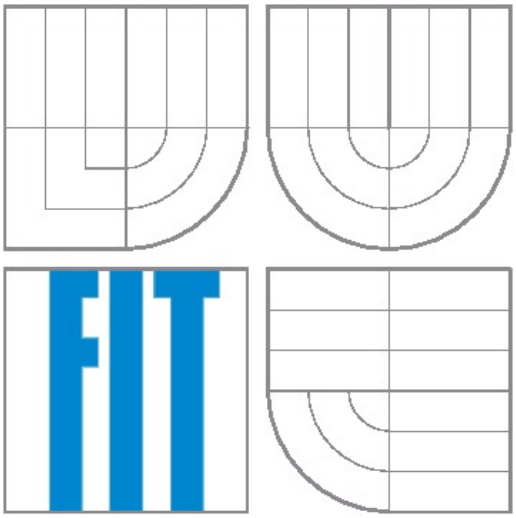


Fakulta informačních Technologií

Vysoké učení technické v Brně

Jednoduchý autentizačný server

Síťové aplikace a správa sítí

Karel Hala

xhalak00

listopad 2012

Obsah

1 Úvod

Tento program je studentský projekt napsaný do předmětu Síťové aplikace a správa sítí (ISA).

Má za úkol autentizaci uživatele na serveru, podle normy RFC 2865 (Radius).

2 Popis

2.1 Všeobecný popis

RADIUS je síťový protokol, který má za úkol centralizovat autentizaci, autorizaci a acounting na serveru.

Je mnoho způsobů jak tyto akce provádět a RADIUS má za úkol tyhle úkony sjednotit. Tento protokol popisuje obsah jedntolivých packetů, práci s nimi a jak by měl programátor postupovat při jeho implementaci, aby dosáhl nejlepšího výsledku.

Protokol pracuje na bázi uživatelských jmen a hesel, heslo je pokaždé šifrováno jednosměrnou hashovací funkcí MD5[1]. Pro zachování bezbečnsosti se pro další šifrování používá takzvané tajné heslo, které je sdíleno mezi serverem a uživatelem.

Skladba packetu je do podrobna vysvětlena v sekci 5.

2.2 Popis implementovaného programu

V tomto programu je však zapotřebí pouze autentizace uživatele. Ověření zda má na serveru účet a zda se jeho heslo shoduje s tím, které má v databázi. Dále kontroluje, zda nebyl packet během své cesty odchycen a popřipadě se někdo nesnaží podvrhnout přihlášení. Proto tento prgram zpracovává pouze jeden typ packetu a to Access-Request. A jako odpověď může odeslat pouze dva packety a to Access-Accept, nebo Access-Reject. Pro kontrolu, zda se nejedná o podvrhnutou komunikaci program ignoruje jakékoliv nadbytečné atributy. A naopak, pokud nebude packet obsahovat všechny povinné atributy, server vždy odešle Acess-Rejected. Do odpovědi dále server odpovídá zprávou zda je uživatel potvrzen, nebo odmítnut na základě zaslaného hesla.

1 Message-digest algortimus. Pouzžvá se pro zahashovani integrity souborů a hesel. Velikost 128 bitů.

3 Použití

Program tiskne několik kontrolních výstupů, všechny jsou tisknuty na standartní chybový výstup.

Program se spouští příkazem **./radauth** dále očekává přepínač -c , který mu určuje kde se nachází konfigurační soubor. Pokud program spustíme bez přepínače -c program očekává ruční zadání konfigurace, která je ukončena koncem souboru.

Konfigurační soubor obsahuje nastavení autentizačního serveru. Tabulka s vysvětlivkami jednotlivých nastavení je v sekci 3.1 Konfigurační soubor.

Ten dále obsahuje cestu k souboru s databází jednotlivch uživatelů. Popis souboru a příklad obsahu souboru je dále popsán v sekci 3.2 Databáze úživatelů.

3.1 Konfigurační soubor

iface=eth0,eth1	Jména jednotlivých zařízení na kterých program naslouchá. [2]
port=11812	Port na kterém program naslouchá.
secret=tajneheslo	Tajné heslo, sdílené uživatelem a serverem
userdb=users.txt	Cesta k databázi uživatelů.

Ošetření jednotlivých speciálních stavů a chybové hlášky jsou popsány v sekc 6.1 Implementace vyhledání a rozdělení konfiguračního souboru..

3.2 Databáze úživatelů

Jednotlivé záznamy v databázi uživatelů jsou odděleny novým řádkem. Heslo je odděleno od uživatele znakem ":". Speciální případy a ošetření chyb je popsáno v 6.2 Implementace práce s databází uživatelů

Karel:123456 [3]
Papai:iapap
Pepa:548
Brano:abc
Bizon:6589

2 Tabulka popisující obsah konfiguračního souboru.

3 Tabulka popisující obsah databáze.

4 Popis RFC

RADIUS Je zkratka představující "Remote Authentication Dial In User Service". Podle specifikace je packet rozdělen podle bitů do jednotlivých bloků 7.1. Každý blok je různě velký. Code označuje kod packetu, každý packet má specifické Code. Například Access-Accept má Code 1, odpoved od serveru Access-Accept má Code 2 atd. (Pro bližší informace[4]).

Identifier je jedinečné číslo dotazu na server. Pod tímto číslem dále taká server odpovídá. Jedna zpráva server<->uživatel má stejný identifikátor.

Length obsahuje celkovou délku packetu. Tato délka je uvedena v Bajtech. Na základě této hodnoty se dají procházet položky packetu.

Authenticator je 128b, což je 16B 8bitových hodnot. Toto pole se používá pro ověření správnosti packetu. Pro každý packet se tato položka vypočítává různě. Ale pro šifrování se vždy používá jednosměrná MD5. Odtud 16B 8bitových hodnot.

Attributes obsahují další atributy potřebné pro práci s daným packetem. Například heslo, login, zprávu a podobne (Pro bližší informace[5]).

Každý atribut obsahuje jeho číslo, délku a hodnotu. Názorně ukázán na obrázku 7.2

5 Výstup programu

Pogram netiskne nic na standardní výstup, ale na chybový výstup vypisuje postupně některé kontrolní výstupy a popřípadě chybové hlášení.

Pravým výstupem programu, lze brát formát packetu. Obsah packetu je znázorněn na obrázku7.1. Kde Code obsahuje 2, nebo 3, podle toho zda byl přístup povolen, nebo odmítnut. Identrifier obsahuje Identifier z příchozího packetu. Length, je délka odchozího packetu. Authenticator obsahuje 16B MD5 hash. Který se vytváří za pomoci MD5(Code+Identifier+Length+Authenticator+Attributies) Authenticator je převzat z říchozího packetu a je to 16B MD5 hash. Attributes obsahuje další atributy, v našem případě pouze jeden atribut a tím je Reply-Message.

---

4 RADIUS. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2012-11-19]. Dostupné z: http://en.wikipedia.org/wiki/RADIUS

5Tutorialspoint: RADIUS-atributy. [online]. [cit. 2012-11-18]. Dostupné z: http://www.tutorialspoint.com/radius/radius_attribute_format.htm

6 Implementace

Program je napsán v C++ používá klasické knihovny a pro komunikaci používá BSD sockety.

6.1 Implementace vyhledání a rozdělení konfiguračního souboru.

Data v konfiguračním souboru jsou od sebe oddělena zalomením řádku. Víceřádkové nastavení a zbytečné řádky v konfiguračním souboru ignoruji. Dále identifikátory nastavení serveru očekávám na začátku noveho řádku.

Program postupně nejdříve vytáhne jednotlivé data z konfiguračního souboru. A poté je otestuje, zda v nich není chyba. Jako je chybný formát portu, neexistující tajné heslo. Neexistující zařízení, neexistující databázi a nekontroluje program při parsování jednotlivých částí, ale až při běhu. Tudíž, pokud zadáte špatný port program tuto zkutečnost sdělí přes standardní chybový vstup a korektně se ukončí.

Dále pokud se programu nepovede najít danné zařízení ukončí se a tuto skutečnost sdělí na standardní chybový výstup. Pohrával jsem si s myšlenkou že pokud se povede připojit, alespoň k jednomu zařízení sdělí program tuto skutečnost, ale bude se snažit dále pracovat s tímto jedním zařízením. Nicméně by mohlo dojít k mylnému přesvědčení že program pracuje v pořádku na všech zařízeních, tudíž jsem tuto myšlenku zavrhl. Pokud budou jména zařízení oddělena více čárkami, nic se neděje program se s touto skutečnosti vypořádá, jako by tam byla jedna čárka. Pokud bude jméno zařízení obsahovat prázdné znaky, program tyto prázdné znaky ignoruje. Pokud bude některá hodnota chybět, program dá uživateli vědět o této skutečnosti a ukončí se.

Program nepracuje s IPv6 standradem, tento standard není stále plně v provozu a pro úplný chod aplikace naprosto stačí pracovat se standardem Ipv4.

6.2 Implementace práce s databází uživatelů

Program jednoduše jednotlivé prvky zařadí do hashovacího pole, kde jako index položky je jméno a hodnota je heslo.

Pokud jméno obsahuje prázdné řetězce ignoruju takového uživatele. Pokud má uživatel prázdné znaky v hesle použiji takové heslo, takže položka v databázi ve tvaru "*pepa:pe pa*" je korektní. Pokud je však u uživatele naprosto prázdné heslo takového uživatele ignoruji. V RFC 2865 je toto chování takto popsáno, tudíž jsem se tímto naprosto řídil.

Pokud bude duplicita uživatelů, druhý záznam uživatele ignoruji.

6.3 Implementace komunikace serveru

Server komunikuje pomocí UDP packetů. Částečnou implementaci inicializace a přijímání packetů jsem převzal ze třetího projektu IPK. Přijde mi znovu opisování již funkční věci zbytečné a zdržovalo by to důležitejší práci.

Server naslouchá na určených zařízeních a zadaném portu. Pokud mu přijdou nějaká data na zpracování, pokusí se je rozluštit, pokud by serveru došel jakkoliv poškozený packet, dotaz zahodí a dále se jím nezabývá.

Přijmu UDP packet o maximální velikosti 65535, jako je to uvedeno zde[6]. Pokud bude packet ale větší než 4096 znaků, jak je uvedeno v RFC[7], zahodím jej. Může se jednat o podvrhnutý packet, nebo o poškozený packet a tím by nastala nekonzistence.

Packet postupně rozluští a rozdělí do struktury se kterou program dále pracuje. Rozdělení do struktury prochází podle 7.1.

Pokud najde uživatele v databázi zašifruje heslo a zkontroluje zda se shoduje s šifrou, která přišla jako atribut heslo v příchozím packetu. Maximální délka hesla je nastavena na 128 znaků. Pokud bude heslo delší než tato konstanta, heslo bude jednoduše oříznuto. Pokud uživatel požaduje heslo delší, jak 128 znaků, velice hluboce se mu omlouvám, ale hesla delší, než tato konstanta program bohužel nepodporuje.

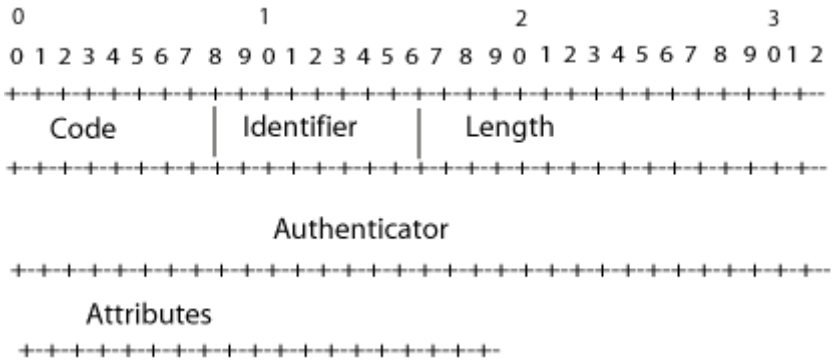
Po ověření shody, nebo neshody hesla, které přišlo v příchozím packetu, s tím, které jsme vygenerovali MD5 hashem z databáze. Vytvoří program odchozí packet, ve kterém sdělí následnou shodu, nebo neshodu hesla. Pokud klient žádá o přístup do databáze s neexistujícím uživatelem, dostane odpověď access-rejected.

Jako povinné atributy v packetu považuji pouze Login, Heslo a NAT-identifier, nebo NAS-IP-Adress. Pokud přijde NAS-Identifier a zároveň i NAS-IP-address ověřím výskyt alespoň jednoho a pokud je alespoň jeden z těchto atributů použit pokračuji v práci. NAS-IP-Address upravím do klasického tvaru xxx.xxx.xxx.xxx. Pokud uživatel nepošle všechny povinné atributy, odmítám jakkoliv pracovat a odpovídám uživateli access-rejected packetem. Jakýkoliv atribut, který je navíc ignoruji a zahazuji.

6UDP packet. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2012-11-19]. Dostupné z: http://en.wikipedia.org/wiki/User_Datagram_Protocol#Packet_structure
7RADIUS: RFC2865. [online]. [cit. 2012-11-19]. Dostupné z: http://tools.ietf.org/html/rfc2865

7 Přílohy

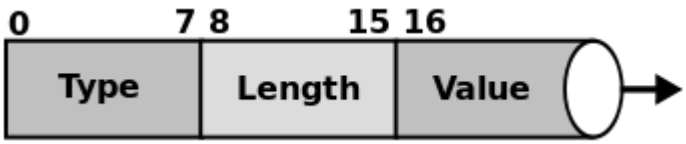
Formát packetu. 0-12 bity, 0-3 Bajty



7.1

Formát atributu v packetu. 0-7b Typ, 8-15b delka, 16-n hodnota.

7.2



8 Závěr

Program se dokáže vypořád s valnou většinou případů. Například při příchodu porušeného packetu může nastat zásah do paměti, kam nemá program přístup. Tento stav je nežádoucí. Proto jsem jej vyřešil odchycením signálu a korektním ukončení aplikace. Je zde možnost dlouhých jumpů. Ale skoky nejsou naprosto košér.

Při implementaci jednotlivých částí jsem čerpal z mnoha zdrojů většinu jsem vyjmenoval v sekci 8.1 Zdroje a Odkazy. Pro implementaci UDP komunikace jsem použil udt.h knihovnu z projektu číslo 3 z IPK, která byla napsána panem Onřejem Ryšavým. Čímž bych mu chtěl touto cestou poděkovat, jelikož mi ušetřil dost času implementací něčeho co jsem již jednou použil. Nicméně jsem musel zasáhnout do tohoto kodu,pro úpravu na stávající zadání, ale to vyžadovalo pouze minimální zásahy.

Při implementaci daného problému jsem se naučil zdatně pracovat s hashovací funkcí MD5. A uvědomil jsem si jak je důležité při implementaci nějaké aplikace myslet dopředu, že možná jednou za několik let bude někdo psát nádstavbu, popřípadě pouze část takovéto aplikace.

8.1 Zdroje a Odkazy

Vzorový příklad implementace celého Radius serveru.

Opensource.apple: radius.c. In: [online]. [cit. 2012-11-19]. Dostupné z: http://www.opensource.apple.com/source/freeradius/freeradius-32/freeradius/src/lib/radius.c

RFC RADIUS.

RADIUS: RFC2865. [online]. [cit. 2012-11-19]. Dostupné z: http://tools.ietf.org/html/rfc2865

Formát packetu RADIUS.

RADIUS: packet format. [online]. [cit. 2012-11-19]. Dostupné z: http://www.tutorialspoint.com/radius/radius_attribute_format.htm

Tutoriál k funkci select.

Tutoriál k funkci selectna root.cz . [online]. [cit. 2012-11-19]. Dostupné z: http://www.root.cz/clanky/sokety-a-c-funkce-select/

Tutiriál k síťovámu rozhraní.

Tutoriál síťové rozhraní na root.cz. [online]. [cit. 2012-11-19]. Dostupné z: http://www.root.cz/clanky/sokety-a-c-sitove-rozhrani/

RADIUS příklad odpovědí ze serveru.

RADIUS: Příklad jednotlivých odpovědí. [online]. [cit. 2012-11-19]. Dostupné z: http://www.xperiencetech.com/radius_manual/radsample.htm

Popis RADIUSu na wikipedii.

RADIUS. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2012-11-19]. Dostupné z: http://en.wikipedia.org/wiki/RADIUS