UNIVERZITA PARDUBICE

FAKULTA ELEKTROTECHNIKY A INFORMATIKY

# BAKALÁŘSKÁ PRÁCE

2016 DenisaPlecháčková

Univerzita Pardubice

Fakulta elektrotechniky a informatiky

# Principy bezpečnosti využívající AAA mechanismus

DenisaPlecháčková

Bakalářská práce

2016

**Univerzita Pardubice**
**Fakulta elektrotechniky a informatiky**
Akademický rok: **2015/2016**

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

| | |
|---|---|
| Jméno a příjmení: | **Denisa Plecháčková** |
| Osobní číslo: | **I12357** |
| Studijní program: | **B2646 Informační technologie** |
| Studijní obor: | **Informační technologie** |
| Název tématu: | **Principy bezpečnosti využívající AAA mechanismus** |
| Zadávající katedra: | **Katedra informačních technologií** |

Zásady pro vypracování:

Cílem práce je vysvětlit principy zabezpečení komunikace s využitím protokolu AAA a na vybrané úlohy připravit ukázkové řešené úlohy. V teoretické části autor představí základní principy AAA protokolu (authentication, authorization and accounting protocol)a zaměří se zejména na představení protokolů RADIUS, DIAMETER, TACACS, TACACS+, PPP, EAP a LDAP.
Na využití, konfiguraci a testování vybraných protokolů autor připraví sadu řešených úloh, které představí jejich silné a slabé stránky, konfiguraci a ověření funkčnosti.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování bakalářské práce: **tištěná**

Seznam odborné literatury:

*** SASU TARKOMA. Mobile Middleware Supporting Applications and Services. Chichester: John Wiley & Sons, 2009. ISBN 9780470745526.**
*** P. Calhoun, J. Loughney, E. Guttman, G. Zorn, J. Arkko, "Diameter Base Protocol", IETF RFC 3588, September 2003.**

Vedoucí bakalářské práce: **Mgr. Josef Horálek, Ph.D.**
Katedra informačních technologií

Datum zadání bakalářské práce: **31. října 2015**
Termín odevzdání bakalářské práce: **13. května 2016**

Univerzita Pardubice
Česká republika

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

Mgr. Josef Horálek, Ph.D.
vedoucí katedry

V Pardubicích dne 31. března 2016

Prohlášení autorky

Prohlašuji, že jsem tuto práci vypracovala samostatně. Veškeré literární prameny a informace, které jsem v práci využila, jsou uvedeny v seznamu použité literatury.

Byla jsem seznámena s tím, že se na moji prácivztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 17. 05. 2016

Denisa Plecháčková

## PODĚKOVÁNÍ

Ráda bych tímto poděkovala vedoucímu své bakalářské práce, Mgr. Josefu Horálkovi Ph.D., za cenné rady a připomínky při konzultacích. Také bych chtěla poděkovat rodičům za jejich podporu při studiu.

## ANOTACE

Tato práce popisuje nejčastěji používané protokoly AAA – RADIUS, Diameter, TACACS, EAP, PPP a LDAP. V praktické části je ověřena funkčnost protokolu RADIUS,TACACS+ a LDAP.

## KLÍČOVÁ SLOVA

AAA, RADIUS, Diameter, TACACS, TACACS+, EAP, PPP, LDAP, autentizace, autorizace, účtování, zabezpečení

## TITLE

Principles of security using AAA mechanism

## ANNOTATION

This bachelor's thesis describes the most commonly used AAA protocols – RADIUS, Diameter, TACACS, EAP, PPP and LDAP. In the practical part, there is verified functionality of RADIUS, TACACS+ and LDAP.

## KEYWORDS

AAA, RADIUS, Diameter, TACACS, TACACS+, EAP, PPP, LDAP, authentication, authorization, accounting, security

# OBSAH

# SEZNAM ILUSTRACÍ A TABULEK

# SEZNAM ZKRATEK A ZNAČEK

| | |
|---|---|
| AAA | Authentication, Authorization, Accounting |
| AES-128 | Advanced Encryption Standard-128 |
| AV | Attribute-Value |
| AVP | Attribute-Value Pair |
| DAP | Directory Access Protocol |
| DIT | Directory Information Tree |
| DN | Distinguished Name |
| DNS | Domain Name Server |
| DSA | Directory System Agent |
| DSE | DSA Specific Entry |
| EAP | Extensible Authentication Protocol |
| EAP-AKA | EAP- Authentication and Key Agreement |
| EAP-IKEv2 | EAP-Key Exchange version 2 |
| EAP-PSK | EAP-Pre-Shared Key |
| EAP-SIM | EAP-Subscriber Identity Modul |
| EAP-SPEKE | EAP-Simple Password-Authenticated Exponential Key Exchange |
| EAP-TLS-PSK | EAP-Transport Layer Security-Pre-Shared Key |
| EAP-FAST | EAP-Flexible Authentication via Secure Tunneling |
| EAPOL | EAP Over LAN |
| EAP-PAX | EAP-Password Authenticated eXchange |
| EAP-TLS | EAP-Transport Layer Security |

| | |
|---|---|
| EAP-TTLS | EAP-Tunneled Transport Layer Security |
| FCS | Frame Check Sequence |
| GPRS | General Packet Radio Service |
| GSM | Global System for Mobile Communications |
| HDLC | High-Level Data Link Control |
| HTTP | Hypertext Transfer Protocol |
| CHAP | Challenge Handshake Authentication Protocol |
| ID | Identification |
| IP | Internet Protocol |
| LCP | Link Control Protocol |
| LDAP | Lightweight Directory Access Protocol |
| LDIF | LDAP Data Interchange Format |
| LEAP | Lightweight Extensible Authentication Protocol |
| MD4 | Message-Digest 4 |
| MD5 | Message-Digest 5 |
| NAI | Network Access Identifier |
| NAS | Network Access Server |
| NCP | Network Control Protocol |
| PAC | Protected Access Credential |
| PAP | Password Authentication Protocol |
| PEAP | Protected Extensible Authentication Protocol |
| PKI | Public Key Infrastructure |

| | |
|---|---|
| PPP | Point-to-Point Protocol |
| RADIUS | Remote Authentication Dial In User Service |
| RDN | Relative Distinguished Name |
| RFC | Request For Comments |
| SASL | Simple Authentication and Security Layer |
| SCTP | Stream Control Transmission Protocol |
| SIM | Subscriber Identity Modul |
| SRVLOC | Service Location Protocol |
| SSL | Secure Sockets Layer |
| TACACS | Terminal Access Controller Access-Control System |
| TCP | Transmission Control Protocol |
| TLS | Transport Layer Security |
| UDP | User Datagram Protocol |
| WEP | Wired Equivalent Privacy |
| XTACACS | Extended Terminal Access Controller Access-Control System |

# 0 ÚVOD

V dnešní době narůstá počet koncových zařízení, což souvisí se vznikem počítačových sítí. S tímto narůstajícím počtem roste i počet útoků, jejichž cílem je například poškodit nebo zcizit data. Z těchto důvodů je třeba sítě zabezpečovat a neustále zvyšovat jejich ochranu.

S vývojem bezpečnostních opatřeních vznikly tzv. protokoly AAA. Protokoly AAA zajišťují ověření totožnosti uživatele, přidělují mu určitá práva v systému a také dokáží sbírat data týkající se uživatele. Předchozí tři části se nazývají autentizace, autorizace a účtování.

Tato práce se zabývá bezpečností počítačových sítí, konkrétně využitím protokolů AAA.Téma jsem si vybrala především z důvodu mého zaujetí počítačovými sítěmi a snahou prohloubit si znalosti o možnostech jejich zabezpečení.

Cílem práce bude popsat principy nejčastěji používaných protokolů AAA a vybrané z nich prakticky ověřit. V první kapitole lze nalézt provedenou rešerši, týkající se tématu protokolů AAA. Vysvětlení samotného pojmu AAA bude věnována druhá kapitola, ve které jsou popsány jednotlivé složky tohoto protokolu – autentizace, autorizace a účtování. V dalších kapitolách se poté věnuji jednotlivým protokolům. Třetí kapitola obsahuje informace o jednom z nejpoužívanějších protokolů, protokolu RADIUS. Čtvrtá kapitola popisuje protokol Diameter, jež je následovníkem protokolu RADIUS. V páté kapitole lze nalézt informace o protokolu TACACS a jeho vývoj v aktuální verzi TACACS+. Jeden z dalších rozšířených protokolů je protokol PPP, obsažený v kapitole šest. Kapitola sedm popisuje protokol EAP a dále typy metod, jež lze využít k autentizaci. V kapitole osm bude rozebrán protokol LDAP a vysvětleny jeho 4 modely, z nichž se skládá. V praktické části práce se budu věnovat ověření funkčnosti protokolu RADIUS, s využitím programu WinRadius. Poté otestuji protokol TACACS+ a jako poslední LDAP, který bude aplikován na virtuální síti.

# 1 LITERÁRNÍ REŠERŠE PRINCIPŮ BEZPEČNOSTI MECHANISMU AAA

Tato kapitola popisuje základní principy mechanismu AAA a pro něj důležité protokoly. Budou zde uvedeny knihy, které se přímo věnují této problematice.

Protokol AAA znamená Authentication, Authorization and Accounting protocol (česky autentizační, autorizační a účtovací protokol), který slouží k zabezpečení počítačových sítí. Tento protokol bude podrobně probrán v následujících kapitolách.

H. Ventura ve své práci Diameter Next generation's AAA protocol vysvětluje základní principy AAA. Autentizace, první krok, vyžaduje od uživatele při přístupu na síť potvrzení jeho identity, nejčastěji uživatelským jménem a heslem nebo také tajným klíčem atd. Ty jsou v případě malé firmy uloženy v místní databázi a porovnávány se zadanými informacemi. Pokud je síť rozsáhlá, databáze se nachází na centrálním serveru. K tomuto serveru směrovač (router) přistupuje nejčastěji pomocí protokolu RADIUS (Remote Authentication Dial In User Service) nebo TACACS (Terminal Access Controller Access Control System). Autor dále popisuje RADIUS jako protokol, který vznikl za účelem „dial-up" PPP a terminálového přístupu k serveru. V současnosti je to jeden z hlavních protokolů AAA pro připojení k síti. Ventura také porovnává výhody a nevýhody mezi protokoly RADIUS a Diameter, který je jeho následníkem. Diameter se skládá z několika částí, a to z Diameter Base Protocol, jeho rozšířením a aplikacemi. Tomuto protokolu autor věnuje celou kapitolu (2004).

Diameter, stejně jako TACACS+, využívá k posílání zpráv protokol TCP (na rozdíl od protokolu RADIUS, který používá UDP). Funguje na principu „peer-to-peer", kdy je vyslán požadavek od klienta agentovi, který poskytuje služby. Požadavek je nakonec zpracován serverem. Touto problematikou se zabývají i Jyh-Cheng Chen a Tao Zhang v knize IP-Based Next-Generation Wireless Networks: Systems, Architectures, and Protocols (2004).

Mezi další AAA protokoly se řadí i výše zmíněný TACACS+, jež je podrobně vysvětlený v knize AAA Identity Management Security, kterou napsali V. Santuka, P. Banga a B. J. Carrol (2011). TACACS je zastaralý protokol, který využíval UDP. Později se vyvinul v rozšířený TACACS (XTACACS) nabízející více funkcí, ale stále používající UDP. Nejnovější TACACS+ je zpětně nekompatibilní s předchozími verzemi. TACACS+ umožňuje šifrovanou komunikaci mezi klientem a serverem. Santuka a další ve své literatuře dále popisují zašifrování zprávy, autentizaci, autorizaci a účtování.

Mechanismu AAA využívá i protokol EAP (Extensible Authentication Protocol). Mogollon ve své knize uvádí, že EAP je autentizační protokol (2007), který umožňuje zavedení nových protokolů mezi žadatelem a autentizačním serverem. Zapouzdřovací technika je známá jako EAP over LAN neboli EAPOL. EAP podporuje mechanismy, jako jsou Kerberos, digitální certifikáty a další. Tyto mechanismy mohou být implementovány tzv. EAP metodami, např. EAP-TTLS, EAP-PEAP, které autor později rozebírá (2007), a v této práci jim bude věnována samostatná kapitola.

Další důležitý protokol se nazývá PPP (Point-to-Point Protocol), který slouží k přenášení paketů mezi dvěma uzly, zajišťující spojení mezi klientem a přístupovým síťovým serverem (Network Access Server, zkráceně NAS). Při vytváření spojení protokol PPP pošle pakety LCP, které nastaví a otestují komunikaci. Následně může být provedena autentizace. Dále jsou rozeslány pakety NCP, jež vyberou a nastaví jeden či více protokolů síťové vrstvy. (Mogollon 2007)

Nesmí se také opomenout důležitý LDAP (Lightweight Directory Access Protocol), který Tuttle a další ve své publikaci definují jako technologii pro přístup ke společným adresářovým informacím. (2004)

Druhým krokem po úspěšné autentizaci je autorizace, která definuje práva uživatele – říká, co smí a co nesmí dělat. Účtování je posledním krokem protokolu AAA, kdy jsou sbírána data o uživateli, například provedené příkazy, čas připojení a odpojení, počet paketů a další. (Ventura 2004)

# 2 PROTOKOL AAA

V této kapitole bude popsán protokol AAA a jeho jednotlivé části, tedy autentizace, autorizace a účtování.

## 2.1 Popis

Protokol AAA poskytuje základní zabezpečení sítě pomocí tří nezávislých služeb: autentizace, autorizace a účtování. Ve většině případů využívá protokol AAA některý z bezpečnostních protokolů, jako je například TACACS+, RADIUS či Kerberos. Protokol AAA stanovuje komunikaci mezi síťovým přístupovým serverem a zabezpečeným serverem, na kterém běží některý z výše zmíněných protokolů. Obecný popis protokolu AAA lze najít v dokumentu RFC 2903. (Podrobnější údaje se nacházejí v dokumentech RFC 2904 a RFC 2906.)

## 2.2 Autentizace

Autentizace je proces ověření totožnosti jedné nebo obou komunikujících stran. (Pužmanová 2004, s. 44) Během ní jsou identifikováni uživatelé, například pomocí jména a hesla, dále probíhá výměna zpráv, požadavků a odpovědí, případně i šifrování. Identifikovanému uživateli je povolen přístup do sítě a k síťovým službám. Při konfiguraci těchto služeb si lze zvolit typy autentizačních metod a v jakém pořadí budou prováděny na předem určených rozhraních. (*CiscoSystems*,2014, s. 15)

## 2.3 Autorizace

Autorizace zajišťuje identifikovanému uživateli přístup do systému a přidělení práv. (Pužmanová 2004, s. 44) Práva jsou reprezentována formou atributů, které jsou vždy porovnávány s databází pro daného uživatele. Výsledek je vrácen AAA pro určení aktuálních možností a omezení uživatele. Tato databáze může být umístěna lokálně na přístupovém serveru (směrovači) anebo vzdáleně na zabezpečeném serveru. Tyto vzdálené zabezpečené servery (remote security servers) používají k přidělení určitých práv uživateli páry atribut-hodnota (Attributte-Value pairs). Všechny autorizační metody musí být definovány skrze AAA příkaz. A stejně jako u výše uvedené autentizace, si lze zvolit autorizační metody a aplikovat je na určená rozhraní. (*Cisco Systems*, 2014, s. 16)

## 2.4 Účtování

Účtování poskytuje metodu pro sběr a odesílání zabezpečených informací serveru. Tyto informace slouží například jako podklady pro vyúčtování, audity a podávání různých zpráv

(o počtu bajtů, počtu paketů atd.). Když je účtování AAA aktivní, přístupový server podává zprávy o aktivitách uživatele (tzv. účtovací záznamy – accounting records) síťovému přístupovému serveru jako je TACACS+ nebo RADIUS. Každý účtovací záznam se skládá z účtovacích AV párů, jež jsou uloženy na kontrolním přístupovém serveru (control access server). Tato data mohou být dále zpracovávána pro síťový management či vyúčtování klienta. Účtování AAA musí být povoleno příkazem aaa. Stejně jako u předchozích dvou služeb, i tato umožňuje volbu z různých účtovacích metod. (*Cisco Systems*, 2014, s. 16)

AAA poskytuje různé druhy účtování:

- Síťové účtování – poskytuje účtování pro protokoly PPP, Serial Line Internet Protocol, Apple Remote Access Protocol.
- Účtování spojení – podává informace o odchozích spojeních od klienta, např. Telnet, Local Area Transport aj.
- Účtování EXEC – poskytuje informace o terminálových spojeních na síťovém přístupovém serveru. Mezi tyto informace patří například adresa IP, datum, uživatelské jméno atd.
- Systémové účtování – poskytuje informace o systémových událostech.
- Příkazové účtování – poskytuje informace o tom, jaké byly provedeny EXEC „shell“ příkazy na přístupovém serveru.
- Účtování zdroje – poskytuje záznamy (records) o autentizovaných začátcích a koncích hovorů. (Santuka a spol. 2011, s. 7)

# 3 RADIUS

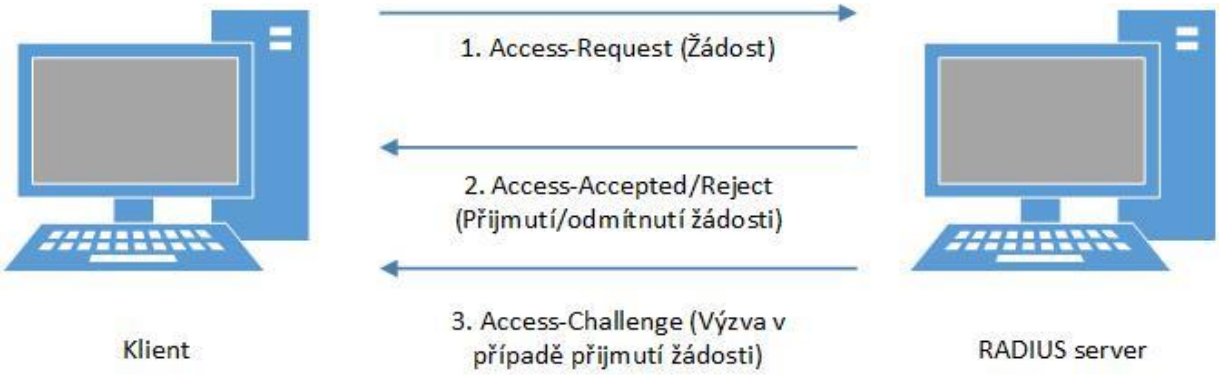
Tato kapitola bude zaměřena na jeden z AAA protokolů, konkrétně RADIUS. Kapitola je rozdělena na dvě podkapitoly. V první podkapitole je popsán protokol RADIUS, princip komunikace a autentizace spolu s autorizací. V druhé podkapitole jsou rozebrány specifické vlastnosti pro protokol RADIUS – RADIUS Accounting a šifrování.

## 3.1 Popis

RADIUS (Remote Authentization Dial-In User Service) je protokol bezpečnostních služeb pro klienty připojované po telefonu. (Naik 1999, s. 75) Jeho cílem je centralizovaná autentizace připojovaných uživatelů. (Dostálek 2001, s. 140) RADIUS používá protokol UDP. Pro autentizaci využívá porty 1645 a 1812, pro účtování potom 1646 a 1813. (Carroll 2004) Jeho úplný popis lze nalézt v dokumentu RFC 2685.

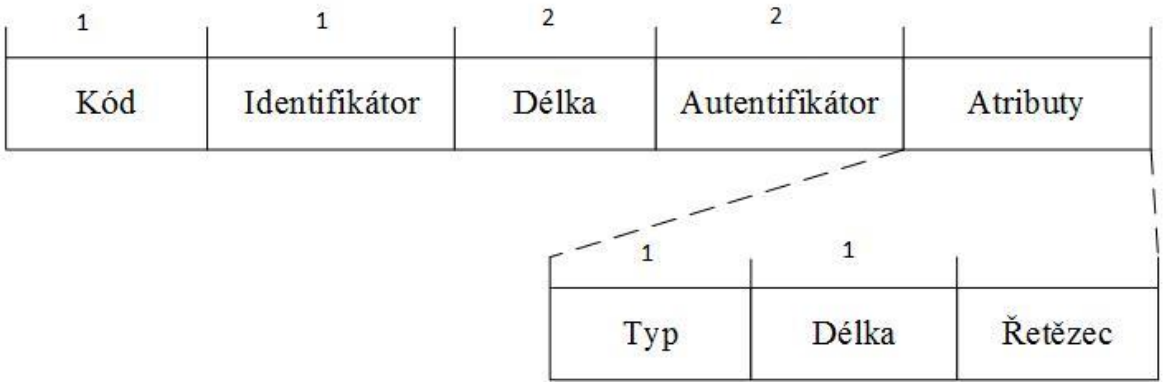
### Komunikace

Protokol RADIUS používá 4 typy zpráv. Prvním typem je „Access-Request“, který slouží pro odeslání požadavku na autentizaci uživatele. Druhým typem je „Access-Accepted“, kterým server RADIUS povoluje přístup do sítě. Dalším je „Access-Reject“, kterým server RADIUS zamítá přístup uživatele do sítě. Posledním typem zprávy je „Access-Challenge“, jež slouží serveru pro odeslání výzvy na zadání hesla. Názorná ukázka této komunikace viz obrázek č. 1. (Dostálek 2001, s. 141)




**Obrázek 1 – Komunikace protokolu RADIUS**

Formát paketu (viz. Obrázek č. 2) obsahuje kód, který specifikuje typ zprávy („Access-Reject“, „Access-Accept“, „Access-Request“, „Access-Challenge“). Pole s názvem identifikátor slouží k párování požadavků a jejich odpovědí na ně. Následuje pole s uvedenou délkou paketu. Předposlední pole, autentifikátor, slouží k prokázání pravosti odpovědi serveru. Poslední pole je vyhrazeno pro atributy. Ty se dělí na další pole. Mezi ně patří typ, určující typ atributu. Poté délka a řetězec (hodnota). Tento řetězec může být textovým řetězcem, řetězcem bajtů, adresou IP nebo časem. (Dostálek 2001, s. 141) Seznam atributů lze nalézt například v dokumentu RFC 2865.




**Obrázek 2 – Formát paketu protokolu RADIUS**

**Zdroj: Přepracováno podle Dostálka**

### Autentizace a autorizace

Protokol RADIUS poskytuje dva druhy autentizace: autentizaci klienta a autentizaci serveru RADIUS. V prvním případě, kdy je autentizován klient při přihlašování k přístupovému serveru, může RADIUS implementovat různé autentizační mechanismy. Například při autentizaci pomocí hesla uživatele je jméno a heslo přenášeno ve zprávě typu „Access-Request“, na kterou server reaguje zprávou „Acces-Accepted“ nebo „Access-Reject“. Jméno a heslo jsou obsaženy v atributech zprávy „User-Name“ a „User-Password“. V případě autentizace jednorázovým heslem je odeslána zpráva „Access-Challenge“ (výzva). Server poté odešle žádost „Access-Request“, ve které požaduje heslo uživatele. Výzva a odpověď jsou spárovány pomocí atributu stav (state). V případě, že samotný server nedokáže ověřit klienta (potřebné informace jsou uloženy na jiném serveru), pracuje server RADIUS jako proxy a požaduje autentizaci od vzdáleného serveru. V druhém případě – autentizace serveru RADIUS – probíhá autentizace pomocí sdíleného tajemství, které zná jen klient a server. (Dostálek 2001, s. 141–143)

## 3.2 Specifické vlastnosti

V této části práce budou popsány části protokolu RADIUS, které jsou pro něj specifické.

### RADIUS Accounting

Pomocí protokolu RADIUS Accounting lze získat údaje o uživatelích. Zaznamenává přihlašování a odhlašování uživatelů, datum a čas přihlášení, množství přenesených bajtů atd. Tyto údaje využívají poskytovatelé Internetu a na základě těchto informací často vznikají faktury pro zákazníky. V protokolu RADIUS Accounting existují dva typy zpráv, a to „Accounting-Request“ a „Accounting-Response“. Zprávy „Accounting-Request“ obsahují informace, které mají být zapsány na server. Zprávami typu „Accounting-Response“ server potvrzuje přijetí zprávy „Accounting-Request“. Protokol RADIUS Accounting využívá atributy z protokolu RADIUS a přidává k nim vlastní atributy týkající se účtování. (Dostálek 2001, s. 144)

### RADIUS Šifrování

Protokol RADIUS zašifruje pouze hesla a zbytek informací zůstává ve formě čistého textu. Proces šifrování probíhá následovně:

1. Hodnota v paketu v poli autentifikátor je zkombinována s hodnotou tajného klíče. Poté je zašifrována pomocí funkce MD5, která vrátí 16 oktetový haš.
2. Heslo uživatele je vyplněno ve zprávě s hodnotou nula, takže dosahuje velikosti 16 oktetů. Haš z kroku 1 je poté „zxorována“ tímto heslem, což vygeneruje šifrovaný text. Ten je dále přenášen na server AAA, na kterém běží protokol RADIUS.
3. Server AAA spočítá vlastní haš a „zxoruje“ ji obdrženým šifrovaným textem, aby získal zpět heslo ve formě čistého textu. (Carroll 2004)

# 4 DIAMETER

Tato kapitola pojednává o protokolu Diameter a je rozdělena do 3 částí. V první části je vysvětlen protokol Diameter, jeho typ zpráv a proces komunikace. Následně je popsána autentizace, autorizace a účtování. V poslední části jsou uvedené specifické vlastnosti protokolu Diameter – princip zachycení chyby, nalezení uzlu a rozdíl mezi „connection“ a „session“.

## 4.1 Popis protokolu Diameter

Diameter je základní protokol pro poskytování služeb AAA. Diameter většinou není používán samostatně. Pro spolehlivé doručení zpráv využívá protokol TCP. Diameter je dvoubodový (point-to-point) protokol. Entita generující požadavek je klient, který také provádí řízení přístupu na okraji sítě. Požadavek může být odeslán agentovi, který poskytuje ochranu, proxy, přesměrování a překladatelské služby. Tento požadavek je poté zpracován serverem. Protokol Diameter používá k identifikaci uživatele a směrovací zprávy tzv. síťový přístupový identifikátor – NAI (Network Access Identifier). Agent odešle požadavek určitému serveru podle jeho směrovací tabulky, která je nazývána oblastní směrovací tabulka (Realm Routing Table). Oblast je chápána jako část NAI poskytující služby uživateli. (Chen 2004)

### Tvar zpráv

Zprávy Diameteru přenášejí informace o protokolu AAA. Informace ve zprávě Diameter je obvykle nazývána atribut, který je tvořen AVP (Attribute-Value Pair) formátem. Existují dva typy zpráv: požadavek a odpověď. V rámci těchto typů používá Diameter příkazy, které jsou od sebe odlišeny pomocí příkazového kódu specifikujícího funkci zprávy. Akce, která má být s každou přijatou zprávou vykonána, je definována příkazovým kódem a atributy. (Nakhjiri, M. a M. Nakhjiri 2005, s. 153) Formát zprávy obsahuje v prvním poli verzi protokolu Diameter. V dalším poli, příznaky příkazu (command flags), existují 4 typy příznaků:

- R (Request) – Znamená, že zpráva je buď požadavek, nebo odpověď.
- P (Proxiable) – Říká, zda může být zpráva zastoupena, předána či přesměrována.
- E (Error) – Ukazuje, že zpráva obsahuje chyby.
- T – Tento příznak ukazuje, zda může být zpráva opakovaně vysílána při selhání.

V dalším poli se nachází kód příkazu (command code), který obsahuje kód spjatý se zprávou. Délka zprávy nese hodnotu o délce zprávy, jak je patrné z názvu. Následuje aplikační ID,které

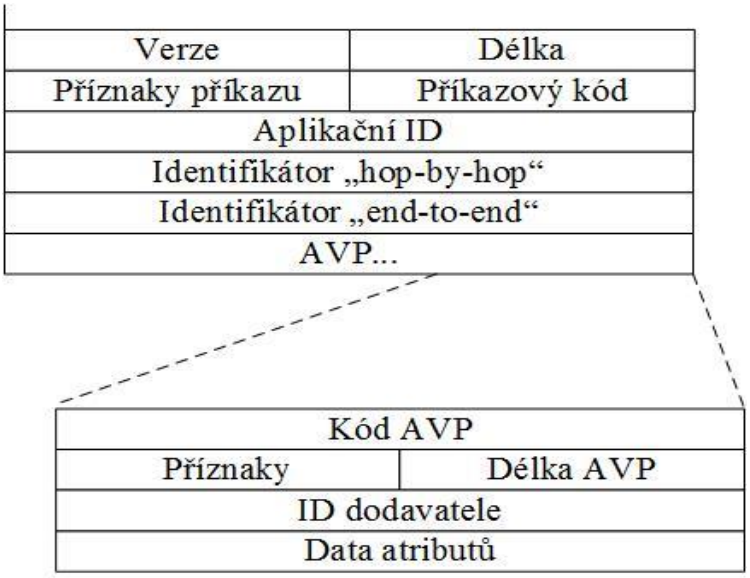
určuje, pro kterou aplikaci je zpráva použita (například účtování). Identifikátor „Hop-by-hop“ slouží k určení požadavků a odpovědí během aktuálního skoku. Odesilatel požadavku se musí ujistit, že identifikátor je v rámci spojení jedinečný. Také se ujišťuje o totožnosti identifikátoru požadavku a jeho odpovědi, tedy že k sobě požadavek a odpověď pasují. Identifikátor „End-to-end“ je používán pro detekci duplicitních zpráv. Identifikátor požadavku musí být shodný s identifikátorem odpovědi. Ten musí zůstat jedinečný alespoň 4 minuty. Jako poslední pole je AVP, které je rozděleno na dalších 5 částí, jak je patrné z obrázku č.3. První je kód AVP, který určuje typ atributu. Tyto hodnoty jsou kontrolovány organizací Internet Assigned Number Authority. Pro protokol Diameter jsou rezervovány čísla od 0 do 255. Příznaky mohou obsahovat následující hodnoty:

- M – Nazýván jako povinný bit. Slouží k určení, zda uzel Diameteru vyžaduje od prvku (peer) podporu atributu při odesílání zprávy. Pokud uzel Diameter obdrží AVP s nastaveným M bitem, ale nerozezná AVP nebo jeho hodnotu, musí tuto zprávu zamítnout.
- P – Označuje potřebu zašifrování pro koncovou (end-to-end) bezpečnost. (Nakhjiri, M. a M. Nakhjiri 2005, s. 154–156)
- R – RRRRR bity jsou rezervované bity a měly by být nastaveny na 0. (Calhoun a spol. 2003, s. 39)
- V – Bit značící, zda je obsaženo pole ID dodavatele (vendor-id). Bit V nemusí být nastaven.

Dalším polem v AVP je jeho délka, obsahující počet oktetů ve zprávě (součet kódu AVP, délky AVP, příznaků AVP, ID dodavatele a data AVP). V poli ID dodavatele lze nalézt zakódovanou hodnotu, přidělenou organizací IANA. Každý dodavatel, který chce implementovat specifické AVP, musí použít vlastní ID spolu s vlastním adresním prostorem. V posledním poli, data atributů, se nachází data specifická pro atributy. Formát tohoto pole musí být shodný s formátem definovaným v RFC. (Calhoun a spol. 2003, s. 39–41)

### Seskupené hodnoty AVP

Diameter umožňuje seskupit hodnoty AVP do sekvence AVP. Rozdíl mezi seskupeným a jednoduchým AVP je v poli dat, kde seskupené AVP obsahuje jedno nebo více AVP. Tyto AVP se chovají jako data. (Calhoun a spol. 2003, s. 49)




**Obrázek 3 – Formát paketu Diameter**

**Zdroj: Přepracováno podle NakhjiriM. a M. Nakhjiri**

### Komunikace

Komunikace v protokolu Diameter probíhá v těchto fázích:

1. Klient odešle požadavek serveru, respektive „Auth-Request“ zprávu.
2. Server obdrží zprávu a do své odpovědi může vložit AVP „Authorization-Lifetime“, které říká, kolik času je třeba pro opětovnou autorizaci klienta. V případě vypršení tohoto limitu server Diameter odstraní spojení ze svého seznamu a uvolní všechny alokované zdroje. Pro výjimečné ukončení spojení se používá AVP „Origin-State-Id“.

Zprávy s ukončením spojení jsou používány u autentizace a autorizace. V případě účtování je použita stop zpráva. Zpráva o ukončení spojení může být odeslána klientem nebo serverem. Má-li být spojení ukončeno, klient Diameter odešle zprávu „Session-Termination-Request“ serveru s odůvodněním, proč má být spojení ukončeno. Naopak, pokud server detekuje, že má být spojení ukončeno, odešle zprávu „Abort-Session-Request“ klientovi Diameter, který může a nemusí spojení ukončit. (Liu a spol. 2006)

### Autentizace a autorizace

Protokol Diameter z důvodu různých autentizačních a autorizačních mechanismů aplikací nedefinuje příkazové kódy a specifické AVP pro autentizaci a autorizaci. Tyto mechanismy si

zařizují aplikace sami. Například v aplikaci přístupového serveru je pro autentizaci a autorizaci použita zpráva „AAA-Request“. (U aplikace SIP je tato zpráva nazývána „User-Authorization-Request“.) (Liu a spol. 2006)

### Účtování

Na rozdíl od autentizace a autorizace je účtování definováno. Účtování v podstatě navazuje na serverem řízený model, což znamená, že zařízení, které generuje účtovací záznamy, sleduje směr autorizačního serveru. Server Diameter informuje příslušného klienta Diameteru, jak často budou účtovací záznamy posílány od klienta na server, anebo zda má být generován účtovací záznam současně s účtovacím spojením.

Existují dva druhy účtovacích záznamů: Pro jednorázové služby je použit EVENT_RECORD. Pokud je služba poskytována v určité periodě, je použit START_RECORD pro začátek spojení, INTERIM_RECORD pro aktualizaci a TOP_RECORD pro ukončení spojení.

Aby nedocházelo k duplicitním záznamům, každá zpráva obsahuje ID spojení a číslo účtovacího záznamu. Uzel Diameteru chovající se jako agent Diametru může tuto informaci použít k detekci duplicitních účtovacích zpráv odeslané serveru. Tím lze zabránit zbytečnému používání serveru a následně problémům v síti a výpadkům u klienta. (Liu a spol. 2006)

## 4.2 Specifické vlastnosti Diameteru

V této podkapitole budou uvedeny části protokolu, které jsou specifické pro Diameter.

### Rozdíl mezi „Connection“ a „session“

„Connection“ (propojení) je fyzické propojení mezi dvěma Diameter uzly. Diameter musí běžet na protokolu TCP nebo SCTP, které poskytují spolehlivý přenos. Diameter uzel musí být propojen minimálně se dvěma dalšími uzly.

„Session“ (spojení) je logické propojení mezi dvěma uzly a může procházet mnoha propojeními. Je to také interakce mezi klientem a serverem za určitý čas. Každé spojení má své unikátní ID spojení, jak již bylo popsáno výše.

### Zachycení chyby

Chyby v Diameteru mohou být dvojího druhu: protokolové a aplikační. Chyby protokolu mohou značit špatný podkladový protokol, který přenáší zprávy Diameteru.Další možností jsou nesprávné směrovací informace či trvalý výpadek sítě. Aplikační chyby vycházejí ze selhání samotného Diameteru a jejich příčin je mnoho. Každá odpověď obsahuje AVP „Result-Code“, díky které si může příjemce zprávy ověřit, zda byla zpráva úspěšně doručena. Pro včasnou detekci výpadku sítě má Diameter zřízen zprávu „Device-Watchdog-Request“. Pokud si dva uzly po určitou dobu nevyměňují žádné zprávy, je pro zjištění problému odeslána tato zpráva. Protokol Diameter používá stejné chybové kódy jako protokol HTTP, pomocí první číslice vrácené hodnoty:

- 1xxx znamená, že požadavek nemůže být splněn a jsou potřebné dodatečné informace.
- 2xxx znamená, že požadavek byl úspěšně proveden.
- 3xxx znamená, že při posílání zprávy se vyskytla chyba protokolu.
- 4xxx znamená, že požadavek nemůže být momentálně uskutečněn.
- 5xxx znamená, že se vyskytla aplikační chyba při odesílání zprávy serveru.

Dalším způsobem, jak zjistit chybu, je AVP „Error-Message“, nesoucí zprávu o chybě nebo AVP „Error-Reporting-Host“, které identifikuje hostitele, u kterého nastala chyba. Po detekci chyby odesílající uzel doručí všechny nedoručené zprávy jinému Diameteru uzlu (tzv. „failover“). Každý Diameter uzel si uchovává kopie svých odchozích zpráv. (Liu a spol. 2006)

### Nalezení uzlu

Diameter dokáže nalézt své uzly dynamicky pomocí SRVLOC nebo DNS. Server odešle „broadcast“ obsahující informace o tom, jaké podporuje aplikace a úroveň zabezpečení. Klienti poté na základě těchto informací vyhledají „first-hop“ uzly, kterým budou odesílat zprávy. Takto nalezené uzly jsou uchovávány ve dvou Diameter tabulkách. První je tabulka uzlů, obsahující adresu, status a zabezpečení nalezeného hostitele. Ve druhé tabulce, směrovací tabulce uzlů, jsou informace jako například „Realm Name“ a „Application Name“, sloužící ke zvolení směrovacího kritéria. Další položkou v této tabulce jsou možné akce (PROXY, RELAY, REDIRECT, LOCAL). Poslední sloupec obsahuje odkaz na vstup do tabulky uzlů.(Liu a spol. 2006)

# 5 TACACS

V této kapitole jsou uvedeny základní informace o protokolu TACACS a jeho vývoj v TACACS+. Následující podkapitoly jsou věnovány aktuálnímu a nejvíce používanému protokolu TACACS+. V podkapitolách lze nalézt formát paketů, komunikaci a operace TACACS+ – autentizaci, autorizaci a účtování.

## 5.1 Popis protokolu TACACS

TACACS (Terminal Access Controller Access Controller System) je protokol pro řízení terminálového přístupu, který nabízí bezpečnostní služby pro uživatele připojované po telefonu. Server si od klienta vyžádá informace o oprávnění uživatele, který toto oprávnění může poslat dále autentizačnímu serveru, který jej prověří. Protokol XTACACS (eXtended TACACS) je rozšířením protokolu TACACS. Oba protokoly využívají pro transport paketů protokol UDP. Protokol XTACACS umožňuje práci více autentizačních serverů a sledování času, po který je uživatel připojen. (Naik 1999, s. 74)

TACACS+ je výsledkem vývoje TACACS a XTACACS. TACACS+ běží na protokolu TCP (spolehlivé doručení) a musí být povolen prostřednictvím příkazů AAA. Server spustí TACACS+ démona, kterého používá ke komunikaci a vytváření paketů pro klienty AAA. TACACS+ je proprietární protokol společnosti Cisco Systems, který používá sdílený tajný klíč. Každá část AAA je prováděna odděleně, kdy jednotlivé části mohou být vázány na vlastní databázový server AAA.(Carroll 2004) Tento protokol šifruje obsah celého paketu, kromě hlavičky, která zůstává ve formě čistého textu. (Wilkins 2015)

### Komunikace

Komunikace mezi serverem a klientem probíhá na protokolu TCP, který zajišťuje spolehlivé doručení paketů. TCP protokol běží na portu 49, přes nějž je navázáno spojení. Mimo jiné poskytuje také potvrzení požadavků, adaptuje se při zahlcení a může měnit šířku pásma. Dokáže také okamžitě zaregistrovat nedostupnost serveru a o nastalé situaci informovat klienta.(Carroll 2004)

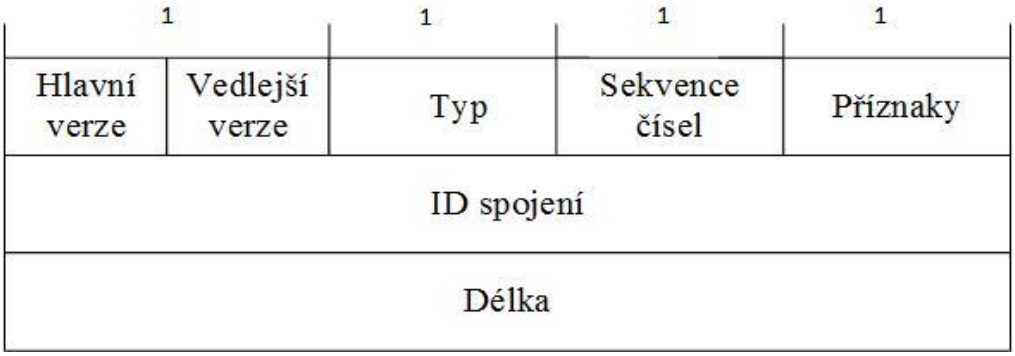
### Formát paketu

Formát paketu protokolu TACACS+ (viz obrázek č. 4) začíná polem s názvem hlavní verze, které obsahuje hlavní číslo verze protokolu TACACS+. Vedlejší verze potom uvádí revizní

číslo TACACS+, jež slouží ke zpětné kompatibilitě. Dalším polem je typ, který rozlišuje tři typy paketů:

- TAC_PLUS_AUTHEN=0x01 → Slouží k autentizaci.
- TAC_PLUS_AUTHOR=0x02 → Slouží k autorizaci.
- TAC_PLUS_ACCT=0x03 → Využíván pro účtování.

Následuje pole sekvence čísel, definující sekvenční číslo aktuálního spojení. Těchto spojení může být více. Počáteční paket má sekvenční číslo 1. Další čísla jsou dále inkrementována až do hodnoty $2^8$-1, kdy je spojení ukončeno a navázáno nové. Pole příznaky obsahuje příznaky TAC_PLUS_UNENCRYPTED_FLAG a TAC_PLUS_SINGLE_CONNECT_FLAG. TAC_PLUS_UNENCRYPTED_FLAG značí, zda je tělo paketu zašifrováno. TAC_PLUS_SINGLE_CONNECT_FLAG říká, zda lze uskutečnit více spojení skrze jedno TCP spojení.ID spojení obsahuje náhodné číslo, udávající aktuální spojení mezi klientem a severem. Toto číslo se v průběhu spojení nemění. Jako poslední pole je délka, která označuje celkovou délku paketu.(Carroll 2004)




**Obrázek 4 – Formát paketu TACACS+**

**Zdroj: Přepracováno podle Carroll**

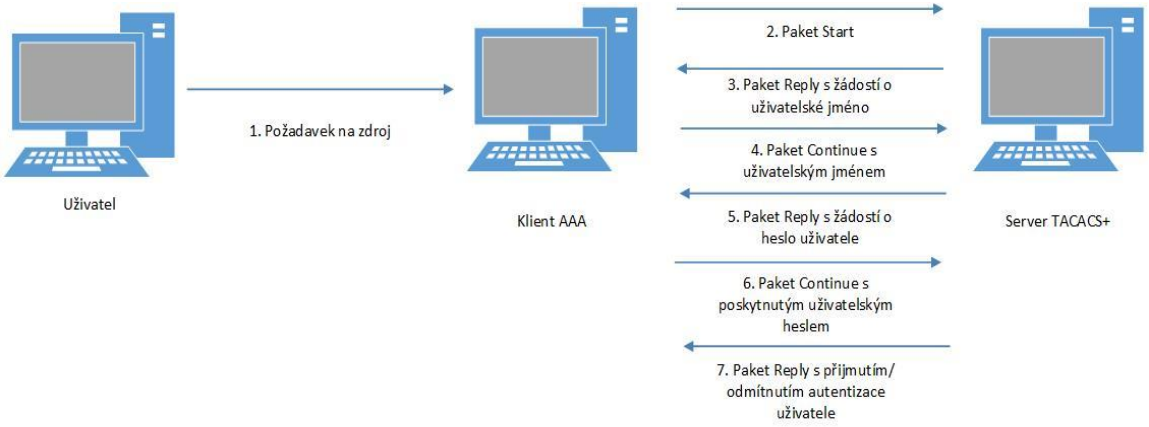
## 5.2 Operace TACACS+

Na rozdíl od protokolu RADIUS umí TACACS+ rozdělit autentizaci, autorizaci a účtování jako jednotlivé funkce. Při autentizaci je identifikován uživatel, při autorizaci jsou mu přidělena práva a při účtování jsou sbírány různé informace.

### Autentizace

Při autentizaci dochází k identifikaci uživatele. V této fázi jsou používány pakety START (při navazování spojení), REPLY (při autentizaci od serveru) a CONTINUE (pro odeslání jména a hesla klienta). Průběh autentizace (viz obrázek č. 5) je následující:

1. Klient AAA obdrží od uživatele žádost o spojení.
2. Serveru AAA, na kterém běží démon, je odeslán paket START, obsahující informace o typu autentizace.
3. Server TACACS+ odešle odpověď klientovi AAA pomocí paketu REPLY. Server požaduje uživatelské jméno.
4. Klient AAA odešle paket CONTINUE serveru TACACS+ spolu s poskytnutým uživatelským jménem.
5. Server TACACS+ pošle zpět paket REPLY s žádostí o heslo uživatele.
6. Klient AAA odešle paket CONTINUE serveru TACACS+ s heslem uživatele.
7. Server TACACS+nakonec odešle klientovi paket REPLY s odpovědí, zda autentizace prošla, neprošla, skončila chybou nebo klient žádá další informace. (Carroll 2004)




**Obrázek 5 – Průběh autentizace TACACS+**

**Zdroj: Přepracováno podle Carroll**

## Autorizace

Autorizace přiděluje uživateli určitá práva. Při autorizaci jsou využívány dva typy zpráv, a to REQUEST (od klienta AAA) a RESPONSE (od serveru AAA). Zpráva RESPONSE v sobě nese odpovědi: FAIL (služby nebyly poskytnuty), PASS_ADD (autorizace byla úspěšná), PASS_REPL (server se rozhodl ignorovat požadavek), FOLLOW (Server AAA žádá klienta, aby odeslal autorizaci jinému serveru.), ERROR (chyba na straně serveru AAA). Průběh autorizace (viz obrázek č. 6) vypadá následovně:

1. V případě, že autentizace proběhla v pořádku a uživatel byl identifikován, obdrží klient AAA od uživatele požadavek zdroje (resource request).
2. Serveru AAA je odeslán paket REQUEST pro službu „shell“.
3. Klientovi AAA je odeslán paket RESPONSE obsahující odpověď, zda autorizace proběhla úspěšně.

4. Klient AAA může poté přidělit nebo zakázat přístup ke službám „shellu“. (Carroll 2004)


**Obrázek 6 – Průběh autorizace TACACS+**

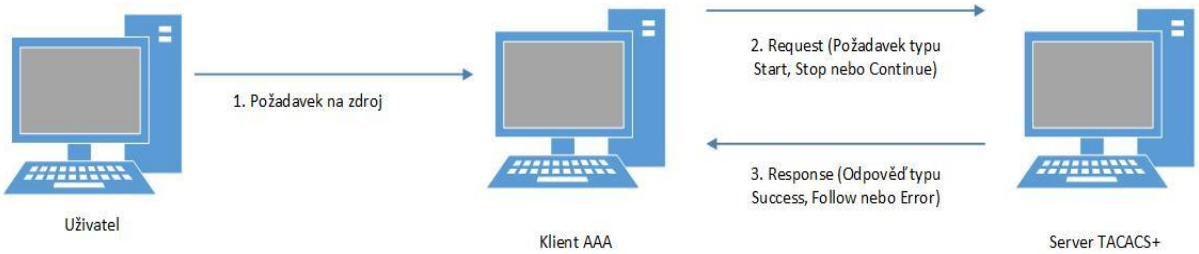
**Zdroj: Přepracováno podle Carroll**

## Účtování

Účtování používá dva typy zpráv – REQUEST a RESPONSE. Ve zprávě typu REQUEST jsou serveru posílány záznamy, kde každá z nich obsahuje AV pár. Existují tři typy záznamů: Start, která označuje začátek služby a obsahuje informace o účtu i předchozí autorizaci. Stop, která znamená přerušení nebo pozastavení služby a obsahující informace k účtu a autorizaci.

Continue, jež je označována jako Hlídací pes (Watchdog). Tento typ zprávy je posílán, když služba běží. Povoluje klientovi posílat informace serveru AAA. Obsahuje další informace, stejné jako předchozí dva typy zpráv. Zpráva typu RESPONSE v sobě může nést tři stavy: SUCCESS (server obdržel záznam od klienta AAA), ERROR (selhalo přidání záznamu do databáze), FOLLOW (žádá klienta AAA, aby poslal záznam jinému serveru AAA). Samotné účtování (viz obrázek č. 7) probíhá takto:

1. Uživatel požádá klienta AAA o poskytnutí zdroje.
2. Klient AAA odešle zprávu typu REQUEST serveru AAA.
3. Server AAA potom odpoví pomocí zprávy RESPONSE. (Carroll 2004)




**Obrázek 7 – Průběh účtování TACACS+**

**Zdroj: Přepracováno podle Carroll**

# 6 PPP

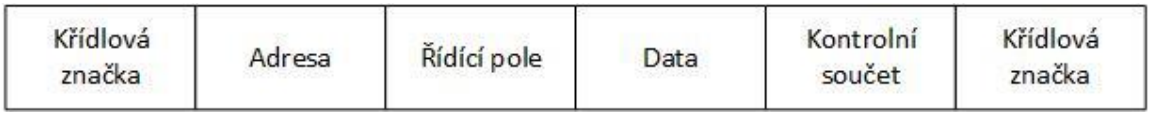
V této kapitole je vysvětlený pojem PPP, uvedený přenos paketů a tvar rámce protokolu PPP. Také se zabývá autentizací, respektive protokoly PAP, CHAP a MS CHAP.

## 6.1 Popis protokolu PPP

Protokol PPP (Point-to-Point Protocol) se využívá pro zapouzdření síťových protokolů na dvoubodových sériových linkách. Popis protokolu PPP se nachází v RFC 1171, jeho rozšíření na PAP a CHAP v RFC 1334. (Naik 1999, s. 70) Protokol PPP nevyužívá řídící signály, může používat jak synchronní, tak i asynchronní přenos dat. Cílem protokolu je umožnit po jedné lince přenos více síťových protokolů současně. (Dostálek a Kabelová 2002, s. 78) Protokol PPP byl navrhnut pro přenos paketů na jednoduchých linkách. Tyto linky poskytují plně duplexní, souběžný, obousměrný provoz a doručování paketů v určitém pořadí. (Mogollon 2007, s. 167) Protokol tvoří celkem tři části:

1. HDLC (High Level Data Link Control)

Protokol HDLC vychází z protokolu SDLC, který sloužil pro synchronní přenos. HDLC byl doplněn o asynchronní přenos, jež využívá převážně protokol PPP. Protokol HDLC slouží k detekci chyb a řízení toku dat. Protokol PPP využívá tvar jeho rámce, viz obr. č. 8.




**Obrázek 8 – Tvar rámce HDLC**

**Zdroj: Přepracováno podle Dostálka a Kabelové**

Tvar rámce HDLC obsahuje v první části křídlovou značku, která uvozuje začátek (i konec) paketu. Obsahuje osm bitů. V případě dvou po sobě jdoucích křídlových značek se paket nezpracovává, protože je prázdný. Za křídlovou značkou následuje linková adresa stanice, které je paket určen. Dalším polem je kontrolní součet, který se spočítá z dat, adresního a řídícího pole. V poli dat jsou přenášena data. V poslední části se nachází řídící pole, jež určuje typ rámce, a tedy i jeho účel. (Dostálek a Kabelová 2002, s. 78–79)

2. LCP (Link Control Protocol)

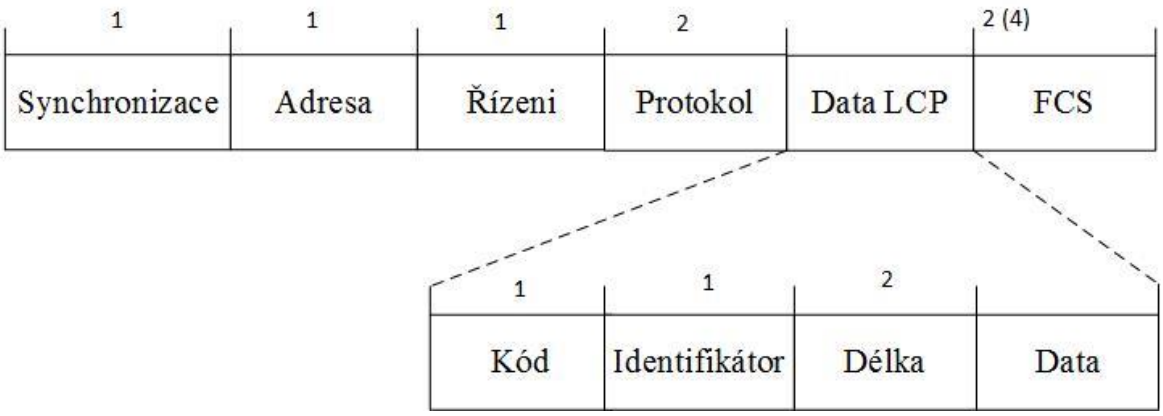
Další částí PPP je protokol LCP, který je využíván především pro navázání spojení, jeho ukončení a výběr algoritmu pro autentizaci. Linka postupně prochází fázemi: odpojena, navazování spojení, autentizace, síťový protokol a ukončování spojení. Ve fázi odpojení linka začíná i končí. Z odpojení přechází do fáze navazování spojení pomocí výměny konfiguračních paketů. Po navázání spojení linka přechází do stavu autentizace, kdy je prokázána totožnost klienta. Ve fázi nazývané „síťový protokol" jsou vysílány pakety NCP (Network Control Protocol) od jednotlivých protokolů. Protokol, který nevyšle svůj paket NCP, nemůže linku vyžívat. V poslední fázi, ukončování spojení, jsou všechny pakety zahozeny (kromě paketů LCP) a linka je ukončena. (Dostálek a Kabelová 2002, s. 79–83)

3. NCP (Network Control Protocol)

Poslední část protokolu PPP tvoří tzv. protokoly řízení sítě. Tyto protokoly jsou využívány pro konfiguraci vyšších síťových protokolů. (Pužmanová 2004, s. 61) Protokoly, které využívají PPP, mají vlastní normy definované v různých RFC (pro každý protokol zvlášť). V paketu PPP se poté použije číslo protokolu definované v normě. (Dostálek a Kabelová 2002, s. 79)

### Tvar rámce PPP

Tvar rámce PPP se skládá z několika částí, viz obrázek č. 9. První je synchronizace, jež označuje začátek rámce. Další částí je adresa; nejčastěji se jedná o konstantní adresu 11111111 (= oběžník). Následuje pole řízení, které obsahuje hodnotu 00000011, značící přenos dat v nečíslovaném rámci. Pole protokol obsahuje číslo protokolu vyšší vrstvy. Část data obsahuje data o maximální délce pole 1500 oktetů (záleží na zvoleném protokolu). V části LCP lze nalézt pole kód (code), který označuje typ paketu. Poté následuje identifikátor, což je náhodné číslo, sloužící k navázání požadavků a jejich odpovědí. Délka (length) značí délku paketu, a poslední pole – data – v sobě nese uživatelská data a případné další možnosti konfigurace. Poslední částí rámce PPP je zabezpečení rámce FCS (Frame Check Sequence), které v sobě nese 16–32 bitů kódu pro lepší zabezpečení. (Pužmanová 2004, s. 62)




**Obrázek 9 – Tvar rámce PPP**

**Zdroj: Přepracováno podle Pužmanové**

### Přenos paketů

PPP nejprve naváže spojení mezi klientem a síťovým přístupovým serverem. Aby byla možná komunikace, je třeba, aby oba konce PPP linky nejdříve poslali pakety LCP (Link Control Protocol). Paket LCP slouží ke konfiguraci a otestování datového spojení. Poté následuje možnost autentizace. Po autentizaci musí PPP poslat pakety NCP pro výběr a nastavení jednoho či více síťových protokolů. Po konfiguraci síťových protokolů může začít komunikace. Spojení je nastavené dokud nevyprší pakety LCP/NCP nebo dokud nenastane nějaká vnější událost. (Mogollon 2007, s. 127)

### Autentizace

Protokol PPP umožňuje různé metody autentizace, které budou popsány v této části práce.

Protokol PAP

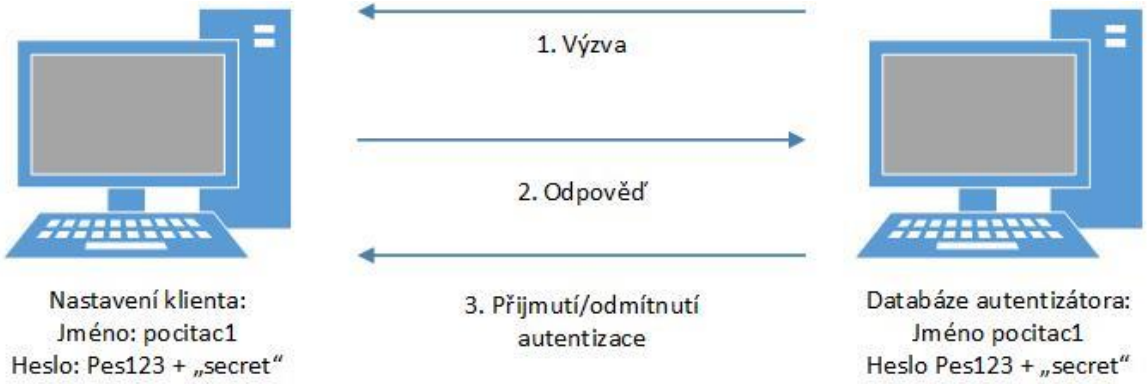
Autentizační protokol PAP (Password Authentication Protocol) poskytuje jednoduchou metodu pro stanovení identity klientů za použití dvoucestného navázání spojení. (Lloyd a Simpson 1992)Hostitel (klient) odešle serveru (autentizátor) žádost o ověření totožnosti, ke kterému přiloží své uživatelské ID a heslo. Autentizátor si ověří jeho totožnost a poté pošle zpět odpověď – přijme/nepřijme žádost o autentizaci. Nevýhodou protokolu PAP je zasílání uživatelského ID a hesla ve formě čistého textu.(Naik 1999, s. 70–71) Ukázka autentizace viz obrázek č. 10.


**Obrázek 10 – Autentizace pomocí PAP**

**Zdroj: Přepracováno podle Pužmanové**

Protokol CHAP

Protokol CHAP (Challenge Handshake Authentization Protocol) opakovaně ověřuje identitu uzlu (peer). (Lloyd a Simpson 1992)Nabízí lepší zabezpečení než protokol PAP. Jednotlivé fáze autentizace řídí autentizátor, který nejprve odešle klientovi výzvu. Klient poté vypočítá s pomocí algoritmu MD5 a hašovací funkce určitou hodnotu, kterou pošle zpět autentizátorovi. Autentizátor také vypočítá hodnotu, kterou následně porovná s hodnotou přijatou od klienta. (Naik 1999, s. 70–71) Ukázka autentizace pomocí CHAP viz obrázek č. 11.




**Obrázek 11 – Autentizace pomocí CHAP**

**Zdroj: Přepracováno podle Pužmanové**

MS CHAP

Microsoft Challenge Handshake Authentization Protocol je protokol podobný CHAP, s rozdílem, ževýpočet probíhá s pomocí algoritmu MD4. Dále také autentizátor nepotřebuje přístup k prostému textu ID a hesla uživatele, čímž je dosáhnuto větší bezpečnosti. (Naik 1999, s. 71)

EAP

EAP je protokol podobný CHAP, ale při navazování spojení dojde na obou stranách k dohodě, že bude použit protokol EAP. Použití konkrétního algoritmu je vyjednáno až protokolem EAP – umožňuje tedy využití libovolného autentizačního mechanismu. (Dostálek a Kabelová 2002, s. 90) Protokol EAP je podrobně popsán v samostatné kapitole.

# 7 EAP

V této kapitole je rozebrán protokol EAP, jeho autentizační mechanismy a metody.

## 7.1 Popis protokolu EAP

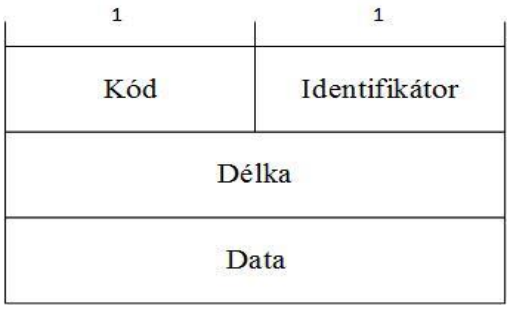
EAP (Extensible Authentication Protocol) je rozšiřitelný protokol, protože k zapouzdření zpráv je možné použít jakékoli autentizační mechanismy. Podporuje technologie, jako jsou chytré karty, Kerberos, digitální certifikáty, jednorázová hesla a další. Autentizační mechanismy je možné implementovat metodami EAP jako jsou například EAP-TLS, EAP-PEAP, EAP-TTLS aj. EAP je specifikován v RFC 3748. Technika zapouzdření EAP paketů mezi žadatelem a autentizátorem v prostředí LAN je známá jako EAP over LANs (EAPOL). (Mogollon 2007, s. 157)

### Typ zpráv

Protokol EAP používá 4 typy zpráv: požadavek EAP („EAP-Request“), odpověď EAP („EAP-Response“), úspěch EAP („EAP-Success“) a selhání EAP („EAP-Failure“). (Nakhjiri, M. a M. Nakhjiri 2005, s. 42)

### Formát EAP paketu

Prvním polem paketu EAP je 1 oktetový kód, který uvádí typ paketu (požadavek, odpověď aj.). Identifikátor slouží k párování požadavku a jeho odpovědi. Pole délka zabírá v paketu 2 oktety a udává celkovou délku paketu. V posledním poli se nachází data, která mohou zabírat 0 a více oktetů. Formát dat je definován podle typu paketu. Formát paketu EAP viz obrázek č. 12. (Aboba a spol. 2004)




**Obrázek 12 – Formát paketu EAP**

**Zdroj: Přepracováno podle Aboba**

## 7.2 EAP metody

Protokol EAP využívá k autentizaci různé metody lišící se typem zabezpečení a požadavky na jeho nasazení. (Antal 2013) EAP metody lze podle Mogollona rozdělit do těchto kategorií: Certifikované (EAP-TLS), hybridní (PEAP, EAP-TTLS), založené na SIM (EAP-SIM, EAP-AKA), se sdíleným klíčem (EAP-PSK, EAP-IKEv2, EAP-TLS-PSK) a založené na heslech (EAP-PAX, EAP-SPEKE).(2007, s. 160-163) Pužmanová doplňuje k těmto metodám tzv. tunelovací metody (EAP-FAST). (2006)

### EAP-MD5

Tato standardizovaná metoda představuje nejslabší metodu zabezpečení komunikace a její použití se nedoporučuje. Proces autentizace uživatele pomocí metody EAP-MD5 probíhá následovně:

1. Autentizátor odešle zprávu „EAP-Request“ žadateli, ve které žádá o prokázání totožnosti.
2. Žadatel v případě souhlasu s autentizací odešle odpověď „EAP-Response“.
3. Autentizátor pošle druhé straně výzvu „EAP-Request“.
4. Žadatel spojí sdílené tajemství s výzvou pomocí funkce MD5 a výsledek odešle odpovědí „EAP-Response“.
5. Autentizátor potvrdí/zamítne totožnost žadatele zprávou „EAP-Success“/„EAP-Failure“. (Dostálek a Kabelová 2002, s. 90)

### Certifikované

EAP-TLS

EAP-TLS (EAP-Transport Layer Security) využívá pro vzájemnou autentizaci digitální certifikáty X.509 s použitím digitálních podpisů. Pro autentizaci je vyžadována certifikace od obou stran. Protokol zabezpečuje autenticitu, integritu zpráv a nepopiratelnost. Je vhodný pro podniky, které již mají zavedenou infrastrukturu PKI. EAP-TLS je definován v RFC 2716. (Mogollon 2007, s. 160)

### Hybridní

PEAP

Protokol PEAP (Protected EAP) nemusí na rozdíl od EAP-TLS využívat digitální certifikáty (klient nevyužívá). Poskytuje ochranu identity uživatele a kryje proces autentizace. Pro zabezpečení kanálu TLS je třeba autentizace založená na certifikaci serveru. Hlavní rozdíl mezi EAP-TTLS a PEAP je, že EAP-TTLS nese jakýkoli autentizační materiál přes AVP skrze záznamy TLS, což umožňuje použít jakýkoliv ověřovací protokol (PAP, CHAP) i EAP metody. Naopak PEAP umožňuje pouze ověřovací metody, které jsou definovány pro použití s EAP. Stejně jako EAP-TTLS se i PEAP vyhýbá posílání nechráněných údajů o uživateli. (Nakhjiri, M. a M. Nakhjiri 2005, s. 262)

EAP-TTLS

EAP-TTLS (EAP-Tunneled TLS) je metoda autentizace využívající k autentizaci serveru digitální certifikát. Klient zde ale používá pouze heslo. Komunikace autentizačních protokolů (PAP, CHAP aj.) se odehrává uvnitř zašifrovaného TLS tunelu. V rámci této komunikace jsou generovány klíče. EAP-TTLS je definována v RFC 5281. (Pužmanová 2006)

### Založené na SIM

EAP-SIM a EAP-AKA

EAP-SIM (EAP Subscriber Identity Module) je metoda založená na symetrickém šifrování, které opakovaně využívá strukturu autentizace GSM (Global System for Mobile Communications). Jejím nedostatkem je chybějící vzájemná autentizace. Odvozený 64 bitový šifrovací klíč také není dostatečně silný pro použití v datových sítích. Pro překonání velikosti klíče jsou použity RAND výzvy generující různé 64 bitové „Kc klíče“, které jsou spojovány za účelem získání delšího klíče. EAP-SIM mohla být implementována z důvodu zpětné kompatibility s miliony již nasazenými GSM/GPRS SIM kartami. EAP-AKA však poskytuje vzájemnou autentizaci, ochranu přehrávání a odvození delších klíčů, a proto je více používána.Popismetody EAP-SIM lze nalézt v dokumentu RFC 4186.(Mogollon 2007, s. 161)

### Založené na heslech

EAP-PSK

EAP-PSK (EAP Pre-Shared Key) je metoda jednoduchá na implementaci i na nasazení do již existující sítě. Pro navázání spojení TLS, na němž je založená, využívá předem sdílené symetrické klíče (PSKs). Šifrování je založeno na AES-128. Podrobnější popis metody se nachází vRFC 4764. (Mogollon 2007, s. 163)

EAP-SPEKE

EAP-SPEKE (Simple Password Encrypted Key Exchange) je další EAP metoda založená na symetrickém šifrování a asymetrickém klíči, poskytující výměnu klíčů pouze mezi stranami ověřenými heslem. EAP-SPEKE je použitelná pouze v případě poskytnutí hesla uživatelem. Je zbytečně složitá pro autentizaci zařízení. Vylepšený EAP-SPEKE podporuje vzájemnou autentizaci, výměnu klíčů a funguje na základě Elliptic Curve Cryptosystems, stejně jako na Diffie-Hellman základě. (Mogollon 2007, s. 163)

**Tunelovací metody**

EAP-FAST

Metoda EAP-FAST (EAP-Flexible Authentication via Secure Tunneling) je založena na bezpečném tunelu mezi klientem a serverem za použití klíče PAC (Protected Access Credential). EAP-FAST je definována v RFC 4851. V tomto dokumentu je uvedeno, že metoda EAP-FAST je odolná vůči slovníkovým a „man-in-the-middle“ útokům.(Pužmanová 2006)

LEAP

LEAP (Lightweight Extensible Authentication Protocol) je proprietární protokol společnosti Cisco Systems. Neřadí se mezi tunelovací metody. LEAP využívá pro zabezpečení komunikace mezi klientem a serverem mechanismus WEP (Wired Equivalent Privacy). (Nakhjiri, M. a M. Nakhjiri 2005, s. 260) Použití tohoto protokolu však není doporučováno ani samotnou společností z důvodu vysoké zranitelnosti (hrozí „offline“ útok). (Antal 2013)

# 8 LDAP

Tato kapitola je rozdělena do dvou částí. V první části je popsán protokol LDAP a jeho komunikace. Ve druhé části jsou uvedeny a rozebrány jeho modely – informační, jmenný, funkční a bezpečnostní.

## 8.1 Popis a principy protokolu LDAP

Protokol LDAP (Lightweight Directory Access Protocol) je otevřený protokol sloužící pro přístup k adresářovým službám.Jeho specifikace se nachází v dokumentu RFC 4511 (verze 3 v RFC 2251), úplný seznam RFC dokumentů týkajících se tohoto protokolu lze nalézt na adrese https://www.ldap.com/ldap-specifications-defined-in-rfcs. LDAP vychází z protokolu DAP (Directory Access Protocol). Protokol LDAP měl zjednodušit určité funkce uvedené ve specifikaci X.500. (Naik 1999, s. 127) V současné době jsou servery založeny na LDAP verzi 3. (Isode 2016)

### Komunikace

Adresář (directory) je utříděný seznam informací o objektech, který využívá ke komunikaci model klient/server. (Tuttle a spol. 2004, s. 8) Klient naváže spojení se serverem a odesílá mu zakódované požadavky. Po přijetí požadavku serverem je provedena akce. Nakonec je vrácen výsledek. Tato komunikace může být synchronní nebo asynchronní. Synchronní komunikace probíhá v reálném čase, kdy je požadavek vyřízen ihned. V případě asynchronní komunikace existuje určitá časová prodleva mezi požadavkem a jeho vyřízením. (Dostálek a Kabelová 2002, s. 520)

Komunikace mezi klientem LDAP a serverem LDAP probíhá následovně:

1. Klient naváže spojení se serverem LDAP, během kterého klient specifikuje jméno uzlu (hostname) nebo adresu IP a číslo portu TCP/IP, na kterém LDAP naslouchá.
2. Autentizace klienta může proběhnout pomocí uživatelského jména a hesla nebo navázání anonymního spojení s výchozími přístupovými právy. Lze navázat i bezpečnější spojení pomocí šifrování.
3. Klient provede operace v adresáři, nejčastěji je to čtení, vyhledávání a úprava dat.
4. Po dokončení všech klientových operací je spojení se serverem ukončeno. (Tuttle a spol. 2004, s. 28)

## 8.2 Modely LDAP

Modely LDAP popisují strukturu záznamů (entry) v adresáři, jejich organizaci a identifikaci. Dále uvádějí možné operace, které lze nad daty provádět a bezpečnostní mechanismy, které chrání informace před neoprávněným přístupem. Protokol LDAP se skládá celkem ze čtyř modelů: informační model, jmenný model, funkční model a bezpečnostní model.

### Informační model

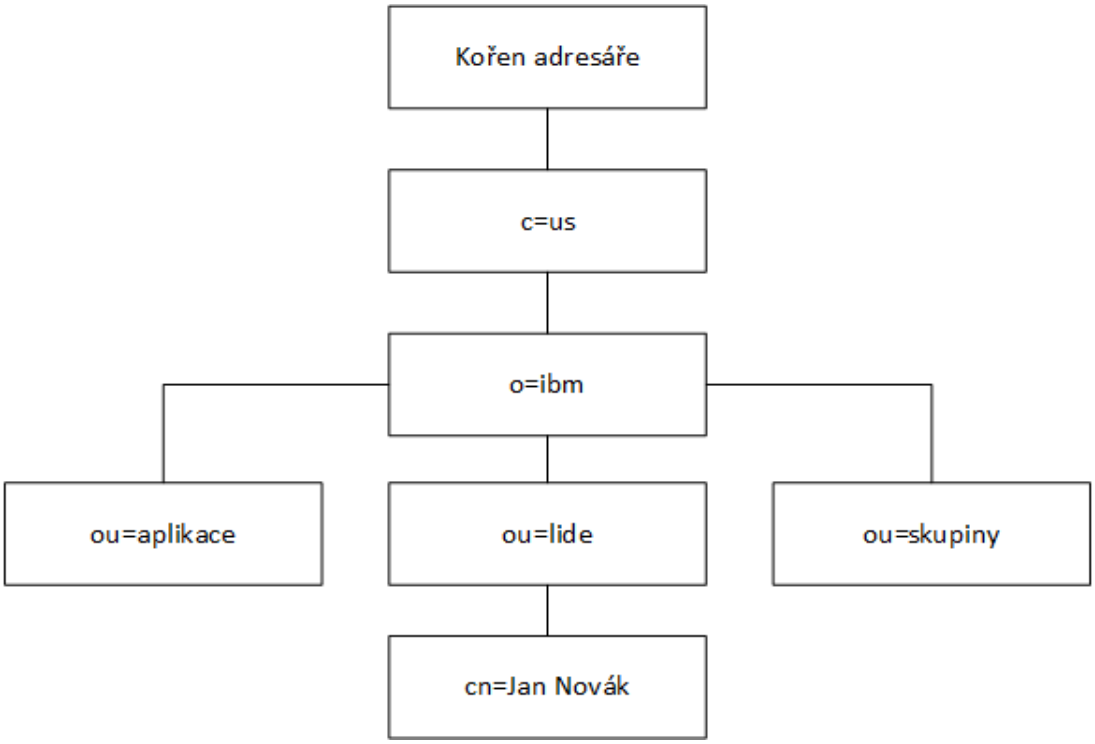
Informační model popisuje strukturu záznamů uložených v adresáři. Každý záznam obsahuje několik atributů. Jednotlivé atributy v sobě nesou typ a jednu nebo více hodnot. Typ je syntakticky daný, popisující jaké typy hodnot budou v atributu uloženy. Dále existuje tzv. objektová třída (objectClass), která určuje typ záznamu. Schémadefinuje, jaká objektová třída a atributy lze u záznamu použít, tedy který typ objektů bude uložen do adresáře. V protokolu LDAP existují podtřídy (subClass) umožňující odvození jednoho objektu od druhého, založeného na principu dědičnosti. Objekt, který nemá předchůdce, je nazývaný kořenem (top). (Tuttle a spol. 2004, s. 5-42)

LDIF (Leigthweigth Directory Interchange Format) je specifikace datového formátu. Pomocí LDIF je možný přenos informací mezi servery, obsahující popis a změny záznamů. (Dostálek a Kabelová 2002,s. 532–533) Umožňuje jednoduchou manipulaci s velkým množstvím dat. (Tuttle a spol. 2004, s. 35) Soubor LDIF se skládá z jednotlivých záznamů. Tyto záznamy mají položky, které specifikují hodnoty atributů. Každá položka je na samostatném řádku. (Dostálek a Kabelová 2002, s. 532–533)

### Jmenný model

Popisuje, jak jsou záznamy v adresáři organizovány a identifikovány. Adresář obsahuje záznamy organizované do stromové struktury, která se nazývá DIT (Directory Information Tree). Ukázka DIT viz obrázek 13. Každý záznam má své jednoznačné jméno (Distinguished Name – DN), podle kterého je zařazen do DIT. Jednoznačná jména se skládají ze sekvence relativních jednoznačných jmen (Relative Distinguished Name – RDN) oddělených čárkou. RDN má tvar < název atributu > = < hodnota>. (Tuttle a spol. 2004, s. 42–47)

V případě, že je DIT rozprostřený na více serverů, obsahuje záznam odkaz (referral) s příslušnou LDAP URL na daný podstrom. Sada záznamů v rámci jednoho podstromu spravovaná jedním nebo více servery je nazývána jmenný kontext. Kořen DIT je tvořen tzv. DSE záznamem (DSA Specific Entry) obsahující informace o serveru LDAP a spravovaných kontextech. V rámci serveru existuje pouze jeden DSE záznam. Server také spravuje tři jmenné kontexty. (Dostálek a Kabelová 2002, s. 520)




**Obrázek 13 – Příklad DIT**

**Zdroj: Přepracováno podle Tuttle a spol.**

### Funkční model

Funkční model popisuje možné operace nad záznamy v adresáři. „LDAP zahrnuje tyto typy operací (každý má definován formát požadavku a formát odpovědi):

- bind – požadavek na provázání relace klienta a s konkrétní identitou (tj. autentizace uživatele),
- unbind – zrušení vazby mezi relací a identitou uživatele,
- search – operace vyhledávání v DIT, v rámci odpovědi klient obdrží jeden z následujících typů – entry (vyhovující položka), reference na jiný server, konec zpracování dotazu,
- modify – změna záznamu,
- add – přidání záznamu,
- del – zrušení záznamu,
- modifyDN – změna DN (jména) záznamu, čímž je možné záznam v rámci DIT „přestěhovat“,

- compare – otestování, zda záznam s konkrétním DN má dotazovaný atribut nějakou hodnotu,
- abandon – zrušení dosud probíhající operace,
- extended – umožňuje přenášet mezi serverem a klientem libovolné zprávy zakódované do řetězce, jde tedy o způsob pro proprietární rozšíření standardní cestou.“(Dostálek a Kabelová 2002, s. 255)

Vyhledávání je nejčastější operací, řídící se podle různých parametrů: báze, rozsah, práce s odkazy, limit času, limit velikosti, jen typy, filtr, seznam atributů aj. (Dostálek a Kabelová 2002, s. 255)

### Bezpečnostní model

Tuttle a spol. ve své knize (2004, s. 53–55) popisují, jak může být informace v adresáři chráněna před neoprávněným přístupem. Při navázání spojení se serverem může klient zadat uživatelské jméno a heslo (nezašifrované) nebo nezadává nic a je navázáno anonymní spojení. Použití prostého textu u hesel se však z důvodu nedostatečného zabezpečení nedoporučuje. LDAP nabízí tři typy zabezpečení:

1. Základní

Klient se autentizuje serveru pomocí DN a hesla. Server následně zjistí, zda se odeslané údaje shodují s údaji v adresáři. Může být použito šifrování Base64, ale to je lehce zlomitelné.

2. SASL (Simple Authentication and Security Layer)

SASL je rozhraní (rámec) pro přidávání autentizačních mechanismů do spojitě-orientovaných protokolů. Každý protokol je reprezentován profilem, který umožňuje spolupráci daného protokolu se SASL. Protokol, který chce využít SASL, musí být rozšířen o příkaz, který zjistí typ autentizace. Pro zašifrování dat může být využita zabezpečená vrstva.

3. SSL (Secure Socket Layer) a TLS (Transport Layer Security)

Protokol SSL byl navrhnut pro autentizaci a ochranu dat. Zapouzdřením „socketu“ TCP/IP je umožněna zabezpečená komunikace pro každou aplikaci, která jej využije. SSL/TSL podporuje serverovou autentizaci, klientskou autentizaci nebo vzájemnou autentizaci. Kromě toho poskytuje ochranu dat posílaných přes síť jejich zašifrováním. To je dosáhnuto pomocí veřejného/privátního klíče. Data zašifrovaná pomocí privátního klíče mohou být dešifrována pomocí klíče veřejného a naopak. Předpokladem je, že server má již tento pár klíčů předem vygenerovaný; obvykle se tak děje při nastavování LDAP serveru.

Interakce mezi klientem a serverem při SSL/TLS autentizaci probíhá následovně:

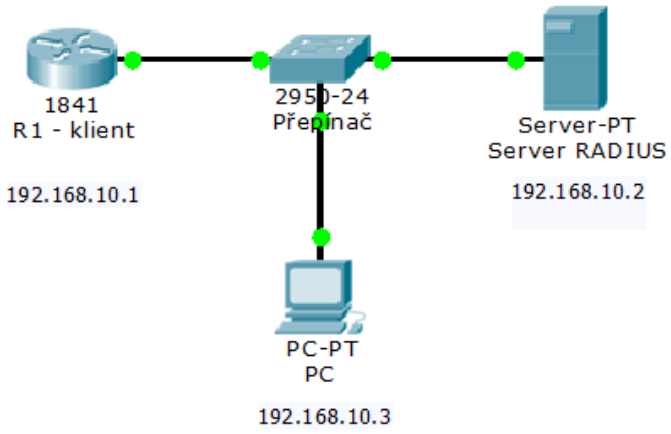
- Klient požádá server o navázání SSL/TLS spojení.
- Server pošle možná nastavení SSL/TLS a certifikát obsahující veřejný klíč, identitu vydavatele, název certifikátu aj.
- Klient poté žádá server o prokázání jeho identity.
- Server pošle zpět zprávu obsahující její přehled (message digest), která je zašifrována privátním klíčem. Přehled je vypočítán z obsahu zprávy použitím hašovací funkce. Klient zprávu dešifruje veřejným klíčem a poté ji porovná se součtem, který si spočítal. Pokud jsou totožné, identita serveru je prokázána a autentizace je ukončena.
- Server se s klientem dohodne na tajném klíči pro zašifrování dat. Klient vygeneruje symetrický klíč, zašifruje ho pomocí veřejného klíče a odešle serveru.
- Server dešifruje tajný klíč a pošle zpět testovací zprávu zašifrovanou tajným klíčem, aby se potvrdilo, že klíč dorazil v pořádku. Poté již může začít komunikace s použitím tajného klíče pro zašifrování dat.(Tuttle a spol. 2004, s. 53–55)

# 9 OVĚŘENÍ FUNKČNOSTI

Tato část práce se věnuje praktickému ověření funkčnosti tří vybraných protokolů. K otestování jsem si zvolila protokoly RADIUS, TACACS+ a LDAP. Protokol RADIUS spolu s TACACS+ patří mezi nejrozšířenější AAA protokoly. LDAP jsem si zvolila z důvodu častého využívání na Unixových systémech. Protokol PPP je také velice používaný. Jeho konfiguraci lze však nalézt v učebních materiálech, a proto se s ním dále nebudu zabývat.

## 9.1 Ověření funkčnosti protokolu RADIUS

Testování proběhlo na síťovém modelu obsahující jeden přepínač (switch), jeden směrovač, který sloužil jako klient, a dva počítače v rolích serveru a uživatele (viz obrázek 14).



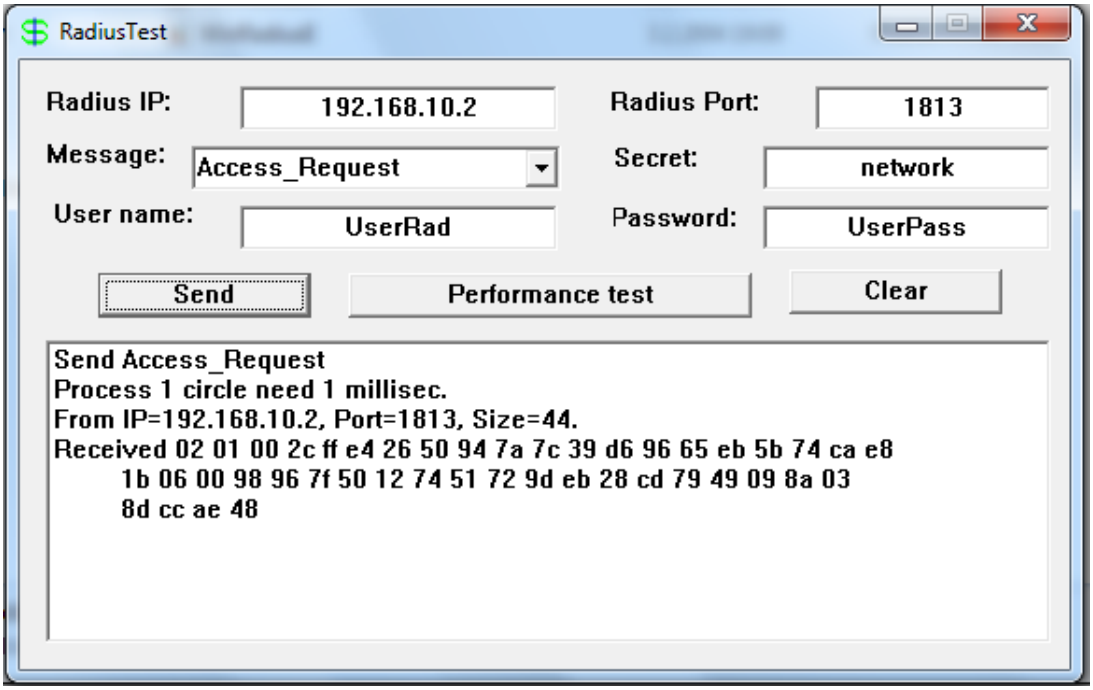

**Obrázek 14 – Síťový model pro otestování protokolu RADIUS**

### Nastavení serveru

Pro provoz RADIUS serveru jsem si zvolila program WinRadius. Po instalaci programu jsem spustila hlavní program „WinRadius.exe“. V této části se nastavuje databáze, klienti aj. V prvním kroku jsem nastavila parametry systému jako je IP adresa serveru, sdílené tajemství a porty pro autentizaci a účtování. Ve druhém kroku jsem nastavila databázi – její jméno, název tabulkyuživatelů, název tabulky pro účtování a název tabulky pro VoIP. V dalším kroku jsem přidala nového uživatele s údaji o jeho uživatelském jméně a hesle. Po provedených změnách je třeba WinRadius vypnout a znovu zapnout.

K ověření správné konfigurace serveru slouží podprogram WinRadiusu zvaný RadiusTest. Po spuštění jsem vyplnila požadované údaje jako je IP adresa serveru, port, na kterém RADIUS naslouchá, sdílené tajemství a údaje o uživateli (nově vytvořeném). Při stisknutí tlačítka

odeslat (send) jsou údaje odeslány serveru a v případě úspěchu se objeví přijaté hexadecimální znaky. Mimo jiné je v tabulce s logy napsáno „User (UserRad) authenticate OK.“, značící úspěšnou autentizaci uživatele.




**Obrázek 15 – Nástroj RadiusTest**

### Nastavení klienta

V této části práce jsem se řídila konfiguračními příkazy od společnosti Cisco Systems. (c2001-2006)Nastavení směrovače, v tomto případě klienta, vyžaduje nastavení lokálního uživatelského jména a hesla, které je použito v případě, že autentizace pomocí RADIUS serveru není dostupná. Také je třeba nastavit adresu IP rozhraní a zapnout ho.

```
R1 (config) # username uzivatel secret heslo
R1 (config) # interface f0/0
R1 (config-if) # ip address 192.168.10.1 255.255.255.0
R1 (config-if) # no shutdown
R1 (config-if) # exit
```

Po základní konfiguraci následuje nastavení autentizace pomocí serveru RADIUS. Příkazem aaa new-model jsou povoleny služby AAA.

```
R1 (config) # aaa new-model
```

V dalším kroku je nastavena autentizace pomocí „Prihlaseni“, uveden použitý protokol pro autentizaci – radius. Příkaz local enable umožňuje použití uživatelského jména a hesla, které jsou uloženy v lokální databázi směrovače, jak již bylo uvedeno výše.

```
R1 (config) # aaa authentication login Prihlaseni group radius local enable
```

Pro správnou funkčnost autentizace je nutné vyplnit stejné tajemství (klíč) na klientovi i serveru. Příkazem radius-server host se nastavuje IP adresa serveru.

```
R1 (config) # enable secret network
R1 (config) # radius-server host 192.168.10.1 auth-port 1813 key network
```

Aby bylo po uživateli vyžadováno přihlášení skrze server, je třeba zadat následující příkazy:

```
R1 (config) # line vty 0 15
R1 (config-line) # login authentication Prihlaseni
R1 (config-line) # exit
```

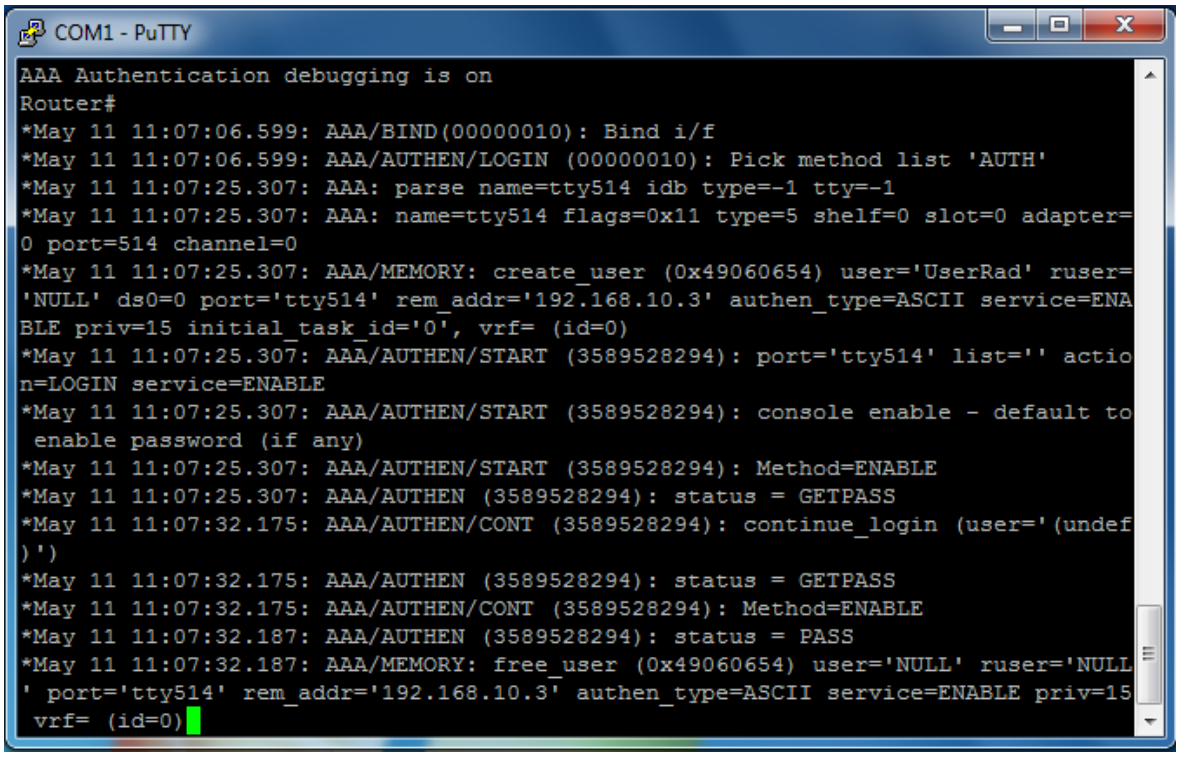
Následující příkazy slouží k vypsání „Jmeno ->“ a „Heslo ->“ v momentě, kdy se uživatel chce přihlásit ke konzoli směrovače. Tyto příkazy nejsou povinné.

```
R1 (config) # aaa authentication username-prompt Jmeno ->
R1 (config) # aaa authentication password-prompt Heslo ->
```

Aby byla vidět probíhající výměna zpráv při autentizaci uživatele, zadala jsem užitečný příkaz:

```
R1 (config) # debug radius authentication
```

Při přihlašování uživatele na klienta probíhá zachytávání autentizačních údajů, viz obrázek č. 16.



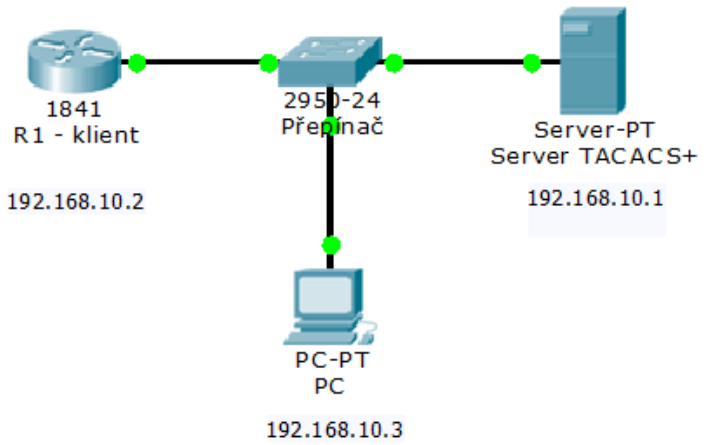

**Obrázek 16 – Příkaz „debug radius authentication“**

### Zhodnocení konfigurace protokolu RADIUS

Nástroj WinRadius byl velice přehledný, nastavení serveru proběhlo rychle a snadno. Konfigurace klienta byla ze začátku matoucí, ale později jsem jej zprovoznila. Uživatel se mohl přihlásit pomocí účtu vytvořeném na serveru, i z lokální databáze. Protokol RADIUS mi přišel nejsnadnější na konfiguraci.

## 9.2 Ověření funkčnosti protokolu TACACS+

V této kapitole bude ověřena funkčnost protokolu TACACS+ na jednoduché síti (viz obrázek č. 17). Nejdříve bude popsán průběh konfigurace serveru TACACS+ a následně konfigurace klienta. Nakonec bude ukázka autentizace uživatele při jeho přihlašování na klienta.




**Obrázek 17 – Síťový model pro otestování protokolu TACACS+**

### Nastavení serveru TACACS+

Jako nástroj pro správu serveru jsem si zvolila program TACACS.net dostupný z http://www.tacacs.net/download.asp. Během instalace tohoto programu bylo třeba zadat tzv. sdílené tajemství (klíč), které musí být stejně nastavené na serveru i na klientovi. V mém případě jsem použila slovo „network“.

Tento nástroj obsahuje hlavní konfigurační soubory „authentication.xml“, „authorization.xml“, „clients.xml“, „tacplus.xml“. K ověření konfigurace lze využít TACTest a TACVerify, které vypíší případné syntaktické i sémantické chyby v konfiguračních souborech.

V souboru „tacplus.xml“ jsem upravila lokální adresu serveru, která byla původně nastavená na „localhost“, tedy 127.0.0.1. Port by měl mít číslo 49. V souboru „authentication.xml“ jsem přidala nového uživatele s heslem a stejně tak i souboru „authorization.xml“. Uživatelé se poté přidávají do skupin (v obou souborech). Skupina je zadávána v klientovi při konfiguraci. Do souboru „clients.xml“ patří adresy klientů. (*TACACS.net,*c2010, s. 3–14)

Konfigurační soubory protokolu TACACS+ lze nalézt v elektronické příloze.

### Nastavení klienta TACACS+

Při nastavování klienta jsem jako zdroj využívala příkazy od společnosti Cisco Systems, Inc. (c2001-2006)Nastavení klienta TACACS+ probíhá podobně jako výše popsané nastavení klienta RADIUS. Nastavuje se lokální uživatelské jméno, heslo a adresa IP. Také zůstává stejný příkaz pro povolení služeb AAA.

```
R1 (config) # username R secret heslo
R1 (config) # interface f0/0
R1 (config-if) # ip address 192.168.10.2 255.255.255.0
R1 (config-if) # no shutdown
R1 (config-if) # exit
R1 (config) # aaa new-model
```

V konfiguraci protokolu RADIUS byla k autentizaci použita nově vytvořená skupina. V případě této konfigurace jsem pro test využila možnost použít výchozí skupinu. Místo group radius je nyní zadáno group tacacs+.

```
R1 (config) # aaa authentication login default group tacacs+ local
```

Příkazem authorization se upravuje nastavení pro autorizaci uživatele.

```
R1 (config) # aaa authorization exec default group tacacs+ if-authenticated
```

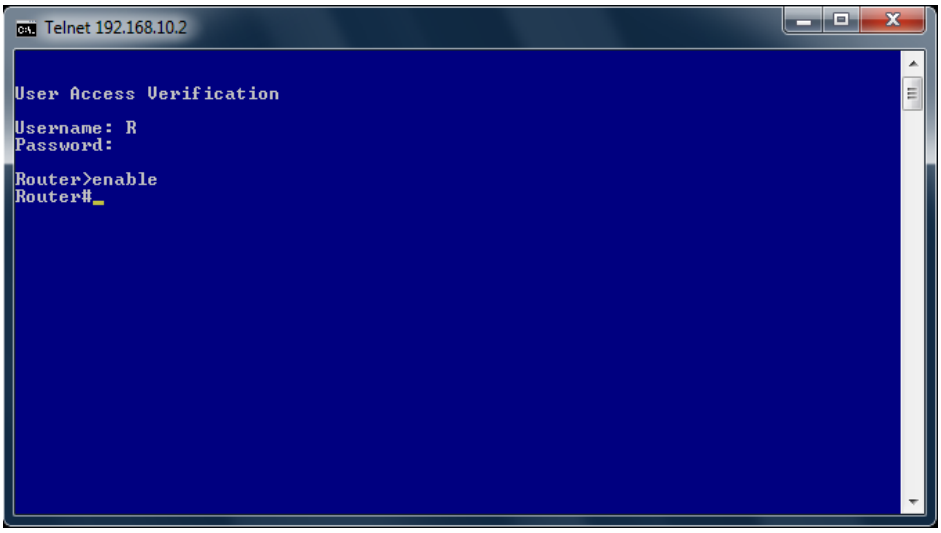
Příkazem tacacs-server host se nastavuje IP adresa serveru, na stejném principu jako RADIUS.

```
R1 (config) # tacacs-server host 192.168.10.2 key net
```

Aby bylo po uživateli vyžadováno přihlášení skrze server, je třeba zadat následující příkazy:

```
R1 (config) # line vty 0 15
R1 (config-line) # login authentication default
R1 (config-line) # authorization exec default
R1 (config-line) # exit
```

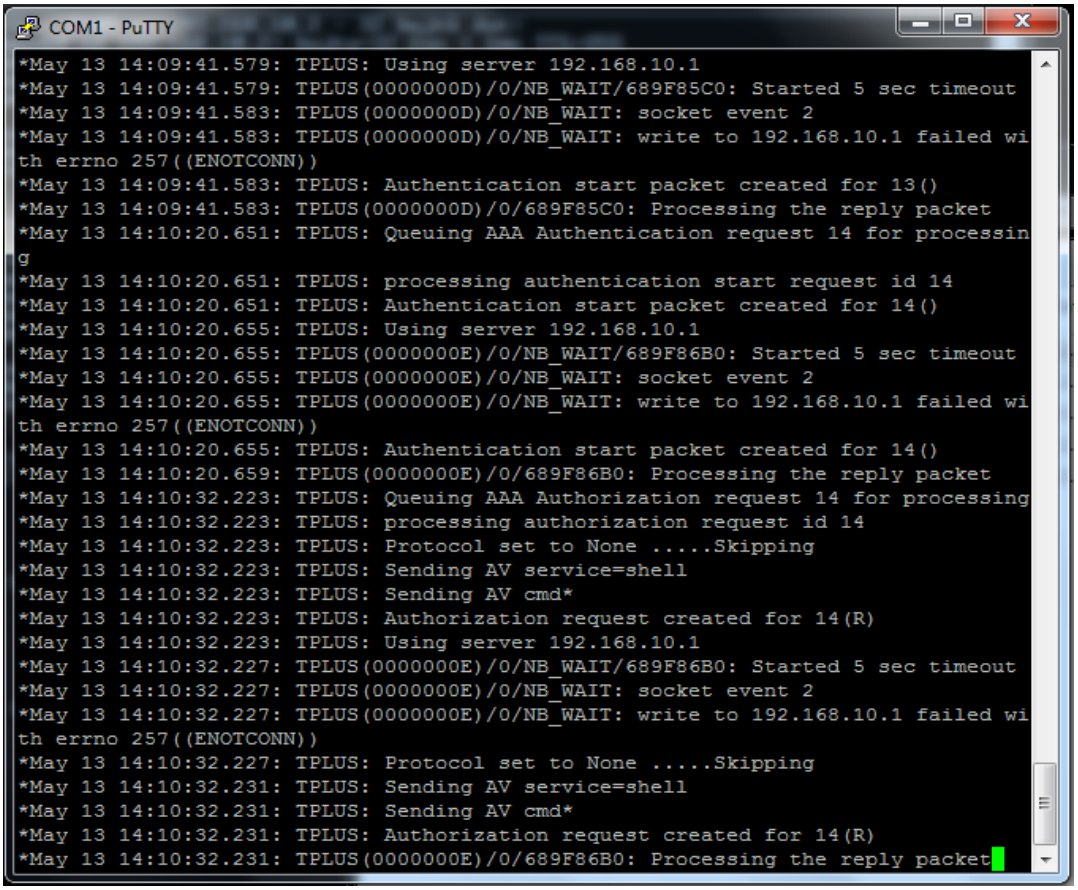
Úspěšné přihlášení uživatele R na klienta přes telnet viz obrázek č. 18.




**Obrázek 18 – Přihlášení uživatele na klienta TACACS+**

Pro sledování stavu komunikace a ověřování existuje příkaz debug tacacs. Po zadání tohoto příkazu je zachytáváno přihlašování uživatelů a na konzoli jsou vypisovány informace o probíhající autentizaci, viz obrázek č. 19.

```
R1 (config) # debug tacacs
```




**Obrázek 19– Příkaz „debug tacacs“**

Ve výpise se nacházejí informace o fázích, kterými server prochází a také jeho adresu IP. Také lze zachytit, jakjsou vytvářeny autentizační a autorizační pakety spolu s jejich odpověďmi.

#### Zhodnocení konfigurace protokolu TACACS+

Úprava konfiguračních souborů serveru TACACS+ byla méně přehlednější než u předchozího protokolu RADIUS, kde šlo využít nástroje WinRadius. Avšak nelze upřít vložené komentáře, které práci s konfiguračními soubory usnadňují a také vysvětlují jejich jednotlivé části. Nastavení klienta probíhalo podobně jako u protokolu RADIUS, a proto jsem s ním již neměla potíže. Uživatel se dokázal přihlásit přes nastavenou skupinu na serveru, s použitím lokálně uložených přihlašovacích údajů. Protokol funguje správně pouze částečně z důvodu nenastavení databáze na straně serveru.

## 9.3 Ověření funkčnosti protokolu LDAP

Testování proběhlo pomocí virtuálního serveru Oracle VirtualBox, na kterém běžel operační systém Debian Jessie. Použila jsem řešení OpenLDAP. Příkazy uvedené v této části práce jsou pro operační systém Debian Jessie (Debian 8), a proto nemusí fungovat na jiné verzi operačního systému.

#### Základní nastavení serveru LDAP

Jako první je třeba nainstalovat server. Instalaci serveru LDAP provedeme pomocí příkazu:

```
# apt-get -y install slapd ldap-utils.
```

Instaluje se démon slapd a také potřebné knihovny. (*Configure LDAP Server*, 2015) Následně spustíme základní konfiguraci serveru LDAP:

```
# dpkg-reconfigure -p low slapd
```

Nastavuje se název domény, název organizace, heslo administrátora, výběr databáze aj. Jako doménové jméno jsem nastavila „svet.cz“, název organizace „svetova-organizace“ a nastavila heslo. Typ databáze jsem zvolila HDB. Následně ověříme, že v souboru `/etc/default/slapd` existuje řádek, který nastavuje adresu a port, na kterém služba poběží a odkomentujeme jej. (Tomas 2014) U mne vypadal takto:

```
SLAPD_SERVICES="ldap://127.0.0.1:389/ ldaps:/// ldapi:///".
```

Poté ověříme počáteční konfiguraci pomocí příkazu slapcat, který vypíše informace o provedeném nastavení:

```
dn: cn=admin,dc=svet,dc=cz
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
description: LDAP administrator
userPassword:: e1NTSEF9L0h4ellKcDFwVDVPU1kva2hBMGpZaldDWWx3VGI3Ry8=
structuralObjectClass: organizationalRole
entryUUID: 4106eb5c-a56d-1035-9d4f-e3074ee4288d
creatorsName: cn=admin,dc=svet,dc=cz
createTimestamp: 20160503112350Z
entryCSN: 20160503112350.165522Z#000000#000#000000
modifiersName: cn=admin,dc=svet,dc=cz
modifyTimestamp: 20160503112350Z
```

Správné fungování démona slapd ověříme skrze nástroj ldapsearch, který vyhledá určitý výraz.V průvodci OpenLDAPSoftware 2.4 Administrator's Guide (2011) je použitý tento příkaz:

```
# ldapsearch -x '' -s base '(objectclass=*)' namingContexts.
```

Nyní již můžeme přidat do adresáře nový záznam. Nejdříve je třeba vytvořit soubor ldif, který vyžaduje nástroj ldapadd. Důležité je vyplnit dc=název_domény a o: vase_organizace. Do souboru ldif si lze uložit různé informace a přidávat s jeho pomocí nové uživatele, domény, záznamy, měnit hesla apod. Zde je ukázka souboru ldif, který přidává nového uživatele s rolí manažera:

```
dn: dc=svet,dc=cz
objectclass: dcObject
objectclass: organization
o: svetova-organizace
dc: svet
dn: cn=Manager,dc=svet,dc=cz
objectclass: organizationalRole
cn: Manager
```

Po vytvoření souboru ldif jej přidáme do adresáře:

```
# ldapadd -x -D "cn=Manager,dc=svet,dc=cz" -W -f prvni.ldif
```

Záznam si následně vyhledáme, abychom zjistili, zda byl opravdu přidán:

```
# ldapsearch -x -b 'dc=svet,dc=cz' '(objectclass=*)'.
```

(*OpenLDAP,* c2011, s. 13-14)

Případně lze použít příkaz uvedený na obrázku č. 20.

```
denca@debian:~$ sudo ldapsearch -x '' -s base '(objectlass=*)' namingContexts
# extended LDIF
#
# LDAPv3
# base <> (default) with scope baseObject
# filter: (objectclass=*)
# requesting:  (objectlass=*) namingContexts
#

#
dn:
namingContexts: dc=svet,dc=cz

# search result
search: 2
result: 0 Success

# numResponses: 2
# numEntries: 1
```

**Obrázek 20– Výpis příkazu „ldapsearch“**

Nakonec příkazem „sudo service slapd status“ zkontroluji, zda démon běží, viz obrázek č. 21.

```
denca@debian:~$ sudo service slapd status
[sudo] password for denca:
● slapd.service - LSB: OpenLDAP standalone server (Lightweight Directory Access
Protocol)
   Loaded: loaded (/etc/init.d/slapd)
   Active: active (running) since Po 2016-05-09 21:09:33 CEST; 3min 59s ago
  Process: 408 ExecStart=/etc/init.d/slapd start (code=exited, status=0/SUCCESS)
   CGroup: /system.slice/slapd.service
           └─497 /usr/sbin/slapd -h ldap:/// ldapi:/// -g openldap -u openlda...

kvě 09 21:09:21 debian slapd[450]: @(#) $OpenLDAP: slapd  (Jan 17 2016 16:... $
                                        buildd@x86-csail-01:/build/open...pd
kvě 09 21:09:22 debian slapd[497]: hdb_db_open: database "dc=svet,dc=cz": ...y.
kvě 09 21:09:33 debian slapd[497]: slapd starting
kvě 09 21:09:35 debian slapd[408]: Starting OpenLDAP: slapd.
Hint: Some lines were ellipsized, use -l to show in full.
denca@debian:~$
```

**Obrázek 21– Výpis příkazu „sudo service slapd status“**

## Metody zabezpečení serveru LDAP

V této části práce je předvedena praktická ukázka nastavení zabezpečení. Toho je docíleno různými metodami – SSL, TLS, SASL. LDAP nabízí i anonymní přístup, kdy není potřeba přihlašovací jméno ani heslo. Existuje i zabezpečení pomocí jména a hesla v podobě čistého textu. Tyto typy zabezpečení však nejsou doporučovány, a proto se jimi tato práce nebude dále zabývat.

OpenLDAP SASL podporuje několik bezpečnostních mechanismů – DIGEST-MD5, PLAIN a EXTERNAL, který se používá spolu s TLS. SASL PLAIN není doporučované zabezpečení. Pro zabezpečení SASL je potřeba nainstalovat Cyrus SASL, a to jak na server, tak i na klienta.(*OpenLDAP,* c2011, s. 143)

DIGEST-MD5 využívá tzv. tajemství, nejčastěji je to heslo. Jako první krok přidáme nového uživatele:

```
# saslpasswd2 -c denca.
```

V dalším kroku je třeba se ujistit, že hesla budou ve formě čistého textu. Otevřením konfiguračního souboru slapd.conf zjistíme nastavení pro hesla:

```
password-hash {CLEARTEXT}.
```
(*OpenLDAP,* c2011, s. 146)

Následující příkazy jsem čerpala od autora Tomas, z webových stránek Lisenet (2001). Pro SSL/TLS zabezpečení si lze nainstalovat OpenSSL, ale v případě Debianu je lepší si vytvořit vlastní certifikáty s balíčkem gnutls-bin. Po jeho instalaci již můžeme spustit generování vlastního klíče:

```
# certtool --generate-privkey --outfile server.key
# certtool --generate-self-signed --load-privkey server.key --outfile
server.crt.
```

Po vygenerování klíče je třeba vytvořit soubor LDIF (např. ldaps.ldif), který bude obsahovat cestu k vytvořeným certifikátům. Příklad souboru ldif pro certifikaci:

```
dn: cn=config
add: olcTLSCACertificateFile
olcTLSCACertificateFile: /etc/ssl/webserver/server-ca.crt
add: olcTLSCertificateFile
olcTLSCertificateFile: /etc/ssl/webserver/server.crt
add: olcTLSCertificateKeyFile
olcTLSCertificateKeyFile: /etc/ssl/webserver/server.key.
```

Příkazem ldapmodify importujeme výše vytvořený soubor.

```
# ldapmodify -Y EXTERNAL -H ldapi:/// -f ./ldaps.ldif.
```

Předchozí nastavení autentizace si ověříme příkazem ldapsearch. V případě, že je hlášena chyba, je třeba začít od začátku.

```
# ldapsearch -Y EXTERNAL -H ldapi:/// -b cn=config -s base|grep TLS
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
olcTLSCACertificateFile: /etc/ssl/webserver/server-ca.crt
olcTLSCertificateFile: /etc/ssl/webserver/server.crt
```

```
olcTLSCertificateKeyFile: /etc/ssl/webserver/server.key.
```

Šifrování hesel algoritmem SSHA se provede nově vytvořeným souborem passwordhash.ldif:

```
dn: cn=config
add: olcPasswordHash
olcPasswordHash: {SSHA}.
```

Po uložení souboru jej importujeme příkazem:

```
# ldapmodify -Y EXTERNAL -H ldapi:/// -f ./passwordhash.ldif.
```

Správné nastavení si lze ověřit opět nástrojem ldapsearch:

```
# ldapsearch -Y EXTERNAL -H ldapi:/// -b cn=config|grep SSHA
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
olcPasswordHash: {SSHA}
olcRootPW: {SSHA}NRBAhLr9Ae0SveMOD8MdiOb1sOmEteSt.
```

Pro přidání nové domény vytvoříme soubor nova.cz.ldif , který může vypadat následovně:

```
dn: dc=nova.cz,dc=svet,dc=cz
o: nova.cz
dc: nova.cz
objectClass: dcObject
objectClass: organization
dn: ou=Users,dc=nova.cz,dc=svet,dc=cz
objectClass: organizationalUnit
ou: Users
dn: ou=Groups,dc=nova.cz,dc=svet,dc=cz
objectClass: organizationalUnit
ou: Groups
dn: cn=sysadmins,ou=Groups,nova.cz,dc=svet,dc=cz
gidNumber: 1000
objectClass: posixGroup
cn: sysadmins

# ldapadd -nx -f ./nova.cz.ldif
# ldapadd -x -D cn=admin,dc=svet,dc=cz -W -f ./nova.cz.ldif.
```

Do nově vytvořené domény lze například přidat nového uživatele (nový soubor „uziv.ldif“):

```
dn: uid=alc,ou=Users,dc=nova.cz,dc=svet,dc=cz
uid: alc
uidNumber: 4567
gidNumber: 4567
cn: Aneta
sn: T
objectClass: posixAccount
objectclass: organizationalPerson
loginShell: /sbin/nologin
homeDirectory: /home/alc

# ldapadd -nx -f ./uziv.ldif
!adding new entry "uid=alc,ou=Users,dc=nova.cz,dc=svet,dc=cz"
# ldapadd -x -D cn=admin,dc=svet,dc=cz -W -f ./uziv.ldif.
```

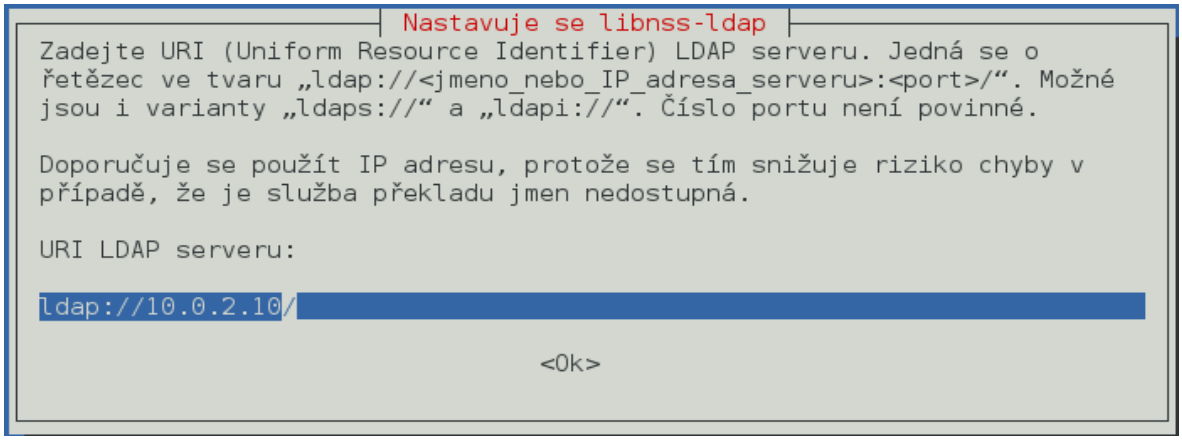
Nástrojem ldappasswd lze novému uživateli změnit heslo. OpenLDAP umožňuje také nastavení přístupových práv pomocí ACL.

### Nastaveni klienta LDAP

Klienta LDAP jsem nastavovala podle webových stránek Server World – Configuring LDAP clients (2015).Na klientovi LDAP jsem nejdříve nainstalovala nástroj libpam a také nástroje pro správu LDAP.

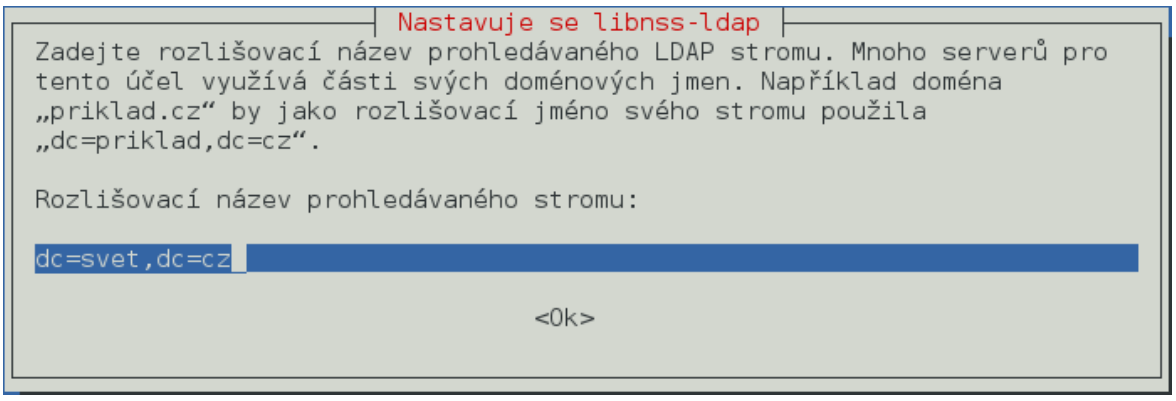
```
# aptitude -y install libnss-ldap libpam-ldap ldap-utils
```

Balíček libnss-ldap vyžaduje vyplnění různých informací, jako je adresa serveru LDAP (viz obrázek č. 22), verze protokolu LDAP, účet a heslo uživatele „root“.




**Obrázek 22– Formulář pro vyplnění adresy serveru LDAP**

Dále se vkládá rozlišovací jméno stromu (viz obrázek č. 23) a účet správce.




**Obrázek 23– Formulář pro vyplnění rozlišovacího jména**

V následujícím kroku jsem upravila konfigurační soubor nsswitch.conf a přidala nové řádky.

```
# sudo gedit /etc/nsswitch.conf

passwd: compat ldap

group: compat ldap

shadow: compat ldap
```

V souboru common-password je třeba na řádku 26 odstranit slovo „use_authtok“.

```
# sudo gedit /etc/pam.d/common-password

password     [success=1 user_unknown=ignore default=die]     pam_ldap.so
try_first_pass
```

Další úprava se nachází v souboru common-session.

```
# sudo gedit /etc/pam.d/common-session
```

V tomto souboru jsem na jeho konec přidala tento řádek, který při prvním přihlášení uživatele vytváří domovský adresář.

```
session optional        pam_mkhomedir.so skel=/etc/skel umask=077
```

Po provedení všech úprav je potřeba se odhlásit a zkusit se přihlásit přes uživatele LDAP. Nyní již lze měnit heslo uživatele, přidávat nové záznamy apod.

### Zhodnocení konfigurace protokolu LDAP

Nastavení protokolu LDAP pro mě bylo nejtěžší částí práce, a to hlavně z důvodu, že jsem předtím na unixovém systému tímto způsobem nepracovala. Konfigurace serveru byla časově náročná, i když věřím, že zkušenější administrátor by ji zvládl rychle a snadno. LDAP byl také těžší na počáteční pochopení struktury a organizaci záznamů. Z hlediska administrátora oceňuji možnost opětovného základního nastavení, kdy v případě špatné konfigurace lze server znovu nastavit. Nastavení klienta bylo jednodušší a rychlejší. Pro běžného uživatele je práce se záznamy snadná.

# 10 ZÁVĚR

Cílem mé bakalářské práce bylo popsat a vysvětlit obecné mechanismy AAA a následně popsat jednotlivé protokoly. V první kapitole jsem obecně vysvětlila pojmyautentizace, autorizace a účtování, jež jsou používány v dalších částech práce. Následující kapitoly byly věnovány samotným protokolům – RADIUS, Diameter, TACACS a TACACS+, PPP, EAP a LDAP . U těchto protokolů jsem nastínila jejich popis a funkce, princip komunikace a formát paketu, popsala složky AAA a případně uvedla specifika daného protokolu.

V praktické části práce jsem se zabývala nejprve protokolem RADIUS, který jsem si vybrala z důvodu jeho velkého rozšíření v oblasti bezpečnosti počítačových sítí. Tento protokol jsem záhy úspěšně nakonfigurovala na představeném síťovém modelu. Dále jsem ověřila funkčnost protokolu TACACS+. Protokol TACACS+ nabízí, dle mého názoru, lepší zabezpečení než protokol RADIUS. Na rozdíl od protokolu RADIUS totiž využívá spolehlivé doručování paketů pomocí protokolu TCP a také zašifrujetéměř celý paket a ne jen heslo.Nakonec jsem si vyzkoušela práci s operačním systémem Debian 8, na kterém jsem zprovoznila server LDAP. Tento protokol je na unixových systémech velice populární, a proto jsem se rozhodla jej také vyzkoušet. Myslím si, že tento protokol nabízí řadurůzných implementací atypů zabezpečení. Je vhodný do podniků, kde je potřeba uchovávat a vyhledávat informace, například o zaměstnancích (jejich jména, adresy, telefonní čísla apod.).

Ve svojí bakalářské práci jsem uvedla základní informace o bezpečnostních protokolech a vybrané z nich prakticky ověřila. Z důvodu objevování nových způsobů, jak narušit bezpečnost sítí, je třeba vyvíjet stále další a lepší bezpečnostní opatření. Protokoly AAA v tomto ohledu zatím poskytují účinné řešení. A jak z této práce vyplývá, je na každém, aby si vybral, které řešení (protokol) mu nejvíce vyhovuje. Bezpečnost sítí se však nesmí podceňovat a je vždy lepší si zvolit zabezpečení větší než menší.

# 11 POUŽITÁ LITERATURA

**ANTAL, Lukáš.** Méně známé slabiny WiFi. *IT Systems* [online]. c2001-2016, 2013 [cit. 2016-04-14]. ISSN 1802-615X. Dostupné z: http://www.systemonline.cz/it-security/mene-zname-slabiny-wifi.htm.

**CALHOUN, P., J. LOUGHNEY, E. GUTTMAN, G. ZORN a J. ARKKO.** Diameter Base Protocol. In: *Diameter Base Protocol* [online]. San Jose: Calhoun, c2003, s. 33-50 [cit. 2016-04-25]. Dostupné z: https://tools.ietf.org/html/rfc3588.

**CARROLL, Brandon**. *Cisco access control security: AAA administrative services*. Indianapolis, Ind.: Cisco Press, c2004. ISBN 1587051249.

**CHEN, Jyh-Cheng a Tao ZHANG**. *IP-based next-generation wireless networks: systems, architectures, and protocols*. New York: Wiley-Interscience, c2004, xxiii, 413 p.

**CISCO SYSTEMS, INC.***Cisco IOS Security Configuration Guide: Release 12.2* [online]. 12.2. San Jose: Cisco Systems, Inc., c2001-2006 [cit. 2016-05-18]. Dostupné z: http://www.cisco.com/c/en/us/td/docs/ios/12_2/security/configuration/guide/fsecur_c.html.

**Configure LDAP Clients**. *Server World* [online]. Server World, c2007-2016 [cit. 2016-05-17]. Dostupné z: http://www.server-world.info/en/note?os=Debian_8&p=openldap&f=3.

**Configure LDAP Server**. *Server World* [online]. Server World, c2007-2016 [cit. 2016-05-17]. Dostupné z: http://www.server-world.info/en/note?os=Debian_8&p=openldap.

**DOSTÁLEK, Libor**. *Velký průvodce protokoly TCP/IP: bezpečnost*. Vyd. 1. Praha: Computer Press, 2001. Profi. ISBN 807226513X.

**KABELOVÁ, Alena a Libor DOSTÁLEK**. *Velký průvodce protokoly TCP/IP a systémem DNS*. 3. aktualiz. a rozš. vyd. Praha: Computer Press, 2002. Všechny cesty k informacím. ISBN 8072266756.

**LDAP Version 3.***Isode* [online]. Isode Ltd., c2002-2016 [cit. 2016-05-1]. Dostupné z: http://www.isode.com/whitepapers/ic-6032.html.

**LIU, Jeffrey, Steven JIANG a Hicks LIN**. Introduction to Diameter: Get the next generation AAA protocol. *IBM* [online]. 2006 [cit. 2016-03-10]. Dostupné z: https://www.ibm.com/developerworks/wireless/library/wi-diameter/.

**LLOYD, B. a W. SIMPSON**. PPP Authentication Protocols. In: *PPP Authentication Protocols* [online]. California: Lloyd, 1992, s. 3-8 [cit. 2016-04-25]. Dostupné z: https://www.ietf.org/rfc/rfc1334.txt.

**MOGOLLON, Manuel**. *Cryptography and security services: mechanisms and applications*. Hershey, PA: CyberTech Pub., c2007, xv, 471 p. ISBN 1599048396.

**NAIK, Dilip C**. *Internet: standardy a protokoly*. 1. vyd. Brno: Computer Press, c1999. Internet. ISBN 8072261460.

**NAKHJIRI, Madjid a Mahsa NAKHJIRI**. *AAA and network security for mobile access: radius, diameter, EAP, PKI and IP mobility*. Hoboken, NJ: John Wiley & Sons, c2005. ISBN 9780470011942.

**OpenLDAP Software 2.4 Administrator's Guide.**ZEILENGA, Kurt a Richard KRUKAR. *OpenLDAP* [online]. Minden, Nevada: OpenLDAP Foundation, c2011 [cit. 2016-05-17]. Dostupné z: http://www.openldap.org/doc/admin24/.

**PUŽMANOVÁ, Rita**. Metody autentizace EAP. *ComputerWorld: Deník pro IT profesionály* [online]. 2006 [cit. 2016-04-14]. Dostupné z: http://computerworld.cz/archiv/metody-autentizace-eap-25352.

**PUŽMANOVÁ, Rita**. *Širokopásmový Internet: přístupové a domácí sítě*. Vyd. 1. Brno: Computer Press, 2004. ISBN 8025101398.

**SANTUKA, Vivek, Premdeep BANGA a Brandon CARROLL**. *AAA identity management security*. Indianapolis, IN: Cisco Press, c2011, xxiii, 443 p. ISBN 1587141442.

**TACACS.NET**. *TACACS.net Installation & Configuration Guide* [online]. San Jose: TACACS.net, c2010 [cit. 2016-05-17]. Dostupné z: www.tacacs.net/docs/TACACS.net_config.pdf.

**Tomas**. Configure LDAP Server. *Lisenet* [online]. Server World, 2014 [cit. 2016-05-17]. Dostupné z: https://www.lisenet.com/2014/install-and-configure-an-openldap-server-with-ssl-on-debian-wheezy/.

**TUTTLE, Steven, Ami EHLENBERGER, RamakrishnaGORTHI, et al.***Understanding LDAP design and implementation* [online]. 2. NY: IBM, c2004 [cit. 2016-04-11]. ISBN 073849786X. Dostupné z: http://www.redbooks.ibm.com/abstracts/sg244986.html.

**VENTURA, Håkan**. *Diameter - Next generation's AAA protocol*. LINKÖPING, 2002. Institutionen för Systemteknik. Vedoucí práce Miguel Garcia, Peter Cederstrand.

**WILKINS, Sean**. TACACS+ vs. RADIUS: Similarities and Differences. *Pearson IT Certification* [online]. Indianapolis: Pearson Education, 2015 [cit. 2016-05-10]. Dostupné z: http://www.pearsonitcertification.com/articles/article.aspx?p=2449614.

# 12 PŘÍLOHY

Příloha A – *Konfigurační soubor „tacplus.xml“*

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Version 1.2 -->
<!-- This is the global configuration file for the TACACS+ Server

If you have multiple NICs or IP addresses on this system (including virtual
adapters like VMWare or VPN),
the server will resolve to the first IP address listed in Ipconfig.
If you have multiple physical or virtual NICs, you will need to manually
hard code the IP address in <LocalIP>,
disconnect any open sessions, and restart the service. You will also need
to use the -s switch in TACTest.

The following logging levels are available: Alert, Critical, Error,
Warning, Notice, Information, and Debug.
Debug generates the most information, and Alert generates the least amount
of logging information.
-->
<Server xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<Port>49</Port>
<LocalIP>192.168.10.1</LocalIP>
<DisabledPrompt>This account has been disabled</DisabledPrompt>
<PasswordPrompt>Password: </PasswordPrompt>
<UserNamePrompt>Username: </UserNamePrompt>
<ExpiredPasswordPrompt>The password for this account has
expired</ExpiredPasswordPrompt>
<IncorrectPasswordPrompt>Incorrect Password</IncorrectPasswordPrompt>
<IncorrectUserOrPasswordPrompt>Invalid username or incorrect
password</IncorrectUserOrPasswordPrompt>
<SocketTimeoutSecs>30</SocketTimeoutSecs>
<AccountingLog RolloverDays="30" RolloverMB="10" DeleteDays="90"
LoggingLevel="Information"></AccountingLog>
<DebugLog RolloverDays="30" RolloverMB="10" DeleteDays="90"
LoggingLevel="Debug"></DebugLog>
<SystemLog RolloverDays="30" RolloverMB="10" DeleteDays="90"
LoggingLevel="Information"></SystemLog>
<!-- <Syslog Host="127.0.0.1" Port="514" MaxLength="1000"
Facility="Local6"></Syslog> -->
<AccountLockoutTries>6</AccountLockoutTries>
<AccountLockoutperiodMins>30</AccountLockoutperiodMins>
<SessionIdleTimeoutMins>15</SessionIdleTimeoutMins>
<TimedCacheExpirySecs>5</TimedCacheExpirySecs>
<OTPSeparator>*</OTPSeparator>
<AuthorizationRuleRequiredForAuthentication>Disabled</AuthorizationRuleRequ
iredForAuthentication>
</Server>
```

Příloha B – *Konfigurační soubor „authentication.xml“*

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Version 1.2 -->
<Authentication xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<UserGroups>

<!-- GENERAL INSTRUCTIONS -->

<!-- This is the configuration file for Authentication Groups.
This file has examples of the types of authentication groups
that are supported below. You can modify them as needed.

The DEFAULT user group defined at the bottom of this file
is a catch-all group that will accept all of the users
configured as Administrators on the local computer as
authorized TACACS+ users.

You have the option of using clear text passwords (ClearText),
or DES encrypted passwords (DES). Use clear text passwords to
get the system up and running, and then change to DES passwords
before deploying in a production environment.
Use the TACDES utility to create DES passwords.

EnablePassword (optional) is the password used for the enable command.
If this is not provided, the login password for the user is used.

CHAPPassword (optional) is used for CHAP/MS-CHAP/ARAP.
If this is not provided then the login password is used.

OutboundPassword (optional) is used for outbound requests.
If this is not provided then the login password is used. -->

<!-- / GENERAL INSTRUCTIONS -->


<!-- FILE GROUP EXAMPLE -->

<!-- This is an example of a File User group.
The File User groups can be used to define users that only exist
within the TACACS+ server. -->

<UserGroup>
<Name>Prvni</Name>
<AuthenticationType>File</AuthenticationType>

            <Users>
                  <User>
                  <Name>user1</Name>
                  <LoginPassword ClearText="heslo" DES=""></LoginPassword>
                  <EnablePassword ClearText="" DES=""></EnablePassword>
                  <CHAPPassword ClearText="" DES=""></CHAPPassword>
                  <OutboundPassword ClearText="" DES=""></OutboundPassword>
                  </User>
                  <User>
                  <Name>user2</Name>
                  <LoginPassword ClearText="heslo" DES=""></LoginPassword>
                  <EnablePassword ClearText="" DES=""></EnablePassword>
                  <CHAPPassword ClearText="" DES=""></CHAPPassword>
                  <OutboundPassword ClearText="" DES=""></OutboundPassword>
```

```
                </User>
            </Users>

</UserGroup>

<!-- ACTIVE DIRECTORY EXAMPLE -->

<!--This is an example is of a Windows Active Directory group.
This group will authenticate using a Windows Domain Controller.
LDAPUserDirectorySubtree is the distinguished name of the subtree that
contains all users.
The LDAPGroupName should point to the name of the AD group.
LDAPAccessUserName and LDAPAccessUserPassword are optional elements and
should be specified if the active directory server
does not allow anonymous  access to the active directory for
authentication.
This username must  have read/write access to Active Directory.

To see the user directory subtree name, you can execute the following
dsquery command on windows server:
Note: The command DSQUERY is only available on Windows Server.
                        C:\>dsquery user -samid USERNAME

To see the list of AD groups the user belongs to, use:
  C:\>dsquery user -samid USERNAME | dsget user -memberof -expand
  You can use the complete DN of the group or just the AD name of the group
in the LDAPGroupName configuration parameter.

-->
<!--
      <UserGroup>
<Name>Network Operations</Name>
<AuthenticationType>Windows_Domain</AuthenticationType>
<LDAPServer>127.0.0.1:389</LDAPServer>
      <LDAPUserDirectorySubtree>cn=Users,DC=contoso,DC=com</LDAPUserDirecto
rySubtree>
      <LDAPGroupName>Network Operations</LDAPGroupName>
      <LDAPAccessUserName>Administrator</LDAPAccessUserName>
      <LDAPAccessUserPassword ClearText="password"
DES=""></LDAPAccessUserPassword>
      </UserGroup>
-->
<!-- / ACTIVE DIRECTORY EXAMPLE -->

<!-- LDAP EXAMPLE -->

<!--This is an example of a LDAP based authentication and can be used to
authenticate against generic LDAP servers or AD servers.
This is similar to the active directory example above but with additional
configuration.

Here is a description of various additional configuration parameters:
LDAPUserObjectClass-The value of the LDAP 'objectType' attribute that
identifies the record as a user.
LDAPUserNameAttribute-The name of the attribute in the user record that
contains the user name.
LDAPMemberOfAttribute- The name of the attribute in the user record that
contains the various groups the user belongs to.
LDAPAuthType-The type of authentication used. Valid values are Anonymous,
Basic, Digest, Dpa, External, Kerberos, Msn, Negotiate, Ntlm, Sicily.
```

```
                  For Active dirctory, usually Ntlm is used. For LDAP,
usually Basic authentication is used.
-->
<!--
      <UserGroup>
<Name>Tech Support</Name>
<AuthenticationType>LDAP</AuthenticationType>
<LDAPServer>127.0.0.1:389</LDAPServer>
      <LDAPGroupName>Tech Support</LDAPGroupName>
      <LDAPUserDirectorySubtree>cn=Users,DC=contoso,DC=com</LDAPUserDirecto
rySubtree>
<LDAPAccessUserName>Administrator</LDAPAccessUserName>
      <LDAPAccessUserPassword ClearText="password"
DES=""></LDAPAccessUserPassword>
      <LDAPUserObjectClass>person</LDAPUserObjectClass>
      <LDAPUserNameAttribute>cn</LDAPUserNameAttribute>
      <LDAPMemberOfAttribute>isMemberOf</LDAPMemberOfAttribute>
      <LDAPAuthType>Basic</LDAPAuthType>
      </UserGroup>
-->
<!-- / ACTIVE DIRECTORY EXAMPLE -->

<!-- LOCALHOST EXAMPLE -->

<!-- This is an example of a Windows Localhost group.
This group will authenticate using the users and groups
configured on the local computer. -->


<UserGroup>
<Name>LokalniSkupina</Name>
<AuthenticationType>Localhost</AuthenticationType>
<LocalhostGroupName>Administrators</LocalhostGroupName>
      </UserGroup>

<!-- / LOCALHOST EXAMPLE -->

<!-- DEFAULT GROUP -->

<!-- The DEFAULT group is included with the installation. It is
the fallback group that will allow all users configured
in the Administrators group on the local computer as TACACS+
users by default.

-->

<UserGroup>
<Name>DEFAULT</Name>
<AuthenticationType>Localhost</AuthenticationType>
<LocalhostGroupName>Administrators</LocalhostGroupName>
</UserGroup>
</UserGroups>
</Authentication>

<!-- /DEFAULT GROUP -->
```

Příloha C – *Konfigurační soubor „authorization.xml“*

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Version 1.2 -->
<Authorizations xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <Authorizations>
      <Authorization>
            <!--This entry will only be processed in the times given below-
->
            <!--<Time>MTWRFSN,04:00-21:00</Time>-->
            <!--This authorization section applies to the following user
groups. In case of conflicting authorization entries for the same group,
the entry which appears first in the file is used.-->
            <UserGroups>
                  <UserGroup>Prvni</UserGroup>
            </UserGroups>
            <!--This authorization section applies to the following client
groups. In case of conflicting authorization entries for the same client
group, the entry which appears first in the file is used.-->
                  <!--If no client groups are specified then the settings
are applied to the specified usergroups irrespective of the clients they
come from-->
            <ClientGroups>
                  <ClientGroup>LOCALHOST</ClientGroup>
                  <ClientGroup>INTERNAL</ClientGroup>
            </ClientGroups>
            <AutoExec>
                  <!--<Set>acl=7</Set>-->
                  <!-- When an exec is started, its connection access list
will be 7.  It will also automatically execute this autocmd. If the cmd
element is not provided then the shell entry is used when the shell is
first invoked.-->
                  <!--<Set>autocmd=telnet 10.1.1.1</Set>-->
                  <!--<Set>priv-lvl=7</Set>-->
            </AutoExec>

            <Shell><!--note that the login and exit commands are always
permitted-->
                  <!--<Permit>configure</Permit>--><!--this will allow
configure command -->
                  <!--<Deny>show running-config</Deny>--><!--this will deny
the user to run 'show running-config' -->
                  <!--<Permit>enable</Permit>--><!--this will allow this
group to run enable command -->
                  <!--<Deny>show bgp all</Deny>-->   <!--this will deny
this group to run show bgp all command -->
                  <!--<Permit>show bgp .*</Permit>--><!--this will deny
this group to run other show bgp commands -->
                  <Permit>.*show.*</Permit><!--This will allow all show
commands -->
                  <Deny>.*</Deny><!--This will deny all other commands -->
      </Shell>
            <Services>
                  <!-- <Service>
                        <Set>service=ppp</Set>
                        <Set>protocol=ip </Set> -->
                        <!--these groups can run IP over PPP only if they
use one of the following mandatory addresses. If they supply no address,
the first one here will be mandated-->
```

```
                        <!--<Set>addr=10.1.1.1</Set>--><!--mandatory
argument-->
                        <!--Their  mandatory input access list number is 5-
->
                        <!--<Set>inacl=5</Set>-->
                        <!--We will suggest an output access list of 10 but
the NAS may choose to ignore or override it-->
                        <!--<SetOptional>outacl=10</SetOptional>-->
                        <!--These are examples of vendor specific
attributes(VSAs)-->
                        <!--<Set>foundry-privlvl=5</Set>-->
                  <!-- </Service> -->
            </Services>
      </Authorization>

      <Authorization>
            <UserGroups>
                  <UserGroup>LokalniSkupina</UserGroup>
            </UserGroups>
            <!--No client group provided so this authorization section
applies to the above user groups from all the clients -->

            <!--this group is allowed to telnet everywhere except from
addresses beginning with 161.-->
            <Shell>
                  <!--<deny>telnet 161\.*</deny>
                  <Permit>telnet .*</Permit>-->
                  <Permit>.*show.*</Permit><!--This will allow all show
commands -->
                  <Deny>.*</Deny><!--This will deny all other commands -->
            </Shell>
      </Authorization>


<!-- DEFAULT PROFILE -->

<!-- The DEFAULT Authorization Profile is added by default with the Server
installation and is used to enable ALL Users full access to ALL Clients.
This group should be removed or commented out before deploying the Server
in a production environment.
-->

      <Authorization>
            <Shell>
                  <Permit>.*show.*</Permit><!--This will allow all show
commands -->
                  <Deny>.*</Deny><!--This will deny all other commands -->
            </Shell>
      </Authorization>

</Authorizations>
</Authorizations>
```

Příloha D – *Konfigurační soubor „clients.xml“*

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Version 1.2 -->
<!--This is the configuration file for TACACS+ clients. A TACACS+ client,
as defined by the RFC, is the client that is
 making a request to the TACACS+ server such as a router, switch, or
firewall-->
<Clients xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <ClientGroups>
            <!-- GENERAL INSTRUCTIONS


This is the configuration file for network clients such as
 routers, switches, firewalls, etc.

This file is read linerally (top to bottom) This means that
 the first match  is applied. This will enable you to configure
 overlapping client Groups. For example, you could specify
 one policy for 192.168.1.1/32 and another policy for
 192.168.1.0/24 and the first match will be applied.

It is recommended to put your more specific client Groups
 first, and the less specific client Groups towards the
 bottom of this file.

This configuration file supports Regular Expressions. This
 can be useful when you want to set policy based on hosthames
 instead of IP Addresses.

Examples:

192.168.1.1                  This matches a single IP address.
192.168.*                    This will match all ip addresses beginning
with 192.168.
192.168.1.1-192.168.1.255    This will match all ip addresses in the
specified range.
192.168.0.0/16               This will match all ip addresses in the
specified CIDR format.
192.168.1.0/255.255.255.0    This will match all ip addresses provided IP-
Subnet configuration.
^OrgSwitch-a.*               This will match all hostnames with with the
prefix 'OrgSwitch-a'.
^switch1$                    This matches a hostname 'switch1'.
switch                       This matches any hostname which has the word
'switch' in it.

For more information on using Regular Expressions refer to the following
links:
  http://www.regular-expressions.info/tutorialcnt.html
  http://www.regular-expressions.info/examples.html
  http://www.regextester.com/


  / GENERAL INSTRUCTIONS -->
            <!-- client GROUP EXAMPLES

Following are some examples of configuring client Groups
 with different names using Regular Expressions.
```

```
<ClientGroup Name="Core Routers">
<Secret ClearText="mysharedsecret" DES=""></Secret>
<Clients>
<Client>.*-rt.$</Client>
</Clients>
</ClientGroup>

<ClientGroup Name="Peering Routers">
<Secret ClearText="mysharedsecret" DES=""></Secret>
<Clients>
<Client>.*-pr.$</Client>
</Clients>
</ClientGroup>

<ClientGroup Name="Switches">
<Secret ClearText="mysharedsecret" DES=""></Secret>
<Clients>
<Client>192.168.1.1</Client>
<Client>192.168.*</Client>
<Client>192.168.1.1-192.168.1.255</Client>
<Client>192.168.0.0/16</Client>
      <Client>192.168.1.0/255.255.255.0</Client>
<Client>^OrgSwitch-a.*</Client>
      <Client>^switch1$</Client>
      <Client>switch</Client>
</Clients>
</ClientGroup>

<ClientGroup Name="Firewalls">
<Secret ClearText="mysharedsecret" DES=""></Secret>
<Clients>
<Client>192.168.*</Client>
<Client>192.168.1.1-192.168.1.255</Client>
<Client>192.168.1.0/16</Client>
<Client>OrgSwitch-a.*</Client>
<Client>192.168.1.1</Client>
</Clients>
</ClientGroup>

<ClientGroup Name="Prvni">
<Secret ClearText="network" DES=""></Secret>
<Clients>
<Client>192.168.10.2</Client>
</Clients>
</ClientGroup>

            / CLIENT GROUP EXAMPLES -->
            <!-- LOCALHOST GROUP

The LOCALHOST Group is added by default. This group is will
 enable the local computer to be a TACACS+ client and run
 the TACTest test tool. -->
            <ClientGroup Name="LOCALHOST">
                  <Secret ClearText="network" DES=""></Secret>
                  <Clients>
                        <Client>127.0.0.1</Client>
                  </Clients>
            </ClientGroup>
            <!-- INTERNAL GROUP
```

```
The INTERNAL Group is added by default. This group will
 enable all non-routeable IP addresses to be TACACS+
 clients without having to explicitly define them.
This is useful in an internal NAT or lab network.-->
            <ClientGroup Name="INTERNAL">
                  <Secret ClearText="network" DES=""></Secret>
                  <Clients>
                        <Client>10.0.0.0/8</Client>
                        <Client>172.16.0.0/12</Client>
                        <Client>192.168.0.0/16</Client>
                  </Clients>
            </ClientGroup>
            <!-- DEFAULT GROUP

The DEFAULT group is added by default and is used as a
 catch-all group to verify operations of the server.
This group will permit ANY network element to use TACACS+.
This group should be removed or commented out
 before deploying the server in a production environment.-->
            <ClientGroup Name="DEFAULT">
                  <Secret ClearText="network" DES=""></Secret>
                  <Clients>
                        <Client>.*</Client>
                  </Clients>
            </ClientGroup>
      </ClientGroups>
</Clients>
```