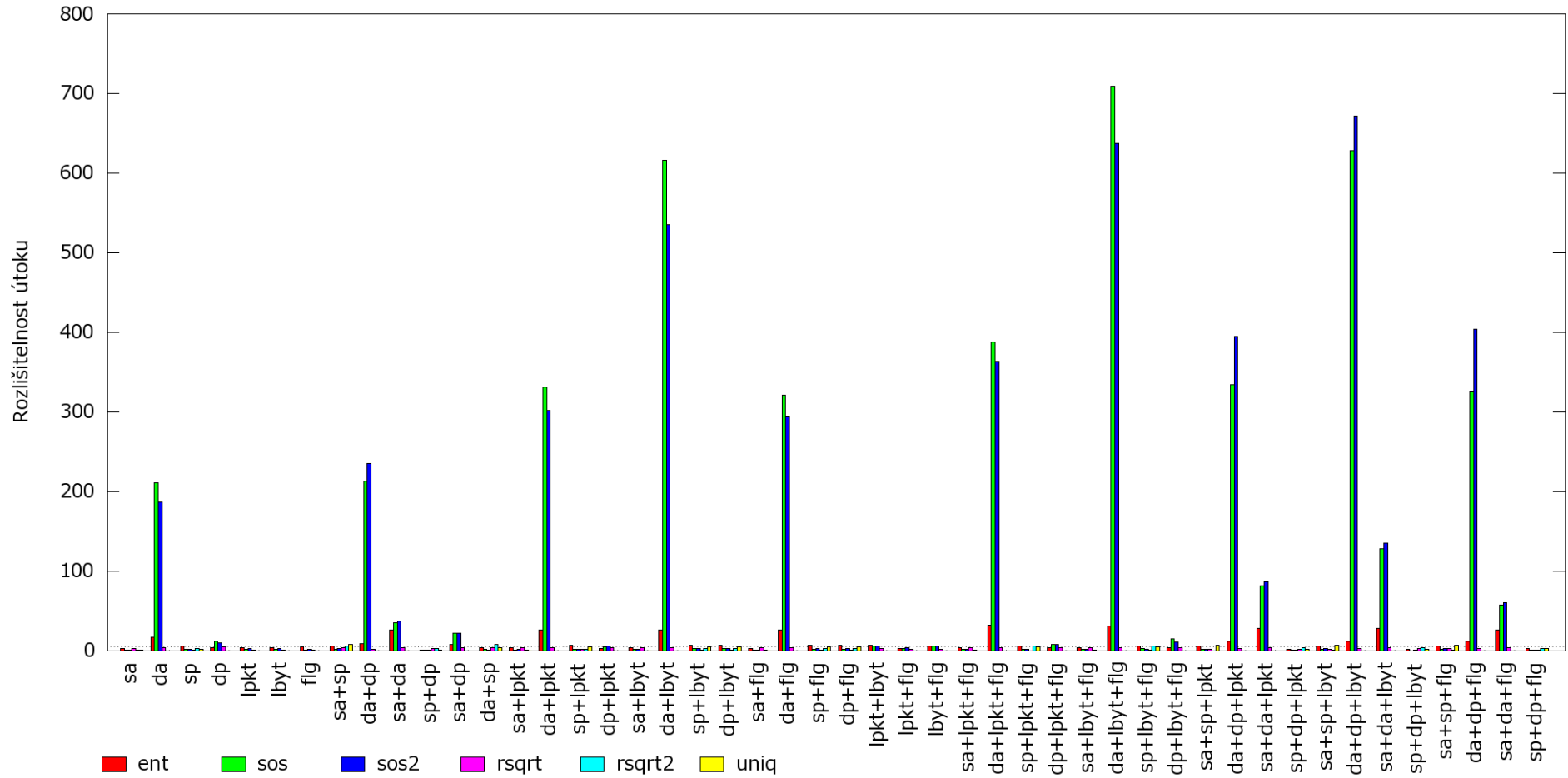
Obrázek C.3: DoS útok, 10 tisíc toků, vzorek dat č. 3

DoS útok, 50k, data 1



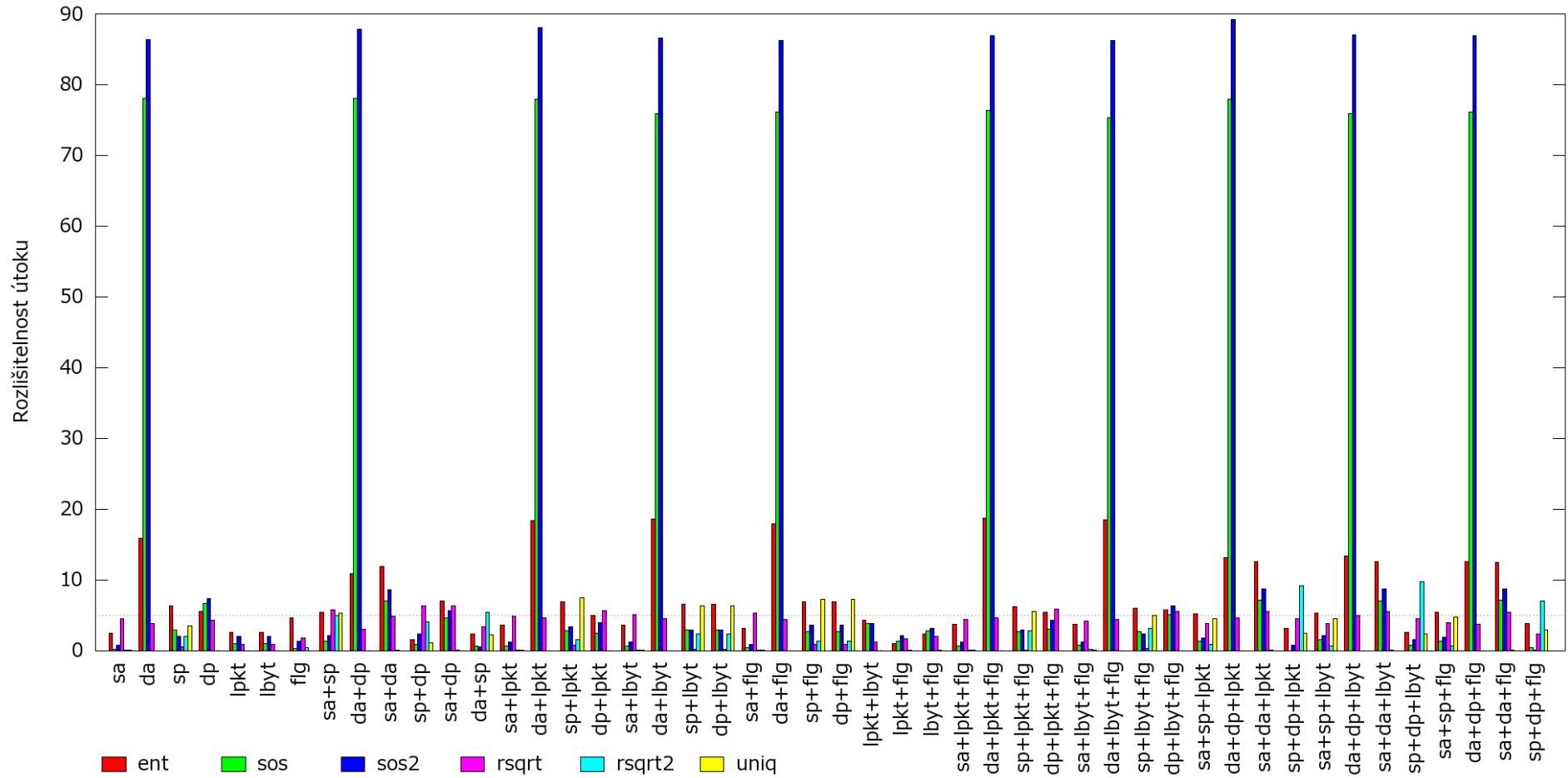
Obrázek C.4: DoS útok, 50 tisíc toků, vzorek dat č. 1

DoS útok, 50k, data 2



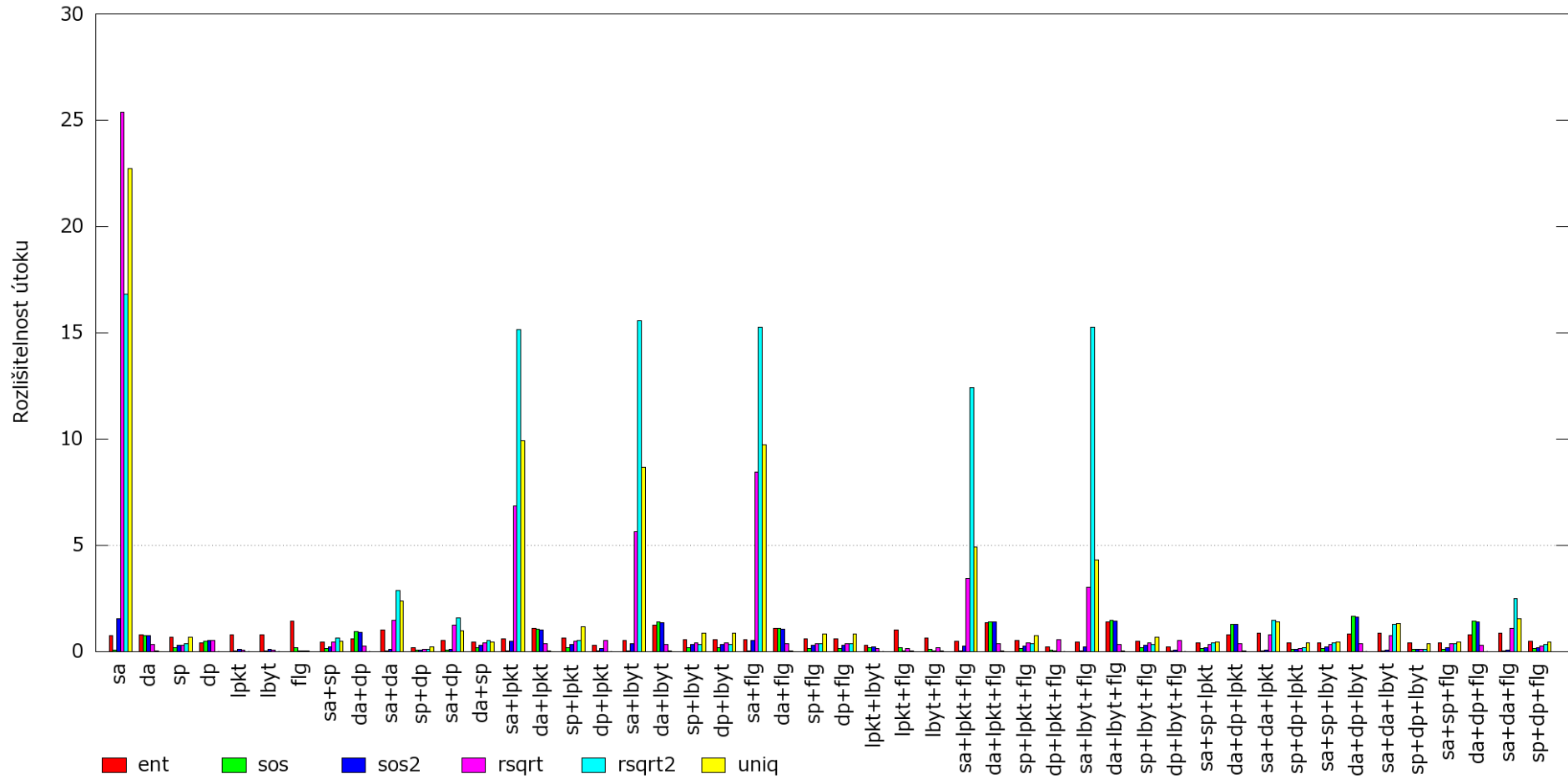
Obrázek C.5: DoS útok, 50 tisíc toků, vzorek dat č.2

DoS útok, 50k, data 3



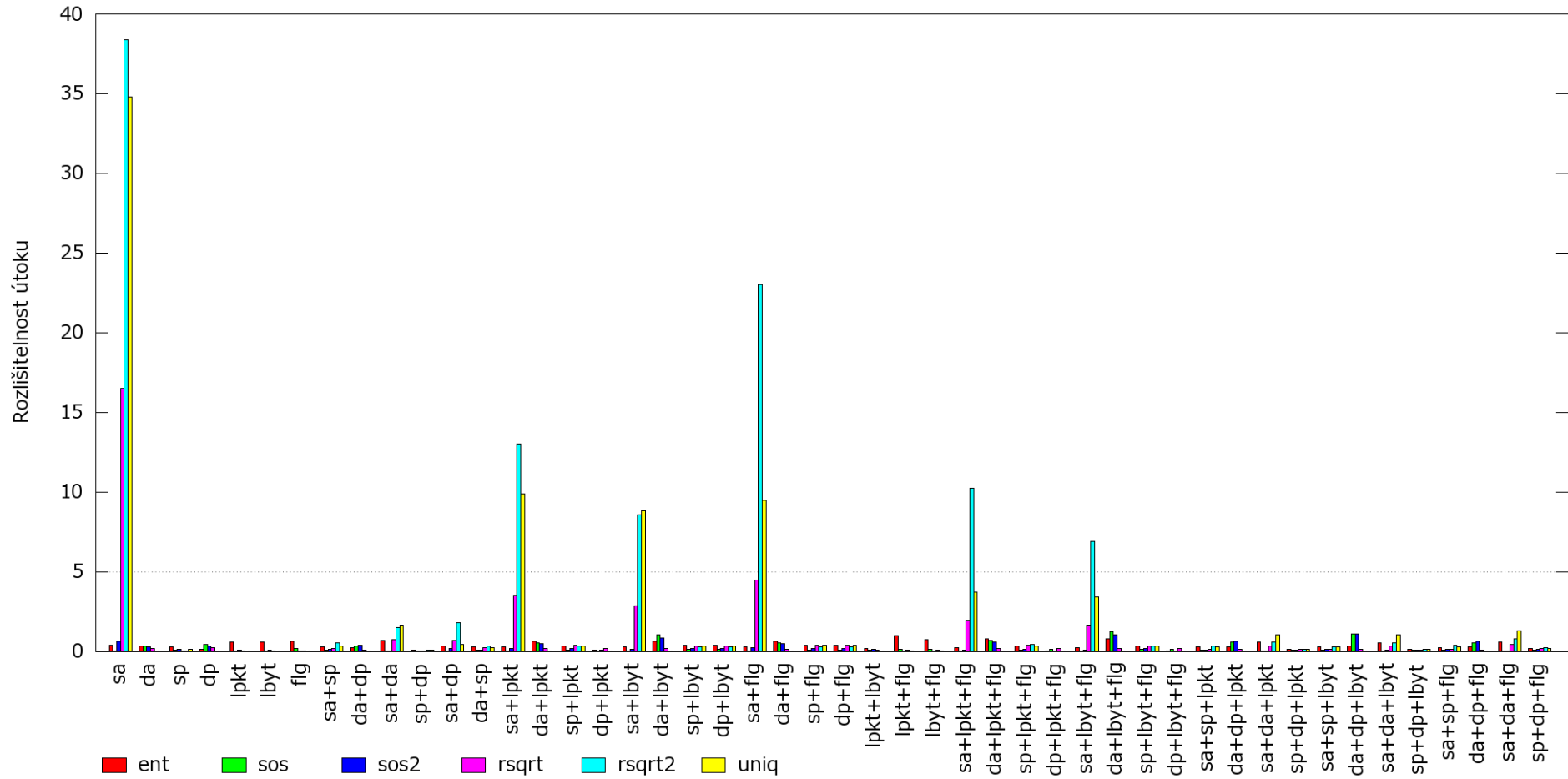
Obrázek C.6: DoS útok, 50 tisíc toků, vzorek dat č. 3

DoS útok s podvrženými adresami, 2k, data 1



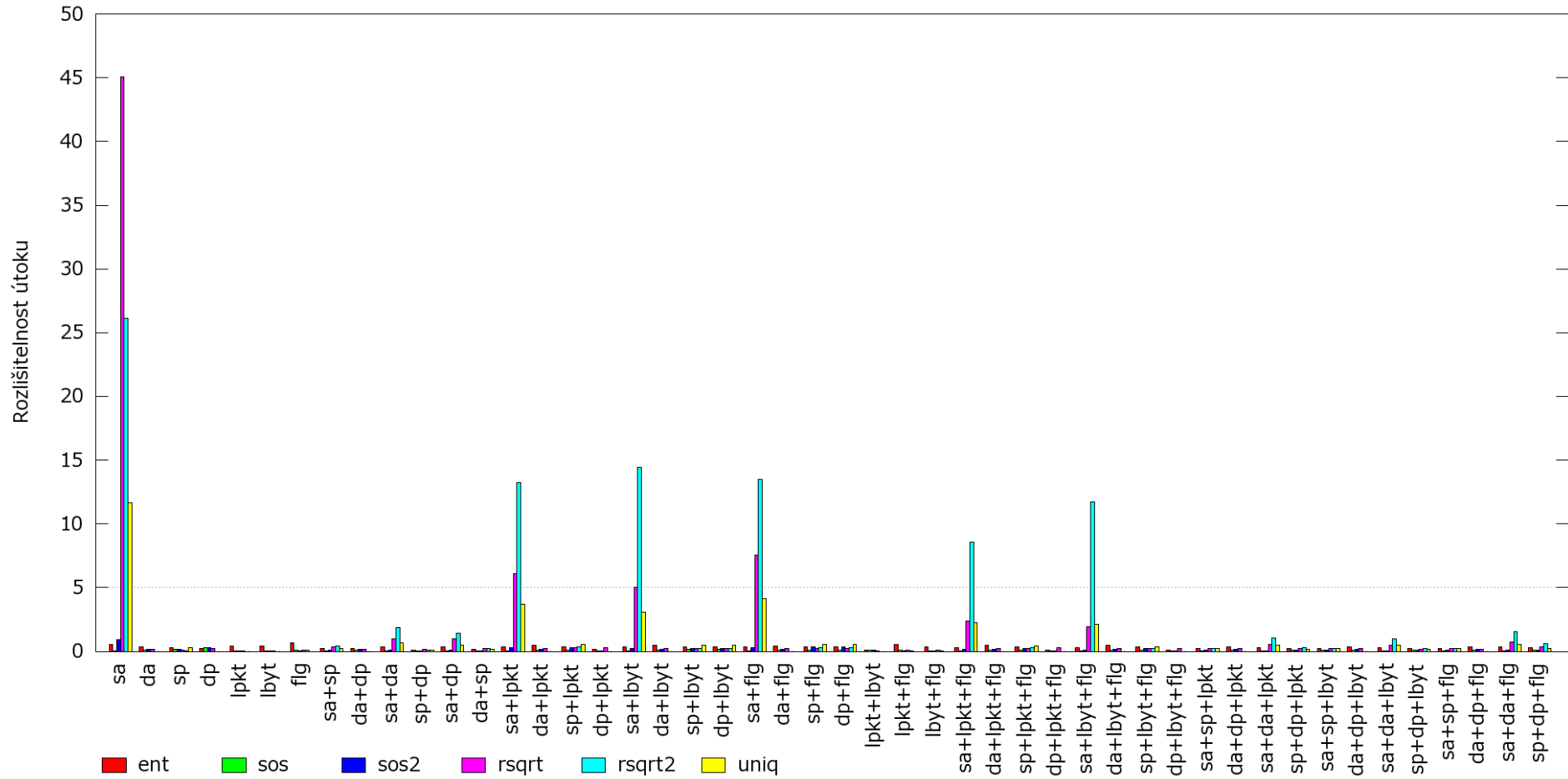
Obrázek C.7: DoS útok s podvrženými zdrojovými adresami, 2 tisíce toků, vzorek dat č. 1

DoS útok s podvrženými adresami, 2k, data 2



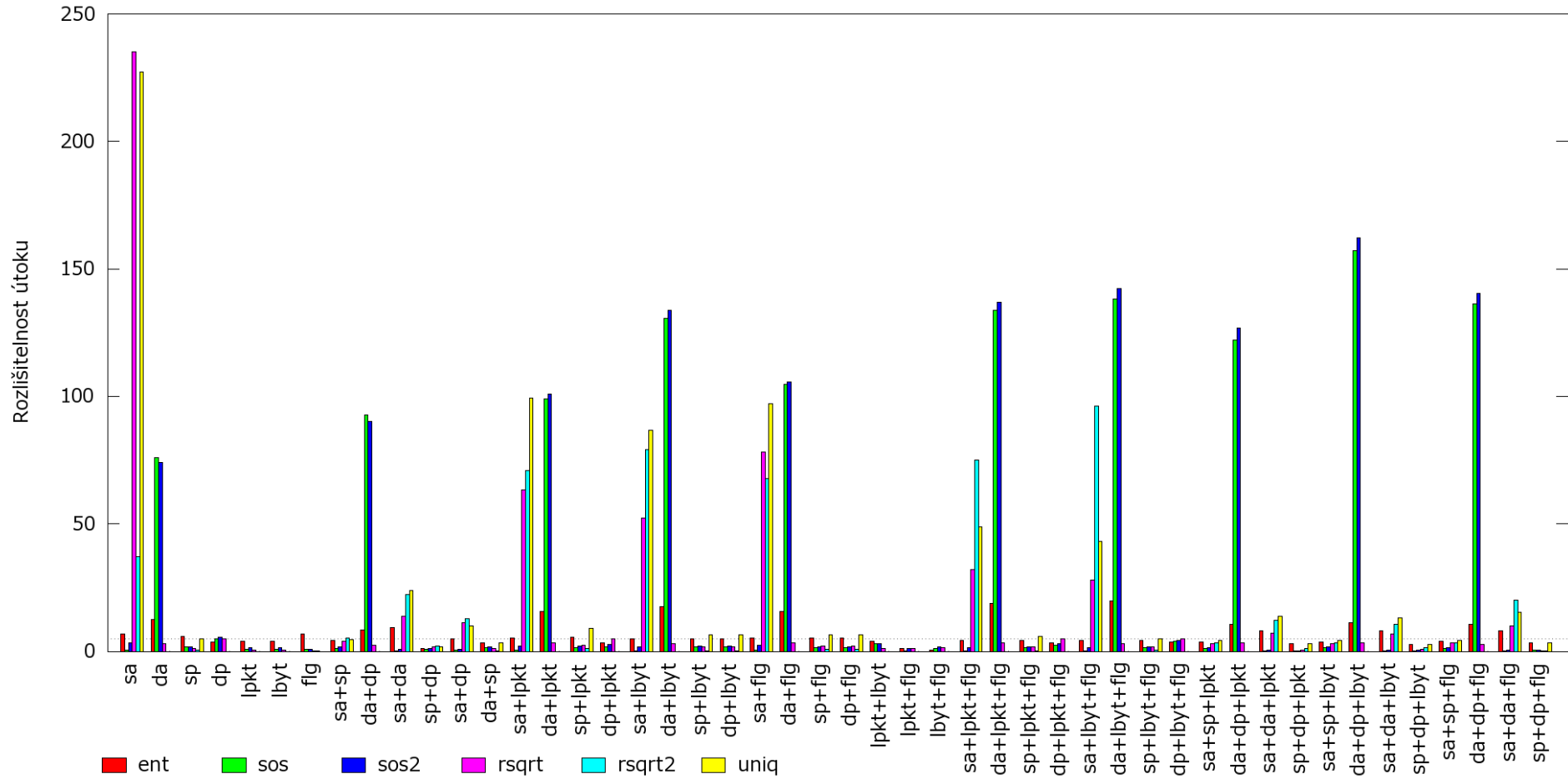
Obrázek C.8: DoS útok s podvrženými zdrojovými adresami, 2 tisíce toků, vzorek dat č. 2

DoS útok s podvrženými adresami, 2k, data 3



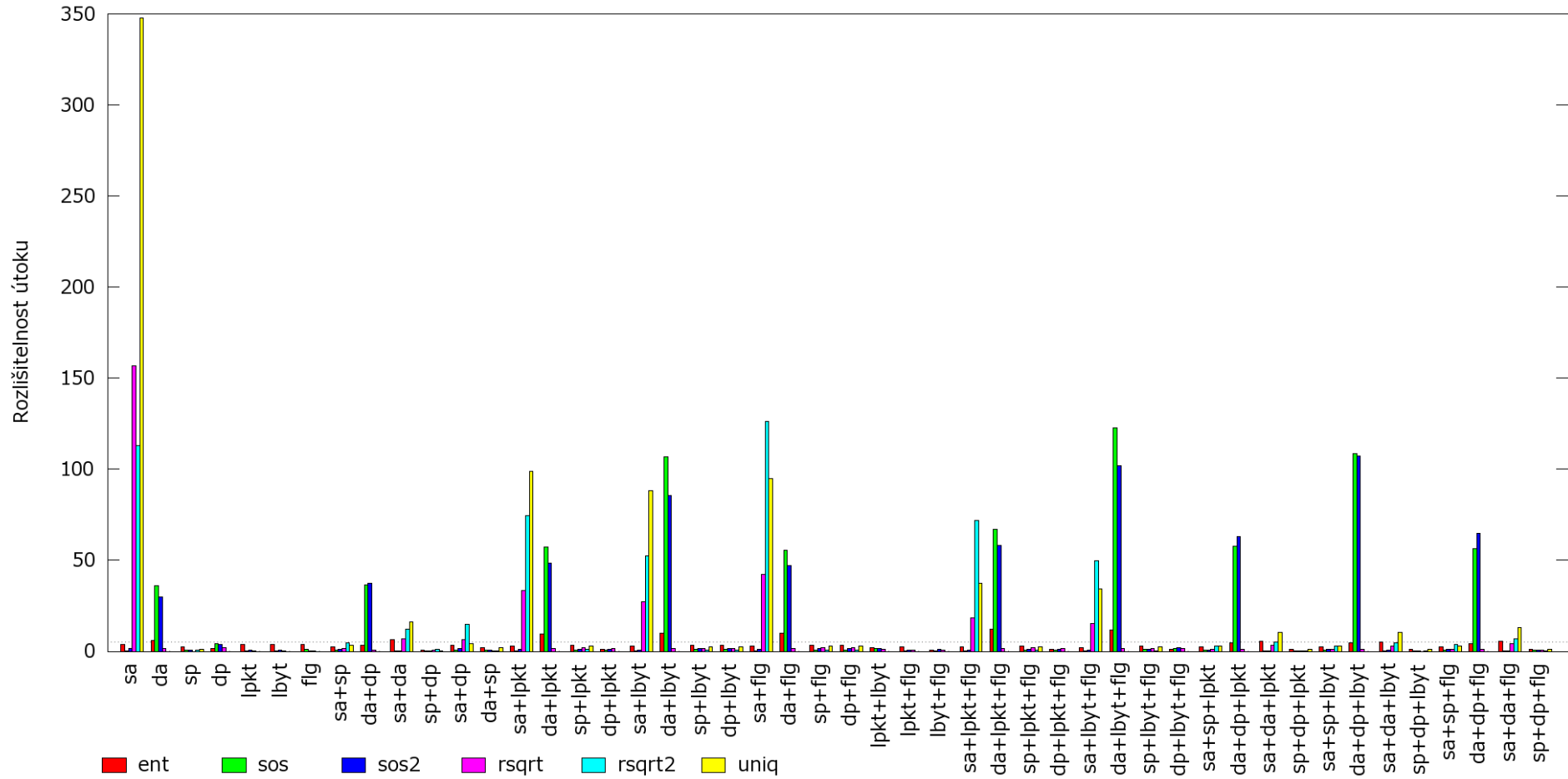
Obrázek C.9: DoS útok s podvrženými zdrojovými adresami, 2 tisíce toků, vzorek dat č. 3

DoS útok s podvrženými adresami, 20k, data 1



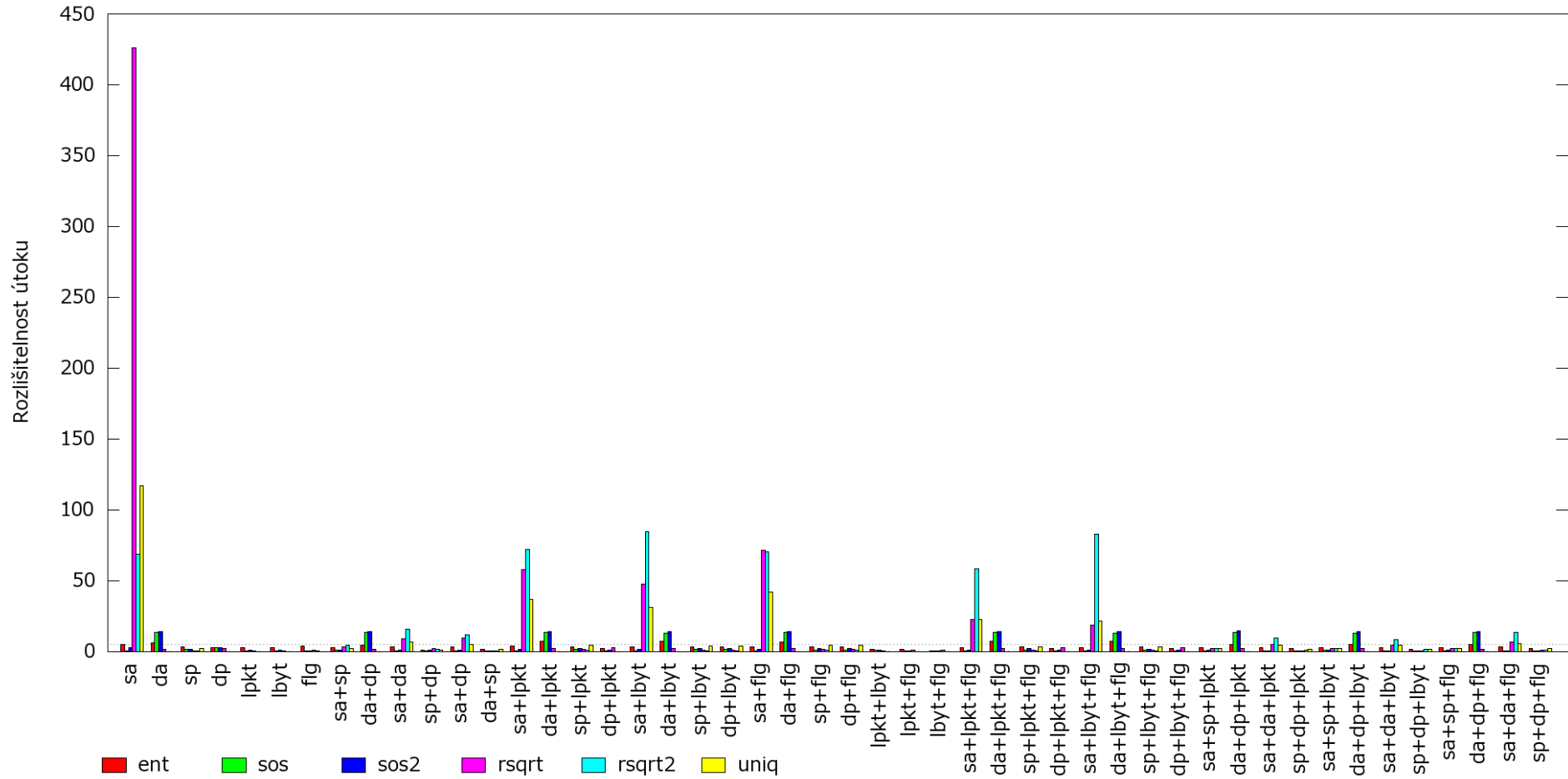
Obrázek C.10: DoS útok s podvrženými zdrojovými adresami, 20 tisíc toků, vzorek dat č. 1

DoS útok s podvrženými adresami, 20k, data 2



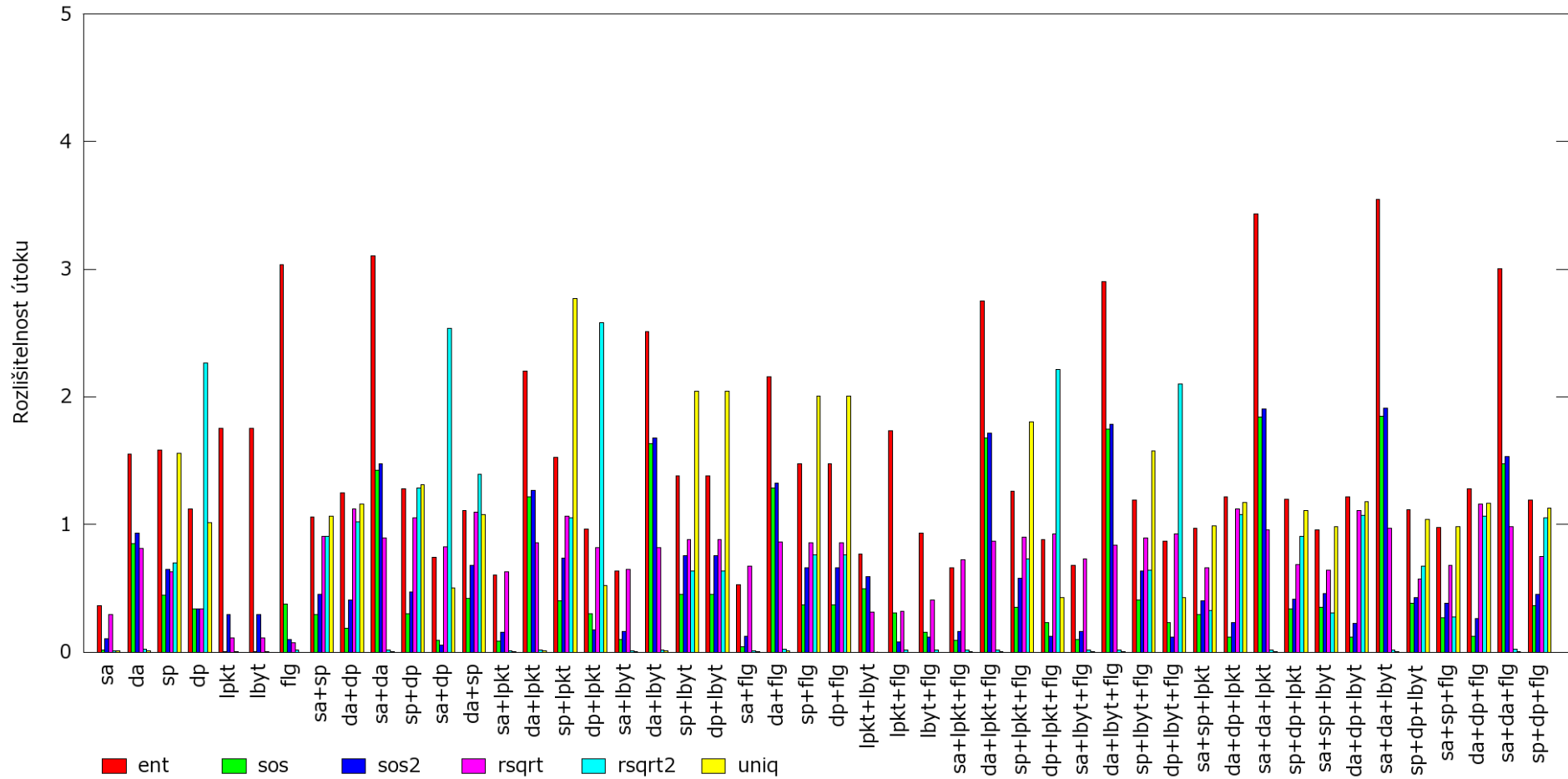
Obrázek C.11: DoS útok s podvrženými zdrojovými adresami, 20 tisíc toků, vzorek dat č. 2

DoS útok s podvrženými adresami, 20k, data 3



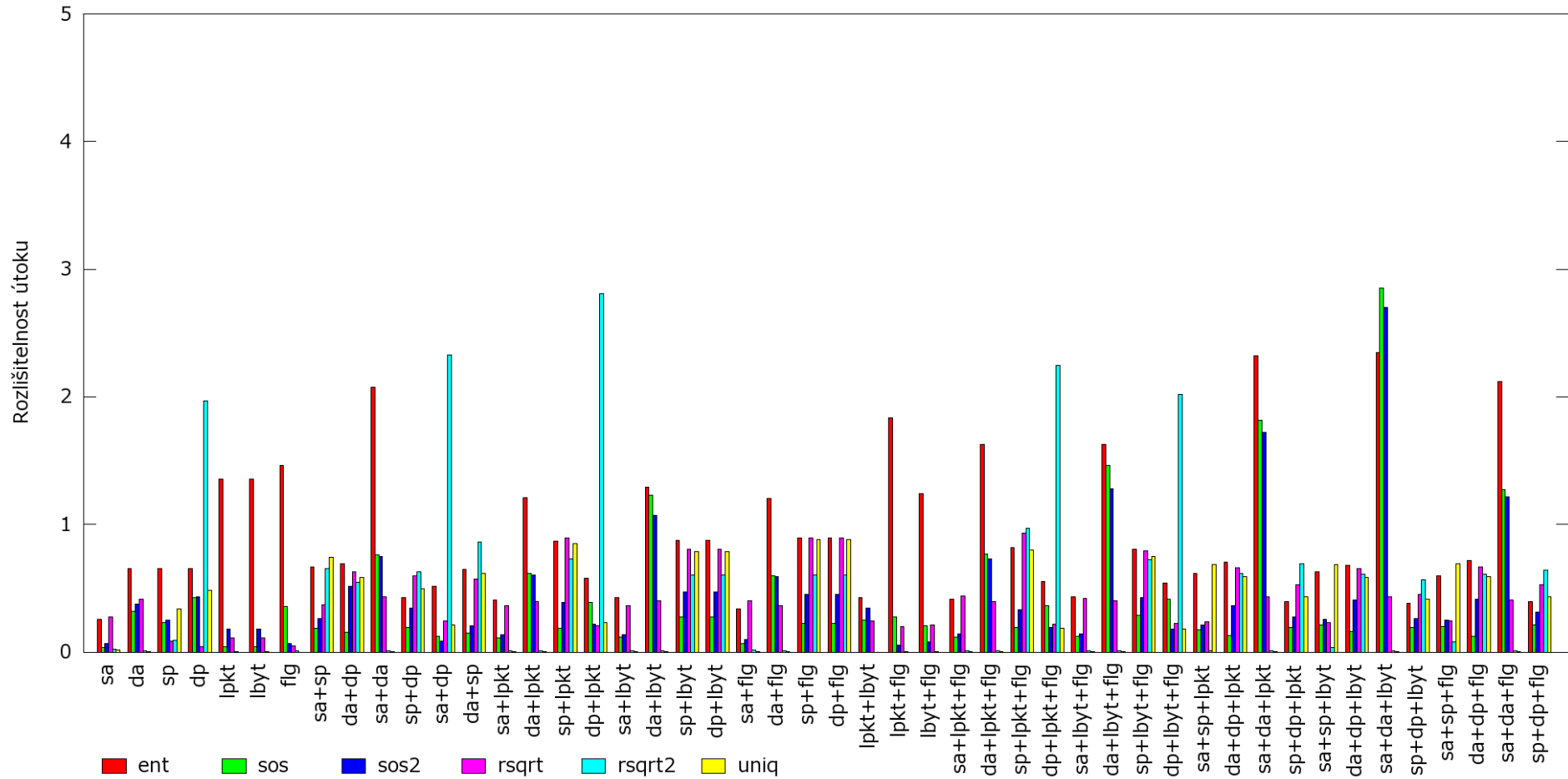
Obrázek C.12: DoS útok s podvrženými zdrojovými adresami, 20 tisíc toků, vzorek dat č. 3

Skenování portů, 5k, data 1



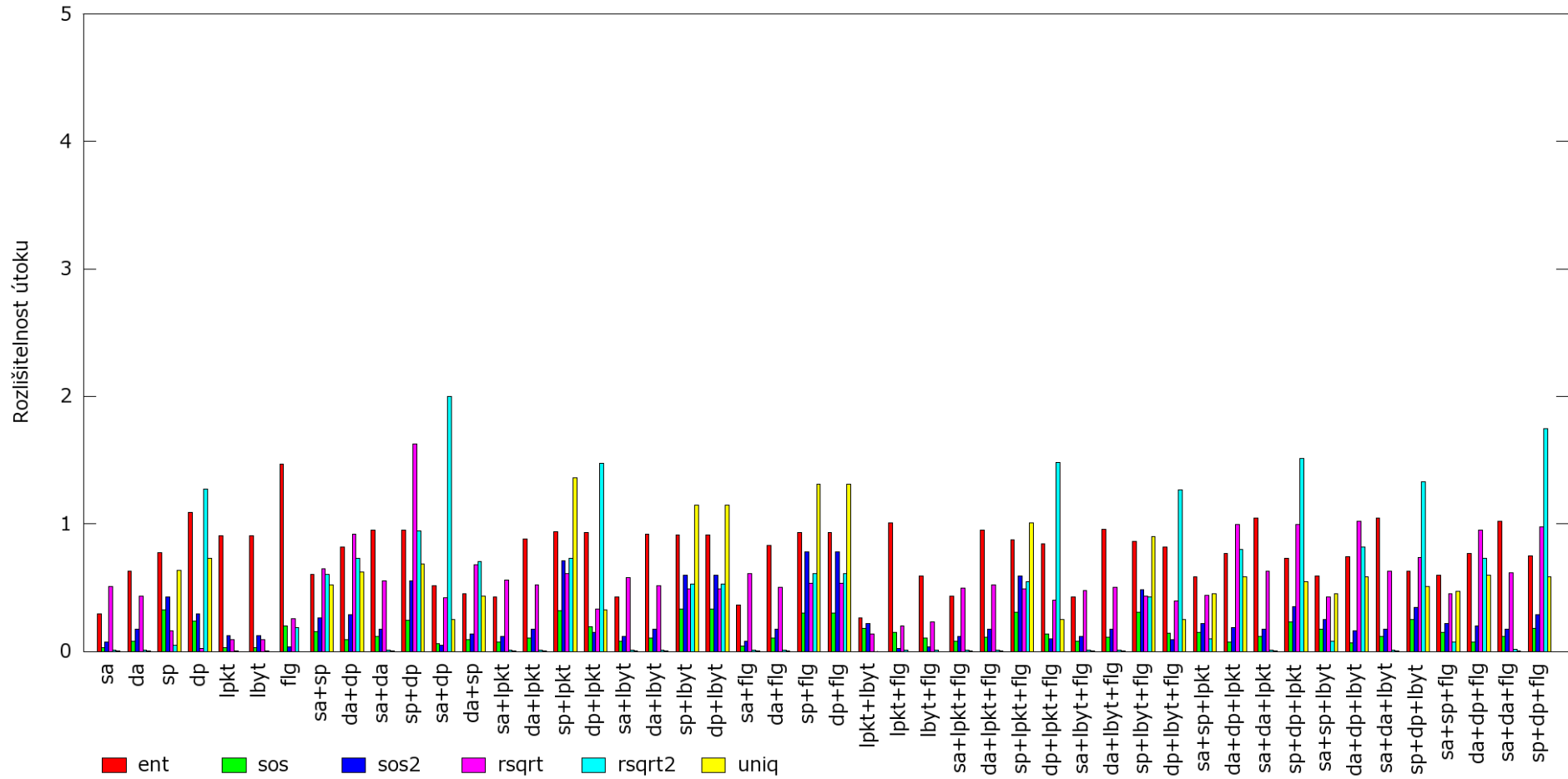
Obrázek C.13: Skenování portů, 5 tisíc toků, vzorek dat č. 1

Skenování portů, 5k, data 2



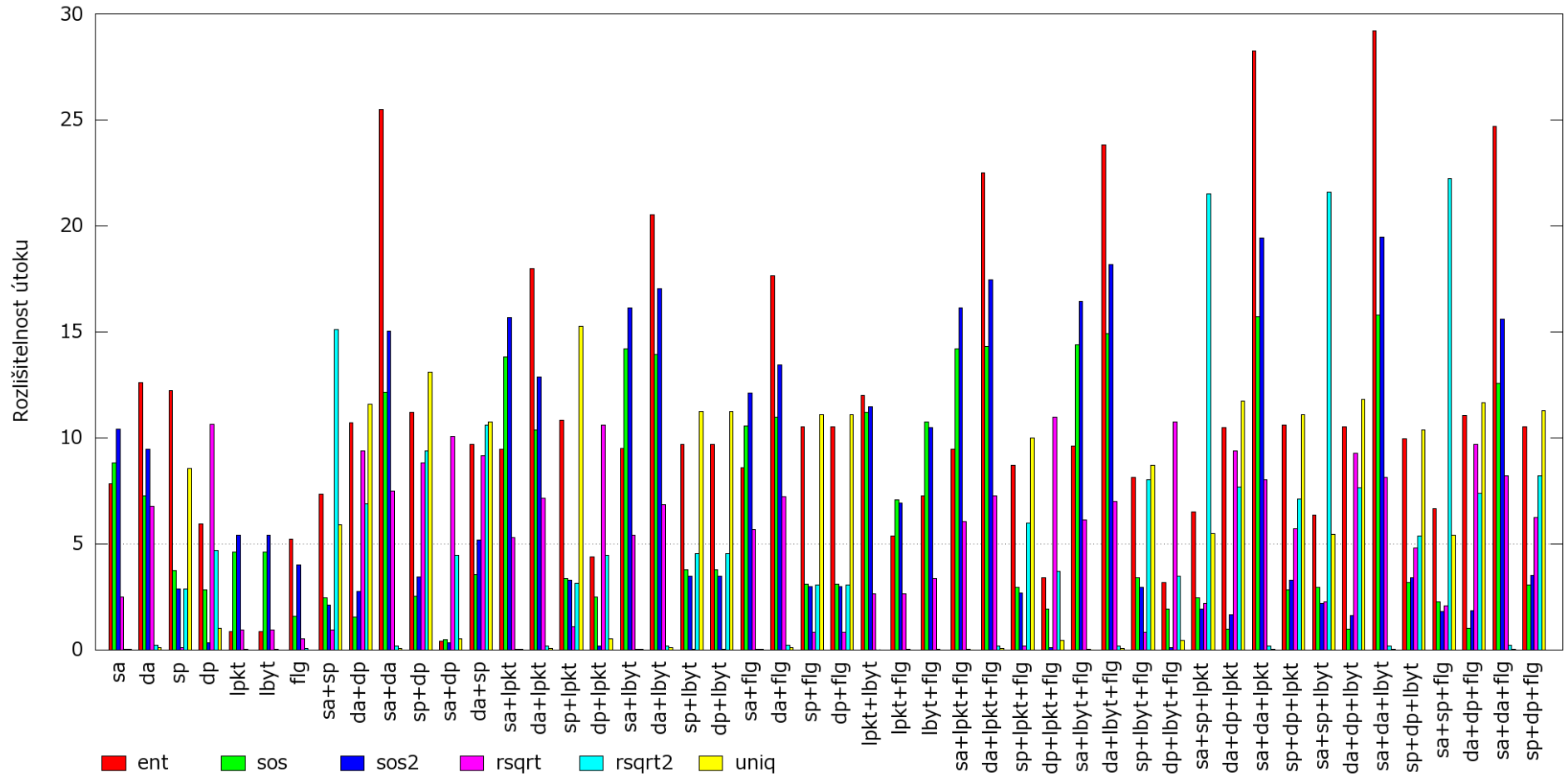
Obrázek C.14: Skenování portů, 5 tisíc toků, vzorek dat č. 2

Skenování portů, 5k, data 3



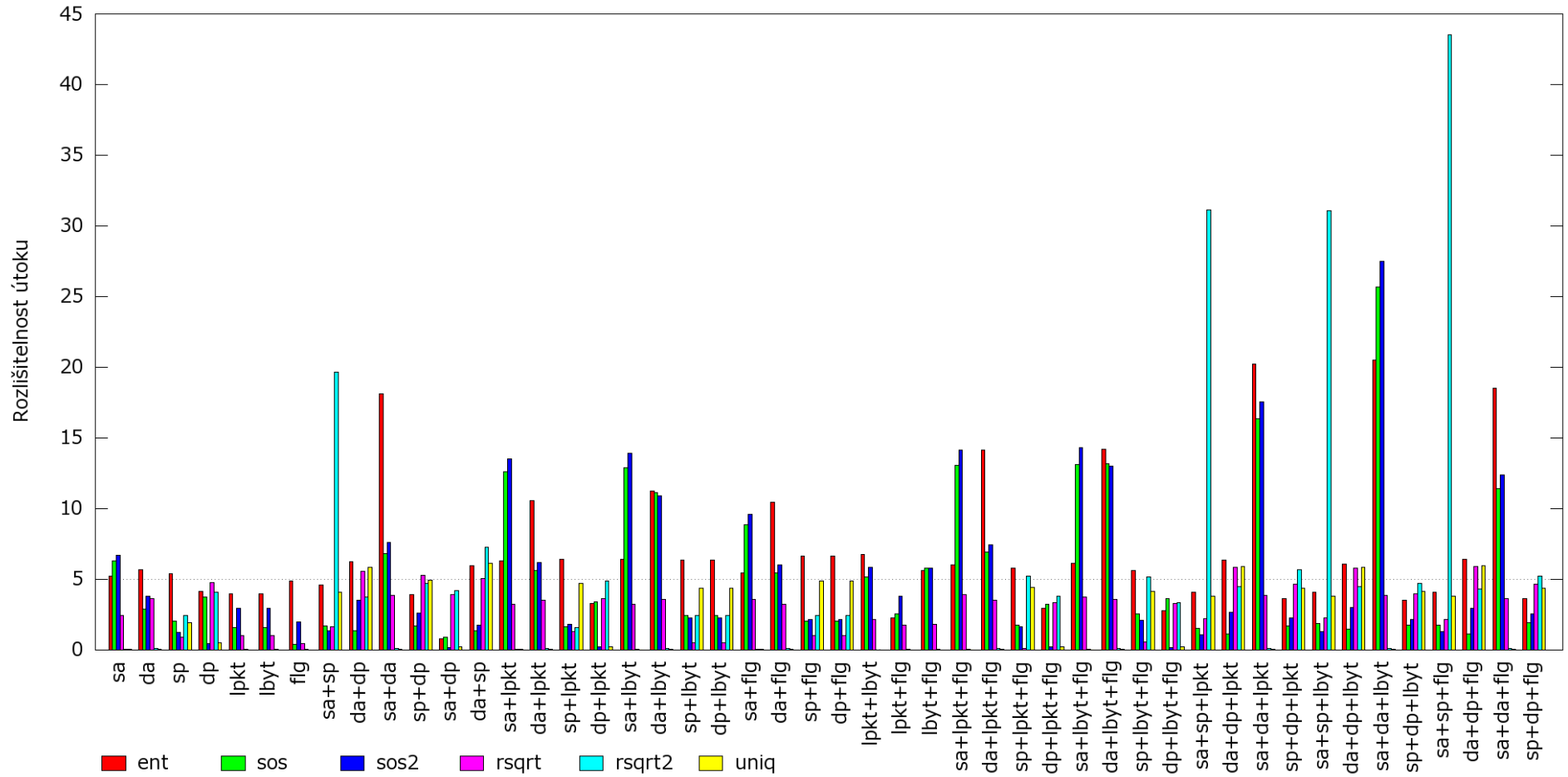
Obrázek C.15: Skenování portů, 5 tisíc toků, vzorek dat č. 3

Skenování portů, 50k, data 1



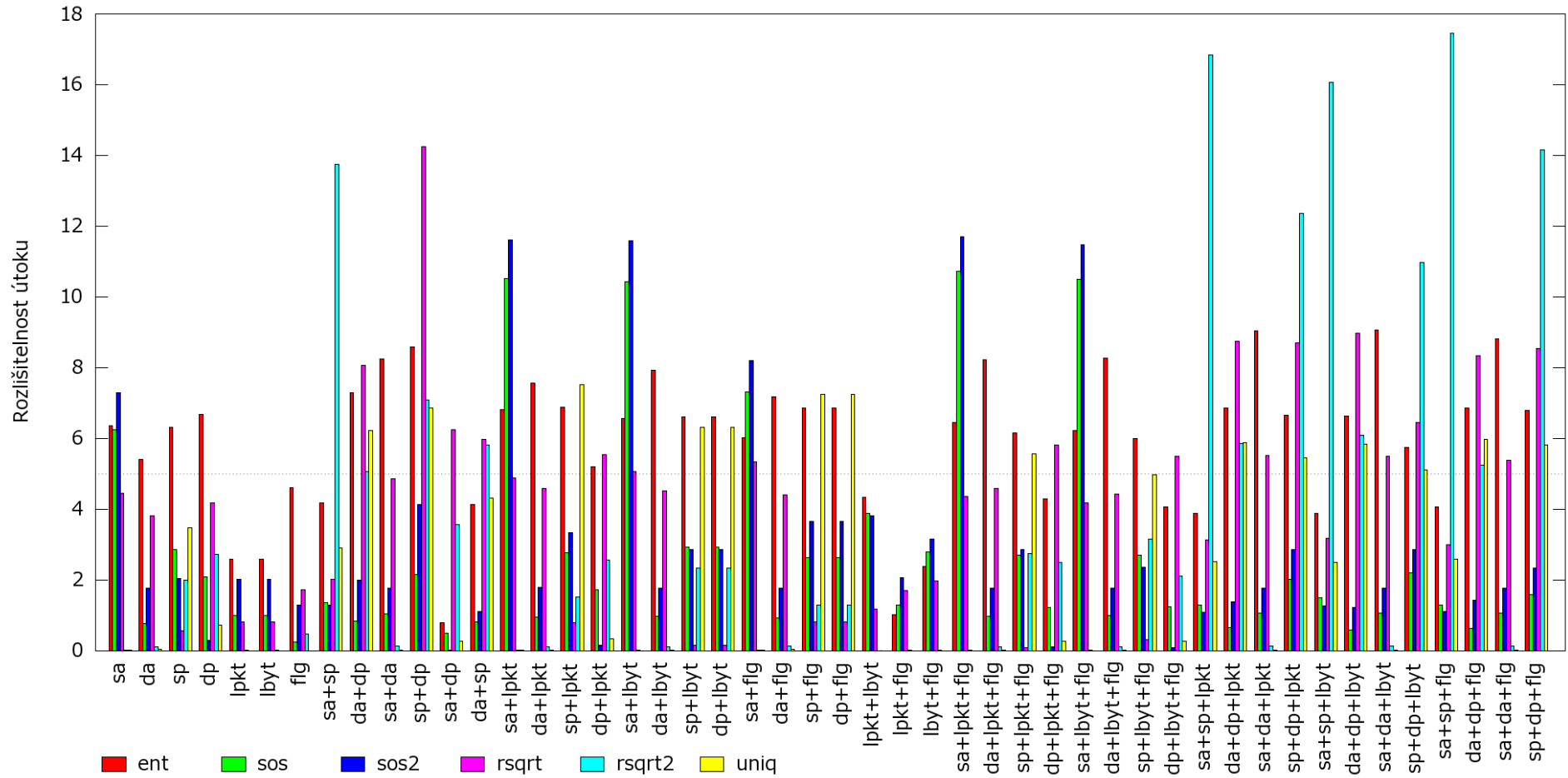
Obrázek C.16: Skenování portů, 50 tisíc toků, vzorek dat č. 1

Skenování portů, 50k, data 2



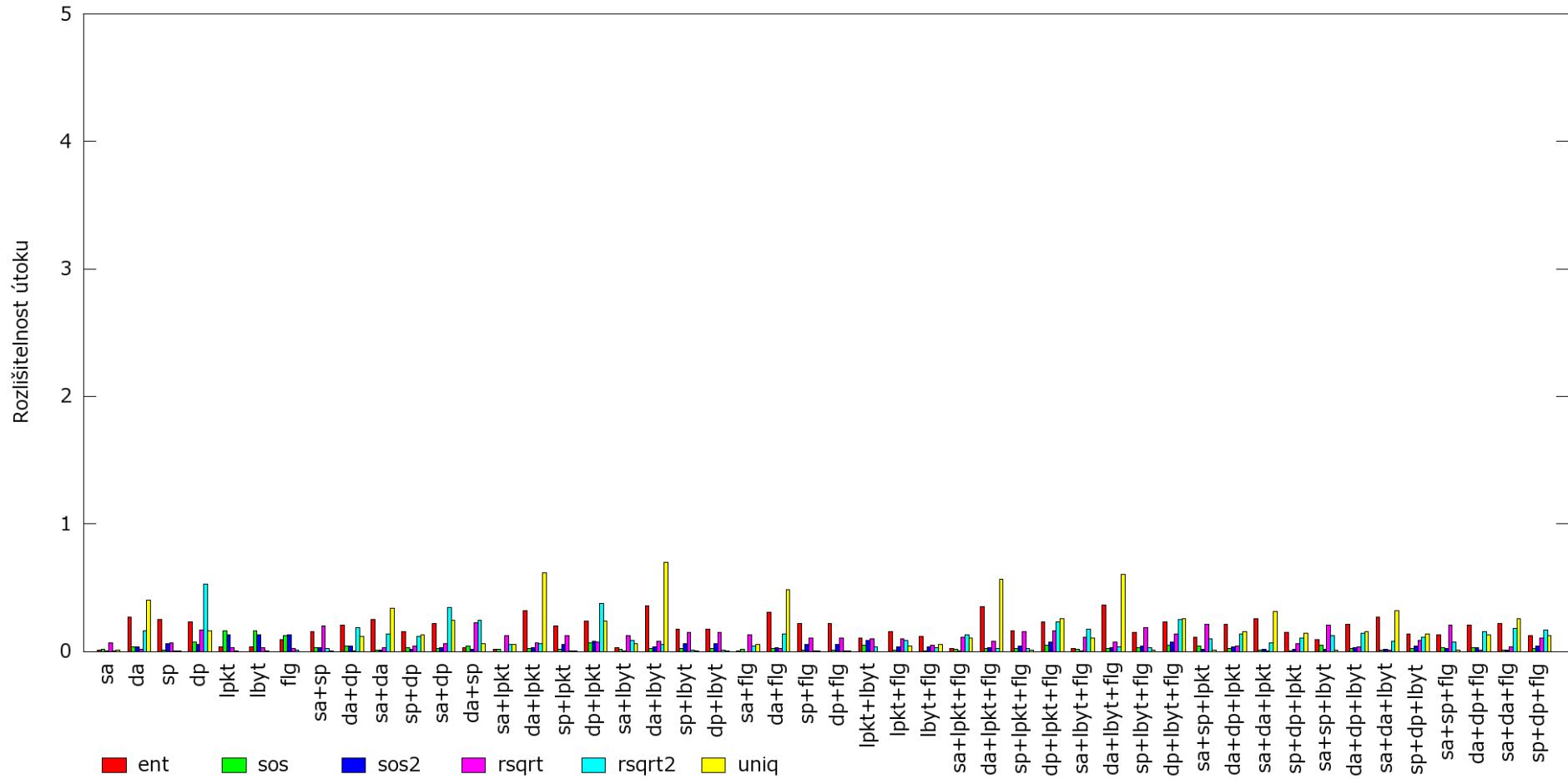
Obrázek C.17: Skenování portů, 50 tisíc toků, vzorek dat č. 2

Skenování portů, 50k, data 3



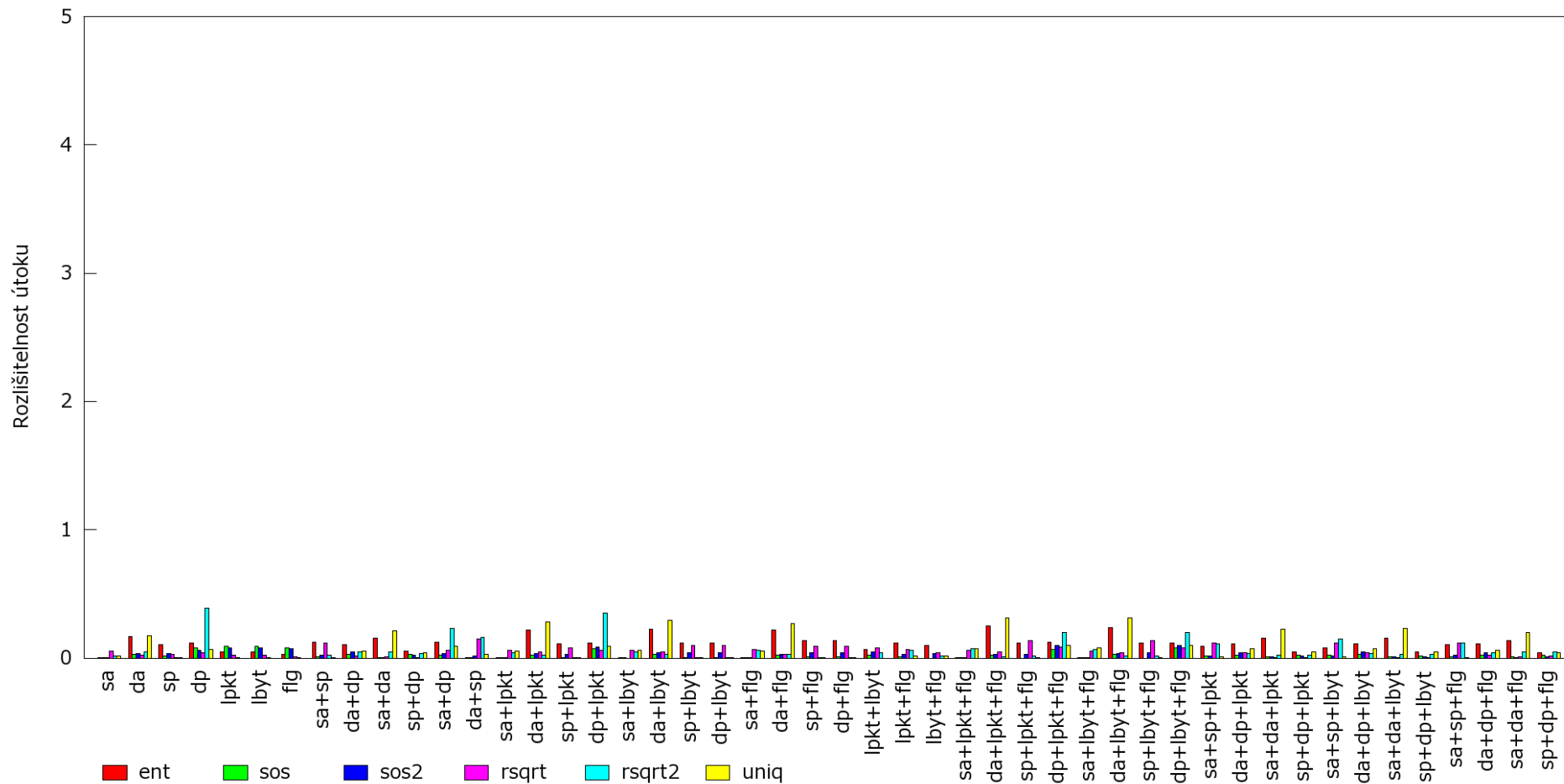
Obrázek C.18: Skenování portů, 50 tisíc toků, vzorek dat č. 3

Výpadek serveru 1, data 1



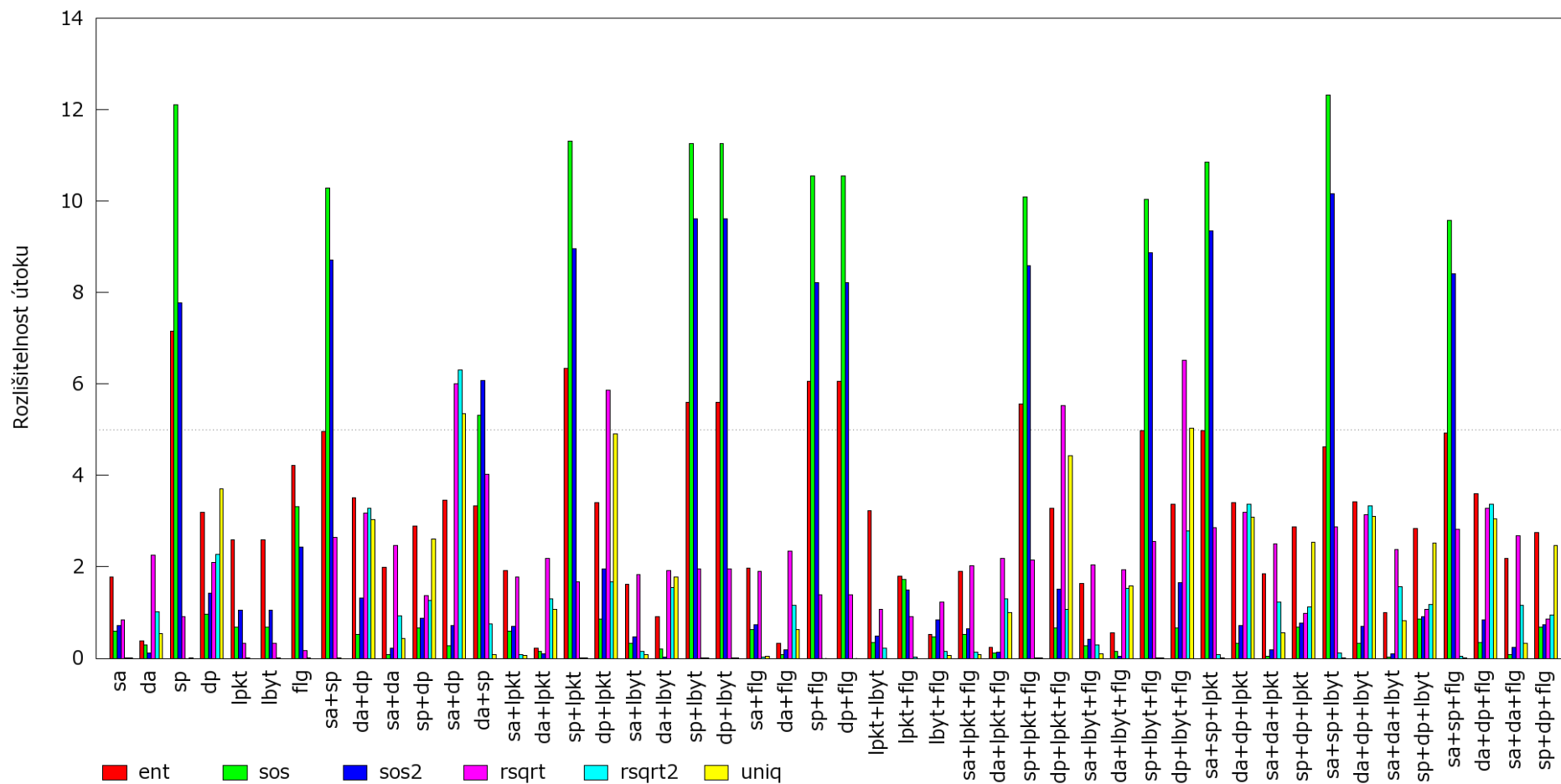
Obrázek C.19: Výpadek serveru 1, vzorek dat č. 1

Výpadek serveru 1, data 2



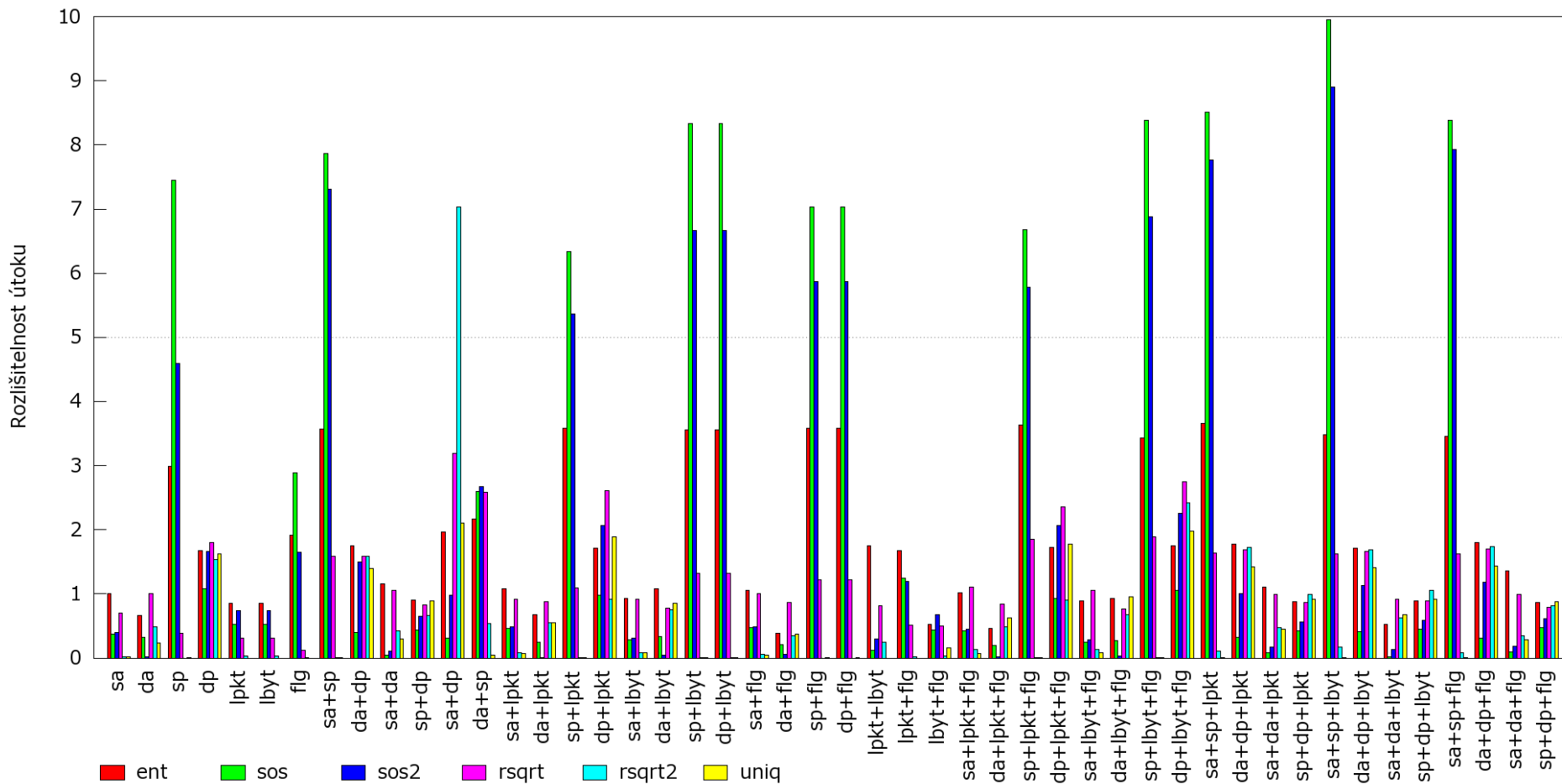
Obrázek C.20: Výpadek serveru 1, vzorek dat č. 2

Výpadek serveru 2, data 1



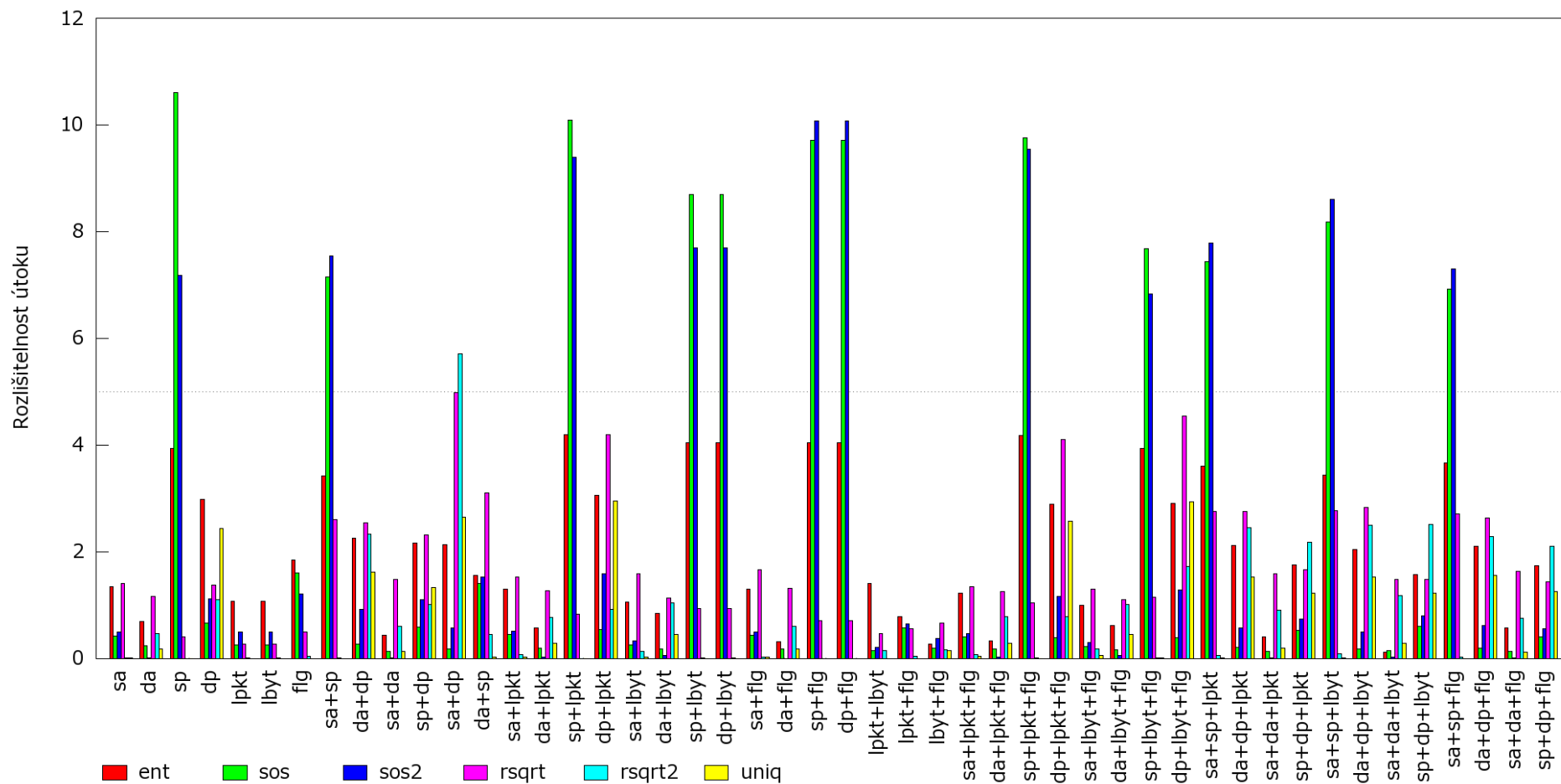
Obrázek C.22: Výpadek serveru 2, vzorek dat č. 1

Výpadek serveru 2, data 2



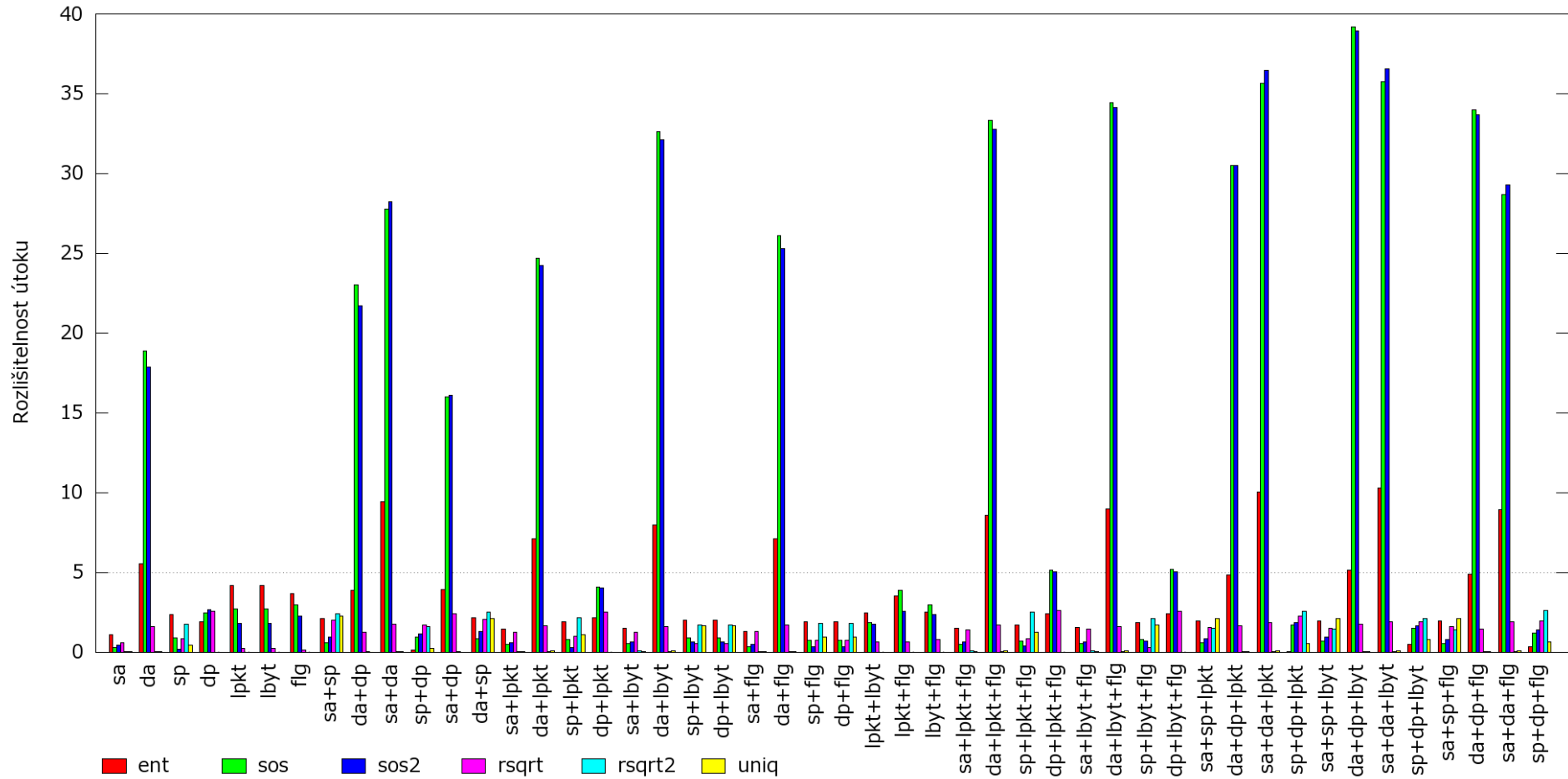
Obrázek C.23: Výpadek serveru 2, vzorek dat č. 2

Výpadek serveru 2, data 3



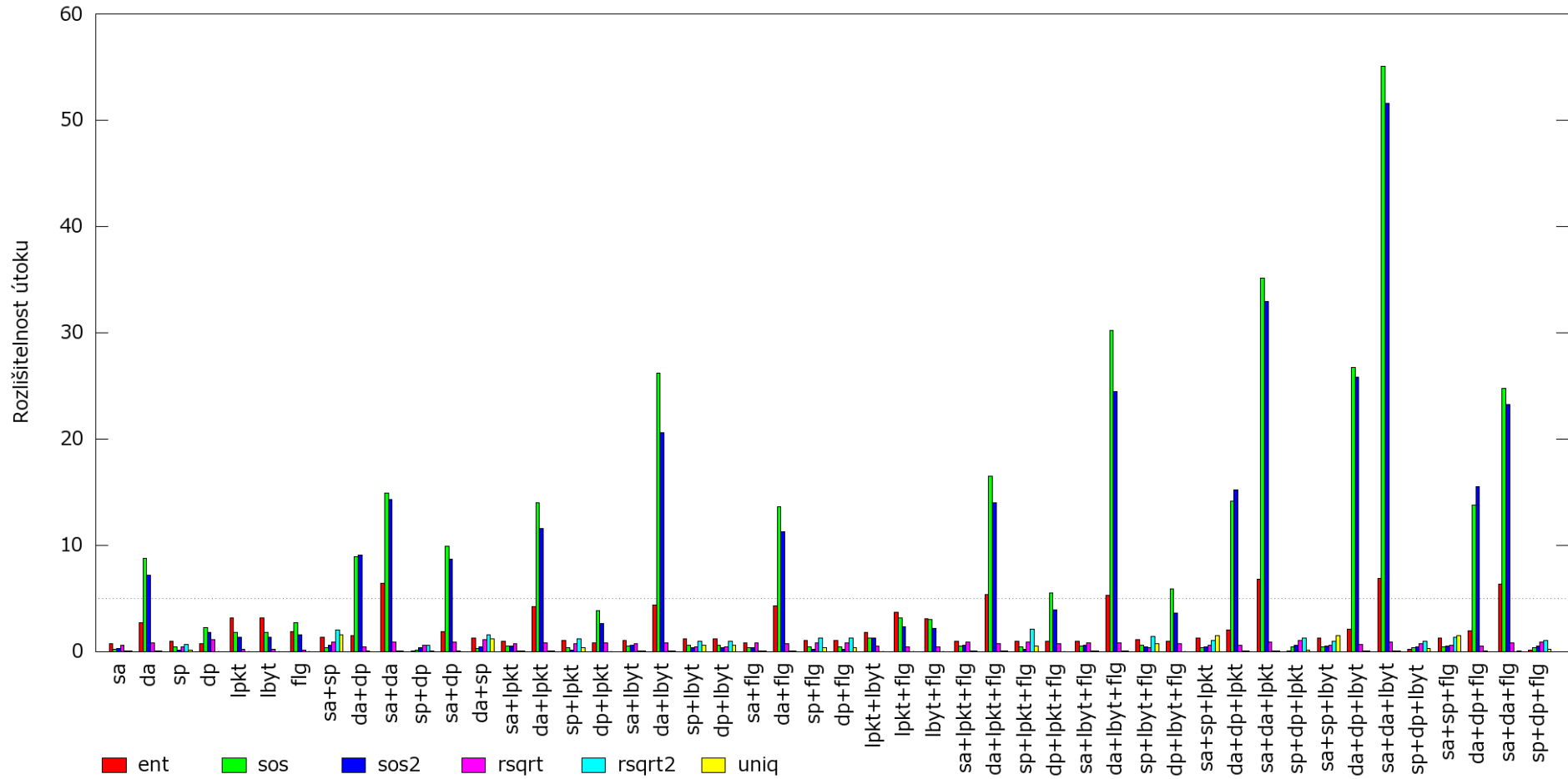
Obrázek C.24: Výpadek serveru 2, vzorek dat č. 3

Reálný DoS útok na web server, data 1



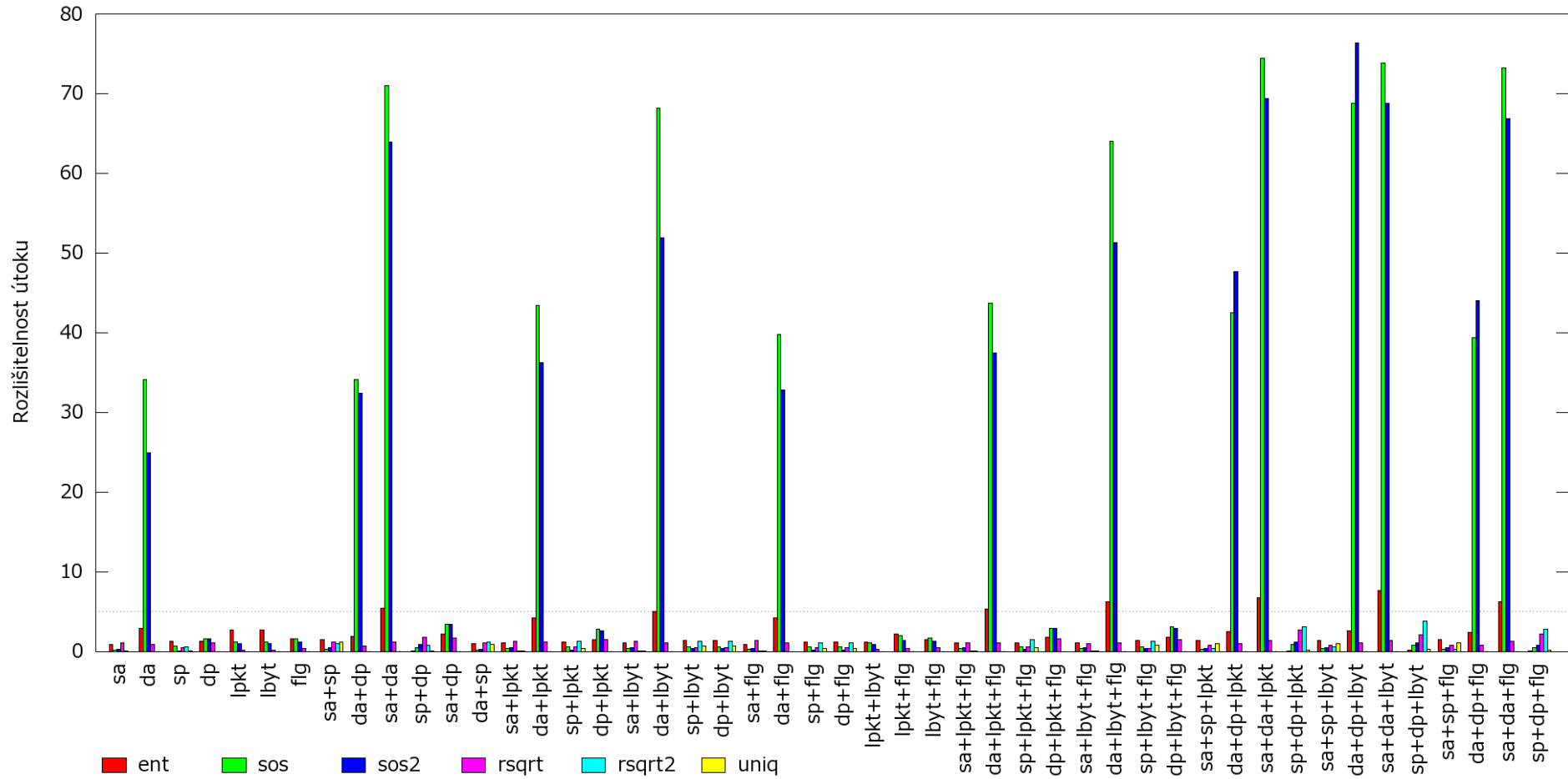
Obrázek C.25: Anomalie z reálných dat, pravděpodobně DoS útok na web server, vzorek dat č. 1

Reálný DoS útok na web server, data 2



Obrázek C.26: Anomálie z reálných dat, pravděpodobně DoS útok na web server, vzorek dat č. 2

Reálný DoS útok na web server, data 3



Obrázek C.27: Anomálie z reálných dat, pravděpodobně DoS útok na web server, vzorek dat č. 3