UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

Analýza protokolů transportní a aplikační vrstvy

David Novotný

Bakalářská práce
2013

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2012/2013**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**David Novotný**
Osobní číslo:	**I09217**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Analýza protokolů transportní a aplikační vrstvy**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování:

Autor práce podrobně představí vybrané protokoly aplikační vrstvy, popíše jejich strukturu a jejich vztah do transportní vrstvy. Na základě podrobné analýzy protokolů, jejich struktury a vztahu k TCP a UDP protokolům využije síťový analyzátor pro odchyt datagramů vybraných protokolů a na základě jejich struktury představí možnosti jejich využití. Autor připraví sadu minimálně 10 úloh pro analýzu a odchyt datagramů vybraných protokolů aplikační vrstvy a jejich vazby na transportní vrstvu TCP/IP modelu.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

***KOZIEROK, Charles M. The TCP/IP guide: a comprehensive, illustrated Internet protocols reference. San Francisco: No Starch Press, c2005, lxxiv, 1539 p. ISBN 15-932-7047-X**
***KABELOVÁ, Alena a Libor DOSTÁLEK. Velký průvodce protokoly TCP/IP a systémem DNS. 5., aktualiz. vyd. Brno: Computer Press, 2008, 488 s. ISBN 978-80-251-2236-5**

Vedoucí bakalářské práce: **Mgr. Josef Horálek**
Katedra softwarových technologií

Datum zadání bakalářské práce: **21. prosince 2012**
Termín odevzdání bakalářské práce: **10. května 2013**

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

Ing. Lukáš Čegan, Ph.D.
vedoucí katedry

V Pardubicích dne 29. března 2013

Prohlášení autora

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

Byl jsem seznámen s tím, že se na moji práci vztahují 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 23. 8. 2013 David Novotný

Poděkování

Tímto bych rád poděkoval Mgr. Josefu Horálkovi za jeho cenné rady a připomínky v průběhu zpracování této bakalářské práce. Také děkuji své rodině za trpělivost a podporu během studia.

Anotace

Tématem bakalářské práce jsou transportní a aplikační protokoly v síťovém modelu TCP/IP. Čtenář bude seznámen s 10 základními protokoly aplikační vrstvy a jejich implementací. Dále bude představen vztah aplikačních protokolů k transportní vrstvě modelu a názorně ukázána komunikace a možná bezpečnostní rizika, související s použitím nezabezpečených mechanismů při komunikaci po síti.

Klíčová slova

aplikační vrstva, transportní vrstva, IPv4, Wireshark, datagram, paket, protokol

Title

Analysis of transport and application layer protocols.

Annotation

The subject of this thesis is transport and application protocols in TCP/IP network model. Reader will be introduced to 10 basic protocols of the application layer and their implementation. Furthemore he will become acquainted with the topic of application protocols in relation to transport layer of a model and the thesis will also demonstrate communication and possible security risks connected with use of nonsecure mechanisms with communicating over network.

Keywords

application layer, transport layer, IPv4, Wireshark, datagram, packet, protocol

Obsah

Seznam zkratek

ARPA	Advanced Research Projects Agency
ASCII	American Standard Code for Information Interchange
BOOTP	Bootstrap Protocol
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
FTP	File Transfer Protocol
GPL	General Public License
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IMAP	Internet Message Access Protocol
IP	Internet Protocol
LAN	Local Area Network
MAC	Media Access Control
POP	Post Office Protocol
RFC	Request For Comments
SMTP	Simple Mail Transfer Protocol
SSH	Secure Shell
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
WAN	Wide Area Network

Seznam obrázků

Úvod

Síťová komunikace je nedílnou součástí našeho každodenního života, ať už čteme došlé zprávy skrze elektronickou poštu, sledujeme on-line zpravodajství oblíbeného portálu nebo spravujeme vzdálený server z domova, či jiného vzdáleného místa, využíváme k tomu Internet a protokoly, které jsou speciálně navrženy pro konkrétní služby. Komunikace v počítačové síti využívá opravdu velké množství protokolů a některé běžně používané budou podrobněji představeny v této bakalářské práci.

Hlavním cílem této bakalářské práce je podrobně představit sadu protokolů, kterou denně využíváme. Celá komunikace bude odchytnuta a představena pomocí nástroje, který dokáže zachytit veškerou příchozí i odchozí komunikaci na daném síťovém rozhraní.

V první části bude představen nástroj Wireshark, který umožňuje sledovat odchozí a příchozí komunikaci. Krátce bude nastíněn proces sledování paketů a jeho využití v rámci počítačové sítě.

Druhá část práce je zaměřena na síťový model TCP/IP, který se dnes využívá v počítačových sítích. Krátce bude představena historie modelu a jednotlivé vrstvy, které jsou v modelu definovány. Dále bude podrobně vysvětlen princip navázání spojení mezi dvěma koncovými zařízeními z pohledu transportní vrstvy modelu.

Třetí část práce obsahuje vybrané protokoly aplikační vrstvy modelu. Tyto protokoly slouží ke konfiguraci síťového rozhraní a komunikaci s dalšími servery, které jsou nedílnou součástí vybraných protokolů.

Čtvrtá část práce se zaměřuje na protokoly elektronické pošty. Bude zde vysvětlen princip odesílání a přijímaní elektronické pošty z pohledu klienta.

Pátá a poslední část této práce je zaměřena na aplikační protokoly, které slouží ke vzdálené konfiguraci a komunikaci mezi vzdálenými počítači. V této části bude představen jeden z nejstarších protokolů, který se využíval k těmto účelům a který nedisponuje žádným bezpečnostním mechanismem. Dále bude představen dnes hojně využívaný protokol v této kategorii, který se považuje za bezpečný. V obou případech bude názorně ukázán rozdíl v nešifrované a šifrované komunikaci.

Pokud není uvedeno jinak, veškerá obrazová dokumentace v bakalářské práci byla vytvořena autorem.

1 Sledování paketů a program Wireshark

Sledování paketů (packet sniffing) je proces, při kterém se zachycují přenášená data po síti. Díky tomuto procesu lze lépe porozumět fungování dané sítě a řešit případné problémy v síťové komunikaci. Sledování a analýza paketů umožňují:

- Porozumět vlastnostem sítě
- Zjistit jednotlivé uživatele sítě
- Zjistit, kdo spotřebovává dostupnou šířku pásma
- Zjistit v jakých časech je síť využita nejvíce
- Identifikovat možné útoky v síti
- Identifikovat neefektivní aplikace

Pro zachycení jednotlivých datagramů, procházejících počítačovou sítí, se používají paketové sniffery. Existuje celá řada nástrojů, lišících se podporovanými protokoly, cenou, podporovaným operačním systémem a nakonec i uživatelskou přívětivostí. Pro účely této práce byl zvolen program Wireshark[1]. Výhodou tohoto programu je v prvé řadě cena.

Program Wireshark je open source software, je šířen pod licencí GPL. Může být použit k osobním i komerčním účelům. Program byl vytvořen studentem informatiky Geraldem Combsem, který jej původně vyvíjel pod názvem Ethereal. Po změně zaměstnání v projektu pokračoval. Poslední zaměstnavatel měl však veškerá práva na značku Ethereal. Cobs se svým vývojovým týmem v polovině roku 2006 přejmenoval projekt na Wireshark. Obliba Wiresharku rychle rostla a dosud přispívá k vývoji programu přes 500 přispěvatelů. Wireshark podporuje přes 850 protokolů, mezi které patří i některé proprietární protokoly typu AppleTalk a BitTorrent a každou další vydanou verzí přibývají podporované protokoly. Mezi další používané nástroje pro sledování paketů patří Microsoft Network Monitor[2] nebo nástroj pracující v příkazové řádce tcpdump[3]. [2]

[1] http://www.wireshark.org/
[2] http://www.microsoft.com/en-us/download/details.aspx?id=4865
[3] http://www.tcpdump.org/

2 Model TCP/IP

TCP/IP je model představující skupinu protokolů členěných do 4 vrstev. Model udává pohled na logické členění a princip fungování síťového programového vybavení. Zahrnuje principy fungování v jednotlivých vrstvách modelu. V současné době je implementován prakticky v každém operačním systému a slouží ke komunikaci v heterogenních sítích. Model je nezávislý na použitém médiu, používá se pro LAN sítě i pro WAN sítě. Pro model platí, že síť je decentralizovaného typu (nemá žádný centrální prvek, který by se mohl stát cílem útoku) a má nespojovaný charakter. [24]

Obrázek 1 – Porovnání ISO/OSI a TCP/IP modelu

2.1 Historie modelu

Model TCP/IP je původně projektem společnosti ARPA, která spadá pod ministerstvo obrany USA, jako komunikační protokol pro sjednocení tehdejší sítě ARPANET. Vznikal koncem 60. let dvacátého století. Původně byl navržen především pro systémy UNIX, dnes je používán prakticky v každém operačním systému. Model zpočátku sloužil pro přední americké univerzity a později se stal součástí navzájem propojených počítačových sítí, dnes označovaných jako Internet. [24]

2.2 Vrstva síťového rozhraní

Vrstva zajišťuje základní přenos dat mezi uzly počítačové sítě. V rámci modelu není tato vrstva blíže specifikována, je závislá na použité přenosové technologii např.: Ethernet, Token Ring, ATM. Vychází se z toho, že není potřeba vymýšlet nová řešení na nejnižší vrstvě, ale použijí se technologie již fungující. [24]

2.3 Síťová vrstva

Tato vrstva již není závislá na použité přenosové technologii a stará se o logickou adresaci paketů, zapouzdření dat, předávání, směrování a doručení jednotlivých datagramů mezi odesílatelem a příjemcem. K tomu slouží IP protokol. Jednotlivé datagramy obsahují v hlavičce IP adresu odesílatele a příjemce datagramu. Služba nezaručuje doručení datagramu, označuje se za nespolehlivou. [24]

Protokol IP verze 4

K označení síťového zařízení na 3. vrstvě TCP/IP modelu se využívá IP adresa. V současné době se využívá stále adresa IP verze 4, která je však postupně nahrazována adresou IP verze 6, kvůli nedostatečnosti adresového prostoru, kterou nabízí verze 4. Adresa IPv4 je čtyřbajtová, skládá se ze čtyř oktetů, tj. 32 bitů. Maximální počet IP adres verze 4 je tedy $2^{32} = 4294967296$ adres.

Jednotlivé IP datagramy se skládají z hlavičky a přenášených dat. Každý řádek IP datagramu má 4 bajty, 0-31 bitů. Hlavička bývá nejčastěji velikosti 20 bajtů, ale může obsahovat volitelné položky, poté bývá větší, než zmiňovaných 20 bajtů. V hlavičce se nachází:

- Verze IP (version) – obsahuje verzi použitého IP protokolu.
- Délka hlavičky (header length) – obsahuje délku hlavičky, délka hlavičky je vždy násobkem 4. V případě volitelné položky se doplní na násobek 4 nevýznamnou výplní.
- Typ služby (type of service) – v praxi tato položka nemá uplatnění. V RFC dokumentech 791 a 1349 lze nalézt návrhy využití. Položka slouží k upřednostnění některých IP datagramů před jinými, k rychlejší odezvě nebo ke zpoždění přenosu.
- Celková délka (total length) – obsahuje celkovou délku IP datagramu. Položka je dvoubajtová, celková velikost IP datagramu může být maximálně 65535 bajtů.
- Identifikace (identification) – obsahuje indentifikaci, kterou vkládá do datagramu operační systém, ze kterého byl datagram odeslán. V případě fragmentace se datagramy patřící k sobě poznají díky tomuto poli, mají nastavené pole identifikace na stejnou hodnotu.
- Příznaky (flags) – toto pole udává, zda je datagram fragmentován či nikoli. Nabývá 3 hodnot: 0, DF (don't fragment – zakazuje fragmentaci), MF (more fragments – signalizuje, zda je posledním fragmentem).
- Posunutí fragmentu (fragment offset) – udává, na jaké pozici začíná fragment v původním datagramu.

- TTL (time to live) – zamezuje nekonečnému šíření datagramu v síti, kde není dostupný příjemce. Na každém směrovači se sníží kladná hodnota TTL o jedna. Pokud TTL nabývá hodnoty 0, datagram je směrovačem zahozen a odesílatel je informován icmp zprávou o vypršení TTL.
- Protokol vyšší vrstvy (protocol) – číselné označení protokolu vyšší vrstvy, kterému je datagram určen.
- Kontrolní součet (header checksum) – kontrolní součet, který se počítá pouze z hlavičky datagramu. Při změně TTL na směrovači se kontrolní součet přepočítává směrovačem, to zvětšuje režii na směrovači
- Adresa odesílatele (source address) – 4 bajtová IP adresa odesílatele.
- Adresa příjemce (destination address) – 4 bajtová IP adresa příjemce.

[30]

Obrázek 2 – IPv4 datagram

Protokol IP verze 6

Nedostatek adresního prostoru u protokolu IPv4 byl znám již v devadesátých letech, kam sahají počátky protokolu IPv6. Protokol je popsán v dokumentu RFC 1883 [16], který vyšel již v roce 1995. Později vyšly další revize protokolu a podpory mobility zařízení. Ta byla dokončena v roce 2004 a revidována v roce 2011 v RFC 6275. Protokol byl navržen tak, aby se předešlo nedostatku adresního prostoru a to navždy. Délka adresy IPv6 je 128 bitů, to znamená, že je k dispozici celkem $3{,}4 \times 10^{38}$ adres. Adresa se skládá ze dvou částí. Prvních 64 bitů určuje síťový prefix a dalších 64 bitů je vyhrazeno pro část hosta. Adresy se dělí do tří skupin:

- Individuální (unicast) – každá tato adresa identifikuje jedno síťové rozhraní.
- Skupinové (multicast) – adresují skupinu zařízení, pokud jsou data adresována této adrese, všechna zařízení skupiny je přijmou.
- Výběrové (anycast) – také označuje skupinu zařízení, data se však pošlou pouze nejbližšímu zařízení ve skupině.

Protokol disponuje autokonfigurací síťového zařízení. Pokud je v síti směrovač s podporovou IPv6, klient vyšle multicast zprávu s dotazem na konfigurační informace. Směrovač poté zašle potřebné informace k nastavení rozhraní. K tomu účelu se využívá protokol Icmp verze 6, který je součástí protokolu IPv6.

Datagram prošel změnami od verze 4. Obsahuje hlavičku a přenášená data. Oproti IPv4 hlavička obsahuje jen ty nejnutnější prvky a hlavička je konstantní délky. Volitelné položky a rozšiřující informace byly přesunuty do rozšířené hlavičky, která nemusí být přítomna v datagramu. Hlavička obsahuje:

- Verze (version) – obsahuje verzi protokolu, v případě IPv6 obsahuje 6.
- Třída Provozu (traffic class) – tato položka slouží k vyjádření priority datagramu nebo k jeho zařazení do přepravní třídy. Úkolem této položky je poskytnout IP služby se zaručenou kvalitou. IPv6 zatím neumí poskytnout parametry jako jsou přenosová rychlost, zpoždění nebo jeho rozptyl. Existují však tzv. diferencované služby, které umožňují zacházet s různými datagramy odlišně. Umožňují prioritní zpracování datagramů před ostatními nebo naopak odložení datagramu až za ostatní.
- Značka toku (flow label) – vyjadřuje, že datagram patří do určitého toku dat společně s dalšími datagramy. To umožňuje směrovači lepší rozhodovaní, jak naložit s datagramem. Tato vlastnost zatím není blíže definována, stále se jedná o experimentální vlastnost.
- Délka dat (payload length) – obsahuje počet bajtů za hlavičkou (hlavička datagramu se do této položky nepočítá). Pokud datagram obsahuje rozšiřující hlavičku, je zahrnuta v této položce.
- Další hlavička (next header) – určuje jaký druh dat nebo jaká hlavička následuje za standartní hlavičkou.
- Maximum skoků (hop limit) – obsahuje počet maximálních průchodů směrovačem. Nahrazuje TTL z IPv4. Na každém směrovači je počet snížen o jedna. Pokud dojde ke snížení až na 0, datagram je zahozen a odesílatel je informován icmp zprávou o vypršení počtu skoků.
- Zdrojová adresa (sources address) – obsahuje 128 bitovou adresu odesílatele
- Cílová adresa (destination address) – obsahuje 128 bitovou adresu příjemce, tato položka společně se zdrojovou adresou zabírají 80% rozsahu celé hlavičky.

[17][30]

Obrázek 3 – IPv6 datagram

2.4 Transportní vrstva

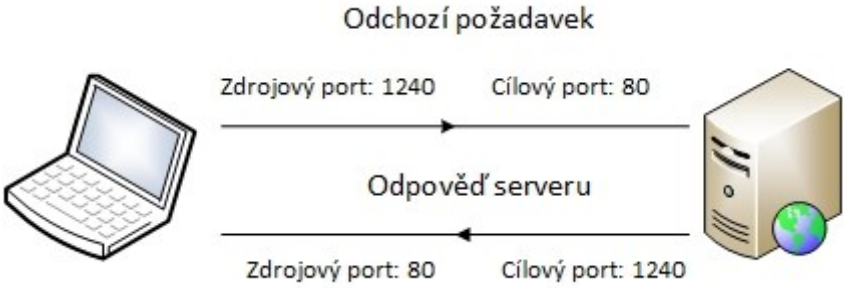
Tato vrstva zajišťuje přenos dat mezi dvěma programy v koncových zařízeních. K tomuto účelu využívá vrstva protokol TCP, který je definován v dokumentu RFC 793 [31] nebo UDP, definovaný v dokumentu RFC 768 [32]. V případě použití protokolu TCP je zaručena spolehlivost přenosu dat, vytvoří se dočasný virtuální okruh po dobu spojení a přenos dat probíhá oběma směry. Každému bajtu na straně odesílatele je přiřazeno pořadové číslo, podle kterého se bajty na straně příjemce sestaví dohromady ve správném pořadí. V případě nedoručení očekávaného bajtu, příjemce oznámí tuto ztrátu. Pro kontrolu správného doručení se využívá kontrolní součet. Transportní vrstva, v případě požadavku může regulovat tok dat oběma směry mezi účastníky přenosu.

Obě strany spojení (klient i server) jsou rozlišeny číslem portu. TCP segment obsahuje položky čísla portu, ty jsou dvoubajtová, mohou nabývat hodnot 0-65535. Rozlišují se porty služby TCP a UDP. Proces s číslem portu 13 protokolu TCP, není ten samý proces, jako proces s číslem portu 13 služby UDP. Na určitém portu může v jednu chvíli komunikovat pouze jedna služba. Porty se dělí obvykle do dvou skupin:

- Dobře známé porty (well known ports) – porty s číslem 1-1023, jsou vyhrazeny pro běžně používané služby, jako jsou http, dns, dhcp, daytime, ssh a další. Port s číslem 0 se ignoruje, tato hodnota je vyhrazena.

- Dočasné porty (epheremal port group) – porty s číslem 1024-65535, porty v tomto rozmezí se generují automaticky. Různé operační systémy je definují odlišně.

Služba na straně serveru naslouchá příchozí komunikaci. Běžně používané serverové služby většinou používají čísla portů v rozmezí 0-1023. Při zahájení komunikace se serverem, operační systém klienta vygeneruje pro komunikaci náhodné číslo portu (jedno z dosud nepřidělených čísel portu v operačním systému) a cílové číslo portu se použije v závislosti na použité službě serveru, ke které se klient hodlá připojit. Server z doručeného paketu zjistí, na jakém portu komunikuje klient a odpověď serveru už směřuje na daný port klienta.



Obrázek 4 – Princip zdrojových a cílových portů

Protokol TCP disponuje spolehlivým doručováním dat, díky tomu je hlavička TCP celkem obsáhlá. Skládá se z:

- Zdrojový port (source port) – číslo portu z odesílající strany paketu.
- Cílový port (destination port) – číslo portu příjemce paketu.
- Pořadové číslo (sequence number) – číslo identifikující segment, díky tomuto poli se v průběhu komunikace může vyžádat opětovné zaslaní chybějících dat.
- Číslo potvrzení (acknowledgement number) – pořadové číslo paketu, které zařízení očekává v komunikaci od druhé strany.
- Délka záhlaví (data offset) – délka záhlaví v násobcích 32 bitů.
- Příznaky (flags) – příznaky identifikující typ odesílaného paketu.
- Velikost okna (window size) – velikost vyrovnávací paměti odesílající strany v bajtech.
- Kontrolní součet (checksum) – slouží k ověření, zda-li nebyl paket porušen.
- Ukazatel naléhavosti (urgent pointer) – pokud byl nastaven příznak URG, v tomto poli jsou další informace o tom, jak se má paket zpracovat.
- Možnosti (options) – volitelné pole s dalšími informacemi.

Pole příznaků obsahuje položky:

- URG – segment nese naléhavá data
- ACK – segment má platné pole pořadové číslo ve všech segmentech kromě prvního
- PSH – segment nese aplikační data, která se mají předat aplikaci
- RST – odmítnutí spojení
- SYN – odesílající strana začíná s novým číslováním segmentů

- FIN – odesílající strana poslala poslední segment, už nehodlá posílat žádné další. Zařízení, které segment přijme, může dál zasílat data. Spojení končí v době, kdy obě strany chtějí ukončit spojení.

[30][31][33]

Obrázek 5 – Záhlaví TCP datagramu

Zahájení komunikace TCP

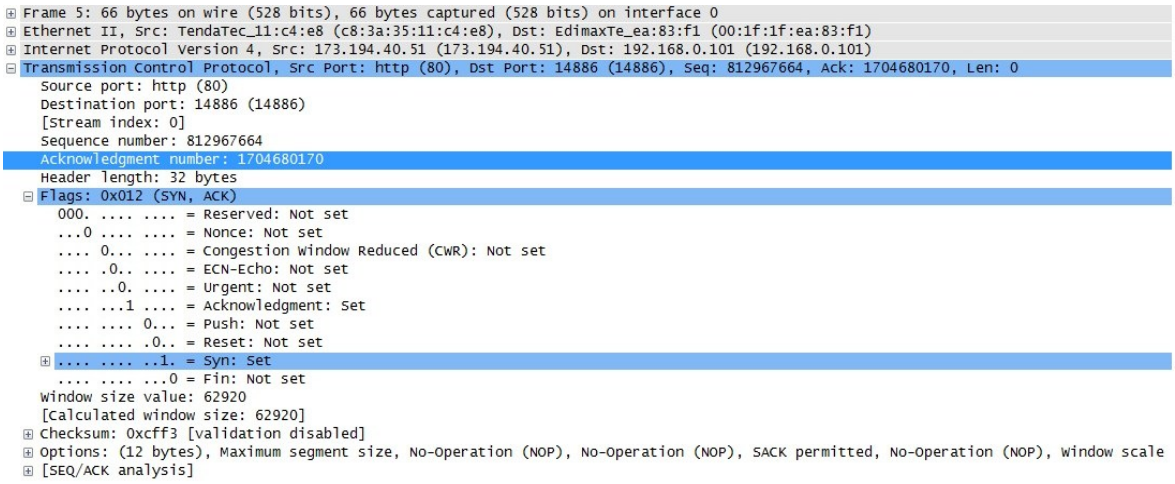
Každá komunikace, v rámci protokolu TCP, začíná trojcestným procesem ověřením mezi účastníky komunikace (three way handshake). Tento proces slouží k ověření, zda je cílový hostitel dostupný, zda cílový hostitel naslouchá na portu, se kterým se snaží klient navázat komunikaci a slouží také k obdržení pořadového čísla segmentu, aby obě strany mohly zachovat správné pořadí datového proudu.

Na začátku komunikace klient odešle TCP paket serveru, tento paket neobsahuje žádná data. V paketu je nastaven příznak SYN a obsahuje pořadové číslo segmentu a maximální velikost segmentu. Operační systém vygeneruje náhodné číslo portu, které klientovi zůstane až do ukončení komunikace. V tomto případě bylo vygenerováno číslo portu 14886. Pořadové číslo segmentu je také generované náhodně, operační systém vybere náhodné číslo v rozsahu 0 až 2^{32}-1.

```
⊞ Frame 4: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
⊞ Ethernet II, Src: EdimaxTe_ea:83:f1 (00:1f:1f:ea:83:f1), Dst: TendaTec_11:c4:e8 (c8:3a:35:11:c4:e8)
⊞ Internet Protocol Version 4, Src: 192.168.0.101 (192.168.0.101), Dst: 173.194.40.51 (173.194.40.51)
⊟ Transmission Control Protocol, Src Port: 14886 (14886), Dst Port: http (80), Seq: 1704680169, Len: 0
    Source port: 14886 (14886)
    Destination port: http (80)
    [Stream index: 0]
    Sequence number: 1704680169
    Header length: 32 bytes
  ⊟ Flags: 0x002 (SYN)
      000. .... .... = Reserved: Not set
      ...0 .... .... = Nonce: Not set
      .... 0... .... = Congestion Window Reduced (CWR): Not set
      .... .0.. .... = ECN-Echo: Not set
      .... ..0. .... = Urgent: Not set
      .... ...0 .... = Acknowledgment: Not set
      .... .... 0... = Push: Not set
      .... .... .0.. = Reset: Not set
    ⊞ .... .... ..1. = Syn: Set
      .... .... ...0 = Fin: Not set
    Window size value: 8192
    [Calculated window size: 8192]
  ⊞ Checksum: 0xbd18 [validation disabled]
  ⊞ Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation (NOP), SACK permitted
```

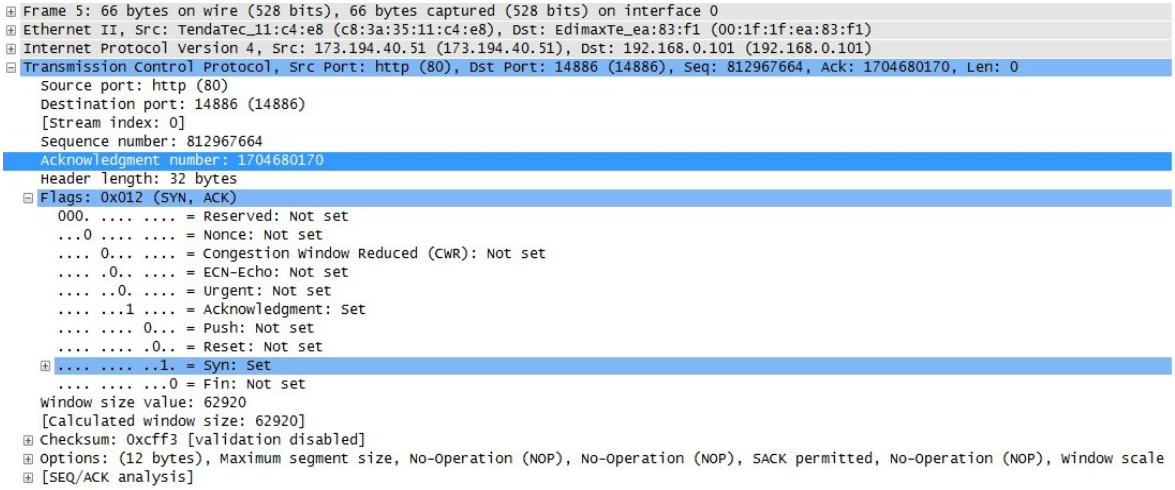
Obrázek 6 – Počáteční SYN paket zaslaný klientem

Server odpovídá paketem s nastaveným příznakem SYN, svým pořadovým číslem segmentu a nastaveným příznakem ACK, který obsahuje číslo segmentu o jedna vyšší, než klient dosud odeslal. Tímto server dává najevo, že následující paket očekává s tímto pořadovým číslem.



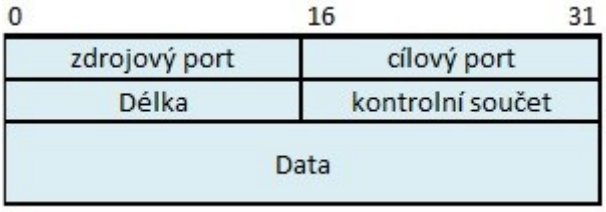
Obrázek 7 – SYN/ACK paket zaslaný serverem

Klient poté zasílá paket pouze s příznakem ACK. Pořadové číslo segmentu je zvětšeno o jedna. Tímto končí proces ustálení komunikace. Obě strany mají potřebné informace a v dalším průběhu komunikace se můžou zasílat aplikační data. Celý proces ověření je k dispozici v přiloženém souboru. [27][30][31]



Obrázek 8 – Poslední fáze ustálení spojení, ACK paket zaslaný klientem

Protokol UDP je nespojovaný protokol, na rozdíl od TCP, se nenavazuje spojení. Odesílatel pošle datagram, ale už není zaručena správnost doručení. Protokol neřeší ztrátu datagramů během přenosu dat. Hlavička protokolu UDP obsahuje pouze nejnutnější informace k přenosu aplikačních dat. Díky tomuto faktu se protokol využívá u služeb, které nevyžadují takovou režii komunikace. [32]



Obrázek 9 – UDP datagram

2.5 Aplikační vrstva

Nejvyšší vrstva TCP/IP modelu představuje jednotlivé aplikační programy (procesy), které slouží k přenosu dat. Procesy využívají protokolů transportní vrstvy. Jednotlivé procesy jsou rozlišeny tzv. portem. Pro síťové spojení aplikace je použito jednoznačné označení portu a použitého protokolu transportní vrstvy. [24]

3 Aplikační protokoly zajišťující konfiguraci a komunikaci

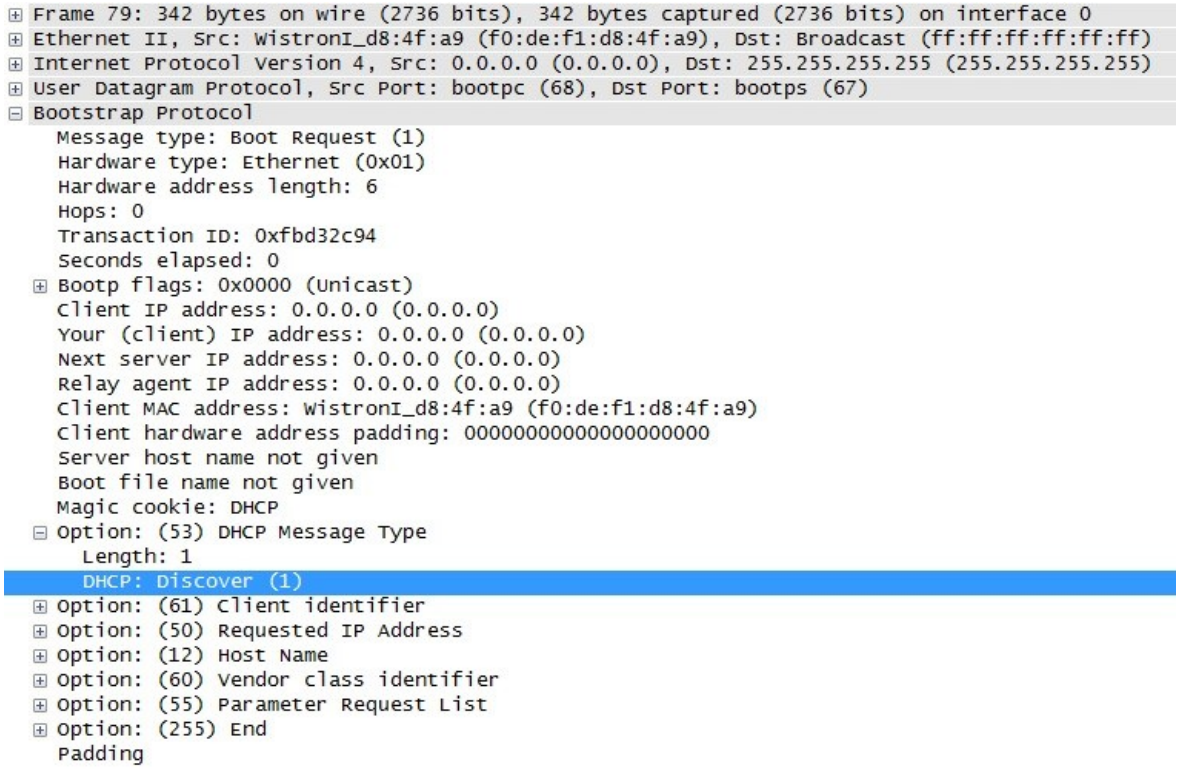
Tato skupina protokolů zajišťuje konfiguraci síťové rozhraní tak, aby zařízení v síti mohla komunikovat s dalšími zařízeními. Bylo by značně nepraktické každé zařízení konfigurovat ručně. Pro tyto účely slouží konfigurační protokoly, které tyto základní informace přidělují na požádání dynamicky v rámci sítě. Každému zařízení je přiřazena IP adresa, maska sítě, adresa brány, která směruje mimo síť, ve které se nachází zařízení a jedna či více adres DNS serverů.

3.1 BOOTP

BOOTP je protokol, který slouží k nastavení základních údajů pro síťové zařízení. Je popsán v RFC 951 [3]. Protokol byl navržen pro zařízení, která nemají vlastní diskový prostor a nemohou uchovávat informace potřebné pro komunikaci po síti. V síti existuje BOOTP server, který má ve své databázi seznam MAC adres zařízení. Server pak na žádost klientů odesílá informace z databáze prostřednictvím UDP protokolu. V zaslané IP konfiguraci je i adresa serveru, ze kterého je možné stáhnout a následně nahrát obraz operačního systému do zařízení. [30]

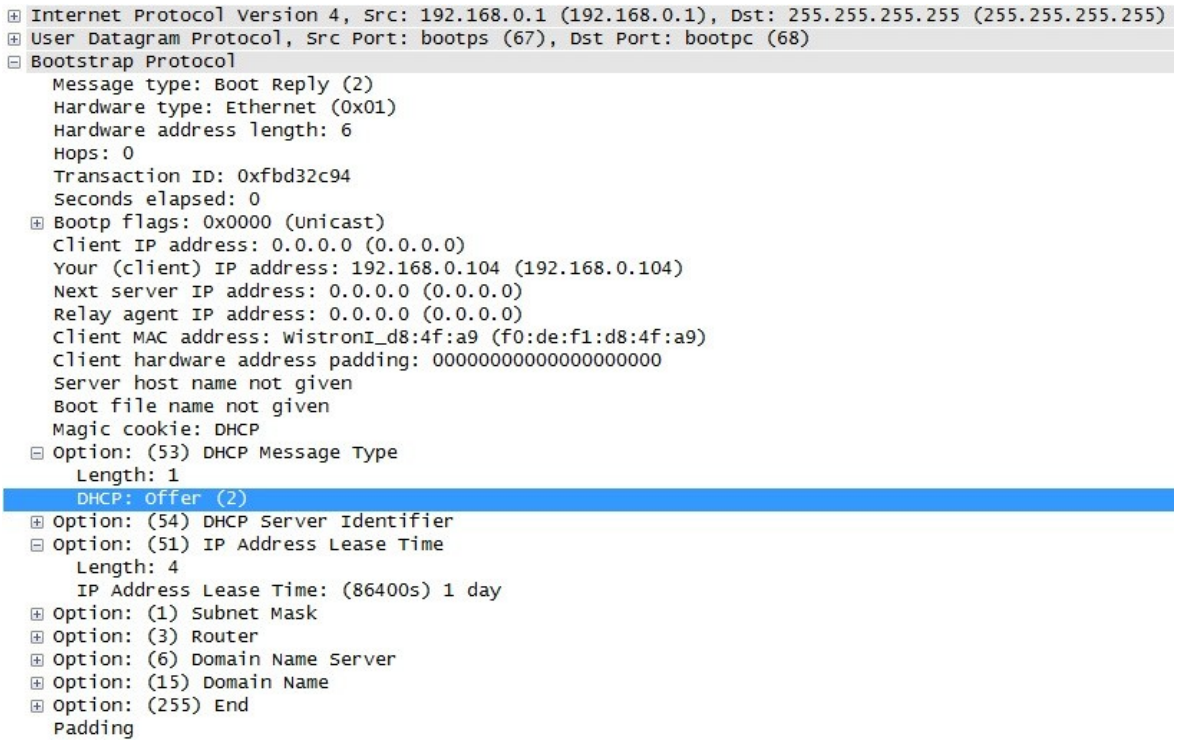
3.2 DHCP

DHCP protokol je rozšíření BOOTP protokolu, je popsán v RFC 2131 [6]. Používá se pro automatické nastavení IP adresy, masky sítě, implicitní brány a DNS serveru zařízením v síti. Platnost údajů je časově omezená, typicky tuto dobu platnosti určuje DHCP server, je však možné ji nastavit manuálně. Server uchovává ve své databázi přidělené IP adresy a jejich dobu platnosti. V zařízeních běží DHCP klient, který se stará o prodloužení platnosti údajů. Pokud DHCP klient po vypršení platnosti nepožádá server o obnovení adresy, server tuto IP adresu může použít pro jiné zařízení, které o ni požádá. V síti existuje jeden či více DHCP serverů sdružených do skupin. Servery naslouchají na UDP portu 67. Klient pak odešle (formou broadcast zprávy – zpráva se odešle na všechna zařízení v dané síti) paket DHCPDISCOVER, který slouží k nalezení DHCP serverů v síti. Tento paket obsahuje fyzickou adresu zařízení a ID transakce, které slouží k identifikaci klienta. Klient zatím nemá přiřazenou IP adresu.



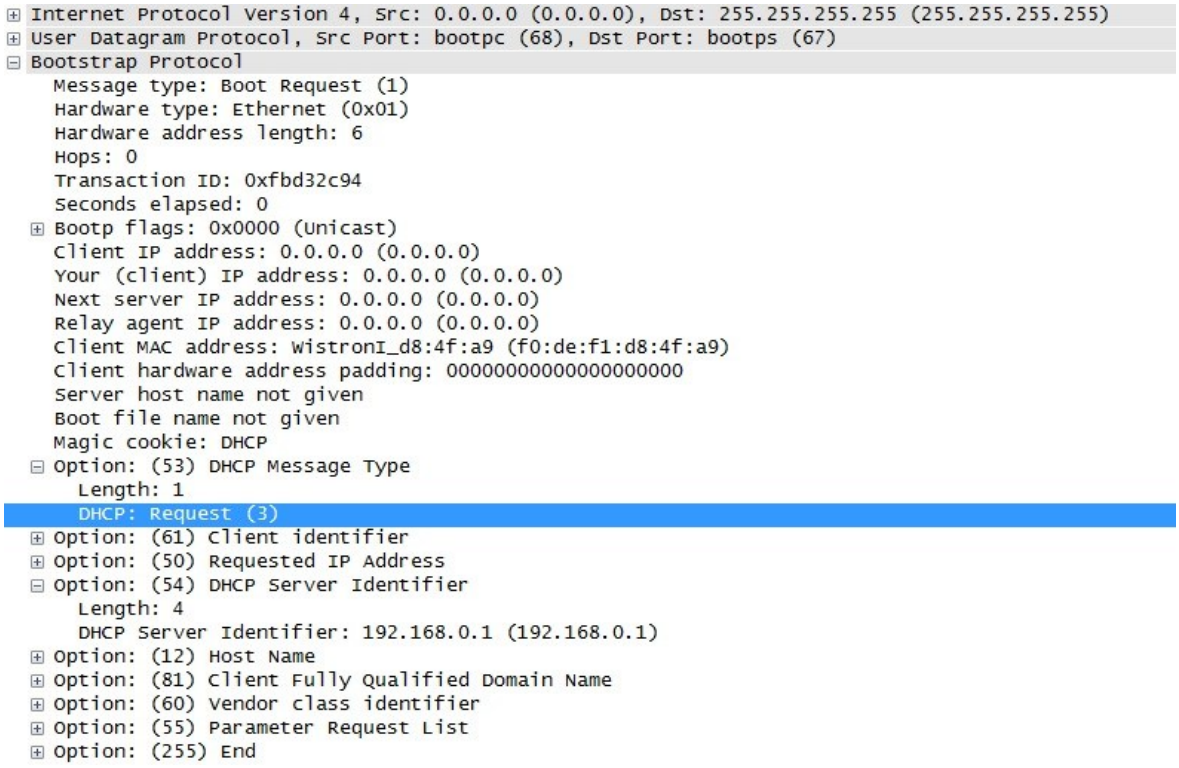
Obrázek 10 – Paket DHCP Discover

DHCP server, který přijme tento paket, jej zpracuje. Server nemusí klientovi nabídnout žádnou IP adresu z několika důvodů. Klientova MAC adresa může být v databázi serveru uvedena jako zakázaná nebo server nemusí mít již žádnou volnou IP adresu. Na DHCPDISCOVER paket odpoví jeden nebo více DHCP serverů, zasláním paketu DHCPOFFER. Pokud byla MAC adresa klienta již přidělena k některé IP adrese, server ji klientovi opět přiřadí, pokud již není přidělená. DHCPOFFER také obsahuje dobu platnosti IP adresy a ID transakce, které je stejné jako ID transakce zaslané klientem, doménové jméno, IP adresu směrovače a masku sítě.



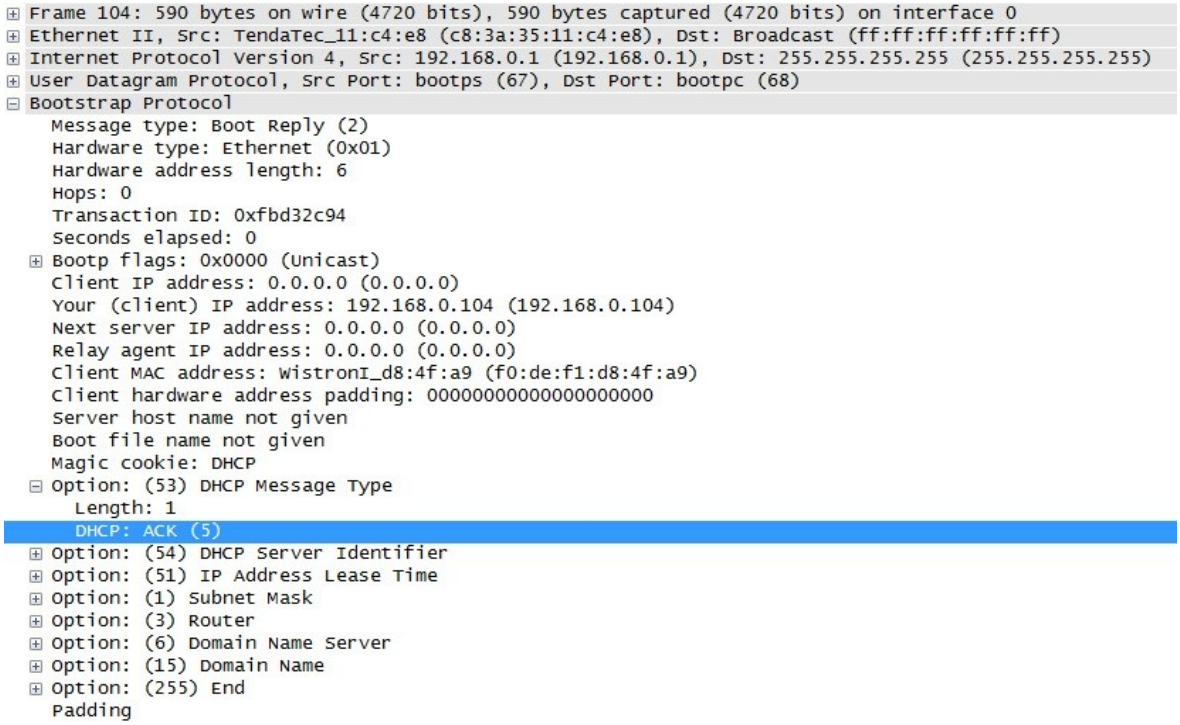
Obrázek 11 – DHCPOFFER paket

V případě obdržení více paketů s nabídkou IP adresy, záleží na nastavení DHCP klienta. Klient může odpovědět na první nabídku nebo čekat na další nabídky a teprve poté si vybrat například podle nejvyšší doby platnosti IP adresy. Poté, co klient zpracuje nabídky, vybere jeden server a zašle mu paket DHCPREQUEST s žádostí o tuto adresu. Paket obsahuje identifikátor serveru a je posílán opět jako broadcast. Ostatní servery tedy zjistí, jakou nabídku si klient vybral.



Obrázek 12 – DHCPREQUEST paket

Server zpracuje request paket a zjistí, jestli do této doby nebyla nabízená adresa přidělena jinému zařízení. Pokud již adresa byla přiřazena jinému zařízení, odpoví klientovi zprávou DHCPNAK, která znamená, že nabízená adresa již není k dispozici. Pokud je adresa k dispozici, server zašle klientovi zprávu DHCPACK a ve své databázi přidělí IP adresu ke klientské MAC adrese.

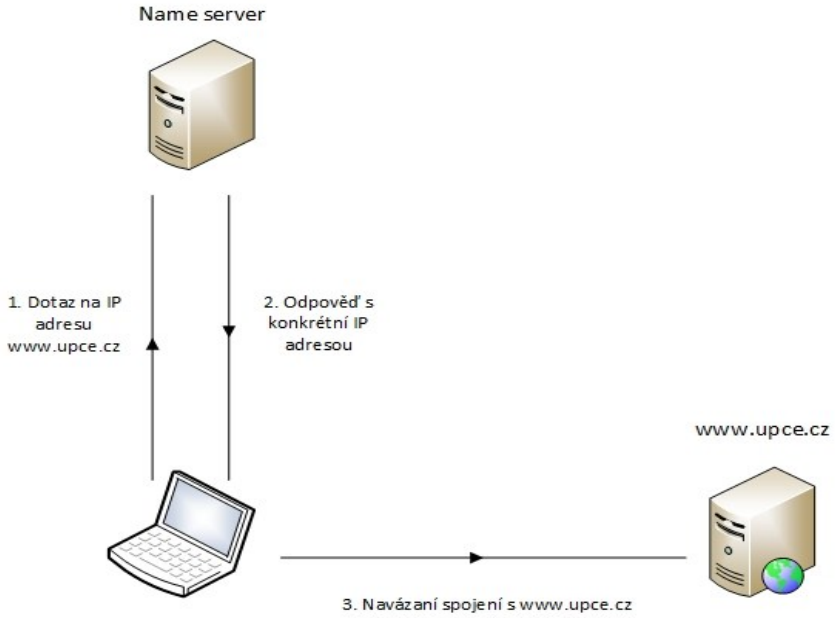


Obrázek 13 – DHCPACK paket

Klient po přijetí DHCPACK ověří, jestli v síti existuje zařízení se stejnou IP adresou vysláním ARP požadavku. Pokud žádné zařízení přidělenou adresu nepoužívá, klient ji začne používat. Pokud by existovalo zařízení se stejnou IP adresou, kterou nabízel server, klient opakuje proces získání adresy od začátku. [4][6][18][30]

3.3 DNS

DNS je systém hierarchických doménových jmen, je popsán v RFC 1035 [5]. DNS slouží k překladu IP adres počítačů na doménová jména, která jsou pro člověka lépe zapamatovatelná. Jména domén jsou uložena v celosvětově distribuované DNS databázi, která je po částech uložena v name serverech.



Obrázek 14 – Překlad doménového jména na IP adresu

Domény se skládají z řetězců oddělených tečkou a jsou hierarchicky rozděleny do několika úrovní, tvoří strom domén. Kořenem stromu je tzv. kořenová doména, značí se tečkou a spadají pod ní domény nejvyšší úrovně (Top Level Domains – TLD). Pro Českou republiku je vyhrazena doména .cz. Další domény, které spadají pod TLD jsou např.: edu, com, net, org. Domény můžou být dále rozděleny na subdomény až do 127 úrovní.

K překladu doménového jména na IP adresu konkrétního počítače slouží komponenta resolver. Při každém požadavku překladu, resolver požádá name server o IP adresu. Pokud name server nemá požadovanou adresu ve své databázi, obrátí se s dotazem na nadřazený name server a akce se opakuje, dokud není nalezen hledaný záznam nebo se zjistí, že daný záznam neexistuje.

DNS využívá služeb protokolu TCP i UDP na portu 53. Pro překlad adres se využívá protokol UDP. V případě že by odpověď DNS byla větší než 512 bajtů, do odpovědi se vloží pouze část nepřesahující 512 bajtů a v záhlaví se nastaví TC (truncated) bit, specifikující, že se nejedná o úplnou odpověď. Klient si může vyžádat celou odpověď prostřednictvím protokolu TCP. DNS paket se skládá z 5 částí:

- Hlavička (header)
- Dotaz (question)
- Odpověď (answer)
- Autoritativní name servery (authority)
- Doplňující informace (additional)

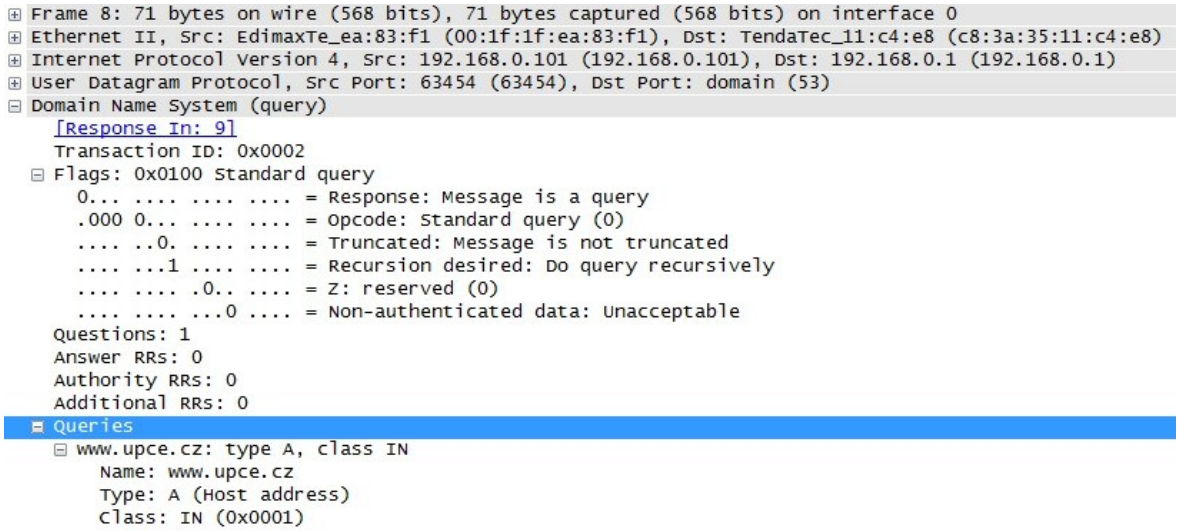
Před zachycením komunikace byla vyčištěna cache paměť DNS resolveru. V operačních systémech Windows k tomuto účelu slouží příkaz:

```
Ipconfig /flushdns
```

V operačním systému Ubuntu lze použít např. příkaz:

```
sudo /etc/init.d/dns-clean start
```

Pro dotaz na překlad doménového jméno serveru byl použit nástroj NsLookUp, který je dostupný v mnoha operačních systémech. Po vyčištění paměti a provedení dotazu na překlad doménového jména www.upce.cz příkazem *nslookup www.upce.cz*. Klient zašle paket DNSQUERY, ve kterém požaduje rekurzivně získat IP adresu.



Obrázek 15 – DNS dotaz na překlad doménového jména

Paket, obsahující odpověď na dotaz, obsahuje kromě hlavičky a zopakovaného dotazu ještě části odpovědí (answers), autoritativní name servery (authoritative nameservers) a dodatečné záznamy (additional records). Část autoritativní name servery obsahuje jména autoritativních serverů, uvedených v záznamu name serveru a část dodatečných informací, která obsahuje většinou IP adresy uvedených autoritativních name serverů. [28][30][33]

Typ dotazu	Hodnota (desítkově)	Význam
A	1	Požadavek na získání IP adresy verze 4
NS	2	Požadavek na získání autoritativních name serverů
CNAME	5	Požadavek na získání věty CNAME
SOA	6	Požadavek na získání věty SOA
WKS	11	Požadavek na získání věty WKS
PTR	12	Požadavek na získání PTR věty
HINFO	13	Požadavek na získání HINFO věty
MX	15	Požadavek na získání věty MX
TXT	16	Požadavek na získání TXT věty
SIG	24	Požadavek na získání věty SIG
KEY	25	Požadavek na získání věty KEY
NXT	30	Požadavek na získání věty NXT
AAAA	28	Požadavek na získání IP adresy verze 6
SRV	33	Požadavek na získání věty SRV
AXFR	252	Požadavek na získání transferu celé zony
IXFR		Požadavek na získání inkremetálního one transferu
*	255	Požadavek na získání všech vět

Obrázek 16 – Hodnoty a typy požadavků na DNS server

```
⊞ Frame 9: 198 bytes on wire (1584 bits), 198 bytes captured (1584 bits) on interface 0
⊞ Ethernet II, Src: TendaTec_11:c4:e8 (c8:3a:35:11:c4:e8), Dst: EdimaxTe_ea:83:f1 (00:1f:1f:ea:83:f1)
⊞ Internet Protocol Version 4, Src: 192.168.0.1 (192.168.0.1), Dst: 192.168.0.101 (192.168.0.101)
⊞ User Datagram Protocol, Src Port: domain (53), Dst Port: 63454 (63454)
⊟ Domain Name System (response)
    [Request In: 8]
    [Time: 0.095658000 seconds]
    Transaction ID: 0x0002
  ⊞ Flags: 0x8180 Standard query response, No error
    Questions: 1
    Answer RRs: 2
    Authority RRs: 2
    Additional RRs: 2
  ⊞ Queries
  ⊟ Answers
    ⊞ www.upce.cz: type CNAME, class IN, cname cms-proxy.upce.cz
    ⊟ cms-proxy.upce.cz: type A, class IN, addr 195.113.124.185
        Name: cms-proxy.upce.cz
        Type: A (Host address)
        Class: IN (0x0001)
        Time to live: 1 minute
        Data length: 4
        Addr: 195.113.124.185 (195.113.124.185)
  ⊞ Authoritative nameservers
  ⊞ Additional records
```

Obrázek 17 – DNS response paket s IP adresou doménového jména

3.4 HTTP

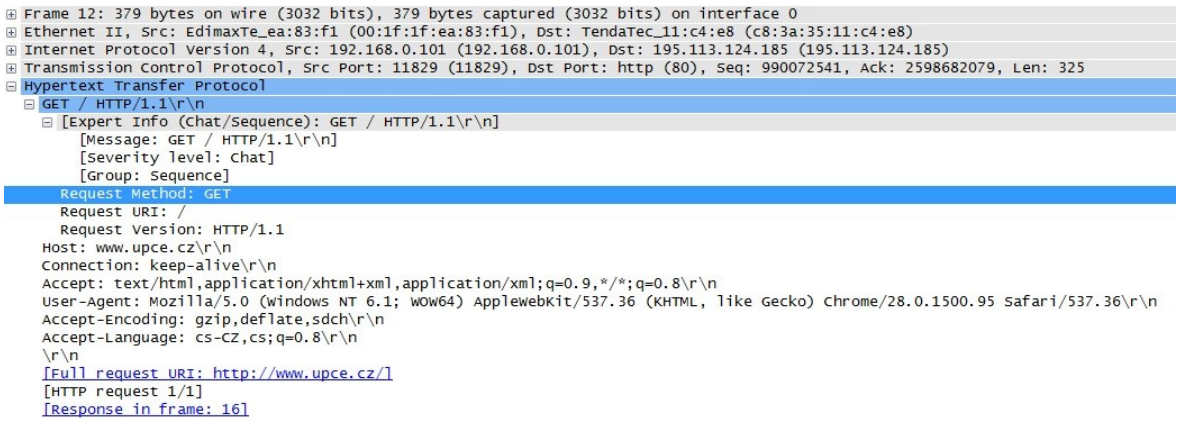
HTTP protokol slouží k výměně hypertextových dokumentů ve formátu html. Aktuálně používaná verze protokolu 1.1 je popsána v RFC 2616 [10]. Tato verze přináší možnost perzistentního spojení. Spojení není ukončeno po vykonání odpovědi serveru, ale server

ještě čeká na další dotazy od klienta. Klient se po obdržení odpovědi rozhodne, jestli spojení ukončí sám nebo zašle serveru další dotazy. Protokol funguje na principu klient/server. Server naslouchá nejčastěji na portu 80 a čeká na požadavky klientů. Komunikaci navazuje klient, odesílá požadavky na konkrétní dokument. Požadavek se skládá z použité metody, URI, verzi protokolu a hlavičky. Existuje několik metod, které se používají pro dotaz na server a to:

- GET – metoda pro zpřístupnění požadovaných dat ze serveru
- POST – metoda pro zaslání dat na server, např. uživatelská data
- HEAD – identická metoda jako GET, nevrací se však tělo zprávy, pouze hlavička
- PUT/DELETE – metoda nahraje/smaže data v serveru
- OPTION – metoda vrací podporované metody serveru
- TRACE – metoda se používá k zjištění cíle požadavku na aplikační vrstvě, koncový příjemce vrací zprávu zpět odesílateli, to je možné použít pro diagnostiku
- CONNECT – metoda použitá při komunikaci skrze proxy, k nastavení komunikace přes SSL

Protokol HTTP je bezestavový, nedokáže uchovávat informace o stavu komunikace. Tato skutečnost je v některých případech nepříjemná. Při komunikaci klienta se serverem v internetovém obchodě je potřeba uchovávat informace o připojených klientech a jejich vzájemné rozlišení. K tomuto účelu slouží HTTP Cookies, o které byl protokol rozšířen. [18][30]

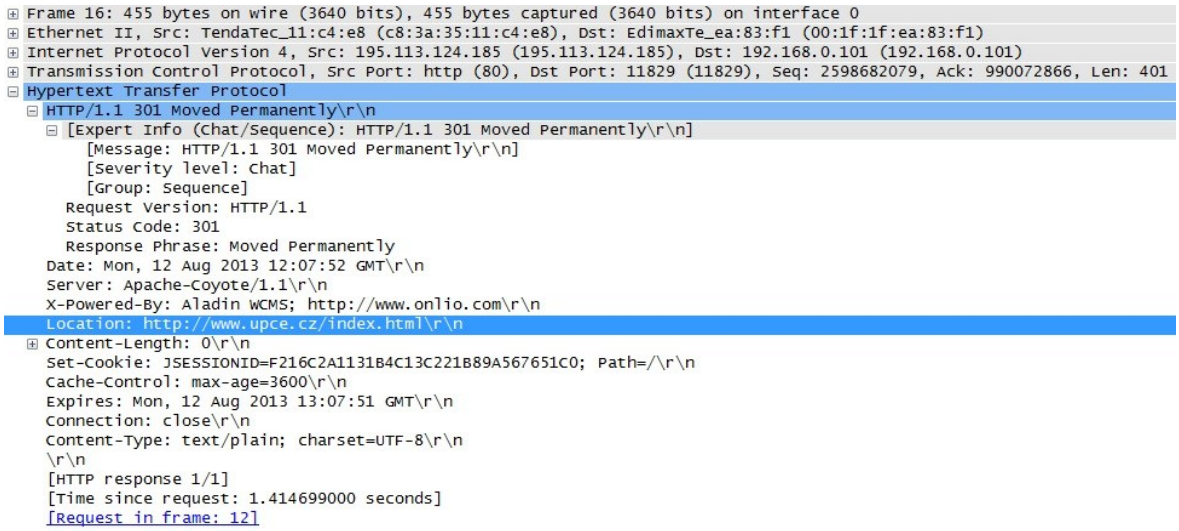
Na začátku komunikace, po ustálení TCP spojení se serverem, klient odešle požadavek pro získání stránky ze serveru. V tomto případě se dotazuje metodou GET webového serveru www.upce.cz. Požadavek obsahuje také informace o prohlížeči, ze kterého byl požadavek odeslán, jaký formát odpovědi klient podporuje, jaký jazyk klient podporuje a typ komprese, kterou klient podporuje.



Obrázek 18 – HTTP požadavek/metoda GET

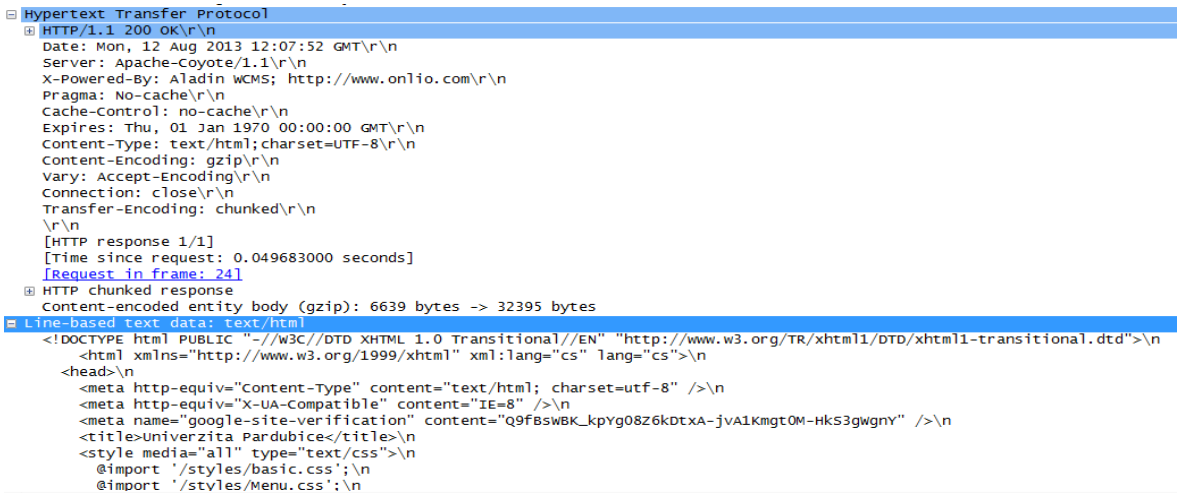
Server tento požadavek zpracuje a zašle klientovi odpověď. Pokud byl požadavek úspěšně zpracován, server zašle data, které si klient vyžádal. V opačném případě informuje klienta odpovědí se stavovým kódem, který blíže specifikuje důvod, proč nebyl požadavek

úspěšně zpracován. Veškeré typy odpovědí jsou popsány v dokumentu RFC 2616. V zachycené komunikaci server zaslal klientovi odpověď se stavovým kódem 301, která znamená, že požadovaná stránka byla přesunuta. Odpověď serveru obsahuje nové umístění, kde se požadovaná stránka nachází.



Obrázek 19 – HTTP odpověď 301 s lokací požadované stránky

Klient zašle nový požadavek GET, který směruje na nově získanou lokaci stránek. Server poté vyhodnotí požadavek jako úspěšný a zašle klientovi požadovanou stránku se stavovým kódem 200, který znamená, že je vše v pořádku. Celá stránka se nachází v sekci Line-based text data.



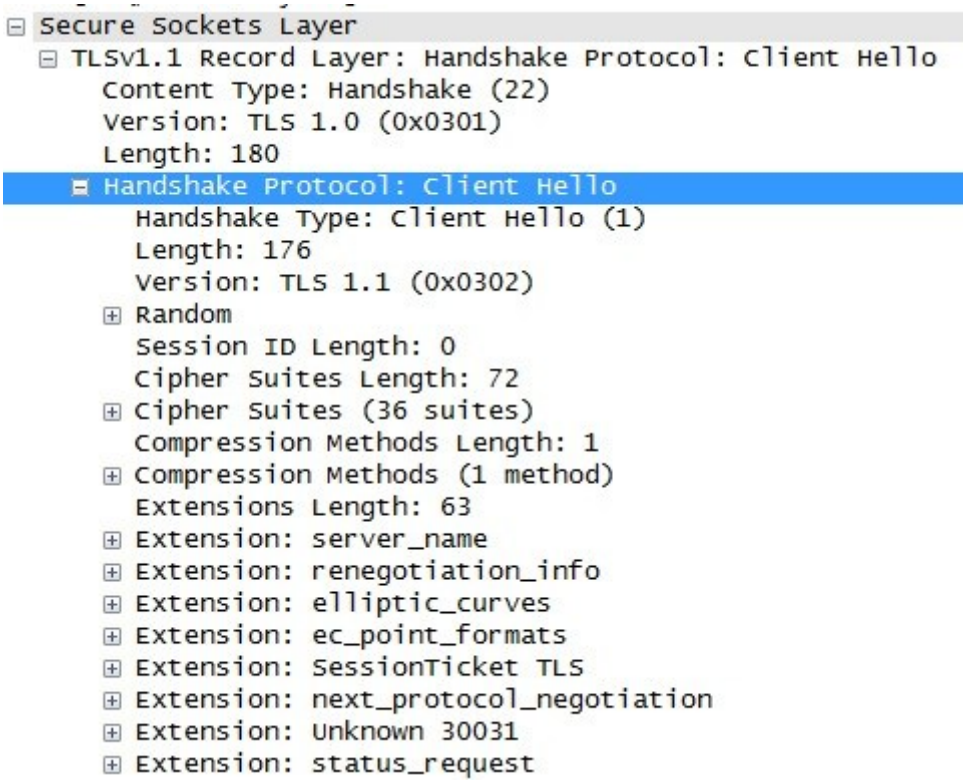
Obrázek 20 – HTTP odpověď 200 s požadovanou stránkou

3.5 HTTPS

HTTPS využívá protokol HTTP a umožňuje zabezpečit spojení a ověřit identitu mezi webovým serverem a webovým prohlížečem pomocí protokolu SSL nebo TLS. Standartní port na straně serveru pro HTTPS je port 443. HTTPS je popsáno v RFC 2818 [9]. Protokol využívá asymetrického šifrování. Webový server i webový prohlížeč si

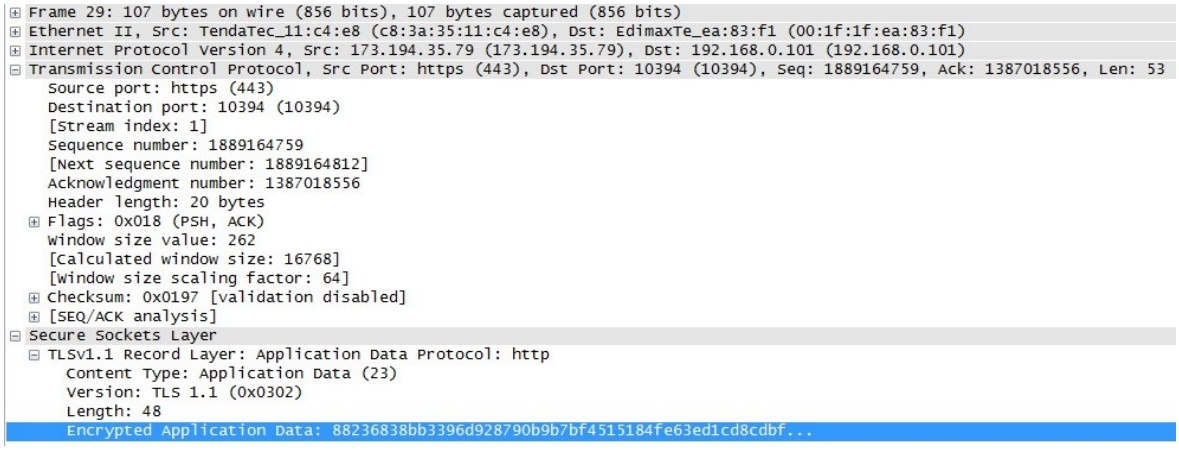
vygenerují dvojici klíčů, veřejný a soukromý. Veřejný klíč může vlastnit kdokoli, naopak soukromý klíč je pečlivě chráněn, před kýmkoli jiným, kromě autora klíče. Existují důvěryhodné certifikační autority, které vydávají klíče a ručí za jejich správnost. Některé známé certifikační autority mají své veřejné klíče již implementované do webových prohlížečů (např. VeriSign, ValiCert, WISeKey).

Při zahájení komunikace klient odešle požadavek ke komunikaci, přes zabezpečené spojení, pomocí SSL/TLS. V požadavku jsou informace o nastavení šifrování, verzi SSL/TLS. V tomto případě se použije TLS verze 1.1.



Obrázek 21 – TLS Hello paket

Server odpovídá na požadavek stejnými informacemi, v odpovědi odešle i svůj certifikát, který obsahuje veřejný klíč serveru. Klient ověří identitu serveru na základě přijatého certifikátu a vygeneruje náhodný symetrický klíč. Tento vygenerovaný klíč zašifruje pomocí veřejného klíče obdrženého v certifikátu serveru a odešle ho zpět serveru. Server rozšifruje přijatý klíč svým soukromým klíčem. Obě strany v této fázi disponují symetrickým klíčem a celá komunikace odteď probíhá šifrovaně, pomocí symetrického klíče. [30]

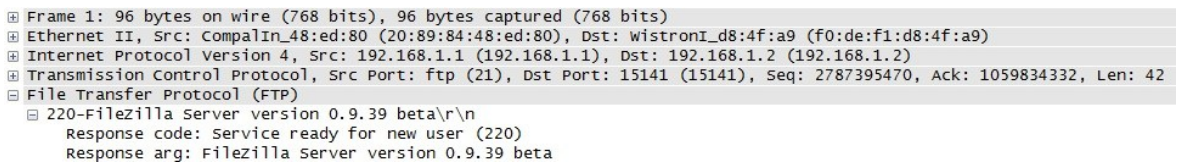


Obrázek 22 – Šifrovaná data v průběhu komunikace

3.6 FTP

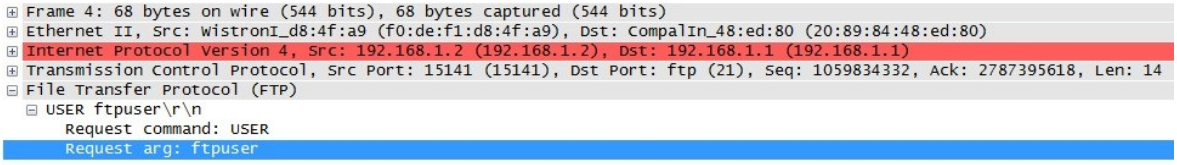
FTP protokol slouží k přenosu souborů mezi dvěma počítači v rámci počítačové sítě. FTP protokol je popsán v RFC 959 [7] a jeho pozdější rozšíření, které definuje bezpečnostní doplňky, je popsáno v RFC 2228 [8]. FTP funguje na principu klient/server. Klienti se připojují na server, kde získávají nebo ukládají svá data. Protokol je interaktivní, může rozlišovat klienty dle přihlašovacího jména a hesla. Klient může procházet adresáře, měnit jejich strukturu v závislosti na přidělených právech. FTP využívá pro svou komunikaci porty TCP/20 a TCP/21. Port 21 slouží k řízení přenosu a výměně příkazů mezi serverem a klientem. Port 20 je určen k přenosu souborů. Při přenosu souborů je komunikace prováděna pouze na portu 20, port 21 je v tu dobu nevyužit. To je nevýhoda při přenosu velkých souborů. FTP ve své základní verzi posílá všechny příkazy a soubory v textové podobě. To je bezpečnostní riziko vzhledem k tomu, že kdokoli může odchytit komunikaci mezi serverem a klientem a získat zasílané heslo. Pro tento případ je vhodnější používat Secure FTP, kdy je využito protokolu SSH. Další možností je použít FTPS, kde se využívá podpory protokolu SSL případně TLS. [30]

Po ustálení TCP spojení mezi klientem a serverem, klient obdrží zprávu se stavovým kódem 220, který znamená, že server je připraven ke komunikaci a klient se může přihlásit pomocí jména a hesla nebo jako anonymní uživatel, pokud to nastavení serveru dovoluje.



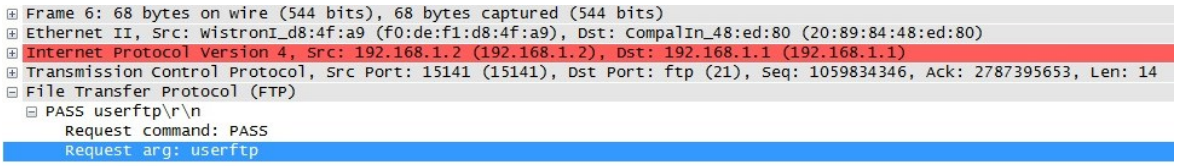
Obrázek 23 – FTP 220 odpověď

Následuje fáze autentizace, kdy uživatel zasílá serveru své uživatelské jméno a heslo. Samotný protokol FTP nevyužívá žádné zabezpečující mechanismy. Uživatelské jméno i heslo může být odchyceno během komunikace. Oba tyto údaje budou ukázány níže. Uživatelské jméno se posílá serveru příkazem USER.



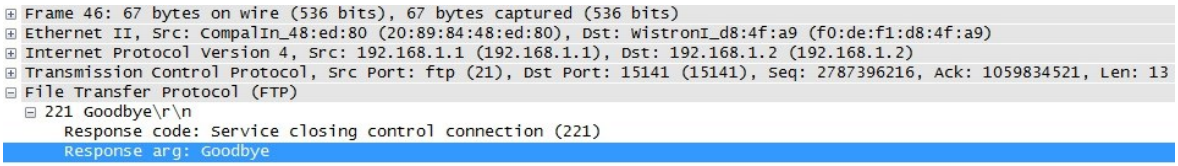
Obrázek 24 – FTP příkaz USER a zachycené uživatelské jméno

Server po přijetí paketu se jménem uživatele zasílá klientovi zprávu se stavovým kódem 331, která říká klientovi, aby zaslal své heslo. Heslo se zasílá příkazem PASS.



Obrázek 25 – FTP příkaz PASS a zachycené heslo

Po úspěšném přihlášení, může dojít k zahájení přenosu dat nebo dalších řídících příkazů protokolů FTP. Spojení ukončuje klient zasláním příkazu QUIT. Po přijetí příkazu server ukončuje spojení a zasílá klientovi zprávu se stavovým kódem 221.



Obrázek 26 – FTP odpověď 221

Touto zprávou končí komunikace mezi klientem a serverem.

4 Aplikační protokoly zajišťující komunikaci elektronické pošty

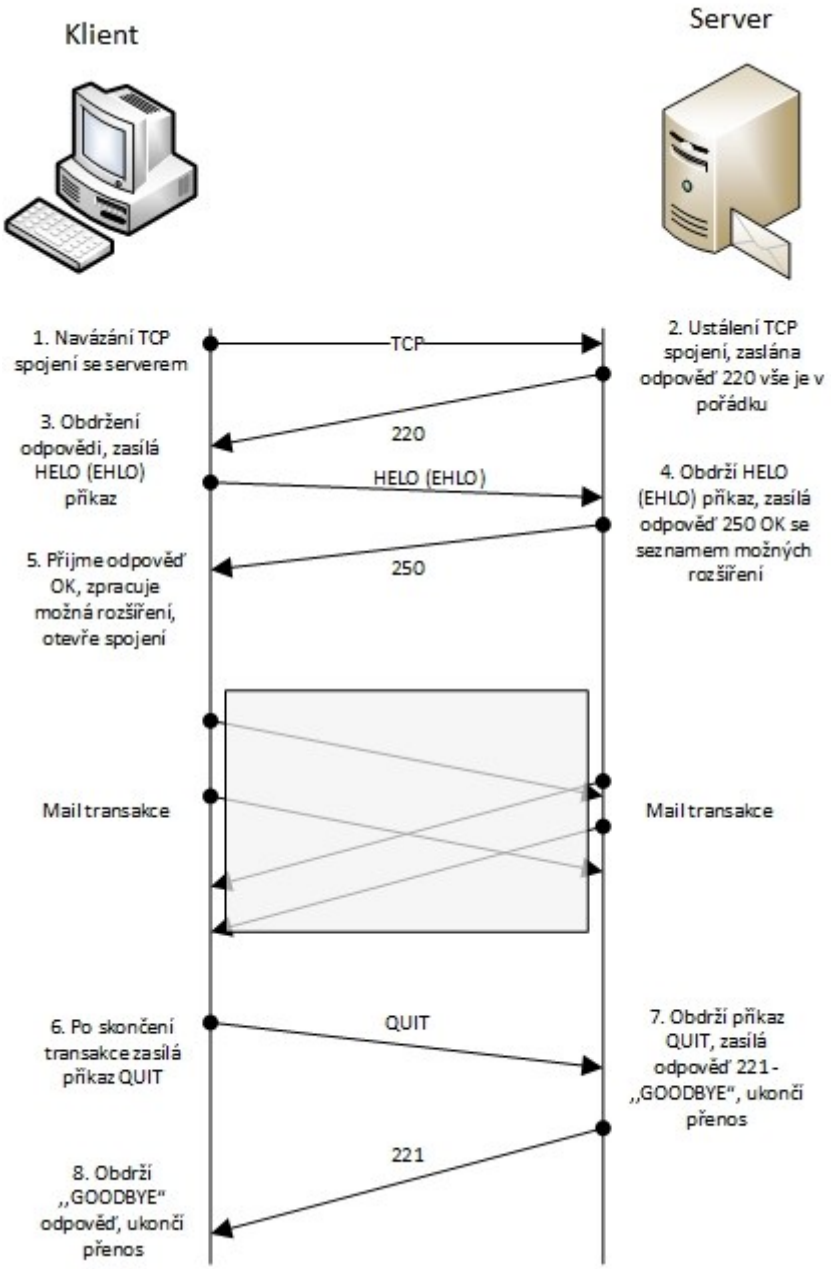
V počátku formování protokolů TCP/IP byla potřeba přenosu krátkých textových zpráv, dnes označovaných názvem „e-mail“. Samotná komunikace pomocí krátkých textových zpráv je starší než Internet samotný. Elektronická pošta vznikala v době počátků sítě ARPANET, tedy před vznikem Internetu samotného. K přenosu elektronické pošty slouží protokol SMTP. Protokol se využívá všude tam, kde se používají protokoly TCP/IP. Může to být Internet jako celosvětová globální síť, tak i internet jako soustava sítí. SMTP může být nasazeno i tam, kde se nevyužívají protokoly TCP/IP, ale jiné síťové a transportní protokoly. Existují i jiné koncepce elektronické pošty. V modelu ISO/OSI se využívá platforma X.400, na které je založen poštovní server MS Exchange od firmy Microsoft nebo Mail602 od firmy Software602. Každá platforma využívá vlastní mechanismy pro odesílání a přijímání zpráv a samotnou formu zprávy. Vzájemně jsou mezi sebou nekompatibilní. Pro kombinování dvou různých koncepcí je potřeba zavést poštovní brány, které zaručí potřebný převod mezi dvěma platformami (příklady poštovních bran mohou být např.: X.400-SMTP, SMTP-Mail602 a další). [30]

Pro účely této práce byly vytvořeny dva účty, ukazka-POP3@seznam.cz a ukazka-IMAP@centrum.cz

4.1 SMTP

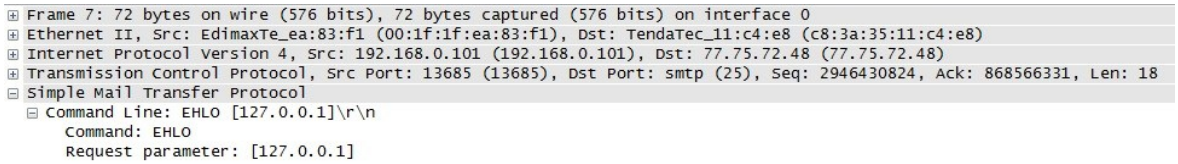
SMTP je protokol, který se používá k přenosu e-mailových zpráv od klienta na server, k přenosu se využívá služby TCP a portu 25. Protokol byl původně publikován ve formě doporučení RFC 788 [23] v listopadu roku 1981. V roce 1982 vyšlo rozšíření protokolu popsané v RFC 822 [26], které definuje formát zprávy. V roce 2001 vyšla další aktualizace popsaná v RFC 2821 [22].

Zpočátku protokol sloužil k přenosu čistě textových zpráv ve formě ASCII kódu. Zprávu tvořily 7 bitové ASCII znaky. SMTP říká, jak se mají tyto zprávy přenášet skrz přenosové cesty, které jsou 8 bitové. 7 bitové kódy znaků se zarovnají směrem k nižším binárním řádům a nejvyšší bit se nastaví na nulu. V případě přenosu 8 bitových znaků (text s českou diakritikou a jiné typy příloh netextového formátu) se zpráva přetransformuje do podoby ASCII textu a korektně se přenese k příjemci. Na straně příjemce dochází k inverzní transformaci zpět do původní podoby zprávy. SMTP zpráva se skládá ze dvou částí. První část obsahuje příjemce a odesílatele zprávy, druhou část tvoří samotné tělo zprávy, které obsahuje text nebo jiné přílohy. Přenášené zprávy jsou chápány jako textové a členěné na jednotlivé řádky pomocí znaků CR a LF. Komunikace mezi příjemcem a odesílatelem je ve formě dialogu, kde se využívá několik metod, kterými protokol disponuje.



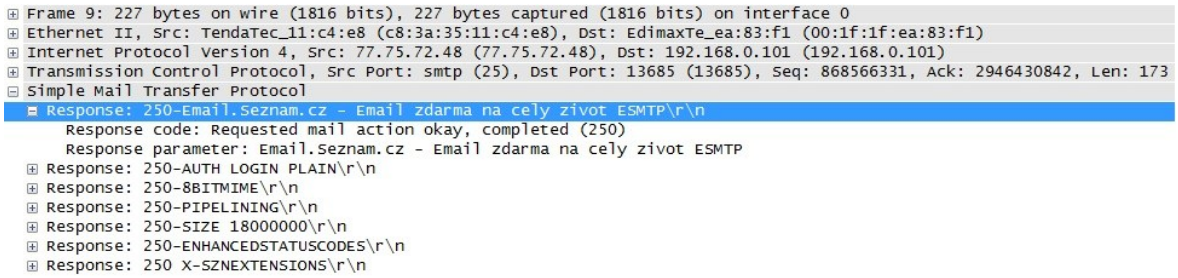
Obrázek 27 – Ukázka navázání a ukončení spojení

Na začátku komunikace si obě strany vymění údaje o své připravenosti přijímat poštu a informace o odesílateli a příjemci zprávy, poté i samotná data zprávy. Klient navazuje spojení se serverem, který naslouchá na portu 25. Odpovědi od serveru jsou čistě číselného formátu v podobě trojmístných desítkových číslic, přenášených ve formě textu. Po obdržení uvítací zprávy od serveru s kódem 220, klient zasílá požadavek HELO, který označuje klientovu identitu.



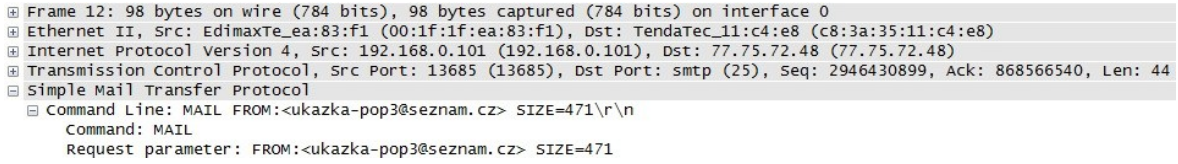
Obrázek 28 – EHLO příkaz

V rozšířené verzi ESMTP se využívá zpráva EHLO, po jejímž obdržení server poskytuje svá rozšíření, která má k dispozici a klient jej může použít. Na příkaz HELO ovšem žádná rozšíření nevrátí. Pokud je server ochoten přijmout zprávu, vrací stavový kód 250.

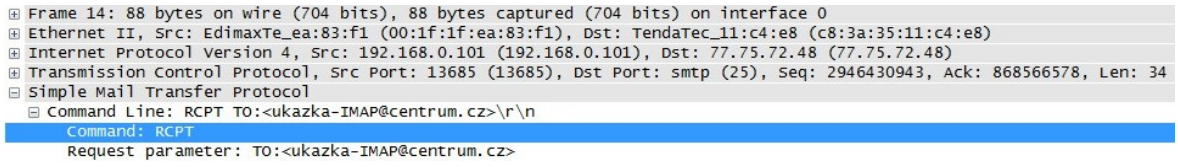


Obrázek 29 – Stavový kód 250 a rozšíření v odpovědi serveru

Poté následuje příkaz MAIL, který určuje informace o odesílateli zprávy a jeden či více příkazů RCPT TO, které určují informace o příjemci, či příjemcích zprávy.

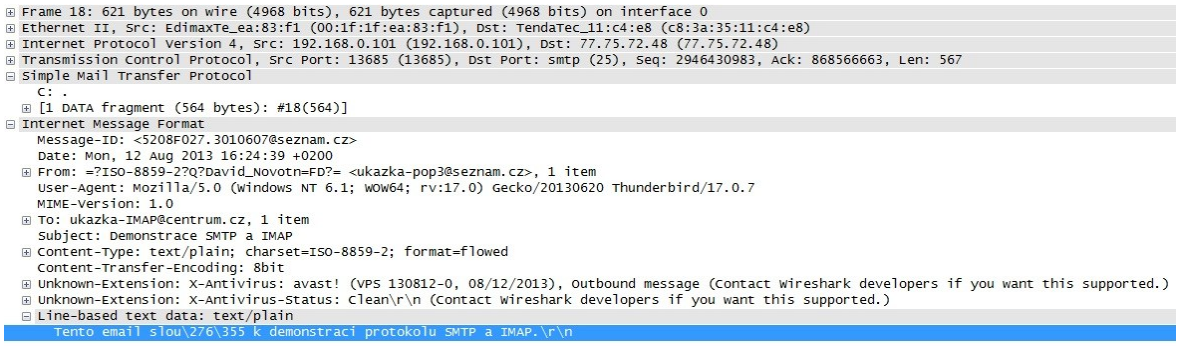


Obrázek 30 – SMTP příkaz MAIL



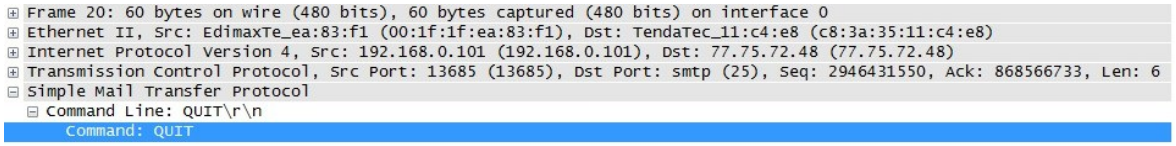
Obrázek 31 – SMTP příkaz RCPT

Poté již následuje přenos samotné zprávy, k tomu slouží příkaz DATA. Zpráva je tvořena dvěma částmi, hlavička a tělo. Obě části se přenášejí po řádcích a celá zpráva je ukončena jedním znakem a to tečkou. V případě výskytu takovéhoto znaku v těle zprávy, se zdvojuje uvozující tečka.



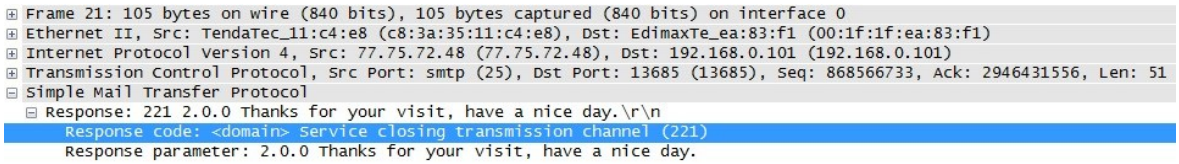
Obrázek 32 – tělo zprávy

Po úspěšném přenosu zprávy následuje kladná odpověď od serveru se stavovým kódem 250. Jakmile klient provede všechny transakce, zasílá příkaz QUIT, kterým dává najevo serveru, že chce ukončit spojení.



Obrázek 33 – SMTP příkaz QUIT

Server po přijetí příkazu QUIT odpovídá zprávou se stavovým kódem 221 a uzavírá spojení s klientem.



Obrázek 34 – SMTP odpověď 221

Kód	Význam
211	System status, or system help reply
214	Help message
220	<domain> Service ready
221	<domain> Service closing transmission channel
250	Requested mail action okay, completed
251	User not local; will forward to <forward-path>
354	Start mail input; end with <CRLF>.<CRLF>
421	<domain> Service not available, closing transmission channel
450	Requested mail action not taken: mailbox unavailable
451	Requested action aborted: local error in processing
452	Requested action not taken: insufficient system storage
500	Syntax error, command unrecognised
501	Syntax error in parameters or arguments
502	Command not implemented
503	Bad sequence of commands
504	Command parameter not implemented
521	<domain> does not accept mail
550	Requested action not taken: mailbox unavailable
551	User not local; please try <forward-path>
552	Requested mail action aborted: exceeded storage allocation
553	Requested action not taken: mailbox name not allowed
554	Transaction failed

Obrázek 35 – Ukázka číselných kódů a jejich významů

Pro výběr cesty zprávy k příjemci, do správné domény, skrze počítačovou síť slouží MX záznam v DNS databázi. Těchto záznamů může být v databázi více. Jednotlivé záznamy mají číselnou hodnotu, která určuje prioritu záznamu. V případě nedosažitelnosti příslušného MX záznamu se využije další záznam s nižší prioritou. [30]

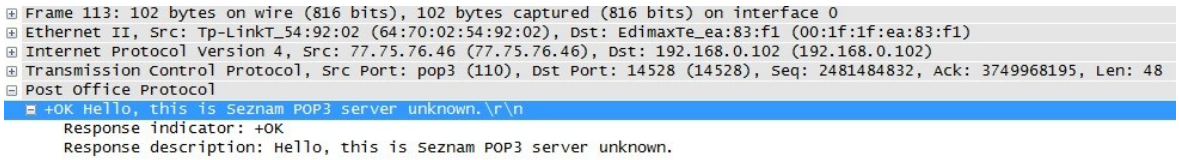
4.2 POP

POP byl navržen jako jednoduchý protokol pro získání pošty ze schránky na SMTP serveru. Protokol pracuje v offline režimu, klient ze serveru získá veškerou poštu do svého zařízení, kde s ní pracuje lokálně, už nemusí být připojen k síti. Původní POP verze 1 je popsána na pouhých pěti stránkách a obsahuje pouze základní sadu příkazů pro ověření identity klienta pomocí uživatelského jména a hesla a následného získání pošty ze serveru. V roce 1985 byla vydána druhá verze protokolu. POP verze 2, je popsána v RFC 937 [19]. Tato verze obsahovala bohatší sadu příkazů a odpovědí, mezi které patřilo např. čtení pouze některých zpráv. V roce 1988 byla vydána třetí verze protokolu POP verze 3, která byla původně popsána v RFC 1081 [20] a následně byla vyvíjena až do roku 1996, kde byl protokol naposled standardizován. Protokol byl zpočátku nezabezpečený a neumožňoval spojení zabezpečit, hesla se přenášela nešifrovaná. Díky tomuto faktu vyšly další RFC dokumenty, které popisují implementaci zabezpečujících mechanismů. Protokol v současné době využívá zašifrovaného spojení skrze SSL nebo novějšího TLS, případně MD5 hash funkce pro přenos hesla.

POP je charakteristický modelem klient/server. Protokol využívá služeb TCP, server naslouchá příchozí komunikaci na běžně známém portu 110. POP obsahuje několik příkazů, které se zasílají v rámci komunikace ve formě ASCII znaků. Formát odpovědi, na rozdíl od SMTP není číselný, ale také formou textu ASCII znaků. Rozlišují se pouze dvě odpovědi a to:

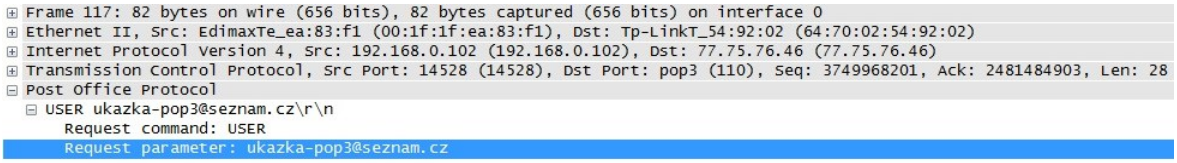
- +OK – pozitivní odpověď, zaslaná v případě úspěšného vykonání akce
- -ERR – negativní odpověď, zaslaná v případě chyby během vykonávání akce

Zahájení komunikace mezi serverem a klientem začíná odesláním požadavku klienta na server a ustálení TCP spojení. Server odpoví uvítací zprávou, která značí, že je server připraven ke komunikaci.



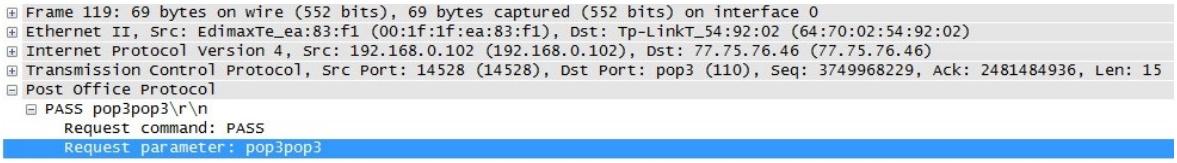
Obrázek 36 – POP3 +OK zpráva

Následuje fáze autentizace, kdy klient prokazuje totožnost uživatele, který se snaží přistoupit ke schránce na serveru. K identifikaci uživatele slouží uživatelské jméno nebo e-mailová adresa a heslo. Pro demonstraci autentizace uživatele nebylo nastaveno zabezpečené spojení. V přiložené souboru je zachycena komunikace s uživatelským jménem i heslem v nezašifrované podobě.



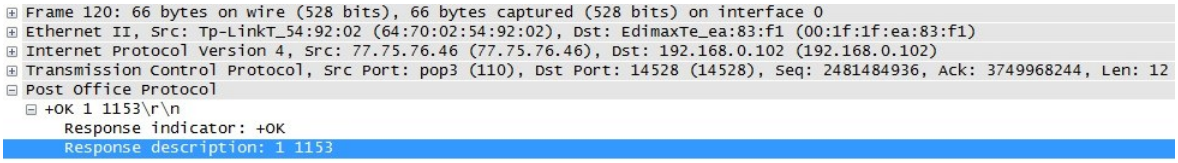
Obrázek 37 – POP3 příkaz USER a uživatelské jméno

Po zpracování uživatelského jména, server odešle +OK zprávu s požadavkem, ve kterém říká, aby klient zaslal heslo.

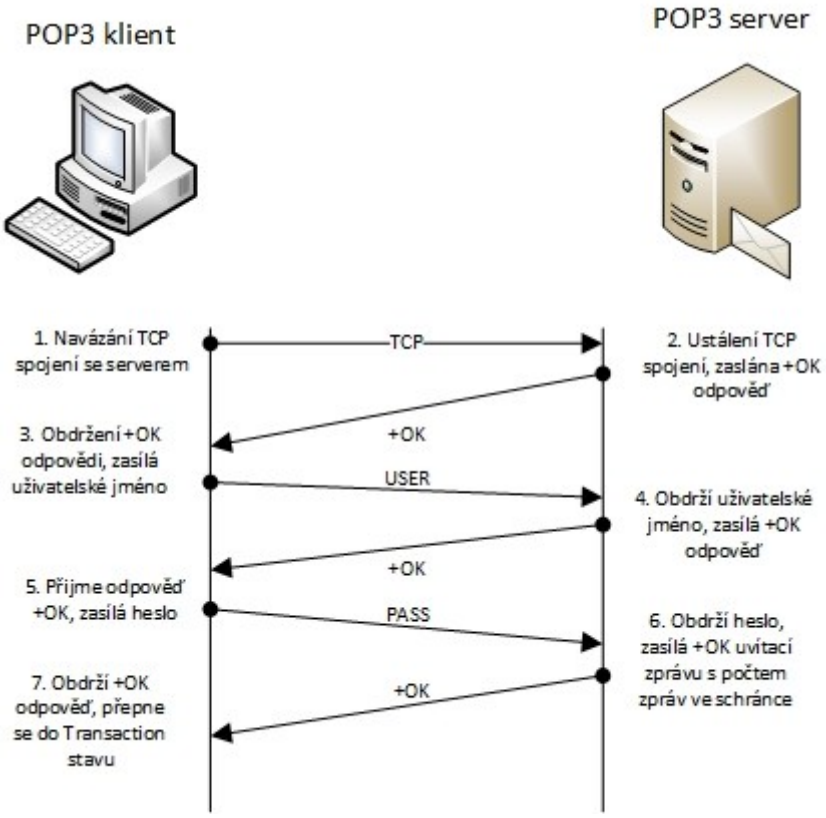


Obrázek 38 – POP3 nešifrovaná podoba hesla

Při úspěšné autentizaci server odpovídá kódem +OK a sděluje klientovi i počet zpráv, čekajících v jeho schránce společně s velikostí zpráv. V zachyceném paketu schránka obsahuje pouze jednu zprávu s celkovou velikostí 1153 bajtů.



Obrázek 39 – +OK zpráva s počtem a velikostí zpráv



Obrázek 40 – Proces autentizace uživatele

Po úspěšné autentizaci klienta následuje fáze transakce. V této fázi dochází ke stažení zpráv ze serveru do zařízení klienta. Většina příkazů definovaných v protokolu POP 3 se využívá právě v této fázi komunikace. Klient zjistí počet zpráv a velikost dat, které obsahují zprávy v bajtech, umístěné ve schránce příkazem STAT.

```
⊞ Frame 121: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
⊞ Ethernet II, Src: EdimaxTe_ea:83:f1 (00:1f:1f:ea:83:f1), Dst: Tp-LinkT_54:92:02 (64:70:02:54:92:02)
⊞ Internet Protocol Version 4, Src: 192.168.0.102 (192.168.0.102), Dst: 77.75.76.46 (77.75.76.46)
⊞ Transmission Control Protocol, Src Port: 14528 (14528), Dst Port: pop3 (110), Seq: 3749968244, Ack: 2481484948, Len: 6
⊟ Post Office Protocol
  ⊟ STAT\r\n
      Request command: STAT
```

Obrázek 41 – POP3 příkaz STAT

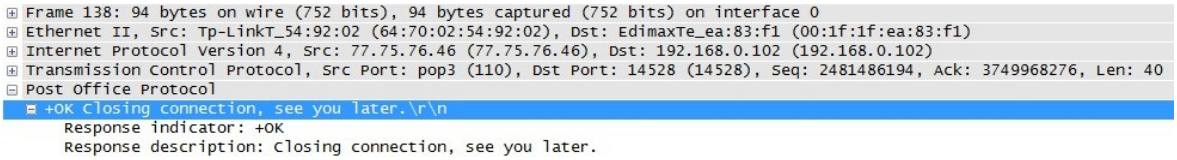
Poté klient odešle příkaz LIST, na který server vrací počet zpráv, které budou poslány klientovi. Následuje příkaz RETR pro získání první zprávy ze serveru.

```
⊞ Frame 133: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface 0
⊞ Ethernet II, Src: EdimaxTe_ea:83:f1 (00:1f:1f:ea:83:f1), Dst: Tp-LinkT_54:92:02 (64:70:02:54:92:02)
⊞ Internet Protocol Version 4, Src: 192.168.0.102 (192.168.0.102), Dst: 77.75.76.46 (77.75.76.46)
⊞ Transmission Control Protocol, Src Port: 14528 (14528), Dst Port: pop3 (110), Seq: 3749968262, Ack: 2481485003, Len: 8
⊟ Post Office Protocol
  ⊟ RETR 1\r\n
      Request command: RETR
      Request parameter: 1
```

Obrázek 42 – POP3 příkaz RETR

Při úspěšném získání zprávy, se zpráva označí příkazem DELE. Tato posloupnost příkazů RETR/DELE se provede k získání všech zpráv ze serveru. Jakmile klient získá všechny zprávy, zasílá příkaz QUIT, signalizující přání skončit s komunikací. Po přijetí příkazu

QUIT, server odstraní všechny zprávy, které byly klientem označeny ke smazání. Nedojde-li k žádnému problému během odstraňování zpráv ze serveru, server zasílá potvrzení o úspěšném odstranění zpráv a informaci o tom, že spojení se schyluje ke konci.



Obrázek 43 – POP3 odpověď serveru a ukončení spojení

Získání všech zpráv ze serveru může trvat pár sekund, ale i několik minut. Tato skutečnost je ovlivněna velikostí zpráv a rychlosti připojení. Protokol umožňuje připojení pouze jednoho klienta ke schránce v jednom okamžiku a po získání všech zpráv ze serveru je klient odpojen.

V současné době patří tento protokol mezi nejpoužívanější protokoly pro práci s poštou, vzhledem k jeho jednoduchosti a nenáročnosti na implementaci. Ovšem jednoduchost tohoto protokolu nemusí být ve všech případech vyhovující a měla za následek vznik jiných a sofistikovanějších protokolů jako je protokol IMAP. [18][30]

4.3 IMAP

IMAP byl navržen pro vzdálenou správu pošty. Na rozdíl od protokolu POP3 pracuje jak v režimu offline, tak i v režimu online a disconnected. První, formálně uznaný standard protokolu IMAP verze 2 byl popsán v RFC 1064 [12] roku 1988. V roce 1991 byla vytvořena nová verze IMAP verze 3, popsána v RFC 1203 [13]. Ta však nebyla trhem přijata a používalo se stále IMAP verze 2. V roce 1994 vyšla dosud poslední verze protokolu IMAP verze 4, která je popsána ve dvou dokumentech. V RFC 1730 popisuje protokol samotný a RFC 1731 [11] popisuje autentizační mechanismy protokolu. Aktuálně používaná verze protokolu je označována IMAP4rev1, popsána v RFC 2060 [15] a rozšířena v RFC 3501 [14].

Protokol umožňuje připojení více klientům k jedné schránce. Všechny změny, které klienti udělají v rámci jedné schránky, jsou viditelné. Protokol definuje příznaky, díky nim je možné uchovávat informace, které zprávy již byly přečtené, na které bylo odpovězeno, nebo které byly modifikovány. Tyto příznaky jsou uchovány na serveru a jsou synchronizovány mezi připojenými klienty k jedné schránce. IMAP také implementuje mechanismus, díky kterému je možné vyhledávat zprávy přímo na serveru, bez nutnosti získáni zpráv do klientova zařízení.

IMAP jako většina ostatních transportních protokolů, pracuje v režimu klient/server. Server naslouchá na dobře známém portu 143. Během komunikace skrze protokol IMAP se rozlišují 4 stavy a to:

1) Not Authenticated State – klient naváže spojení se serverem, zatím neproběhla autentizace klienta

2) Authenticated State – byla dokončena autentizace klienta. Klient nyní může provádět operace nad schránkami, ale před pracováním s jednotlivými zprávami, musí zvolit jednu schránku.

3) Selected State – v tomto režimu má klient vybrán aktuální schránku se zprávami, ve které provádí operace v rámci zvolené schránky. Po dokončení požadovaných operací může spojení ukončit nebo se vrátit do stavu Authenticated a vybrat jinou schránku pro zpracování operací.

4) Logout State – do tohoto stavu se klient může dostat z kteréhokoli předchozího stavu vydáním příkazu LOGOUT nebo po delší době, kdy je komunikace neaktivní a server nepřijímá žádné požadavky.

Při zahájení komunikace se serverem, server určuje, v jaké fázi komunikace se klient nachází. Ve většině případů se bude jednat o Not Authenticated State, ale v některých případech, server už zná identitu klienta. Klient mohl být autentizován pomocí externích mechanismů, které nejsou součástí protokolu IMAP. V takovém případě bude aktuální fáze komunikace v Authenticated State. Příkazy a odpovědi jsou stejně jako u protokolu POP zasílány ve formě ASCII znaků a ukončeny sekvencí CRLF.

IMAP využívá systém označování příkazů a odpovědí. Před každý příkaz se přidává unikátní příznak k explicitnímu označení. Příznaky jsou ve formě řetězců s rostoucím číslem. IMAP definuje první příkaz příznakem a0001, druhý a0002 a u dalších příkazů se postupně zvyšuje hodnota. Pokud server zasílá odpověď, je označena odpovídajícím příznakem příkazu, kterého se odpověď týká. Ne všechny odpovědi jsou označeny příznakem.

Kódy odpovědí od serveru se dělí na dvě kategorie.

Result Codes

- OK – pozitivní odpověď na vykonaný příkaz od klienta, zasílána s příznakem příkazu, na který odpovídá. Při zahájení komunikace je zaslána bez příznaku, jako uvítací zpráva serveru.

- NO – negativní odpověď na příkaz. V případě, že je opatřena příznakem, značí selhání prováděného příkazu. Pokud je bez příznaku, server hlásí obecné varování z jeho strany.

- BAD – označuje chybovou zprávu. Společně s příznakem značí chybu vykonaného příkazu.

- PREAUTH – zasílána bez příznaku. Na začátku komunikace označuje klienta, že není potřeba jeho autentizace a přesouvá komunikaci do fáze Authenticated.

- BYE – zasílána před ukončením komunikace. Je bez příznaku a odpovídá na příkaz LOGOUT signalizující ukončení spojení od klienta.

Response Codes

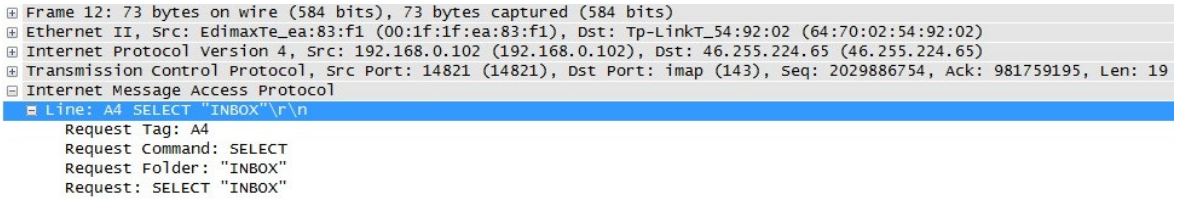
Tyto odpovědi slouží k předání většího množství informací klientovi. Zahrnují textovou informaci, popisující stav komunikace.

- ALERT – výstražná zpráva, informující o něčem důležitém
- BADCHARSET – zaslána v případě použité nepodporované znakové sady
- CAPABILITY – seznam možností serveru, může být zaslána v uvítací zprávě
- PARSE – zaslána v případě chyby, během rozebírání MIME hlavičky
- PERMANENTFLAGS – zasílá seznam příznaků zpráv, které klient může trvale změnit
- READ-ONLY – signalizuje, že schránka je pouze pro čtení
- READ-WRITE – signalizuje, že schránka je v režimu pro čtení a zápis
- TRYCREATE – zaslána v případě neexistující schránky
- UIDNEXT – vyžaduje další identifikátor, který očekává, v podobě desítkového čísla. Identifikátory jednoznačně rozlišují zaslané zprávy.
- UIDVALIDITY – zaslána s desítkovým číslem, označujícím identifikátor platnosti, který slouží k potvrzení jedinečnosti zprávy
- UNSEEN – říká klientovi, že zpráva která je označena, nebyla dosud přečtena

Po navázání spojení se serverem a ustálení spojení skrze TCP, server ve většině případů neví nic o identitě klienta. Tato fáze slouží pouze pro autentizaci klienta a přesun do fáze Authenticated State, klient se v této fázi musí autentizovat. K tomuto účelu slouží tři příkazy a to:

- LOGIN – v parametru příkazu se zasílá uživatelské jméno a heslo ve formě textu. Vzhledem k tomu, v jaké formě se údaje zasílají, tato metoda není bezpečná.
- AUTHENTICATE – v parametru příkazu se specifikuje, jaký mechanismus hodlá klient využít k autentizaci
- STARTTLS – bez parametrů, klient chce využít TLS metodu k autentizaci

Ve fázi Authenticated State, klient může pracovat pouze s jednotlivými schránkami a nemá přístup ke zprávám. V této fázi klient specifikuje určitou schránku, se kterou hodlá dále pracovat. V jeden čas během spojení může klient pracovat pouze s jednou schránkou, tu určuje pomocí příkazů SELECT nebo EXAMINE.



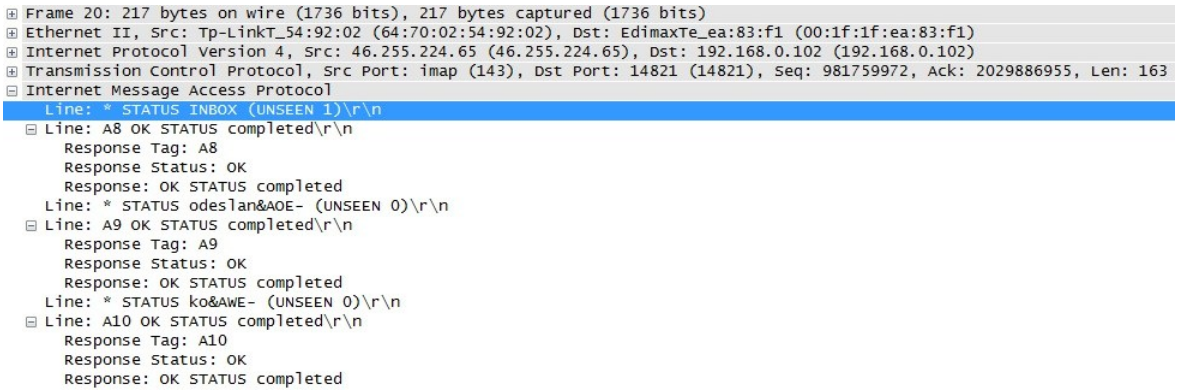
Obrázek 44 – Příkaz SELECT pro výběr schránky

Po výběru schránky se komunikace přesouvá do stavu Selected State a klient může vydávat příkazy pro práci s jednotlivými zprávami. Tato fáze trvá tak dlouho, dokud klient vydává příkazy sloužící k manipulaci se zprávami. Tuto fázi mohou ukončit tři akce.

1. Klient nehodlá provádět další změny a zašle příkaz LOGOUT k ukončení spojení.

2. Klient zašle CLOSE příkaz, který znamená ukončení práce s vybranou schránkou, ale neukončí spojení mezi serverem. Fáze komunikace se přesune do stavu Authenticated a dosavadní schránka je zavřena.

3. Klient vybere novou schránku pomocí příkazů SELECT nebo EXAMINE. Dosud otevřená schránka se automaticky uzavře a otevře se nová vybraná schránka. Stav komunikace zůstává ve fázi Selected State. [28][30]



Obrázek 45 – Informace o počtu nepřečtených zpráv ve schránce

5 Aplikační protokoly ke vzdálené konfiguraci a komunikaci

Tato skupina transportních protokolů dovoluje klientovi interaktivně přistupovat k jiným zařízením skrze počítačovou síť a vzdáleně na nich vykonávat příkazy. Tyto protokoly nejsou využívány běžnými uživateli, ale jsou nesmírně nápomocné pro administrátory, kteří mohou pracovat na vzdálených zařízeních bez nutnosti fyzického přístupu.

5.1 Telnet protokol

Protokol byl navržen vědci v době, kdy standartní protokolová sada TCP/IP ještě nebyla ve formě, v jaké je dnes. Hlavním důvodem byl problém přístupu k jinému počítači a práci na něm, bez toho aniž by uživatel musel být fyzicky u daného počítače. Telnet byl poprvé představen roku 1969 v síti ARPANET. Původní verze protokolu byla definována v RFC 97 [1] roku 1971. Protokol byl stále vyvíjen a bylo vydáno několik dalších RFC dokumentů, poukazujících na možné implementace a vylepšení. Finální verze protokolu byla publikována v RFC 854 [29] v roce 1983. Od té doby vyšlo ještě několik RFC dokumentů zabývajících se možnostmi nastavení a mechanismů pro autentizaci uživatele.

Vzhledem k potřebě přenášet příkazy z jednoho počítače k druhému tak, aby příkazy zůstaly stejné na obou počítačích i přes různorodost zpracování a reprezentace dat na odlišných zařízeních a v odlišných operačních systémech, byl protokol vytvořen na bázi tří základních koncepcí.

Network Virtual Terminal (NVT)

NVT je virtuální terminál, sloužící k univerzální komunikaci mezi klientem a serverem. Telnet klient převede příkaz z původní podoby do formy NVT a odešle na server. Server příkaz převede z NVT do podoby, kterou server používá. Opačným procesem probíhá komunikace ze serveru ke klientovi. Tímto je zaručena vzájemná kompatibilita mezi různorodými zařízeními.

Možnosti a možnost dohodnotí

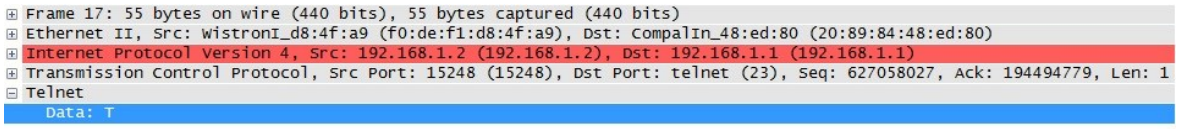
V případě, že jedno ze zařízení při komunikaci hodlá používat pokročilejší služby a příkazy, které druhé zařízení nerozpozná, Telnet definuje sadu mechanismů, které slouží k vzájemné dohodě mezi oběma zařízeními. Pokud nedojde ke sjednocení použitých příkazů, komunikace přejde do základního režimu, kde se využijí pouze základní metody a příkazy.

Souměrné funkce

Telnet funguje v režimu klient/server, během komunikace obě strany můžou vysílat a přijímat data zároveň.

Protokol využívá služeb TCP a server naslouchá na dobře známém portu 23. Po navázání spojení, je spojení udržováno po celou dobu komunikace. Tato doba může být i v řádech několika dnů. Zpočátku spojení se serveru předají data v podobě uživatelského jména a

hesla. Data se zasílají v nešifrované podobě. Každý znak je přenášen samostatně. Celá komunikace je znázorněna v přiloženém souboru.



Obrázek 46 – První znak zachyceného hesla

Telnet zasílá data v podobě 7 bitových US ASCII znaků. Díky jeho možnostem nastavení je však možné v komunikaci použít i jinou znakovou reprezentaci. Při zasílání jednotlivých znaků bylo potřeba sjednotit používání kombinací, které se liší v závislosti na použitém operačním systému. V unixových systémech kombinace kláves CTRL + C přeruší daný program, ale v systémech typu Windows tato kombinace zkopíruje data do schránky. Tyto situace jsou řešeny univerzální sadou kódů, která je definována protokolem samotným. Příkaz se na straně klienta převede do definovaného Telnet kódu a na straně serveru přeloží do podoby, která odpovídá použitému operačnímu systému. Tyto příkazy jsou zasílány ve stejném proudu, jako jsou zasílány data. Jsou reprezentovány speciální hodnotou bytu v rozsahu 240 – 254. U příkazů, v proudu dat, předchází ukončovací znak, který rozlišuje samotná data od příkazu. Tento znak je označován jako Interpret As Command (IAC). [30]

5.2 SSH

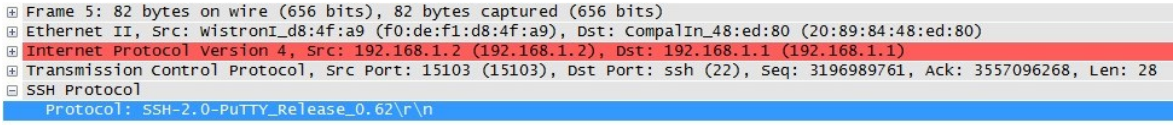
SSH protokol byl vytvořen za účelem zabezpečení síťové komunikace. Měl nahradit protokoly jako je Telnet, Rsh, Rlogin a další, které data neposílají v šifrované formě. Protokol je relativně jednoduchý na implementaci. První verze SSH verze 1 byla zveřejněna roku 1995. V roce 1996 byl vydán SSH verze 2, který je nekompatibilní s SSH verze 1. SSH verze 2 vylepšuje zabezpečení protokolu a je popsána v RFC dokumentech 4250-4256.

SSH protokol funguje na principu klient/server, využívá služeb TCP a server naslouchá na portu 22. Struktura SSH-2 se skládá ze tří částí:

- Protokol Transportní vrstvy
- Uživatelský autentizační protokol
- Protokol spojení

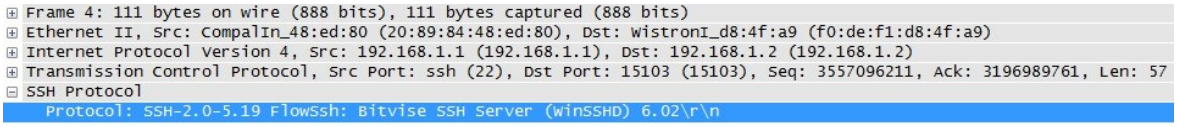
Transportní vrstva protokolu SSH-2

Tato vrstva zajišťuje důvěrnost a integritu dat, autentizaci serveru a počáteční výměnu klíčů mezi klientem a serverem. Po navázání spojení si klient a server vymění pakety obsahující řetězec „*SSH-protoversion-softwareversion SP comments CR LF*“. Řetězec obsahuje informaci o použité verzi protokolu, verzi softwaru a ukončovací znaky SP, CR a LF.



Obrázek 47 – Paket s verzí protokolu a použitým sw na straně klienta

Server odpoví klientovi vlastním paketem se stejnou informací.



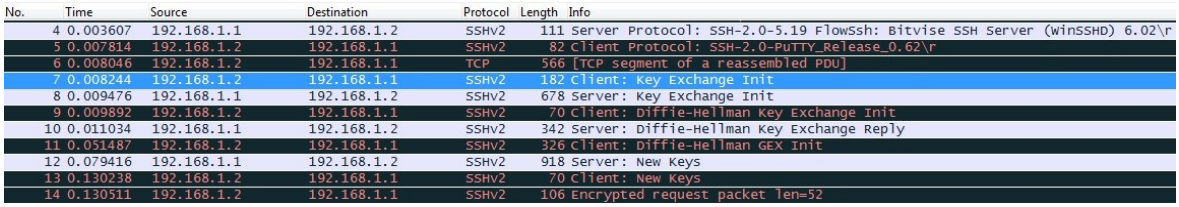
Obrázek 48 – Paket s verzí protokolu a sw na straně serveru

Tyto informace slouží k výměně klíčů pomocí Diffe-Hellman algoritmu. Následuje dohodnutí použitého šifrování. Obě strany si vymění SSH_MSG_KEXINIT pakety, které obsahují seznam podporovaných šifrovacích algoritmů. Vybere se první algoritmus v seznamu klienta, který je podporován i na straně serveru.

ŠIFRA	NUTNOST POUŽITÍ
3des-cbc	požadovaná
blowfish-cbc	volitelná
twofish256-cbc	volitelná
twofish-cbc	volitelná
twofish192-cbc	volitelná
twofish128-cbc	volitelná
aes256-cbc	volitelná
aes192-cbc	volitelná
aes128-cbc	požadovaná
serpent256-cbc	volitelná
serpent192-cbc	volitelná
serpent128-cbc	volitelná
arcfour	volitelná
idea-cbc	volitelná
cast128-cbc	volitelná

Obrázek 49 – Definované šifrovací algoritmy

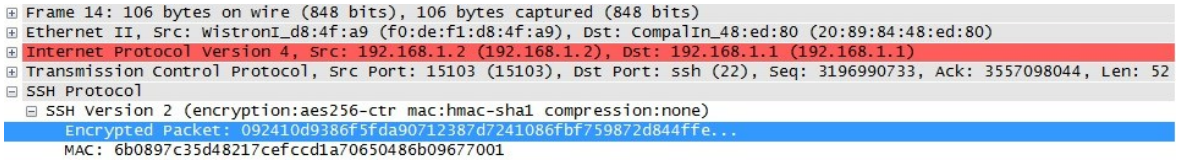
V dalším kroku dochází k výměně klíčů obou stran. Jakmile dojde k výměně klíčů, klient i server si vymění pakety SSH_MSG_NEWKEY, které signalizují dokončení výměny klíčů. Výměna klíčů je znázorněna na obrázku Obrázek 50, pakety s číslem 7 až 13.



No.	Time	Source	Destination	Protocol	Length	Info
4	0.003607	192.168.1.1	192.168.1.2	SSHv2	111	Server Protocol: SSH-2.0-5.19 FlowSsh: Bitvise SSH Server (WinSSHD) 6.02\r
5	0.007814	192.168.1.2	192.168.1.1	SSHv2	82	Client Protocol: SSH-2.0-PuTTY_Release_0.62\r
6	0.008046	192.168.1.2	192.168.1.1	TCP	566	[TCP segment of a reassembled PDU]
7	0.008244	192.168.1.2	192.168.1.1	SSHv2	182	Client: Key Exchange Init
8	0.009476	192.168.1.1	192.168.1.2	SSHv2	678	Server: Key Exchange Init
9	0.009892	192.168.1.2	192.168.1.1	SSHv2	70	Client: Diffie-Hellman Key Exchange Init
10	0.011034	192.168.1.1	192.168.1.2	SSHv2	342	Server: Diffie-Hellman Key Exchange Reply
11	0.051487	192.168.1.2	192.168.1.1	SSHv2	326	Client: Diffie-Hellman GEX Init
12	0.079416	192.168.1.1	192.168.1.2	SSHv2	918	Server: New Keys
13	0.130238	192.168.1.2	192.168.1.1	SSHv2	70	Client: New Keys
14	0.130511	192.168.1.2	192.168.1.1	SSHv2	106	Encrypted request packet len=52

Obrázek 50 – Proces výměny klíčů

Na závěr této části komunikace klient zasílá SSH_MSG_SERVICE_REQUEST paket, který vyžaduje použití Uživatelského autentizačního protokolu nebo protokolu spojení.



Obrázek 51 – SSH request paket

Od této chvíle probíhá komunikace zašifrovaně. Veškerá komunikace je uvedena v přiloženém souboru.

Uživatelský autentizační protokol

Tento protokol specifikuje, jakými metodami je klient autentizován na straně serveru. Klient zašle serveru SSH_MSG_USERAUTH_REQUEST, ve kterém předá informace a autentizaci. Paket obsahuje uživatelské jméno, název služby a jméno autentizační metody, kterou hodlá využít. Pokud server přijme autentizaci, odpoví klientovi zprávou SSH_MSG_USERAUTH_SUCCESS. V případě neúspěchu autentizace zašle zprávu SSH_MSG_USERAUTH_FAILURE, která obsahuje seznam autentizačních metod možných pro pokračování komunikace. Server může také požadovat dodatečné autentizační metody:

- Publickey – metoda založená na výměně klíčů. Klient pošle serveru svůj veřejný klíč, který slouží k rozšifrování zprávy.
- Password – klient zasílá heslo v podobě textu, komunikace je zašifrována přes transportní protkol SSH.
- Hostbased – klient zašle podpis, vytvořený pomocí soukromého klíče, server podle podpisu ověří celé zařízení. Server věří, že uživatel je ověřen na straně klienta.

Protokol spojení

Tento protokol definuje koncept jednotlivých kanálů a jejich požadavků v rámci komunikace. Spojení může mít více kanálů, v každém kanálu obě strany definují číslo kanálu. Tyto čísla kanálů nemusí být stejná na straně serveru ani klienta. SSH rozlišuje 4 druhy kanálů:

- Session – může být vzdálený program na serveru, shell, ftp. Po vytvoření kanálu se spustí program na vzdáleném počítači.
- x11 – kanál pro X Window systém, který slouží k vytvoření uživatelského grafického rozhraní (GUI). Programy běží na serveru, ale zobrazují se na uživatelském počítači.
- forwarded-tcpip – kanál vyhrazený pro přesměrování portů na straně serveru.
- direct-tcpip – kanál pro lokální přesměrování portů na straně klienta.

Jedna z výhod SSH protokolu je možnost tunelování paketů nezabezpečeného TCP spojení do zabezpečeného spojení skrze SSH. K odlišení procesů jsou využity čísla portů jednotlivých protokolů. V současné době má protokol široké využití u vzdálené správy, přenosu souborů, penetračních testů a dalších situací, kdy je vhodně použít zabezpečené spojení. [21][25][30]

Závěr

Tato bakalářská práce byla zaměřena na představení a popis struktury vybraných protokolů aplikační vrstvy modelu TCP/IP a jejich vztahu k transportní vrstvě.

V první části práce byl představen jeden z používaných nástrojů ke sledování paketů na síťovém zařízení. Dále zde byl popsán proces sledování paketů a jeho možné využití, při sledování komunikace v síti.

Ve druhé část byl popsán síťový model TCP/IP a jeho jednotlivé vrstvy. Podrobněji byla popsána transportní vrstva a síťová vrstva modelu a protokoly, které na daných vrstvách pracují. V síťové vrstvě byly představeny dva protokoly, které slouží k adresaci paketů. V transportní vrstvě byl podrobně představen protokol TCP. U TCP protokolu byla zachycena a vysvětlena komunikace, její zahájení a ustálení komunikace mezi klientem a serverem. Byl zde vysvětlen princip generování portů na straně klienta i na straně serveru. Dále zde byl popsán protokol UDP a jeho datagram.

Třetí část práce byla zaměřena na aplikační protokoly, které slouží k nastavení rozhraní a jeho správné komunikaci se zařízeními v Internetu. Byl zde vysvětlen a ukázán princip získání údajů potřebných pro komunikaci. Dále byl popsán princip a ukázán způsob překladu doménových jmen na IP adresy. Byl zde také představen FTP protokol, pro přenos dat a protokol HTTP, který slouží k získání webových stránek ze serveru. Společně s ním bylo představeno jeho vylepšení v podobě HTTPS protokolu, který zajišťuje šifrovaný přenos dat.

Ve čtvrté části práce byl vysvětlen a ukázán princip odesílání a přijímání elektronické pošty. Byly zde podrobně představeny protokoly, které slouží k získání elektronické pošty ze serveru do vlastního zařízení. Dále zde byl představen protokol, který slouží k odeslání elektronické pošty.

V poslední, páté, části byly podrobně představeny protokoly pro vzdálenou konfiguraci. V této části byla ukázána nevýhoda nešifrované komunikace, kterou nabízí protokol Telnet. V přiložené ukázce byla odchytnuta komunikace a v nešifrované podobě jsou vidět přihlašovací údaje klienta. Druhým popsaným protokolem byl protokol SSH. V zachycené komunikaci bylo názorně vidět, že tento protokol již používá šifrovaný přenos a v komunikaci nelze vyčíst přihlašovací údaje klienta.

Cílem práce bylo podrobně představit a popsat strukturu aplikačních protokolů. Vybrané protokoly byly odchyceny a jejich komunikace byla znázorněna v práci a na přiloženém disku CD-ROM.

Literatura

[1] A FIRST CUT AT A PROPOSED TELNET PROTOCOL. MELVIN, John T. a Richard W. WATSON. *Internet Engineering Task Force (IETF)* [online]. 1971 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc97.txt

[2] SANDERS, Chris. *Analýza sítí a řešení problémů v programu Wireshark*. 1. vyd. Brno: Computer Press, 2012, 288 s. ISBN 978-80-251-3718-5.

[3] BOOTSTRAP PROTOCOL (BOOTP). CROFT, Bill a John GILMORE. *The Internet Engineering Task Force (IETF)* [online]. 1985 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc951.txt

[4] DROMS, Ralph a Ted LEMON. *DHCP příručka administrátora*. Vyd. 1. Brno: Computer Press, 2004, 490 s. ISBN 80-251-0130-4.

[5] DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION. MOCKAPETRIS, P. *The Internet Engineering Task Force (IETF)* [online]. 1987 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc1035.txt

[6] Dynamic Host Configuration Protocol. DROMS, R. *The Internet Engineering Task Force (IETF)* [online]. 1997 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc2131.txt

[7] FILE TRANSFER PROTOCOL (FTP). POSTEL, J. a J. REYNOLDS. *The Internet Engineering Task Force (IETF)* [online]. 1985 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc959.txt

[8] FTP Security Extensions. HOROWITZ, M. *The Internet Engineering Task Force (IETF)* [online]. 1997 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc2228.txt

[9] HTTP Over TLS. In: RESCORLA, E. *The Internet Engineering Task Force (IETF)* [online]. 2000 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc2818.txt

[10] Hypertext Transfer Protocol -- HTTP/1.1. FIELDING, R., J. GETTYS, J. MOGUL, H. FRYSTYK, L. MASINTER, P. LEACH a T. BERNERS-LEE. *The Internet Engineering Task Force (IETF)* [online]. 1999 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc2616.txt

[11] IMAP4 Authentication Mechanisms. MYERS, J. *The Internet Engineering Task Force (IETF)* [online]. 1994 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc1731.txt

[12] INTERACTIVE MAIL ACCESS PROTOCOL - VERSION 2. CRISPIN, M. *The Internet Engineering Task Force (IETF)* [online]. 1988 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc1064.txt

[13] INTERACTIVE MAIL ACCESS PROTOCOL - VERSION 3. In: RICE, J. *The Internet Engineering Task Force (IETF)* [online]. 1991 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc1203.txt

[14] INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1. CRISPIN, M. *The Internet Engineering Task Force (IETF)* [online]. 2003 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc3501.txt

[15] INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1. CRISPIN, M. *The Internet Engineering Task Force (IETF)* [online]. 1996 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc2060.txt

[16] Internet Protocol, Version 6 (IPv6) Specification. DEERING, S. a R. HINDEN. *The Internet Engineering Task Force (IETF)* [online]. 1995 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc1883.txt

[17] IP verze 6 (1). SATRAPA, Pavel. *Lupa.cz - server o českém Internetu* [online]. 14. 9. 2000 [cit. 2013-08-20]. Dostupné z: http://www.lupa.cz/clanky/ip-verze-6-1/

[18] YAN, Ke. *Network protocols handbook*. Vyd. 2. Saratoga: Javvin, 2005, 341 s. ISBN 09-740-9452-8.

[19] POST OFFICE PROTOCOL - VERSION 2. BUTLER, M., J. POSTEL, D. CHASE, J. GOLDBERGER a J. K. REYNOLDS. *The Internet Engineering Task Force (IETF)* [online]. 1985 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc937.txt

[20] Post Office Protocol - Version 3. ROSE, M. *The Internet Engineering Task Force (IETF)* [online]. 1988 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc1081.txt

[21] Protocol Basics: Secure Shell Protocol: The Internet Protocol Journal, Volume 12, No.4 - Cisco Systems. STALLINGS, William. *Cisco Systems, Inc* [online]. 2010 [cit. 2013-08-20]. Dostupné z: http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_12-4/124_ssh.html#reference1

[22] Simple Mail Transfer Protocol. KLENSIN, J. *The Internet Engineering Task Force (IETF)* [online]. 2001 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc2821.txt

[23] SIMPLE MAIL TRANSFER PROTOCOL. POSTEL, J. B. *The Internet Engineering Task Force (IETF)* [online]. 1981 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc788.txt

[24] Síťový model TCP/IP. PETERKA, Jiří. *Jiří Peterka: archiv článků a přednášek* [online]. 1992 [cit. 2013-08-20]. Dostupné z: http://www.earchiv.cz/a92/a231c110.php3

[25] BARRETT, Daniel J, Richard E SILVERMAN a Robert G BYRNES. *SSH, the secure shell: the definitive guide*. 2nd ed. Sebastopol, CA: O'Reilly, c2005, xviii, 645 p. ISBN 05-960-0895-3.

[26] STANDARD FOR THE FORMAT OF ARPA INTERNET TEXT MESSAGES. CROCKER, David H. *The Internet Engineering Task Force (IETF)* [online]. 1982 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc822.txt

[27] TCP/IP - navázání a ukončení spojení. BOUŠKA, Petr. *SAMURAJ-cz.com: počítačové sítě, Cisco, Microsoft, VMware, administrace* [online]. 13.09.2007 [cit. 2013-08-20]. Dostupné z: http://www.samuraj-cz.com/clanek/tcpip-navazani-a-ukonceni-spojeni/

[28] FOROUZAN, Behrouz A. *TCP/IP protocol suite*. 4th ed., international student ed. Boston, MA: McGraw-Hill Higher Education, 2010. ISBN 978-007-0166-783.

[29] TELNET PROTOCOL SPECIFICATION. POSTEL, J. a J. REYNOLDS. *Internet Engineering Task Force (IETF)* [online]. 1983 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc854.txt

[30] KOZIEROK, Charles M. *The TCP/IP guide: a comprehensive, illustrated Internet protocols reference*. San Francisco: No Starch Press, c2005, lxxiv, 1539 p. ISBN 15-932-7047-X.

[31] TRANSMISSION CONTROL PROTOCOL: DARPA INTERNET PROGRAM PROTOCOL SPECIFICATION. INFORMATION SCIENCES INSTITUTE UNIVERSITY OF SOUTHERN CALIFORNIA. *The Internet Engineering Task Force (IETF)* [online]. 1981 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc793.txt

[32] User Datagram Protocol. POSTEL, J. *The Internet Engineering Task Force (IETF)* [online]. 1980 [cit. 2013-08-20]. Dostupné z: http://www.ietf.org/rfc/rfc768.txt

[33] KABELOVÁ, Alena a Libor DOSTÁLEK. *Velký průvodce protokoly TCP/IP a systémem DNS*. 5., aktualiz. vyd. Brno: Computer Press, 2008, 488 s. ISBN 978-80-251-2236-5.