VŠB - Technická univerzita Ostrava
Fakulta strojní
Katedra automatizační techniky a řízení

Vývoj a nasazení SW prostředků pro analýzu přístupů do sítě VŠB-TU Ostrava

Development and Deployment of Software Tools for Analyzing of Network Access at VŠB-TU Ostrava

Vedoucí b.p.: Ing. Fojtík David, Ph.D.
Konzultant: Ing. Martin Pustka, Ph.D.
Řešitel: Jakub Kalnik

Ostrava 2017

VŠB - Technická univerzita Ostrava
Fakulta strojní
Katedra automatizační techniky a řízení

Zadání bakalářské práce

Student: **Jakub Kalnik**

Studijní program: B2341 Strojírenství

Studijní obor: 3902R001 Aplikovaná informatika a řízení

Téma: Vývoj a nasazení SW prostředků pro analýzu přístupů do sítě VŠB-TU Ostrava
Development and Deployment of Software Tools for Analyzing of Network Access at VŠB-TU Ostrava

Jazyk vypracování: čeština

Zásady pro vypracování:

1. Nastudujte a popište základní principy ověřování uživatelů v sítích protokolem RADIUS.
2. Navrhněte a realizujte systém pro sběr informací z Radius serverů sítě VŠB, ukládání dat realizujte s využitím SQL databáze.
3. Navrhněte a realizujte webovou aplikaci pro uživatelskou analýzu dat v SQL databázi, kterou bude využívat správa sítě pro běžný provoz.
4. Zhodnoťte výsledky své práce a navrhněte další rozvoj systému.

Seznam doporučené odborné literatury:

FARANA, R., SMUTNÝ, L., VÍTEČEK, A. 1999. Zpracování odborných textů z oblasti automatizace a informatiky. 1. vyd. Ostrava: VŠB-TU Ostrava, 1999. 68 s. ISBN 80-7078-737-6.
MOLINARO, A. SQL: kuchařka programátora. Přeložil Jakub MUŽÍK. Brno: Computer Press, 2009. ISBN 978-80-251-2617-2.
SATRAPA, P. Perl pro zelenáče. Praha: Neocortex, c2000. ISBN 80-86330-02-8.
SATRAPA, P. IPV6: internetový protokol verze 6. 3., aktualiz. a dopl. vyd. Praha: CZ.NIC, c2011. ISBN 978-80-904248-4-5.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. David Fojtík, Ph.D.**

Datum zadání: 09.12.2016

Datum odevzdání: 15.05.2017

Renata Wag

doc. Ing. Renata Wagnerová, Ph.D.
vedoucí katedry

doc. Ing. Ivo Hlavatý, Ph.D.
děkan fakulty

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA Fakulta strojní 3 OSTRAVA

Místopřísežné prohlášení studenta

Prohlašuji, že jsem celou bakalářskou práci včetně příloh vypracoval samostatně pod vedením vedoucího bakalářské práce a uvedl jsem všechny použité podklady a literaturu.

V Ostravě 15. 5. 2017 .....

.....
podpis studenta

Prohlašuji, že

- jsem byl seznámen s tím, že na moji diplomovou (bakalářskou) práci se plně vztahuje zákon č.121/2000 Sb., autorský zákon, zejména § 35 – užití díla v rámci občanských a náboženských obřadů, v rámci školních představení a užití díla školního a § 60 – školní dílo.
- beru na vědomí, že Vysoká škola báňská – Technická univerzita Ostrava (dále jen „VŠB-TUO“) má právo nevýdělečně ke své vnitřní potřebě diplomovou (bakalářskou) práci užít (§ 35 odst. 3).
- souhlasím s tím, že diplomová (bakalářská) práce bude v elektronické podobě uložena v Ústřední knihovně VŠB-TUO k nahlédnutí a jeden výtisk bude uložen u vedoucího diplomové (bakalářské) práce. Souhlasím s tím, že údaje o kvalifikační práci budou zveřejněny v informačním systému VŠB-TUO.
- bylo sjednáno, že s VŠB-TUO, v případě zájmu z její strany, uzavřu licenční smlouvu s oprávněním užít dílo v rozsahu § 12 odst. 4 autorského zákona.
- bylo sjednáno, že užít své dílo – diplomovou (bakalářskou) práci nebo poskytnout licenci k jejímu využití mohu jen se souhlasem VŠB-TUO, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly VŠB-TUO na vytvoření díla vynaloženy (až do jejich skutečné výše).
- beru na vědomí, že odevzdáním své práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, bez ohledu na výsledek její obhajoby.

V Ostravě 15.5.2017

Jakub Kalnik
podpis

Jakub Kalnik
Fügnerova 313
Bohumín 1, 735 81

Anotace

Kalnik, J. *Vývoj a nasazení SW prostředků pro analýzu přístupů do sítě VŠB-TU Ostrava: bakalářská práce*. Ostrava: VŠB – Technická univerzita Ostrava, Fakulta strojní, Katedra automatizační techniky a řízení, 2017, 40 s. Vedoucí práce: Fojtík, D.

Tato bakalářská práce se zabývá analýzou přístupů na síť VŠB-TU, což je rozsáhlá počítačová síť poskytující služby tisícům uživatelů, stanic a mobilních zařízení. Do analýzy spadá identifikace klíčových údajů, jejich získání, ukládání a prezentace koncovému uživateli. V práci je popsán princip autentizace uživatelů v této síti a problematika získání požadovaných informací. Dále je v práci provedena analýza rozšiřitelnosti již nasazených softwarových prostředků. Praktická část se zaměřuje na konfiguraci produkčních serverů a vývoj nových softwarových prostředků, které budou zajišťovat požadované funkcionality.

Klíčová slova: RADIUS, síť, přístup, IPv6, EDUROAM

Annotation

Kalnik, J. *Development and Deployment of Software Tools for Analyzing of Network Access at VŠB-TU Ostrava: Bachelor Thesis*. Ostrava: VŠB – Technical University of Ostrava, Faculty of Mechanical Engineering, Department of Control Systems and Instrumentation, 2017, 40 p. Head: Fojtík, D.

This thesis deals with analysis of accesses into VŠB-TU network which provides services to thousands of users, workstations and mobile devices. Analysis includes identification, acquisition, storage and presentation of valuable data. The thesis explains how the users authenticate and how to collect the data. There is also analysis of extensibility of currently used software. The practical part mainly focuses on production servers configuration and development of new software which will provide requested functionalities.

Keywords: RADIUS, network, access, IPv6, EDUROAM

Obsah

Seznam použitých zkratek

AAA	*Authentication, Authorization, and Accounting* Zkratka pro autentizaci, autorizaci a účtování
ARP	*Address Resolution Protocol* Protokol pro dohledání MAC adresy na základě znalosti IPv4 adresy
API	*Application Programming Interface* Rozhraní využívané programátory pro tvorbu aplikací
CAS	*Central Authentication Service* Protokol umožňující přistupovat k více službám na základě jednoho přihlášení
DHCP	*Dynamic Host Configuration Protocol* Protokol umožňující automatizované přidělování IPv4 adres
DHCPv6	*Dynamic Host Configuration Protocol version 6* Protokol umožňující automatizované přidělování IPv6 adres
DNS	*Domain Name System* Systém umožňující překlad doménových jmen na IPv4/IPv6 adresy
IPv4	*Internet Protocol version 4* Protokol sloužící k adresaci počítačových sítí
IPv6	*Internet Protocol version 6* Protokol sloužící k adresaci počítačových sítí
LDAP	*Lightweight Directory Access Protocol* Protokol sloužící k přístupu a ukládání dat na adresářovém serveru
MAC	*Media Access Control* Identifikátor síťového zařízení
NAS	*Network Access Server* Přístupový server
PHP	*PHP Hypertext Preprocessor* Programovací jazyk
PL/SQL	*Procedural Language/Structured Query Language* Nadstavba jazyka SQL
RADIUS	*Remote Authentication Dial-In User Service* Síťový protokol umožňující centralizovanou autentizaci, autorizaci a účtování
SNMP	*Simple Network Management Protocol* Protokol sloužící ke správě a dohledu síťových zařízení

SQL	*Structured Query Language* Strukturovaný dotazovací jazyk pro práci s relačními databázemi.
TCP	*Transmission Control Protocol* Protokol zajišťující spolehlivé doručení dat v IP sítích.
UDP	*User Datagram Protocol* Protokol pro doručování dat v IP sítích.
VLAN	*Virtual Local Area Network* Virtuální lokální síť.
VPN	*Virtual Private Network* Technologie k propojení privátních sítí přes sítě veřejné.
VŠB	Vysoká škola báňská.
VŠB-TUO	Vysoká škola báňská - Technická univerzita Ostrava

1 Úvod

Cílem bakalářské práce je vyvinout a popsat informační systém, který budou denně používat síťoví administrátoři na VŠB-TU. Jejich potřeba disponovat takovým systémem vychází zejména z časové náročnosti dohledávání záznamů při běžné práci, podpoře uživatelů, ale i jejich blokací. Čas strávený nad dohledáním identity připojeného zařízení se dnes pohybuje až okolo patnácti minut a nový systém by měl umožnit tuto dobu zkrátit ideálně pod jednu minutu. Dále by měl systém umožnit snadnou analýzu přístupů do sítě, která je s aktuálními programovými prostředky velmi pracná.

Důvodem, proč je vůbec potřeba tyto informace shromažďovat a vyhledávat v nich, je zpřehlednění a zrychlení provozní podpory uživatelů a také zrychlení řešení bezpečnostních incidentů.

Všichni uživatelé resp. jejich zařízení se do univerzitní sítě přihlašují pomocí svého jedinečného osobního jména a hesla. Tyto přihlašovací údaje ověřují přístupová zařízení (AP, přepínače) za pomoci RADIUS serveru, proto první kapitola bude věnována popisu RADIUS protokolu a to v rozsahu nutném pro pochopení a realizaci cílů této práce. Další kapitoly práce budou věnovány samotnému vývoji a implementaci programových nástrojů.

2 Ověřování přístupů do sítě

RADIUS (z anglického *Remote Authentication Dial-In User Service*) je síťový protokol umožňující centralizovanou autentizaci, autorizaci a účtování (AAA model) (HASSELL, J. 2002). RADIUS protokol bývá nasazován ve větších sítích kvůli možnosti přidělit každému uživateli jedinečné přihlašovací údaje a centrálně je uchovávat. Tento princip je výhodnější než zabezpečení přístupů do sítě pomocí předsdílené fráze (z anglického *pre-shared-key*). Rizikem tohoto zabezpečení je, že umožňuje připojení do sítě komukoli se znalostí hesla. Taková slabina je pak zneužitelná potenciálním útočníkem k útokům na vnitřní/vnější síť se zachováním vysoké míry anonymity.

2.1 Architektura

Pro lepší pochopení souvislostí je potřeba vědět, jak konkrétně jsou RADIUS servery na síti VŠB nasazeny a co jednotlivé pojmy znamenají. Zapojení RADIUS serverů na univerzitní síti je zjednodušeně znázorněno na obrázku 2.1. Je zde také oranžovou a červenou barvou znázorněno, jak bude do existující architektury implementován vyvíjený informační systém.

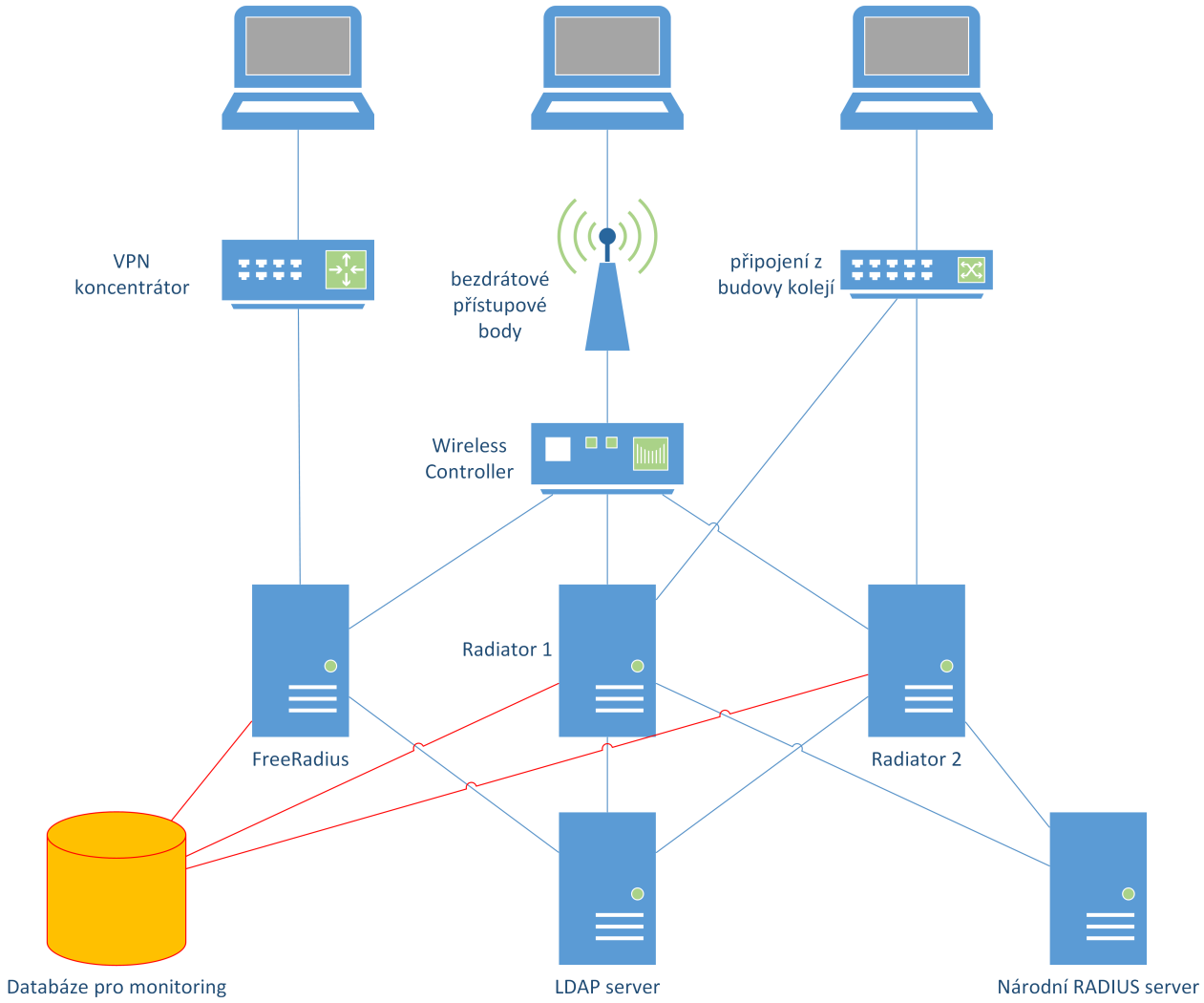
RADIUS server

Serverem rozumíme počítač poskytující služby klientům (model klient-server) (Server Definition 2014). Na univerzitní síti jsou k tomuto účelu používány programy *Radiator* a *Freeradius*. Radiator ověřuje uživatele EDUROAM, Freeradius ověřuje uživatele připojující se pomocí VPN klienta a uživatele s dočasnými účty (hosté univerzity v rámci konaných akcí).

Klient, který s RADIUS serverem komunikuje, je nazýván *přístupový server* - známý pod zkratkou NAS (z anglického *Network Access Server*) (HASSELL, J. 2002). Jedná se většinou o síťový přepínač, Wi-Fi access-point (AP) nebo VPN koncentrátor.

EDUROAM

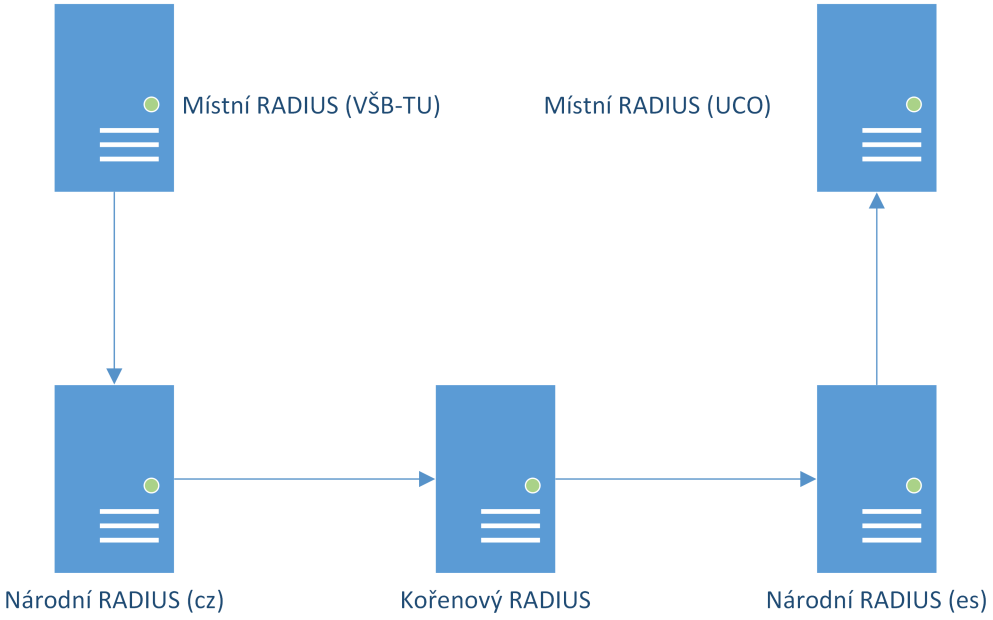
EDUROAM (**EDU**cation **ROAM**ing) je projekt, který vznikl v roce 2002 v Nizozemsku (KRČMÁŘ P.,CALETKA O. 2016) (Fungování roamingu 2016). Uživatelům



Obrázek 2.1: Nasazení RADIUS serverů na síti VŠB-TUO

členské organizace umožňuje připojení k síti v jiné organizaci, než je jeho domovská. Nejvíce je rozšířen ve Wi-Fi sítích akademických institucí. Pokud budu uvažovat modelový případ, kdy na VŠB-TU přijede v rámci zahraniční stáže student, například ze Španělska, bude se moci připojit svými přihlašovacími údaji bez nutnosti si vytvářet nový dočasný účet v síti VŠB. Téhle funkcionality je docíleno pomocí hierarchie autentizačních vazeb zapojených v projektu. Místní RADIUS server uživatele z cizí organizace pozná dle tzv. *realmu*, který se připojuje za uživatelské jméno.

Řetězec identifikující uživatele vypadá obecně takto: *abc0123@uco.es*. *abc0123*. Představuje přihlašovací jméno v rámci domovské instituce a *uco.es* je realm identifikující konkrétní instituci. Autentizace „roamujícího“ uživatele probíhá finálně až v jeho domovské organizaci, které jsou předány k ověření přihlašovací údaje. Místní RADIUS server se s danou organizací spojuje za pomoci národního nebo kořenového RADIUS serveru. Znalost tohoto principu je důležitá zejména proto, že se musí zohlednit při návrhu blokací uživatelů.



Obrázek 2.2: Ověření cizího uživatele (KRČMÁŘ P.,CALETKA O. 2016)

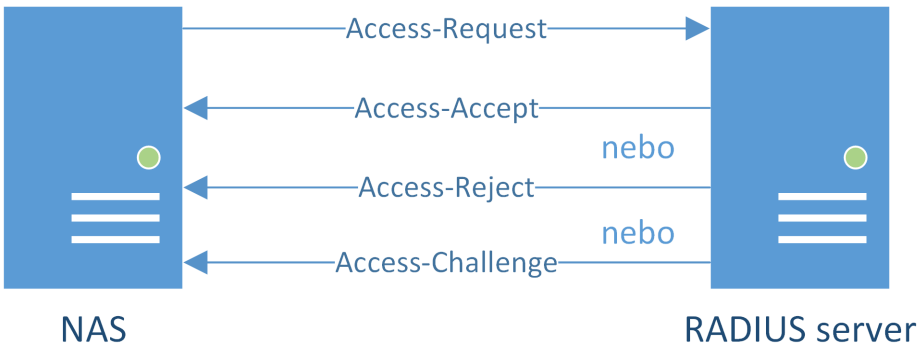
2.2 AAA model

Význam pojmu AAA model byl již zmíněn v úvodu kapitoly 2. Pro dosažení cílů této práce je potřeba pochopit jednotlivé aspekty AAA modelu a jeho implementaci v protokolu RADIUS. Nejvíce se je třeba zaměřit především na autentizaci (pro docílení blokace uživatelů) a účtování (sběr informací).

Autentizace

Při připojení zařízení do sítě je jeho veškerý síťový provoz blokován (GNU Radius Reference Manual 2008). Jedinou vyjímku tvoří komunikace s NAS serverem. NAS uživatele pomocí *EAP-over-LAN* vyzve k představení. Uživatel odešle své jméno i heslo a NAS tyto přihlašovací údaje zapouzdří do *Access-Request* paketu, který odešle RADIUS serveru, jenž je ověří vůči databázi oprávněných uživatelů. Pokud jsou přihlašovací údaje v pořádku, odpoví RADIUS server paketem *Access-Accept* a NAS započne uživatelskou relaci. V opačném případě odpoví paketem *Access-Reject* a uživateli je přístup odepřen. V situaci, kdy žádná odpověď nepřijde, je požadavek několikrát zopakován. NAS může také požadavky přesměrovat na záložní RADIUS server.

Existuje také možnost místo dvou uvedených odpovědí RADIUS serveru poslat paket *Access-Challenge*(Remote Authentication Dial In User Service (RADIUS) 2000). Pomocí tohoto paketu se RADIUS server uživatele dotáže na dodatečné informace. Pro uživatele to může znamenat, že se mu na zařízení zobrazí otázka, na kterou musí odpovědět. RADIUS server poté odpovídá opět pomocí *Access-Accept*,*Access-Reject* nebo *Access-Challenge* paketu.



Obrázek 2.3: Výměna paketů při autentizaci

RADIUS server si při ověřování údajů dělá záznamy o jejich výsledku. Samotný záznam může vypadat například takto:

```
Sun Oct 9 18:21:47 2016: Login OK:[kal0178@vsb.cz]
(CSID D7-C2-9A-E0-2F-D2 NAS C2960-KOLT92-GH/172.19.17.189)
```

Z tohoto záznamu lze vyčíst, jaký uživatel žádal o přístup do sítě (kal0178@vsb.cz), kdy a odkud o něj žádal (v tomhle případě přepínač C2960-KOLT92-GH) a MAC adresa uživatelova zařízení (D7-C2-9A-E0-2F-D2). Tyto informace jsou však nedostačující, protože například úplně chybí, jaká IP adresa byla uživateli přidělena nebo jak dlouho byl na síti připojen.

Autorizace

Po úspěšné autentizaci je možno pomocí přístupového serveru uživatele autorizovat k použití různých služeb. NAS se v případě žádosti o autorizaci opět chová jako prostředník a při vyřizování žádosti kontaktuje server, který poskytuje žádanou službu a sdělí mu, že je uživatel tuto službu oprávněn používat (HASSELL, J. 2002). Ten k ní musí mít samozřejmě dohodnut přístup. V síti VŠB-TU je při tomto kroku uživateli přiřazeno VLAN-ID, které určuje do jaké vnitřní sítě bude přiřazen. Protože v síti VŠB-TU jiná autorizace pomocí přístupového serveru není implementována, tudíž ji nelze využít k cíli téhle práce, nebudu tuto problematiku více rozvádět.

Účtování

Hned ze začátku je třeba podotknout, že slovo účtování vychází z anglického *accounting* a nikoliv *billing*. Slovo účtování je tedy odvozeno od uživatelského účtu, proto se jedná o operace k uživatelskému účtu vztažené.

Účtování je pro samotnou analýzu přístupů ten nejcennější zdroj informací. Po úspěšné autentizaci NAS kromě autorizace započne uživatelskou relaci a v průběhu této relace NAS na RADIUS server odesílá tzv. *accounting* pakety. Do těchto paketů se přikládají atributy, což jsou data ve tvaru *název-hodnota*. Nejpoužívanější atributy jsou standartizovány v RFC 2865 (Remote Authentication Dial In User Service (RADIUS) 2000), k těmto základním atributům se však, je-li to třeba, může přidat libovolné množství atributů nestandartizovaných.

Typy účtovacích paketů

1. **Accounting start**

   Odesílá se po autentizaci uživatele.

2. **Accounting update**

   Odesílá se průběžně během uživatelské relace.

3. **Accounting stop**

   Odesílá se po ukončení uživatelské relace.

Ve všech paketech se nacházejí stejné atributy a mění se pouze časové hodnoty. Jedině u Accounting stop paketu jeden údaj přibývá a to důvod ukončení relace (Acct-Terminate-Cause). V praxi ovšem narazíme na problém, že různá síťová zařízení do stejného atributu vkládají odlišné údaje. Například VPN koncentrátor do atributu *Calling-Station-Id* vkládá IP adresu, ze které se uživatel připojuje. Oproti tomu při připojení uživatele pomocí bezdrátového bodu, kde funkci přístupového serveru zajišťuje *Wireless Controller*, je jako hodnota atributu *Calling-Station-Id* uvedena MAC adresa zařízení. Může také nastat případ, kdy u klientů s IPv6 konektivitou dojde k začátku účtování dříve, než je klientovi pomocí DHCP přidělena IPv4 adresa a tu je třeba později doplnit z *Accounting update* paketů. Po konzultaci s administrátory sítě jsme došli k sadě atributů, které se budou uchovávat, a tyto pro přehlednost uvádím v tabulce 2.1. Některé atributy jsou zde uvedeny vícekrát z důvodu různé interpretace jednotlivých síťových prvků.

Tabulka 2.1: Atributy accounting paketu

atribut	**hodnota** (příklad)	**popis**
Acct-Session-Id	57fa79ba/4d:67:ca:af:8b:b5/6271137	ID relace
User-Name	abc0123@vsb.cz	uživatelský login + organizace
Framed-IP-Address	158.149.50.24	přidělená IPv4 adresa
Calling-Station-Id	158.149.50.24	přidělená IPv4 adresa
Calling-Station-Id	4d:67:ca:af:8b:b5	MAC adresa zařízení
Calling-Station-Id	62.129.36.134	vzdálená IPv4/IPv6 adresa (VPN)
Called-Station-Id	LAP1131-KOL-A1:eduroam	jméno NAS
NAS-Identifier	C2960-KOLB9-IJ	jméno NAS
NAS-IP-Address	158.149.50.254	IPv4 adresa NAS
Acct-Status-Type	Start	typ účtovacího paketu
Timestamp	2016-10-13 09:15:14+02	časová značka

Záznamy o účtování server ukládá do textových souborů. Jestliže však chceme v těchto souborech něco dohledávat, tak se dostáváme do nepříjemné situace. Při velkém počtu uživatelů bývají tyto soubory velmi velké a nepřehledné. Hledání v nich také ztěžuje

fakt, že na velikých sítích bývá RADIUS serverů z důvodů redundance několik. Když si představíme situaci, kdy administrátor dohledává několik dní starý záznam v textovém souboru s desetitisíci záznamy na několika serverech, tak je záhy jasné, že tento postup není pro běžný provoz vhodný. Cílem tedy bude tato data z RADIUS serverů extrahovat a ukládat do relační databáze. Znázornění, jak bude navrhovaný systém zakomponován do existující sítě můžeme vidět na obrázku 2.1.

2.3 Protokol IPv6

Ačkoli by se mohlo zdát, že již víme, kde hledat všechna pro nás zajímavá data, tak v účtování jeden velmi důležitý údaj chybí a to - IPv6 adresa. IPv6 je moderní protokol používaný současně s protokolem IPv4. IPv6 je v některých ohledech od IPv4 velmi odlišný, i když oba vznikly za cílem umožnit počítačům v síti komunikovat. IPv6 se dnes nasazuje převážně kvůli zaplnění adresního prostoru IPv4.

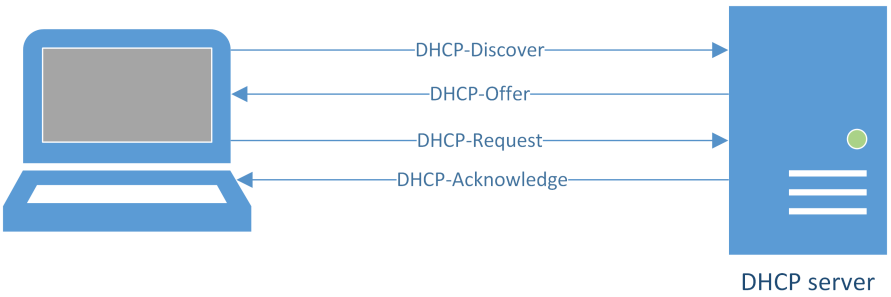
Ke správné funkci jakéhokoli zařízení v síti je potřeba, aby bylo seznámeno se síťovými parametry. Mezi nejdůležitější síťové parametry patří IP adresa, maska sítě, výchozí brána a adresy DNS serverů. U protokolu IPv4 je nejběžnější způsob získání těchto parametrů pomocí DHCP protokolu. Komunikaci s DHCP serverem zahajuje zařízení odesláním paketu *DHCP-Discover* (DHCP 2016). Protože zařízení o síti zatím nic neví, tento paket přijde všem zařízením, které se na síti nacházejí. Na tuto žádost zareaguje pouze DHCP server a to paketem *DHCP-Offer*, kde se klientovi identifikuje a nabídne mu síťové parametry s informací o délce platnosti těchto parametrů. Klient na tuto odpověď reaguje paketem *DHCP-Request*, kde žádá o přidělení nabídnutých parametrů. Tato odpověď je opět adresována všem zařízením v síti, protože klient pořád není oprávněn tyto parametry užívat. Konverzaci zakončí DHCP server paketem *DHCP-Acknowledge*, čímž přidělené parametry nabývají platnosti a klient je může po dobu platnosti využívat. Všechny přidělené parametry si DHCP server ukládá do vlastního logu.

Bezstavová konfigurace

Bezstavová konfigurace je nový způsob získávání komunikačních parametrů při připojení zařízení do sítě (SATRAPA, P. 2011). Narozdíl od stavové konfigurace, kdy veškeré komunikační parametry přiděluje DHCP server nebo se zadávají manuálně (a tudíž lze vše relativně jednoduše dohledat), u bezstavové konfigurace přechází část procesu na samotné koncové zařízení. V IPv6 síti s povolenou bezstavovou konfigurací si část síťových parametrů přidělí zařízení samo na základě ohlášení směrovače. Informace, které se v ohlášení směrovače nacházejí, se liší podle konfigurace. Při bezstavové konfiguraci si však vždy výslednou IPv6 adresu určí zařízení. V současné době není technicky možné konečnou IPv6 adresu žádným způsobem ovlivnit. Proto se tato IPv6 adresa nemůže nacházet nikde v záznamech serverů, pomocí kterých by administrátoři mohli

toto zařízení identifikovat.

Zamezení problému s identifikací zařízení komunikujícího pomocí protokolu IPv6 by se dalo poměrně jednoduše dosáhnout pomocí DHCPv6 serveru. Tohle řešení má ovšem jednu velkou nevýhodu a to chybějící podporu ze strany některých operačních systémů, které konfiguraci pomocí DHCPv6 serveru nepodporují, a tak by jim bylo zamezeno pomocí IPv6 protokolu komunikovat. Kvůli tomuto omezení budu muset využít toho, že i když nemám žádnou informaci o využitých IPv6 adresách v mé síti, tak pořád veškerý provoz probíhá přes univerzitní síťové prvky, bez kterých by komunikace uživatele nebyla možná. Nabízí se například aktivní dotazování směrovače na jeho tabulku sousedů (neighbour cache), obdobu ARP tabulky, kterou používají zařízení komunikující pomocí protokolu IPv4.



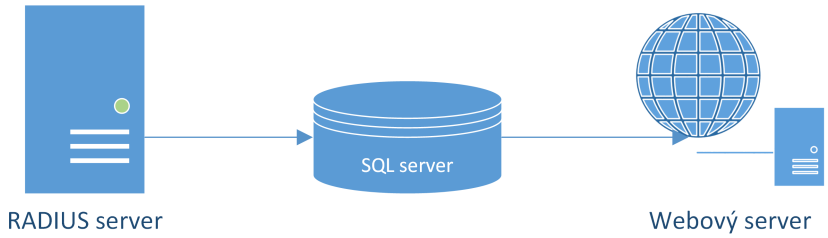
Obrázek 2.4: Konverzace klienta s DHCP serverem

3 Volba programových prostředků a konfigurace serverů

Jak jsem již zmínil v předchozí kapitole, tak ve výchozím stavu se účtování zaznamenává do textového souboru. Mou snahou by tedy mělo být co nejvýhodnější cestou tato data ukládat do databáze. Tohoto lze docílit buď pomocí syntaktické analýzy (parsováním) textového souboru, nebo změnou konfigurace RADIUS serveru. Protože RADIUS servery Radiator a FreeRadius používané na VŠB podporují odesílání dat na SQL servery, budu využívat tyto vlastnosti.

IPv6 adresy uživatelů jsou získávány ze směrovačů pomocí SNMP protokolu, kde se požadovaná data nacházejí v tabulce sousedů. Pomocí tohoto způsobu ale nelze mít všechna data okamžitě a musí být zvolen interval, kdy se data budou ze směrovačů získávat. Tento interval musí být dostatečně krátký na to, abychom byli schopni archivovat všechny adresy, které se na síti vyskytly, a zároveň dostatečně dlouhý, abychom příliš nezatěžovali směrovače a databázi. Nevýhoda tohoto řešení je, že se tabulky stahují vždy celé a v databázi bude velké množství duplicitních dat, které bude nutné pravidelně agregovat.

Jeden z provozních požadavků na tento informační systém byl, aby běžel na Linuxovém serveru. Byla zvolena distribuce Debian, která se vyznačuje velmi vysokou spolehlivostí a stabilitou. Jako databázový server jsem zvolil PostgreSQL. Tento databázový server jsem zvolil především kvůli datovým typům. PostgreSQL obsahuje datové typy pro MAC adresu a IPv4/IPv6 adresu. Prezentaci dat administrátorům bude zajišťovat webový server Apache.



Obrázek 3.1: Obecný návrh informačního systému

3.1 Konfigurace serveru Radiator

Server Radiator byl navržen jako modulární program (Radiator® RADIUS Server 2015). Jedna z jeho konfiguračních možností umožňuje při zpracovávání požadavků spustit skript napsaný v jazyce Perl, kterým můžeme upravit chování celého procesu. Protože popis celého konfiguračního souboru nespadá do rozsahu této práce, budu popisovat jen nejdůležitější parametry z toho důvodu, aby úpravy byly na již fungujícím serveru replikovatelné.

Pro řešení zadaných cílů se je třeba zaměřit na tzv. klauzuli handler, která specifikuje, jaké operace se mají provést pro určité skupiny připojujících se uživatelů. Pro lepší představu uvádím část konfigurace sloužící k odbavení uživatelů VŠB.

```
<Handler Realm=/^vsb\.cz$/ix >
        AuthBy LDAP_VSB
        AuthLog vsbusers
        AcctLogFileName /var/log/radiator/radiator-detail
        WtmpFileName /var/log/radiator/wtmp
        RejectHasReason
</Handler>
```

V uvedené části konfiguračního souboru můžeme vidět, že pro uživatele s realmem vsb.cz bude provedeno ověření hesla pomocí LDAP (*AuthBy*), umístění textových souborů pro log (*AuthLog* pro autentizaci,*AcctLogFileName* pro účtování a *WtmpFileName* pro *Access-Request* pakety) a že v případě selhání autentizace se uživateli odešle důvod nezdaru (RejectHasReason). Dle dokumentace (Radiator® RADIUS Server 2015) lze do konfigurace přidat návěstí *AuthBy INTERNAL*, kde můžeme ovlivnit vytváření odpovědi na příchozí účtovací pakety. V našem případě odpověď měnit nechceme, můžeme ale využít toho, že v této fázi můžeme spustit námi napsaný skript, kterému server předá všechny účtovací atributy ve formě asociativního pole (Satrapa, P. c2000). Poté již není problém pomocí skriptu všechna potřebná data odeslat do databázového serveru.

Pro docílení blokace jsem původně použil skript, který se spustil při přijetí paketu *Access-Request* a dotazoval se databáze na seznamy blokovaných uživatelů a MAC adres. Radiator má bohužel takovou vlastnost, že skript spustil a ukončil pro každý paket zvlášť, což významně navýšilo čas potřebný pro vyřízení jednoho požadavku. Server požadavek běžně vyřídí v čase kolem 1ms. Při spouštění skriptu čas potřebný k vyřízení požadavku vzrostl na asi 70ms. Tento problém se projeví při vysoké zátěži, kdy server nakupené požadavky nestihne vyřídit včas a uživatelé se nejsou schopni autentizovat.

Pro implementaci blokace jsem nakonec použil klauzuli *AuthBy FILE*, která je primárně určená k autentizaci uživatelů, jejichž údaje jsou uloženy v textovém souboru. Výsledek

autentizace lze poté negovat pomocí konfiguračního parametru *Blacklist*. Samotná konfigurace může vypadat následovně:

```
<AuthBy FILE>
        Identifier check_users
        NoCheckPassword
        NoEAP
        NoDefault
        Blacklist
        Filename /etc/radiator/utils/blacklistUsers.txt
</AuthBy>
```

Pro blokaci MAC adres stačí vytvořit stejnou sekci s jedinou změnou a to přidáním parametru *AuthenticateAttribute Calling-Station-Id*. Ten nám umožní kontrolovat místo uživatelských jmen MAC adresy. Výsledná konfigurace tedy může vypadat následovně:

```
<Handler Realm=/^vsb\.cz$/ix >
        <AuthBy INTERNAL>
                AcctHook file:"/etc/radiator/utils/sqlradacct.pl"
        </AuthBy>
        AuthByPolicy ContinueUntilReject
        AuthBy check_users
        AuthBy check_mac
        AuthBy LDAP_VSB
        AuthLog vsbusers
        AcctLogFileName /var/log/radiator/radiator-detail
        WtmpFileName /var/log/radiator/wtmp
        RejectHasReason
</Handler>
```

Aktualizaci souborů zajišťuje jednoduchý skript, který si data načte z databáze a soubory přepíše. Radiator změnu souboru detekuje dle data poslední úpravy, takže není třeba službu restartovat. Skript je spouštěn pomocí plánovače Cron.

Rychlostní optimalizace

Již jsem se zmínil o problému, který nastal při implementaci blokací. Radiator skripty spouští a ukončuje zvlášť pro každý paket. Blokaci jsem mohl z většiny implementovat konfiguračně, ale u sběru dat z účtování jsem se tomuto přístupu snažil vyhnout, abych neztratil flexibilitu, kterou mi vlastní zpracování dat před jejich vložením do databáze nabízí. Aby doba běhu skriptu byla co nejkratší, tak jsem vytvořil jednoduchý proxy server. Skript tak nemusí pokaždé navazovat šifrované TCP spojení a čekat na provedení požadovaných operací databáze. Tuto starost jsem přenesl na proxy server, který si s databází udržuje perzistentní spojení a předpřipravené dotazy. Činnost skriptu jsem omezil pouze na naformátování potřebných údajů a sestavení UDP paketu, který je na

proxy server odesílán přes místní smyčku. Proxy server se stará o čtení dat z vyrovnávací paměti (anglicky buffer) a jejich vkládání do databáze. Další přidaná hodnota proxy serveru je ta, že v případě výpadku databáze přijatá data ukládá do mezipaměti (anglicky cache). Takto může systém fungovat bez jejich ztráty. Proxy server je kvůli své povaze napsán jako démon.

3.2 Konfigurace serveru FreeRadius

Server FreeRadius narozdíl od serveru Radiator nedisponuje takovou flexibilitou, která umožňuje upravovat chování za pomocí vlastních skriptů. Lze však docílit stejného chování jako v předchozím případě za použití vhodné konfigurace. Pro spolupráci FreeRadius serveru s databází je třeba ke standartní instalaci doinstalovat modul *freeradius-postgresql* (Guide/SQL HOWTO 2016). Po dokončení instalace je třeba v konfiguračním souboru *radiusd.conf* odkomentovat řádek *$INCLUDE sql.conf*, čímž podporu SQL povolíme. V souboru *sql.conf* se specifikují údaje nutné pro připojení k databázi. Nejdůležitější řádky jsou zejména tyto:

```
database    = "postgresql" #Název databázového programu
server      = "sixmon.vsb.cz" #Hostname
login       = "radius" #Uživatelské jméno
password    = "heslo" #Heslo
radius_db   = "radiusdb" #Název databáze
acct_table1 = "radacct" #Název tabulky pro účtování
$INCLUDE dialup.conf #Konfigurační soubor s~SQL dotazy
```

V konfiguračním souboru *./sites-enabled/default* v sekci *accounting* odkomentováním řádku *sql* určíme, že komunikace s databázovým serverem má probíhat pro účtování.

Dále je třeba upravit konfigurační soubor *dialup.conf*, kde specifikujeme, jak mají dotazy odesílané na SQL server vypadat. Pro každou událost se zde konfiguruje primární a alternativní dotaz pro případ, že primární selže. Tato vlastnost mi umožnila vypořádat se s problémem, který jsem uvedl v kapitole 2.2, a to tím, že přístupové servery do stejného atributu vkládají odlišné hodnoty. V našem případě máme v atributu *Calling-Station-Id* MAC adresu nebo IPv4 adresu, protože FreeRadius odbavuje klienty sítě *tuonet-guest* a zároveň klienty připojující se přes VPN.

Na databázovém serveru jsem v tabulce pro účtování určil (viz. kapitola 3.3), že sloupec *CallingStationId*, do kterého se tato hodnota ukládá, je datového typu *macaddr* (datový typ pro MAC adresu). Když tedy FreeRadius na databázový server odešle dotaz, čímž se snaží do sloupce *CallingStationId* vložit IPv4 adresu, dotaz je odmítnut a FreeRadius odešle alternativní dotaz, kde je hodnota atributu vkládána již správně do sloupce *externalip*.

K docílení blokace uživatelů je třeba vytvořit nový konfigurační soubor, kde specifikujeme údaje pro připojení k databázovému serveru. Proto stačí zkopírovat již funkční soubor sql.conf, uložit ho pod jiným názvem (zvolil jsem název *sql-blacklist.conf*) a pouze upravit hodnoty jednotlivých konfiguračních parametrů. Tento soubor se zahrne do hlavního konfiguračního souboru *radiusd.conf* pomocí řádku *$INCLUDE sql-blacklist.conf*. Pro samotnou blokaci se v souboru *sites-enabled/default* do sekce *post-auth* přidají tyto řádky:

```
#Blokace uživatelského jména
if ( "%{tolower:%{User-Name}}" == "%{sql-blacklist:
                SELECT userName FROM blacklist WHERE
                username=lower('%{User-Name}')}")
{
        reject
        update reply {
                        Reply-Message = "Your account has been blocked."
                        }
}

#Blokace MAC adresy
if ( "%{tolower:%{User-Name}}" == "%{sql-blacklist:
                SELECT mac FROM blacklist WHERE
                mac=lower('%{Calling-Station-Id}')}")
{
        reject
        update reply {
                        Reply-Message = "Your account has been blocked."
                        }
}
```

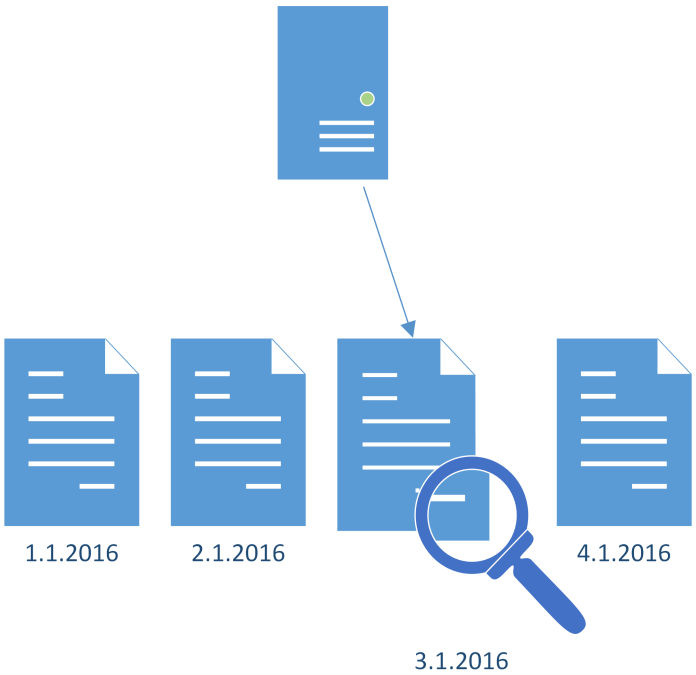
3.3 SQL databáze a obsluhující skripty

Databáze byla navržena s ohledem na nárazovou zátěž, která je způsobena denním režimem studentů a způsobem zjišťování jejich IPv6 adres. Největší špičky se očekávají v ranních hodinách, kdy studenti přichází do školy a jejich zařízení se připojují do sítě. Při měření, které jsem prováděl, bylo běžné, že ve špičkách databáze zpracovávala až 1130 dotazů za sekundu. Toto měření jsem prováděl pomocí programu *PgBadger* a měření bylo uskutečnitelné díky tomu, že v době psaní bakalářské práce již systém běžel v pilotním režimu.

Dle požadavků databáze musí zajišťovat:

1. kontrolu, zda připojující se uživatel není blokován
2. vkládání dat získaných z účtování
3. vkládání dat získaných z tabulek sousedů
4. vyhledávání dat administrátorem

Kvůli rozdílnosti získávaných dat jsou vytvořeny 3 databáze, které jsou používány nezávisle na sobě. Při návrhu databáze bylo třeba zajistit, aby v tabulkách nebylo velké množství dat. Pokud bychom nechali všechna data v jedné tabulce, tak bychom časem zaznamenali obrovskou ztrátu výkonu na databázovém serveru. Trvání jednotlivých operací by se rostoucím počtem dat postupně prodlužovalo z milisekund až na desítky sekund. Jako řešení tohoto problému jsem zvolil tzv. *partitioning* tabulek, což znamená, že se tabulka rozdělí na více menších tabulek podle určitého pravidla. V našem případě je pro každý den vytvořena nová tabulka. Podmínka pro existenci záznamu v téhle tabulce je, že se datum pořízení záznamu musí shodovat s datem, pro které je tabulka určena. Toto rozdělení je výhodné zejména kvůli tomu, že při vyhledávání dat administrátorem, kdy je na server odeslán příkaz *SELECT*, nebudou prohledávány všechny tabulky, ale jen ty, kde se záznamy opravdu mohou nacházet. Protože se záznamy neuchovávají navždy, ale po určité době se mažou, tak je možné mazat rovnou celé tabulky. V opačném případě by se musel kontrolovat záznam po záznamu, což by mělo za následek další ztrátu výkonu. Konkrétní provedení jednotlivých databází je popsáno v podkapitolách 3.3, 3.3 a 3.3.



Obrázek 3.2: Vyhledávání dat nad tabulkou, kde je aplikován partitioning

Účtování

Databáze určená pro účtování obsahuje jednu tabulku určenou pro neukončené účtování (denní tabulka) a druhou pro archivaci. Nad denní tabulkou se nejčastěji provádějí příkazy *INSERT* pro vkládání získaných dat a příkazy *SELECT*, kdy se pomocí *Accounting-Update* paketů kontroluje, zda při začátku účtování v atributech nechyběla přiřazená IPv4 adresa. Protože při kontrole známe identifikátor sezení (*AcctSessionId*), můžeme záznam hledat podle tohoto identifikátoru, který je jako jediný v denní tabulce indexován. Kvůli zajištění co nejmenšího počtu řádků v denní tabulce se již ukončené relace v noci přesouvají do historické tabulky. Protože ne všechny relace jsou v době agregace ukončené, musí se denní tabulka kontrolovat řádek po řádku. Kontroluje se zejména to, zda řádek obsahuje nenulové hodnoty pro sloupce *AcctStartTime* a *AcctStopTime*. Takový záznam skript vloží do historické tabulky a z denní ho smaže. Agregaci zajišťuje skript napsaný v programovacím jazyce Perl. Protože je po celou dobu využíváno předpřipravených dotazů, je pro agregaci několika tisíc záznámů používáno jen dvou dotazů, což zajistí větší rychlost agregace.

Tabulka 3.1: Tabulka pro data z účtování

název	**datový typ**	**index**	**index - archív**	**popis**
radacctid	bigserial	ano	ano	id v rámci tabulky
AcctSessionId	text	ano	ano	id relace
UserName	text	ne	ano	login uživatele
FramedIPAddress	inet	ne	ano	IPv4 adresa
CallingStationId	macaddr	ne	ano	MAC adresa zařízení
AcctStartTime	timestamp with timezone	ne	ano	začátek relace
AcctStopTime	timestamp with timezone	ne	ano	konec relace
externalip	inet	ne	ano	externí IPv4/IPv6 adresa (VPN připojení)
ipv6pref	cidr	ne	ne	prefix IPv6 adresy přidělené VPN koncetrátorem
ipv6id	text	ne	ne	suffix IPv6 adresy přidělené VPN koncetrátorem
framedipv6	inet	ne	ano	IPv6 adresa přidělená VPN koncetrátorem
calledstationid	text	ne	ne	identifikátor přístupového serveru
nasidentifier	text	ne	ne	identifikátor přístupového serveru

IPv6 adresy

V součtu se v tabulkách sousedů běžně nachází několik tisíc párů MAC a IPv6 adres. Protože získávání informací o IPv6 adresách vyskytujících se na síti probíhá v pravidelných intervalech stahováním celých tabulek sousedů, bude tento proces hlavním důvodem špiček na databázovém serveru. Tabulka, do které tato data přicházejí, proto musí být co nejvíce efektivní. Z tohoto důvodu byla vytvořena tabulka určená pouze ke sběru dat. Protože jsme se chtěli vyhnout tomu, aby se v tabulce u jednotlivých párů aktualizoval čas, kdy byly v síti spatřeny, tak se nad touto tabulkou při sběru informací spouští pouze příkaz *INSERT*. Indexace nad tabulkou neprobíhá, což příkaz *INSERT* zrychlí.

Jako další optimalizační metodu jsem zvolil užití tzv. předpřipravených dotazů (z anglického *prepared statements*), které zajistí lepší spolupráci mezi skriptem vkládajícím data a databází. Při použití předpřipraveného dotazu se na databázový server dotaz pošle pouze jednou a poté se odesílají pouze data, která si server do dotazu dosazuje. Server tak nemusí neustále dokola *parsovat* a připravovat jeden a ten samý dotaz (ŽÁK, K. 2004).

Tabulka 3.2: Neagregovaná tabulka párů MAC a IPv6 adres

název	**datový typ**	**index**	**popis**
sweep_time	time	ne	čas sběru informací
ipv6	inet	ne	IPv6 adresa
mac	macaddr	ne	MAC adresa
date	date	ne	datum sběru informací

Protože by vyhledávání nad tabulkou bylo neefektivní, tak se večer, kdy je minimální provoz, tabulka agreguje. Můžeme ji proto nazvat jako denní. Agregaci zajišťuje agregační skript napsaný v jazyce Perl. Agregační skript nalezne čas, kdy byl pár MAC-IPv6 viděn poprvé a naposled. Tento řádek vloží do tabulky sloužící k archivaci záznamů. Tato tabulka je indexována a rozdělena (*partitioning* podle data) pro zajištění rychlosti vyhledávání. Agregační skript těsně před ukončením smaže denní tabulku pomocí příkazu *TRUNCATE* a z archivu smaže staré tabulky. Jelikož při bezpečnostních incidentech dotazy na dohledání uživatele přichází většinou až po několika dnech, tak neefektivnost vyhledávání nad denní tabulkou nepředstavuje problém. Také se domnívám, že by toto řešení nemělo mít v budoucnu výkonnostní problémy, když vezmu v potaz pravděpodobnost, že IPv6 provoz bude narůstat a ne klesat.

Tabulka 3.3: Agregovaná tabulka pro páry MAC-IPv6

název	**datový typ**	**index**	**popis**
id	bigserial	ano	id v rámci tabulky
ipv6	inet	ano	IPv6 adresa
mac	macaddr	ano	MAC adresa
first_seen	time	ne	první výskyt páru IPv6/MAC
last_seen	time	ne	poslední výskyt páru IPv6/MAC
date	date	ne	datum pořízení záznamu

Blokace

Logika blokace byla již přibližně popsána v kapitole 3.1. Při první implementaci funkcionality pro blokování uživatelů na síti byla vytvořena jednoduchá tabulka se sloupci pro

MAC adresu a pro login uživatele. Pro lepší orientaci v datech byly přidány ještě další sloupce, které se na funkci nepodílejí, ale administrátorům poskytují dodatečné informace, jako je čas blokace, kdo jí provedl a důvod blokace. Po provedení prvních blokací jsem byl nucen jejich logiku dodatečně upravit z důvodu, že někteří uživatelé blokace obcházeli tím, že přesvědčili spolužáky, aby jim dali své přihlašovací údaje. Webové rozhraní proto při blokaci loginu uživatele projde databázi pro účtování a nalezne všechny MAC adresy, pod kterými se kdy uživatel připojil. Tyto MAC adresy jsou poté blokovány také a jsou od ostatních odlišeny pomocí sloupce *owner*. Díky tomu můžou být při odblokaci snadno nalezeny a z tabulky smazány. Vedlejší efekt tohoto přístupu je, že můžou být zablokována i zařízení, která blokovaný uživatel nevlastní. Systém na tento případ administrátora upozorní a je na něm, zda takové zařízení do blokace zahrne.

Tabulka 3.4: Tabulka pro blokace

název	**datový typ**	**index**	**popis**
id	bigint	ano	id v rámci tabulky
username	text	ne	login uživatele
realm	text	ne	realm uživatele
mac	macaddr	ne	MAC adresa zařízení uživatele
blockedby	text	ne	osoba, která uživatele zablokovala
blocked	text	ne	osoba, která uživatele odblokovala
reason	text	ne	důvod blokace
owner	text	ne	vlastník zařízení

4 Webové rozhraní

Webové rozhraní slouží jako jednoduchá cesta, jak se samotným systémem pracovat. Obsluha se tedy nebude muset připojovat přímo k databázi a žádaná data získávat pomocí ručně psaných SQL dotazů. Výsledky vyhledávání jsou navíc obohacené o reverzní DNS překlady IPv4 adres a výrobce síťových karet. Obsluha může vyhledávat účtovací logy a blokované uživatele. Mezi kategoriemi vyhledávání se přepíná pomocí postranního menu. Webové rozhraní navíc umožňuje kromě autentizace i autorizaci uživatele, který v rozhraní bude pracovat.

4.1 Vyhledávání

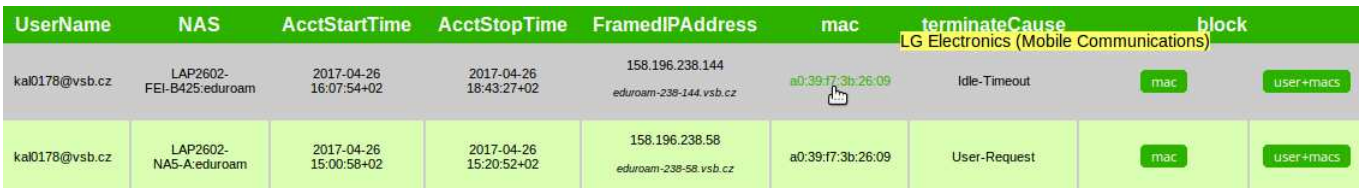
Vyhledávání v logu účtování využijí především členové bezpečnostního týmu při dohledávání identity zařízení, které figurovalo v bezpečnostním incidentu. V praxi se však v logu hledá i kvůli provozní podpoře uživatelů. Často se totiž stává, že uživatel, který se kvůli problému obrátí na helpdeskové pracoviště, neoznámí, že byl jeho problém vyřešen a operátor helpdesku neví, jestli může požadavek uzavřít. Dále se dá pomocí tohoto logu velmi rychle vyloučit problém s autentizací na síti a lze se zaměřit na hledání problémů jiných.

K vyhledávání v logu slouží jednoduchý formulář, kde lze zadat více vyhledávacích kritérií. Nalezeny budou záznamy, které obsahují právě všechna zadaná kritéria. Pro provedení vyhledávání musí být vyplněno alespoň jedno kritérium a vymezená časová oblast, ve které se má záznam přibližně nacházet. Kdyby nebyla zadaná časová oblast, nemohlo by se využít partitioningu tabulek v SQL databázi a tím by docházelo k nadměrné zátěži. Tato podmínka je také důležitá kvůli tomu, že prohlížeče kolabují, když dojde k výpisu velkého množství záznamů (řádově tisíců). Pro snadnější zadávání správného formátu data do formuláře byly do rozhraní implementovány projekty *Momentum* a *Pikaday*, díky nimž si obsluha datum navolí pomocí kalendáře (viz obr. 4.1).

Na obrázku 4.2 můžeme vidět jak zahájit hledání jednotlivých záznamů, aktuálně tedy dle uživatelského jména. Výsledek tohoto vyhledávání můžeme vidět na obrázku 4.3.

Obrázek 4.1: Formulář pro vyhledávání v logu účtování

Obrázek 4.2: Vyhledávání dle zadaných kritérií



Obrázek 4.3: Výsledky vyhledávání v logu účtování

Z výsledku můžeme vyčíst následující informace:

1. **UserName**

   Jméno přihlášeného uživatele. Jméno je uloženo přesně tak, jak ho uživatel zadal, včetně velikosti písmen.

2. **NAS**

   Místo, kde se uživatel přihlásil. Z řetězce v prvním řádku například vyčteme, že se uživatel přihlásil do sítě Eduroam na fakultě elektrotechniky a informatiky (FEI), v místnosti B425 a šlo o bezdrátové připojení přes Wi-Fi access-point LAP2602.

3. **AcctStartTime a AcctStopTime**

   Začátek a konec uživatelské relace.

4. **FramedIPAddress**

   Přidělená IPv4 adresa a její reverzní DNS záznam.

5. **mac**

   MAC adresa síťové karty připojeného zařízení a jméno výrobce síťové karty. Zde se pravděpodobně jedná o mobilní telefon.

6. **TerminateCause**

   Důvod ukončení relace. Nejčastější důvody jsou následující:

   Idle-Timeout - Ztráta signálu.

   User-Request - Manuální odhlášení uživatele.

   Admin-Reset - Nestandartní chování připojeného zařízení (relace ukončena aktivním prvkem).

   Reauthentication-Failure - Zařízení nereagovalo na žádost o reautentizaci, nebo uživatel nemá v zařízení uložené přihlašovací údaje a při reautentizaci je zadal špatně.

Na nalezené údaje lze také klepnout a tím zahájit nové hledání. Ve výsledcích se může objevit záznam z VPN připojení (viz. obr.4.4). V těchto výsledcích přibudou následující informace:

1. **externalIP**

   IPv4/IPv6 adresa z které se uživatel na VPN připojil.

2. **framedIPv6**

   IPv6 adresa přidělená VPN koncetrátorem. Tuto adresu lze přidělit díky tomu, že se uživatel připojuje přes VPN klienta. VPN klient IPv6 adresu nastaví tzv. staticky. Nedochází proto k náhodnému výběru adresy na straně uživatelova zařízení.

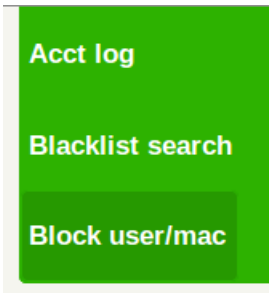
UserName	AcctStartTime	AcctStopTime	FramedIPAddress	externalIP	framedIPv6	block
kal0178	2017-04-29 18:26:23+02	2017-04-29 18:30:28+02	158.196.192.29 *vpnz14.vsb.cz*	90.182.31.3	2001:718:1001:111::5b	user+macs

Obrázek 4.4: Výsledky vyhledávání v logu účtování - VPN připojení

4.2 Blokace

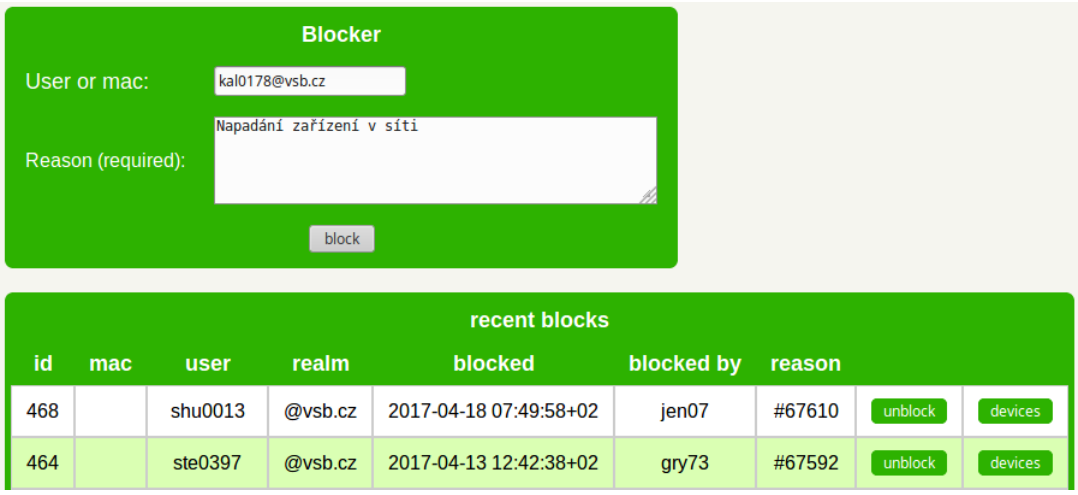
Blokaci uživatele lze provést pomocí tlačítka z výsledků vyhledávání (viz. obr.4.2) nebo přes menu klepnutím na odkaz *Block user/mac* (viz. obr.4.5). V případě, že je blokace provedena z výsledků vyhledávání, je formulář předvyplněn. Ve formuláři (viz. obr.4.6) pro blokaci se vyplňuje blokovaný uživatel nebo MAC adresa a důvod blokace.

V praxi se do důvodu blokace běžně vkládá číslo ticketu z vnitřního ticketovacího systému. V případě, že je blokován uživatel, systém nalezne MAC adresy zařízení, z kterých se uživatel přihlásil. Tyto adresy uvede v informační tabulce (viz. obr.4.7) a upozorní na zařízení, z kterých se přihlašovalo více uživatelů. Pokud administrátor na takovéto upozornění narazí, provede kontrolu (například dle počtu přihlášení), zda se opravdu jedná o zařízení blokovaného uživatele a nedošlo tak k blokaci někoho jiného.

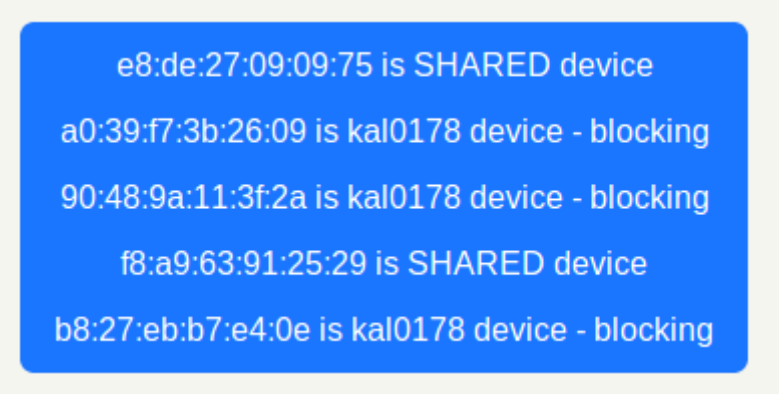


Obrázek 4.5: Menu uživatelského prostředí

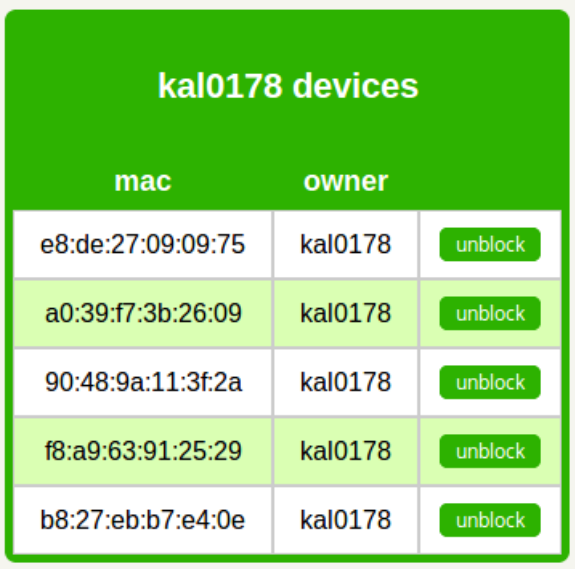
Na obrázku 4.6 můžeme vidět také seznam blokovaných uživatelů. K blokaci se automaticky přidává informace o tom, kdy byla blokace provedena, a osobní číslo administrátora, který blokaci provedl. Z tohoto seznamu lze blokace zrušit pomocí tlačítka *unblock*. Také lze pomocí tlačítka *devices* odblokovat jednotlivá zařízení. To se použije, pokud se i přes kontrolu při blokaci zablokovalo zařízení jiného uživatele.



Obrázek 4.6: Formulář pro blokaci uživatele



Obrázek 4.7: Informační tabulka po blokaci uživatele



Obrázek 4.8: Seznam zařízení blokovaných spolu s uživatelem

5 Popis vytvořených programových prostředků

V této kapitole jsou popsány jednotlivé součásti informačního systému a vazby mezi nimi. Kapitola je členěna na podkapitoly dle hlavního účelu jednotlivých částí systému.

5.1 Skripty pro sběr informací a jejich agregaci

ipv6monitor.pl

Připojuje se pomocí SNMPv2c protokolu na routery definované v poli *ip_list*. Přes tzv. *bulk-request* stahuje páry IPv6 adresa - MAC adresa a ty vkládá do databáze (tabulka *ipv6_daily*). Skript ignoruje *link-local* adresy - tj. adresy patřící do sítě FE80::/10.

Skript je umístěn na databázovém serveru a je pravidelně každých 5 minut spouštěn pomocí plánovače úloh Cron.

sqlradacct.pl

Je spouštěn RADIUS serverem RADIATOR při každém přijetí účtovacího paketu. Skript si od RADIUS serveru převezme vybrané atributy přijatého účtovacího paketu a ty přes místní smyčku pomocí UDP protokolu odešle na proxy server *acctProxy.pl*, odkud jsou atributy vloženy do databáze.

ipv6agregator.pl

Slouží pro agregaci dat z tabulky *ipv6_daily*. Nalezne první a poslední výskyt páru IPv6 adresa - MAC adresa a ten vloží do tabulky *ipv6_master*. Po úspěšné agregaci smaže neagregovanou tabulku a agregovanou tabulku starší než tři měsíce.

Skript je umístěn na databázovém serveru a je pravidelně spouštěn o půlnoci pomocí plánovače úloh Cron.

acctAgregator.pl

Slouží pro agregaci dat z tabulky *radacct*. Nalezne všechny ukončené relace a ty přesune do tabulky *accounting*. Odhaluje relace, které nebyly korektně ukončeny (na RADIUS server nepřišel paket *Accounting-Stop*). Nakonec smaže agregovanou tabulku, která je starší než tři měsíce.

Skript je umístěn na databázovém serveru a je pravidelně spouštěn o půlnoci pomocí plánovače úloh Cron.

acctProxy.pl

acctProxy.pl je jednoduchý proxy server. Naslouchá na volitelném UDP portu, kde mu skript sqlradacct.pl zasílá data, která mají být vložena do databáze.

Proxy server zahazuje data, která nebyla poslána přes místní smyčku nebo neúplná data. Server si udržuje trvalé šifrované spojení na databázový server. SQL dotazy jsou kvůli vyššímu výkonu předpřipravené.

V případě ztráty spojení na databázový server se spojení pokouší obnovit a přijatá data si ukládá v mezipaměti. Při obnově spojení data z mezipaměti vkládá v poměru 1:10 s nově přijatými daty. Pokud by dat v mezipaměti bylo mnoho a jejich vložení by trvalo dlouho, mohlo by dojít k přetečení vyrovnávací paměti.

Server běží jako démon na pozadí a je spouštěn automaticky při startu operačního systému. Je možné ho ovládat pomocí init skriptu *accounting*.

blacklistGenerator.pl

Skript získá seznam blokovaných uživatelů z databáze a vytvoří soubor naformátovaný pro použití se serverem Radiator. V případě, že je databázový server nedostupný, ponechá původní soubor nedotčen.

Skript je umístěn na RADIUS serveru a je každou minutu spouštěn pomocí plánovače úloh Cron.

5.2 Skripty pro monitoring

Skripty pro monitoring byly navrženy jako moduly pro informační systém Nagios. Ten umožňuje automatizovanou kontrolu jednotlivých služeb a v případě problému upozorní

systémové administrátory. Moduly signalizují stav služby pomocí návratových kódů. Významy jednotlivých kódů jsou následující:

- Kód 0 - v pořádku
- Kód 1 - upozornění
- Kód 2 - kritická chyba

Proto budu v popisu modulů uvádět, které kontroly provádí a jaký návratový kód vrátí, pokud modul odhalí problém.

check_acccounting

Je modul určený ke kontrole proxy serveru *acctProxy.pl*. Modul je spouštěn na RADIUS serveru. Kontroluje zda:

1. Běží démon (kód 2).
2. Se lze připojit k databázovému serveru (kód 1).
3. Zda se testovací data odeslána modulem na proxy server uloží do databáze (kód 2).

check_postgresql

Je modul určený ke kontrole databázového serveru, kde je také spouštěn. Modul kontroluje zda:

1. Se lze připojit k databázovému serveru (kód 2).
2. Se v denních tabulkách nachází nová data (kód 1).
3. Se provedla agregace (kód 1).

5.3 Webový server

Tabulka 5.1: Seznam PHP skriptů

název	popis
block.php	Stránka sloužící k blokování uživatelů.
credentials.php	Obsahuje funkce pro připojení se k databázi.
devices.php	Stránka pro zobrazení blokovaných zařízení uživatele.
functions.php	Funkce, které volají jednotlivé stránky.
index.php	Stránka sloužící k vyhledávání logů.
unblock.php	Zajišťuje odblokování uživatelů a hledání v historii blokací.
vendor.php	Zjišťuje výrobce síťové karty podle MAC adresy přes externí API.

Tabulka 5.2: Seznam javascript skriptů

název	popis
showHide.js	Minimalizuje/maximalizuje jednotlivé tabulky s výsledky vyhledávání.
vendor.js	Zobrazí výrobce síťové karty při najetí myši nad MAC adresu.

Tabulka 5.3: Použité knihovny třetích stran

název	popis
CAS	Umožňuje přihlášení uživatele přes školní systém jednotného přihlašování.
pikaday.js	Použito pro implementaci kalendáře.
moment.js	Použito pro implementaci kalendáře.

5.4 Databázový server

Seznam databází

Tabulka 5.4: Seznam databází

název	tabulky	obsah
radiusdb	radacct, accounting	Data z účtování.
ipv6db	ipv6_daily, ipv6_master	Data z tabulek sousedů.
blacklist	blacklist, blacklistarchive	Seznam blokovaných uživatelů.

Účty

Tabulka 5.5: Tabulka účtů pro přístup k databázovému serveru

účet	oprávnění
radius	Příkazy INSERT, UPDATE nad tabulkami v databázi radiusdb.
blacklist	Všechny příkazy nad tabulkami v databázi blacklist. Příkaz SELECT nad všemi tabulkami v databázi radiusdb.
ipv6usr	Příkaz INSERT nad tabulkou ipv6_daily.
web	Příkaz SELECT nad všemi tabulkami ve všech databázích.
agregator	Všechny příkazy nad všemi tabulkami v databázích radiusdb a ipv6db.

Partitioning tabulek

O partitioning tabulek se starají procedury napsané v PL/SQL. Tyto funkce jsou spouštěny pomocí spouštěčů (z anglického *trigger*). Kromě procedur určených k partitioningu tabulek uvádím i proceduru sloužící k uložení IPv6 adresy přidělené VPN koncentrátorem, která ze serveru FreeRadius přichází rozdělená do dvou sloupců (jako prefix a suffix IPv6 adresy). Adresa pak může být uložena jako datový typ *inet* (datový typ určený pro IPv4/IPv6 adresy).

Tabulka 5.6: Seznam procedur a spouštěčů

procedura	**spouštěč**	**účel**
ipv6_partition_function	ipv6_trigger	Partitioning tabulky ipv6_master.
acc_partition_function	acc_trigger	Partitioning tabulky accounting.
ipv6acc	ipv6acc_trigger	Sloučí prefix a suffix IPv6 adresy.

6 Závěr

Cílem práce bylo vyvinout informační systém, který umožní analýzu přístupů do sítě VŠB-TU a blokaci vybraných uživatelů. Bylo třeba identifikovat zdroje požadovaných informací, jejich získání a ukládání. Tento systém pak byl implementován do produkčního prostředí univerzity.

V úvodní kapitole jsem popsal nejdůležitější aspekty ohledně RADIUS protokolu, které jsou klíčové pro vývoj a implementaci požadovaných funkcionalit. Dále jsem přibližně popsal vazby mezi jednotlivými síťovými prvky, které se na autentizaci uživatelů podílí a stručně popsal jejich princip. Nakonec jsem, v návaznosti na identifikaci jednotlivých zařízení, popsal zásadní rozdíl mezi protokolem IPv4 a protokolem IPv6.

Po teoretickém úvodu jsem v následující kapitole zanalyzoval rozšiřitelnost nasazených softwarových prostředků umožňující autentizaci uživatelů. Po zhodnocení jejich možností jsem programy překonfiguroval, abych upravil jejich chování. Kvůli funkčnosti této konfigurace jsem pro program RADIATOR vyvinul programový modul, který z něj přeposílá požadované informace.

Ke správné funkci modulů bylo třeba také navrhnout databázi s ohledem na co nejvyšší rychlost zpracování dotazů. K zajištění této vlastnosti jsem vytvořil agregační skripty pro oddělení aktuálních a historických dat. Pro zajištění větší rychlosti vyhledávání historických dat byla použita technika zvaná *partitioning*. Pro rychlejší vkládání dat do databáze jsem se snažil co nejvíce využívat předpřipravených dotazů. Uvedené optimalizační metody jsem v kapitole také teoreticky popsal.

Kvůli absenci informací o IPv6 adresách jsem naprogramoval skripty, které pravidelně stahují tabulky sousedů ze směrovačů. Nakonec jsem popsal princip blokací jednotlivých uživatelů. Implementace těchto nástrojů také zahrnovala úplnou konfiguraci nově nainstalovaného SQL serveru a webového serveru.

V předposlední kapitole jsem popsal mnou vyvinuté webové rozhraní, které přehledně zobrazuje uložené údaje a umožňuje velice snadným způsobem uživatele zablokovat. Webové rozhraní také umožňuje autentizaci a autorizaci pověřených pracovníků.

V poslední kapitole jsem zdokumentoval všechny použité programové prostředky včetně vazeb mezi nimi. Do popisu jsem zahrnul vysvětlení účelu jednotlivých skriptů a databázových procedur. Dále je zde seznam uživatelů, kteří smí do databáze přistupovat, a jejich oprávnění. V kapitole jsou také popsány skripty určené pro monitoring jednotlivých služeb.

Systém bych chtěl nadále rozvinout tím, že bych rozšířil funkčnost webového rozhraní o automatické odesílání e-mailů zablokovaným uživatelům. Administrátoři by je tak nemuseli při každém incidentu psát sami, čímž by jejich práce byla ještě více zefektivněna. Dále bych chtěl zobecnit zdrojový kód, aby mohl být systém hladce nasazen i v jiných sítích.

Všechny uvedené nástroje již byly otestovány a implementovány do stávající sítě. Nástroje jsou úspěšně a rutinně používány v provozu při řešení bezpečnostních problémů i při technické podpoře uživatelů.

7 Literatura

DHCP 2016. Netacad.com [online]. [cit. 2016-11-27]. Dostupné z: https://static-course-assets.s3.amazonaws.com/RSE503/en/index.html#10.0.1.1.

Fungování roamingu 2016. Eduoram.cz [online]. 2012 [cit. 2016-11-27]. Dostupné z: https://www.eduroam.cz/cs/uzivatel/fungovani_roamingu.

GNU Radius Reference Manual 2008 [online]. Free Software Foundation, 2008 [cit. 2016-11-27]. Dostupné z: https://www.gnu.org/software/radius/manual/radius.pdf

Guide/SQL HOWTO 2016. The FreeRADIUS Server Project and Contributors [online]. 2016 [cit. 2017-01-11].
Dostupné z: https://wiki.freeradius.org/guide/SQL-HOWTO

HASSELL, J. 2002. *RADIUS*. 1005 Gravenstein Highway North, Sebastopol, CA 95472: O'Reilly Media, 2002. ISBN 0-596-00322-6.

KRČMÁŘ P.,CALETKA O. 2016. *Roamingová síť eduroam*: připojení po celém světě [online]. 2016 [cit. 2016-11-27]. Dostupné z:
https://www.root.cz/clanky/roamingova-sit-eduroam-pripojeni-po-celem-svete/

Radiator® RADIUS Server 2015 [online]. Open System Consultants Pty., 2015 [cit. 2016-12-30].
Dostupné z: https://www.open.com.au/radiator/ref.pdf

Remote Authentication Dial In User Service (RADIUS) 2000. The Internet Engineering Task Force [online]. 2000 [cit. 2016-11-27].
Dostupné z: https://tools.ietf.org/html/rfc2865.

SATRAPA, P. c2000. *Perl pro zelenáče: [naučte se programovat v Perlu].* 1. Praha: Neokortex. Bestseller for all. ISBN 80-863-3002-8.

SATRAPA, P. 2011. *IPv6: internetový protokol verze 6*. 3., aktualiz. a dopl. vyd. Praha: CZ.NIC, c2011. CZ.NIC. ISBN 9788090424845.

Server Definition 2014. TechTerms [online]. 2014 [cit. 2016-11-27]. Dostupné z: http://techterms.com/definition/server

ŽÁK, K. 2004. *PostgreSQL: připravené dotazy a oddělení dat od dotazů*. Root.cz [online]. 2004 [cit. 2016-12-30]. Dostupné z: https://www.root.cz/clanky/postgresql-pripravene-dotazy-a-oddeleni-dat-od-dotazu/