IPSec na Linuxu, krok za krokem - 1

17.3.2006 00:01 | Jan Vondráček

Pracuji už dlouhou dobu s Cisco zařízeními, routery, pixy a další. Můj takový koníček je zkoušet nahrazovat služby, která Cisco zařízení poskytují, Linuxem. Chci se s vámi podělit o své zkušenosti.

Úvod a teorie ipsecu

Pracuji už dlouhou dobu s Cisco zařízeními, routery, pixy a další. Můj takový koníček je zkoušet nahrazovat služby, která Cisco zařízení poskytují, Linuxem. Určitě není žádný problém spojit dva cisco routery proti sobě různými způsoby, ale spojit proti sobě zařízení různých výrobců už bývá problém. V internetovém prostředí často na takovouto nutnost narazíte, protože každý z nás si nemůže dovolit koupit Cisco router. Při takových pracech často velmi dobře pochopíte fungování nejrůznějších technologií i fungování Linuxu.

O jednu takovou technologii se s vámi chci teď podělit. Návodů na rozchození ipsecu pod Linuxem jsem četl spoustu, ale pomocí žádného se mi nepodařilo spojit přes ipsec linuxový stroj proti Cisco zařízení. Určitě to musí jít, protože ipsec je standard a už jsem to i viděl v činnosti. Ovšem všechny návody spojovaly pouze dva Linuxy proti sobě a využívaly manuální spojení ipsecem bez IKE (internet key exchanger) a to mi nevyhovuje. Ipsec je podporován i ve spoustě dalších levných necisco zařízeních a tam byste manuální spojení ipsecu jen opravdu marně hledali. Takže se do toho pustíme a nejdříve si povíme něco o teorii, co ipsec je a samozřejmě se zmíním i o manuálním spojení.

Pro Linux existuje spousta programů, které řeší bezpečnou komunikaci. Některé se dokonce honosí, že fungují na ipsecovém standardu, ale jejich implementace ipsecu není plná, a proto se nemohou spojit s jinými programy. Program, který podporuje standard je dle mého takový, který vám dovolí sestavit vpn mezi ciscem, linxovým strojem a třeba Windows. Ano, nedivte se, i Windows mají implementován ipsec, psalo ho pro ně přímo Cisco 😊 Jako nejvhodnější, a v podstatě jediný, se mi pro Linux jeví projekt Openswan. Je to pokračování projektu Freeswan. Implementace ipsecu je zde plná, i když v dokumentaci trochu vázne. A tím se zde budeme zabývat.

Teorie ipsecu

Co je to vlastně ipsec? Je to služba, která umožňuje dvěma zařízením komunikovat bezpečně, tedy šifrovaně. Co je to ale bezpečně? Bezpečně znamená nejen tak, aby nikdo neviděl obsah paketů, které posíláme, ale také, abychom přišli na to, že se nám spojení snaží někdo rušit a i v případě, že by někdo šifrování prolomil, jsme o tom věděli. Toto vše je v ipsecu implementováno. Ipsec má dvě fáze svého běhu - *Main mod* a *Quick mod*.

V Main modu pracuje IKE, který ověří obě strany šifrovaného spojení a ustanoví prvotní bezpečnou komunikaci. V této fázi se nepřenáší zatím žádná data. Cisco používá pro tuto část ISAKMP, Openswan Pluto.

Quick mod se pak již používá pro samotný přenos dat. U Cisca se to nazývá ipsec, Openswan Klips.

Existuje ještě jeden mód, *Agressive*. Je to zjednodušené spojení main a quick s trochu ořezanou bezpečností. Tento mód se používá pro připojení klientů, kteří nemají pevnou IP adresu, třeba připojení z dial-upu a podobně. Omezení bezpečnosti je ale vyváženo ještě dodatečnou autorizací (xauth), ale o tom si povíme dále.

Ipsec je jen o matematice. A je nutno si pamatovat jedno základní pravidlo: Bezpečná jsou jen ta čísla, která jsem si potvrdil vlastním výpočtem. A proto při počítání klíčů se skutečně všechny výpočty provádějí duplicitně, na obou stranách. Ono by to jinak ani nešlo.

Ipsecové spojení je pouze bod-bod. Pokud budete mít tři místa, která mezi sebou budete chtít spojit, vyjdou vám tři ipsecová spojení, na každé ipsecové bráně dvě - metoda spojení do full mesh. Další způsob, jak spojit tři místa, je, zvolit si jednu centrálu a ostatní místa připojit jen do ní. Výhodou je jen jeden tunel na pobočce, ale více na centrále. Pobočky se mezi sebou vidí přes centrálu, ale data tekoucí mezi pobočkami zatěžují linku centrály 2x - metoda hvězda. Při konstrukci sítě si toto musíte uvědomit a podle toho zvolit patřičnou metodu.

Pro pochopení funkce ipsecu si musíme povědět něco o základních pojmech a parametrech, které budeme nastavovat:

hash SHA-1 nebo MD5	**Hash je jednosměrná funkce, která nám z textu nebo čísla vyrobí šifru. Tímto způsobem se uchovávají hesla v Linuxu, v zašifrované podobě. Při přihlášení se pak heslo, které zadáváte, zahashuje a porovná s uloženou hashí. V ipsecu se toto používá pro kontrolu, že paket nebyl modifikován. Po vytvoření, zašifrování paketu se číslo jeho velikosti prožene hashí a připojí se k paketu. Při přijetí paketu se číslo opět kontroluje, tj. velikost paketu se zahashuje a spočítaná hash se porovná s obdrženou.**
DH group Dieffiel-Hellmanovo číslo	**Je to velké prvočíslo používané pro počítání klíčů. A skupina nám prozradí, jak velké to číslo má být, a samozřejmě jak časově a výkonnostně náročný výpočet to je. U každé skupiny je definována šíře bitového základu.** **DH group 2 = 1024 bitů** **DH group 5 = 1536 bitů** **DH group 14 = 2048 bitů** **DH group 15 = 3072 bitů** **DH group 16 = 4096 bitů** **DH group 17 = 6144 bitů** **DH group 18 = 8192 bitů** **Skupina 1 není openswanem podporována z důvodu bezpečnosti, číslo je příliš malé. A například Cisca nemají, krom nejnovějších IOSů, podporu více než 5. Může se vám tedy stát, že příliš staré nebo hodně ořezané zařízení nebude s novým fungovat.**
klíče	1. **Preshare key - sdílený klíč je textový klíč, který je svázán s IP adresami a používá se při obou fázích. Zahrnuje se do výpočtů jak autentizačních klíčů, tak i šifrovacích klíčů. Je to jediná konstanta ve výpočtech, obě strany tento klíč musí obsahovat. Generování je snadné - něco si napište, kus textu, pozdrav a nebo jen uhoďte do klávesnice. Že se vám to zdá jednoduché a málo bezpečné? Ano, není to nejbezpečnější metoda, ale je nejsnazší a nejuniverzálnější na rozchození.** 2. **RSA klíče jsou dalším typem klíčů. Asi je znáte z SSH. U nich je trochu problém, že se musí generovat v každém zařízení. Jsou bezpečnější, ale v praxi jsem je nikdy nepoužil.** 3. **Posledním typem jsou certifikáty od certifikační autority. Obě zařízení se musí spojit s třetím, který jim vydá certifikát. Je to metoda nejbezpečnější, protože klíč není uložen v ipsecovém zařízení. V praxi jsem to viděl v činnosti, ale také nikdy nepoužil.**
DES	**Šifrovací algoritmus s délkou klíče 56 bitů. Dnes je již dost zastaralý, protože jeho rozlousknutí při dnešních výkonech počítačů trvá asi 20 minut. Pokud jej používáme, měli bychom nastavit životnost klíče tak na 10 minut. Openswan DES vůbec nepodporuje z důvodů malé bezpečnosti.**
3DES	**Silnější šifra s délkou klíče 168 bitů, šifruje jako DES, ale 3x dvěma klíči. Nejdříve prvním, pak druhým a na konec zase prvním. Tímto mechanizmem se sice dostaneme na dnes nerozlousknutelnou dobu, ale výpočetní náročnost je také trojnásobná. Efektivní délka klíče se tak smrskne na 112 bitů.**
AES	**Odlišný algoritmus od DESů s délkou klíče 128, 192 nebo 256 bitů. Je mnohem bezpečnější a také rychlejší. Dnes nejvhodnější pro používání.**

PFS	**Perfect Forward Secrecy - říká, jestli se v quick módu mají použít čísla vygenerovaná v main módu. Pokud je na yes, čísla se generují znovu. Je to bezpečnější, ale trochu náročnější na výkon a čas při sestavování spojení. Při dnešních výkonech je zdržení nepostřehnutelné.**
tunelovací a transportní mód	**Při transportu dat se ipsec může chovat dvěma způsoby. Zašifruje datovou část a zbytek, IP záhlaví, nechá bez změny, a nebo zašifruje celý paket a přidá novému paketu nové IP záhlaví. První, transportní, mód vyžaduje, aby pakety mezi ipsecovými zařízeními mohly chodit i bez šifrování. Druhý, tunelovací mód, umožňuje propojit dvě sítě s neveřejnými adresami. Rozdíl je ve větším overheadu v tunelovacím módu a mohou se pak vyskytnout problémy s fragmentací paketů. Na Linuxu jsem se ale s problémy s fragmentací nesetkal. Na Cisco routerech problémy s fragmentací bývají a tam se to řeší tak, že se vytvoří gre tunel a šifrují se až pak gre pakety. Je to nelogické, ale funguje to. Cisco zařízení mají při použití přídavných šifrovacích modulů, VPN akcelerátorů, až 5x větší výkon při tunelovacím módu. Na to je také třeba pamatovat.**  **transportní mód**  **tunelovací mód**

IKE, ISAKMP, PLUTO

Jak již jsem psal výše, IKE se stará o počáteční navázání spojení a především o autentizaci. Autentizací zde rozumíme to, že pokud chci s někým, dle konfiguračního nastavení, navázat spojení, musím si být jistý, že je to on. A především, že naši počáteční domluvu nikdo nerušil a i pokud poslouchal, že mu to není nic platné. Je důležité si uvědomit, že musíte vytvořit zabezpečené spojení skrz prostředí, kde vás může kdokoliv slyšet. Pro lepší představu si vemte místnost plnou lidí a vy si potřebujete s někým promluvit tak, aby z vaší komunikace nemohl nikdo jiný těžit. Tomu, že vás uslyší, bohužel nemůžeme zabránit.

Main mód za účelem navázání spojení sestavuje spojení na UDP 500. Port 500 je cílový i zdrojový. Někde se také můžete setkat s pojmenováním, že je navázán jeden obousměrný tunel, samotná data pak proudí dvěma jednosměrnými tunely navázanými quick módem. To je dáno asymetrickým šifrováním ipsecu, zašifrujete jedním klíčem, ale rozšifrovat musíme druhým. U asymetrické šifry se klíč na zašifrování nedá použít na rozšifrování zprávy.

V rámci main módu se tedy posílá šest zpráv. Výsledkem těchto zpráv je autentizace obou stran a šifrovací klíč pro posílání prvních navazovacích zpráv quick módu. Nejdříve vysvětlím, jak onen šifrovací klíč vznikne. Pro zjednodušení to ukážu na skutečně jednoduchém příkladu se symetrickou šifrou. Symetrická šifra má jeden klíč pro šifrování i dešifrování. Příklad, který uvádím, je velice jednoduchý, a proto, jestli to bude někdo zkoušet, zapomeňte na komplexní čísla a také na kalkulačky. To aby vynikla výpočetní i časová náročnost tohoto procesu.

5 * 2 * 3 = 30

Zde jsme si vytvořili klíč 30 a pro zjednodušení si vezměte zprávu napsanou v čistém textu. Každému znaku přiřaďte číslo, třeba podle ASCII tabulky. Pro zašifrování vám stačí přičíst ke každému číslu ASCII znaku klíč 30 (pokud dojdeme nad 255, skáčeme zpátky na nulu) a vyjde vám z toho již dosti obtížně čitelný text. Počítače to provádějí tak, že klíč je binární sada 1 a 0, a mezi daty a klíčem se provádí logická funkce XOR. Pokud takto zašifrovanou textovou zprávu někdo dostane a ví, že má od každého ASCII znaku zašifrované zprávy odečíst 30, výsledek dostane poměrně rychle. Pokud je ale klíč složen ze dvou velkých prvočísel, je řešení v rozumném čase prostě nemožné. A na tomto podobném principu, ovšem složitější v tom, že klíče jsou dva, funguje i ipsec.

Klíč tedy máme, ale jak jsme se k němu dostali, aniž by to někdo zjistil, že klíč je 30, i když poslouchá naši domluvu? Odpověď je právě v tom násobení. Při sestavování klíče si strana A vygenerovala číslo, zde 5. Strana B číslo 2. Strana A pošle straně B svoje vygenerované číslo a strana B udělá pro stranu A to samé. Číslo 3 je v příkladu sdílený klíč. Obě strany tedy mají všechna čísla a mohou vypočítat daným vzorcem, který je všeobecně znám, šifrovací klíč. Asi si někdo řekne, že to je nesmysl, když znám vzorec a že jsem odposlechl čísla 5 a 2, mohu si klíč sestavit. To je omyl, nevíte třetí číslo, ze kterého se šifrovací klíč skládá. Takže si můžete dát do hromady 5*2*x=šifrovací klíč, co vám to ale pomůže? I v případě, že budeme za klíč uvažovat čísla 0-9, získáme 10 různých klíčů a pomocí nich se musíme pokusit zprávu přečíst. Kolik úsilí a času nám zabere takto rozšifrovat byť i jedno slovo? Pokud nejste génius, zcela jistě 100x déle, než kdybyste klíč znali. Zpráva mezitím už třeba pozbyde platnosti. Dalším opatřením, které se zde implementuje, je, že klíč platí jen po určitou dobu nebo že platí jen pro přenesení určitého objemu dat. Při výměně dat mezi A a B stanovíte znovupočítání klíče každou minutu a nebo 10 slov. Po uplynutí minuty nebo přenesení deseti slov se šifrovací klíč počítá nový a čísla z příkladu 5 a 2 se změní. Pokud byste chtěli zprávu rozšifrovat jako záškodník, musíte si také dát pozor na včasnou změnu klíče, jinak by vám zase vycházely nesmysly.

Navíc pokud ipsec pojme podezření, že mu došel poškozený paket a že by mohlo dojít k narušení bezpečí obsahu - paket raději zahodí a iniciuje nové počítání klíče. Přesto, že je náš příklad matematicky pro první třídu, již to tak jednoduché není, že? 😇

Jednoduchý příklad pro počítání klíče v main módu máme za sebou a teď se tedy můžeme vrátit k oněm šesti zprávám:

1. Pokud chce strana A navázat spojení, pošle straně B své schopnosti. Třeba: "Použiji sdílený klíč, hashovací algoritmus MD5, kryptovací metodu AES o velikosti klíče 128 bitů, pro generování náhodného velkého prvočísla (v příkladu jsme generovali čísla 5 a 2) DH skupinu 5, klíč bude platit hodinu nebo bude pro přenesení 100 MB dat." Tento soubor schopností se nazývá *policy* a zařízení jich může poslat více s různými kombinacemi.
2. Strana B odpoví straně A, jakou policy akceptovala. Policy jde jako čistý text výše zmíněným obousměrným tunelem. Do našeho příkladu by se dala zanést jako domluva, jak velká čísla budeme generovat a jakou metodou budeme posouvat znaky - šifrovat. Vzoreček pro výpočet šifrovacího klíče je pevně dán standardem.
3. Před vysláním třetí zprávy dojde na základě parametrů z policy k vygenerování našeho velkého prvočísla. Generování jakéhokoliv náhodného čísla je pro počítač problém, není na to stavěn, a tak se jako generátory náhodnosti berou různé zdroje. Brumy ze zdroje a nebo, jak jsem nedávno četl o společnosti Via, z teploty cpu 😇 Toto číslo se pošle straně B.
4. Strana B také vygeneruje číslo, ze kterého se šifrovací klíč počítá, a pošle ho straně A. Upozorňuji, že toto je dost zjednodušený model, protože čísel se kvůli kontrole a výpočtu pomocí skutečného vzorečku generuje více.
5. Před dalším paketem se provede výpočet klíče. Pak se vezmou známá data, vypočítaným klíčem se zašifrují a pošlou straně B. Paket jde stále obousměrným tunelem, ale už v šifrované podobě.
6. Strana B také spočítá klíč a pošle ho straně A. Zároveň vezme spočítaný klíč a pokusí se s ním přečíst paket poslaný ve 5. zprávě od A. Pokud se to povede straně A i B, obě prohlásí quick mód za úspěšně ukončený.

Tímto se obě strany bezpečně poznaly a vytvořily šifrovací klíče pro přenos zpráv pro druhý mód.

Jestliže vám navazování ipsecu selže v této fázi, bude určitě chyba v konfiguraci. Nejjednodušší způsob, jak problém odhalit je zapnout debug. Zde vám jak Openswan tak i Cisco velmi přesně popíší, při jaké zprávě došlo k chybě. V debugu totiž přímo uvidíte jednotlivé stavy MM (main modu).

Jak může dojít k chybě při spojování? Pokud používáte Linux proti Linuxu, tak asi nijak, když si dáte pozor na stejné parametry v konfiguraci. Ale když spojujete zařízení různých výrobců nebo

platforem, může nastat problém. Každé ipsecové zařízení má určité defaultní parametry, které nejsou vidět v konfiguraci, a problém může být na světě. Proto pro jistotu všechny parametry raději určete přímo sami, je o možný problém méně. Openswan má kupříkladu defaultně DH skupinu 5, ale Cisco má defaultně skupinu 2; než mi to došlo, chvilku to trvalo.

Jen pro úplnou představu přikládám skutečné vzorečky, které se používají, i když by asi někdo našel mnohem lepší. Generovaných proměnných a počítání je tam opravdu o dost více. Každá fáze je oddělena prázdným řádkem. PRF je pseudonáhodná funkce, SKYID je pak už vlastní šifrovací klíč. Hash na konci je poslední zpráva s kontrolou šifrování. Index i nebo proměná a je iniciator, index r nebo proměná b je responder.

$$CKY-I = md5\{(src\ ip, dest\ ip), Random\ Number\}$$
$$CKY-R = md5\{(src\ ip, dest\ ip), Random\ Number\}$$

$$DH\ Public\ Value = x_a = g^a \bmod p$$
$$DH\ Public\ Value = x_b = g^b \bmod p$$

$$Initiator\ Nonce = N_i$$
$$Responder\ Nonce = N_r$$

$$Shared\ Secret = (X_b)^a \bmod p = (X_a)^b \bmod p = g^{ab}$$

$$SKYEID = PRF(Pre-shared\ Key, N_i|N_r)$$
$$SKYEID_d = PRF(SKYEID, g^{ab}|CKY-I|CKY-R|0)$$
$$SKYEID_a = PRF(SKYEID, SKYEID_d|g^{ab}|CKY-I|CKY-R|1)$$
$$SKYEID_e = PRF(SKYEID, SKYEID_a|g^{ab}|CKY-I|CKY-R|2)$$

$$Hash-I = PRF(SKYEID, CKY-I, CKY-R, Pre-shared\ Key(PK-I), SA\ Payload, Proposal+Transforms, ID_i)$$
$$Hash-R = PRF(SKYEID, CKY-I, CKY-R, Pre-shared\ Key(PK-R), SA\ Payload, Proposal+Transforms, ID_r)$$

Vzorečky

Příště: IPSEC, KLIPS, druhá fáze quick mód.

Související články
IPSec v kernelu 2.6 - I
IPSec v kernelu 2.6 - II
Zabezpečení sítí Cisco
Bezpečnost v Linuxu
Velký průvodce protokoly TCP/IP: Bezpečnost
Šifrované filesystémy - I
Šifrované filesystémy - II
IPTraf - sledování sítě v reálném čase
Linuxové DMZ - I

Odkazy a zdroje
Domovská stránka Openswanu
Dokument readme k xauthu
OpenVPN na Rootu
Ipsec v kernelu 2.6 na abclinuxu

Další články z této rubriky
IPSec na Linuxu, krok za krokem - 2
IPSec na Linuxu, krok za krokem - 1
Březnové jarní Abíčko
SSL - 2 (elektronický podpis)
Ruby pro začátečníky - 3 (datové typy)

Portál AbcLinuxu, přihlášen michaels, 24. březen 2006 09:17

IPSec na Linuxu, krok za krokem - 2

dnes 00:01 | Jan Vondráček

Pracuji už dlouhou dobu s Cisco zařízeními, routery, pixy a další. Můj takový koníček je zkoušet nahrazovat služby, která Cisco zařízení poskytují, Linuxem. Chci se s vámi podělit o své zkušenosti. Druhá fáze, quick mód, se stará o výpočet vlastního šifrovacího klíče určeného pro přenos dat. Quick mód posílá, jako Main mód, tři zprávy.

IPSEC, KLIPS

Druhá fáze, quick mód, se stará o výpočet vlastního šifrovacího klíče určeného pro přenos dat. Quick mód posílá, jako Main mód, tři zprávy. Zprávy jdou ale již pomocí dvou jednosměrných šifrovaných tunelů, šifrovací klíč pro přenos těchto zpráv je použit z 1. fáze. Tato část je vlastně trochu ořezaný Main mód:

1. **Posílají se policy s vygenerovanými čísly pro druhou fázi. Zde je pro nás důležitý parametr pfs, viz tabulka pojmů. První paket jde od A do B.**
2. **Strana B pošle svá čísla a policy straně A.**
3. **A pošle B testovací data zašifrovaná již klíčem pro přenos dat. Tato zpráva se již nepotvrzuje a po ní už jedou data.**

Quick mód, a pak i samotný přenos dat, probíhá protokolem ESP. To je protokol z rodiny ip a má číslo 50 (jen pro upřesnění TCP má číslo 6, viz /etc/protocols). Tento protokol nemá porty, a proto může být občas problém jej propustit natem, ale iptables s tím nemá problémy, cisca to mají až od nejnovějších iosů. Tady máte příklad, jak dostat ipsec skrz iptablesový firewall (vIPa - je veřejná ip strany A, pIPb - neveřejná ip strany B atd.):

```
iptables -t nat -A  PREROUTING -i eth0 -p 50 -j DNAT --to-destination pIPb
iptables -t nat -A  PREROUTING -i eth0 -p udp --dport 500 -j DNAT --to-destination pIPb:500
iptables -A FORWARD -s vIPa -d vIPb -p udp --dport 500 -j ACCEPT
iptables -A FORWARD -s vIPa -d vIPb -p 50 -j ACCEPT
```

V případě, že nepřekládáme a ipsecový stroj má veřejnou IP, natovou částí se vůbec nezabýváme a řetězec FORWARD nahradíme řetězcem INPUT. V tomto případě je eth0 jako veřejný interface a síť je za něj maškarádována. Defaultní politika řetězce INPUT i FORWARD je DROP.

NAT-T

Nat traverzal je způsob, jak dostat ipsec skrz PAT (port address translation), pokud nemáte možnost na překladovém stroji provést statický nat. Příkladem PATu je třeba maškaráda.

Ipsec zjišťuje přítomnost natu ve fázi Main módu, UDP projde patem bez potíží. Pokud je nat-t na zařízení podporován (a podpora je zapnuta), je při první zprávě připojena hash zdrojové IP adresy stroje. Na druhé straně se obdržená hash porovná s nově vytvořenou hashí a pokud se neshodují, stojí v cestě nat. Ipsec se pak v druhé fázi přepíná na jiný protokol než ESP, a to na UDP 4500, který natem projde již bez potíží. Openswan podporu nat-t má, jen se musí zapnout. Cisca ji mají defaultně zapnutou, zapíná se sama podle potřeby.

Program Openswan

V Cisco, nebo jiných zařízeních, je ipsec součástí IOSu, do Linuxu jej musíme dodat. Openswan sestává ze tří částí. První jsou shellové skripty, které usnadňují spouštění. Jsou bashové a perlové, i když perl není pro samotnou funkci nebo kompilaci nutný, a tudíž nemusí být nainstalován. Druhá část jsou binární programy jako pluto pro sestavování spojení. A poslední je

kernelový modul, který provádí vlastní šifrování. Protože se o šifrování dat stará jádro, je to velmi výkonné řešení. Můj pokusný stroj, P120 na 512kb lince, byl při plném vytížení linky zatížen zhruba na 10%, což se výkonnostně vyrovná Cisco PIXu 506.

Openswan můžete získat jako balíček pro svou distribuci, ale pokud používáte jiný kernel, než který je v distribuci, stejně vám to k ničemu nebude. Nejjednodušší je program zkompilovat, pro kompilaci budete potřebovat tyto balíčky:

binutils-2.14.90.0.6-i486-1	**iproute2-2.4.7_now_ss020116_try-i386-4**
gcc-g++-3.2.3-i486-1	**kernel-headers-2.4.22-i386-1**
gcc-objc-3.2.3-i486-1	**kernel-source-2.4.22-noarch-3**
gcc-3.2.3-i486-2	**make-3.80-i386-1**
glibc-2.3.2-i486-1	**m4-1.4-i386-2**
gmp-4.1.2-i486-2	**perl-5.8.0-i486-5**

Seznam balíčků je pro Slackware, ale ostatní distribuce to budou mít podobné. Uvádím zde i balík iproute2, který obsahuje program ip. Openswan ho využívá pro to, aby po sestavení spojení automaticky naroutoval přenášenou síť do ipsecového tunelu. Kernel-source můžete použít samozřejmě jiný, než máte právě na stroji spuštěný, ale verzi kernelu je nutno přesně dodržet. U zbytku programů jen nesmíte kompilovat na vyšší verzi gcc, než používá váš systém, jinak je to ok.

Kompilujeme: `make KERNELSRC=/usr/src/linux/ programs module` **pro kernel 2.4 i pro kernel 2.6. Pokud potřebujeme jen modul, pak samozřejmě vynecháme parametr** `programs`**. Pak už zbývá jen** `make install`**, kernelový modul musíte nakopírovat manuálně. Já jsem modul dal do** `/lib/modules/2.4.24/kernel/crypto/`**, nezapomeňte na** `depmod -a`**, než modul nahrajete** `modprobe ipsec`**. Modul ipsec.o se nachází v podadresáři modobj, tam, kde máme zdroj openswanu.**

Méně zkušeným správcům doporučuji, aby své stroje měli vždy postavené na balíčcích příslušné distribuce. Jednak z důvodu přehlednosti, ale také bezpečnosti. Představte si, že postavíte stroj jako bezpečnou bránu a necháte na něm kompilátor i se zdrojovými kódy. Potenciální útočník, pokud se do stroje dostane, má tak k dispozici veškeré nástroje pro tvorbu vlastních programů a nahrazování původních programů. A hackování linuxových strojů je především o tomhle. Proto určitě pro ostrý provoz kompilujte na jiném stroji a openswan instalujte jako balíček. Pro vytvoření instalačních balíčků existuje spousta programů, třeba Checkinstall. Ovšem ten si vytváření balíčku pro openswan vylámal zuby. Co teď? Řešení je jednoduché a zvládne ho každý. Vytvořte si adresář s adresářovou strukturou podle přiloženého souboru, do adresářů nakopírujte zkompilované programy a balíček vám pak vytvoří krásně makepkg. Makepkg je pro Slackware, ale ostatní distribuce také obsahují příslušné nástroje na tvorbu balíčků. Mnou vytvořený balík vám také dám k dispozici, moduly pro kernel 2.4.24 a 2.4.22 se po instalaci nakopírují do homu roota, odkud je musíte ještě podle jádra zkopírovat k ostatním modulům.

Jestli máte v pořádku nainstalováno a zaveden modul ipsecu, můžeme pokročit dále. Příkazem `ipsec verify` **můžete otestovat, podle dokumentace k Openswanu, zda vše v pořádku běží. Testuje to perlový skript a myslím, že to ani nemusíte dělat, není to k ničemu. Test vám bude třeba řvát na nepřítomnost rsa klíčů, záznamy v DNS a dalších. Jenže proč generovat RSA klíče, když je nebudeme používat? Ne, uděláme to jednodušeji. Jen si ověřte přes lsmod, že máte modul ipsec zaveden,** `ifconfig -a` **vám musí ukázat nová rozhraní ipsec0 až ipsec4 a vrhneme se rovnou na konfiguraci. Po nakonfigurování stačí napsat** `ipsec setup start`**, případně stop nebo restart a VPN se spustí. Při spuštěné VPN si status můžete zkontrolovat příkazem** `ipsec whack --status`**, více příkazů potřebovat nebudete.** 😇

Konfigurace VPN

Konfiguraci VPN si ukážeme na vzorovém příkladu. Máme dvě lokality, kde na jedné straně máme Cisco router, do internetu připojený sériovou linkou. Na druhé straně linuxové PC připojené přes wifi. Každá strana má vlastní přístup do internetu, potřebujeme je tedy propojit jen jako dvě LAN.

Na linuxové straně musíme provést konfiguraci ve dvou souborech /etc/ipsec.secrets, kde je klíč. A /etc/ipsec.conf, kde je konfigurace spojení.

```
konfigurace Linuxu
------------------

/etc/ipsec.secrets
213.211.4.208 194.15.7.21: PSK "abc123"

/etc/ipsec.conf
config setup
        interfaces="ipsec0=eth0"
        klipsdebug=none
        plutodebug=none

conn cisco
        type=tunnel
        authby=secret
        left=213.211.4.208
        leftsubnet=192.168.17.0/24
        leftnexthop=213.211.4.1
        leftid=213.211.4.208
        right=194.15.7.21
        rightsubnet=192.168.1.0/24
        ike=3des-md5
        esp=3des-md5
        keyexchange=ike
        pfs=yes
        auto=start


konfigurace cisca
-----------------

crypto isakmp policy 1
 encr 3des
  hash md5
  authentication pre-share
  group 5
 crypto isakmp key abc123 address 213.211.4.208 no-xauth
!
!
crypto ipsec transform-set MYSET esp-3des esp-md5-hmac
!
crypto map MYMAP local-address Loopback0
crypto map MYMAP 1 ipsec-isakmp
 set peer 213.211.4.208
 set transform-set MYSET
 set pfs group5
 match address 101
!
interface Loopback0
 ip address 194.15.7.21 255.255.255.255
!
interface Serial0
 ip unnubered Loopback0
 crypto map MYMAP
!
ip route 0.0.0.0 0.0,0.0 Serial0
!
access-list 101 permit ip 192.168.1.0 0.0.0.255 192.168.17.0 0.0.0.255
```

Na tomto si vysvětlíme jednotlivé položky konfigurace. Další položky, které zde nemáme, si můžete přečíst v dokumentaci v souboru ipsec.conf. Konfigurace má dvě sekce, `config setup`**, která je společná pro všechna spojení, můžeme ji nazvat globální. A pak konkrétní část pro každé spojení. Každé spojení může tak mít jiné parametry enkrypce.**

`interfaces` **určuje, na kterém interface ipsec visí. Pokud budete mít tunelů více, můžete tam dát** `%defaultroute` **(interface s defaultní routou) nebo jednotlivé tunely oddělit mezerou** `ipsec0=eth0 ipsec1=eth0`**. U cisca je to pověšení cryptomapy na interface, zde** `Serial0`**.**

`klipsdebug` **a** `plutodebug` **je chování openswanu z hlediska debugování. To je užitečné při problémech, ale pro praktický život určitě ne, debug generuje ohromné množství logů. Ale určitě doporučuji, aby si každý alespoň jednou debug zapnul a pak se pořádně na log podíval, aby viděl, jak se ipsec spojuje.**

`nat_traversal` **parametr zde není uveden, defaultně je ne.** `nat_traversal=yes` **podporu zapne a zároveň doporučuji použít leftid (viz dále).**

`type=tunnel` **zapíná tunelovací mód, další volbou je mód** `transport`**. Více říká tabulka na začátku článku s popisem terminologie. U cisca je tunelovací mód výchozí, mění se v transform setu.**

`authby=secret` **určuje, že autentizace bude prováděna sdílenými klíči. Pozor defaultně je volba na RSA.**

`left...` **a** `right...` **jsou parametry pro IP adresy. Není řečeno, že levá strana musí být lokální a pravá vzdálená, to proto, aby se stejný konfigurační soubor nechal použít na obou stranách VPN, tedy pokud spojujeme dva openswany. Která strana je lokální, si openswan určí podle adresování interfacůů. Subnet určuje provoz, který se bude kryptovat, u cisca je to access-list 101. Je možno zadat i položku** `%any`**, což po spojení ipsecu dá defaultní routu do ipsecového tunelu. Samotný parametr left a right určuje konce ipsecového tunelu. Id se používá především u natu nebo pokud máme více interfacůů, kterými může provoz odcházet. Takto přesně určíme, jaká IP se má pro daný tunel použít. Ekvivalent u cisca je** `crypto map local address`**. Next hop není nutno použít.**

`keyexchange` **určí, jaká metoda se má použít pro sestavení tunelu, zde automatická pomocí ike. U cisca je to volba ipsec-isakmp v parametru vytváření cryptomapy.**

`pfsgroup` **sice uvedená není, je použitá defaultní skupina 5, ale takto ji můžeme také měnit.** `pfsgroup=1024` **je tedy skupina2. Tento parametr mění skupinu pro ike i ipsec.**

`ike` **nastavuje policy, nebo-li metodu šifrování, dh skupiny a hashování pro Main mód. U cisca to vidíme v** `isakmp policy 1`**. Přípustné hodnoty jsou:**

- **3des-md5-modp1536, 3des-md5-modp1024 (zde navíc měníme i dh skupinu na 2), 3des-md5, 3des-md5-96 (96 říká že MD5 hash 96-bitová)**
- **3des-sha1-modp1536, 3des-sha1-modp1024**
- **aes128-sha1, aes128-sha1-modp1536, aes256-sha1-modp1024 (u aesu je ještě nutno určit šířku klíče, šířky klíče jsou uvedeny v tabulce pojmů)**
- **aes128-md5, aes128-md5-modp1536**
- **!aes256-md5 (nařizuje striktní použití aes 256 a md5)**

`esp` **nastavuje parametry pro quick mód a pro přenos dat. Hodnoty jsou stejné jako pro ike. U cisca se tyto parametry nastavují v crypto mapě.**

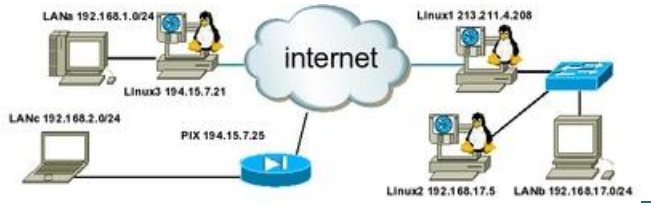
`pfs` **zapne pfs pro quick mód.**

`auto`**, tento parametr bude nahazovat tunel hned po startu ipsecu (ipsec setup start). To budeme asi také chtít, abychom to nemuseli dělat manuálně.**

`keylife` **parametr zde není uveden, určuje životnost šifrovacího klíče. Defaultně je 8 h, maximálně 24 h, což je i defaultní hodnota v ciscu. Mění se keylife=24h. Security asociation zůstává při znovupočítání klíče sestavena, probíhá tedy znovu jen quick mód.**

`ikelifetime` **říká, jak dlouho bude sestavená security asociation - neboli jak dlouho si zařízení budou věřit, že se ověřila. Tato hodnota bývá delší než životnost šifrovacího klíče.**

Tímto máme nakonfigurováno, pak stačí dát jen `ipsec setup start` **a spojení se nám vytvoří. Po sestavení spojení se nám objeví nový interface, zde ipsec0. Openswan pomocí programu ip sám naroutuje síť uvedenou jako subnetlan do tunelu. Linux přistupuje k interfacu jako k fyzickému, lze ho tedy měřit přes mrtg nebo iptraf. Restart, třeba při přidání nového tunelu, uděláte překvapivě** `ipsec setup restart` **a ipsec zastavíte** `ipsec setup stop`**. Sestavené tunely uvidíte ve výpisu** `ipsec whack --status`**.**



V druhém příkladu vám ukážu konfiguraci pixu a také konfiguraci složitější VPN. Linuxové stroje zde máme tři a síť má topologii hvězdy. Stroj Linux1 není ipsecový, běží na něm nat a stroj dělá bránu do internetu. Stroj 2 je ipsecová centrála, končí na něm dva tunely. Pix a Linux3 jsou normální VPN pobočky.

Linux1

Je brána do internetu, firewalluje a natuje maškarádou. Proto na něm musíme udělat prostupy a překlady. Použijeme iptables a v konfiguraci bude tohle:

```
iptables -t nat -A  PREROUTING -i eth0 -p 50 -j DNAT --to-destination 192.168.17.5
iptables -t nat -A  PREROUTING -i eth0 -p udp --dport 500 -j DNAT --to-destination 192.168.17.5:500
iptables -A FORWARD -s 194.15.7.21 -d 213.211.4.208 -p udp --dport 500 -j ACCEPT
iptables -A FORWARD -s 194.15.7.21 -d 213.211.4.208 -p 50 -j ACCEPT
iptables -A FORWARD -s 194.15.7.25 -d 213.211.4.208 -p udp --dport 500 -j ACCEPT
iptables -A FORWARD -s 194.15.7.25 -d 213.211.4.208 -p 50 -j ACCEPT
route add -net 192.168.1.0/24 gw 192.168.17.5
route add -net 192.168.2.0/24 gw 192.168.17.5
```

Linux2

Na tomto stroji nesmíte zapomenout zapnout routování (`echo 1 > /proc/sys/net/ipv4/ip_forward`**).**

```
/etc/ipsec.secrets
213.211.4.208 194.15.7.21: PSK "ajipI$%A*773jHU"
213.211.4.208 194.15.7.25: PSK "prilis_zluty_kun"

/etc/ipsec.conf
config setup
        interfaces=%defaultroute
        klipsdebug=none
        plutodebug=none

conn Linux3
        type=tunnel
        authby=secret
        left=192.168.17.5
        leftsubnet=192.168.17.0/24
        leftnexthop=192.168.17.1
        leftid=213.211.4.208
        right=194.15.7.21
        rightsubnet=192.168.1.0/24
        ike=aes256-md5
        esp=aes256-md5
        keyexchange=ike
        pfs=yes
        auto=start
```

```
conn pix
        type=tunnel
        authby=secret
        left=192.168.17.5
        leftsubnet=192.168.17.0/24
        leftnexthop=192.168.17.1
        leftid=213.211.4.208
        right=194.15.7.25
        rightsubnet=192.168.2.0/24
        ike=3des-md5
        esp=3des-md5
        keyexchange=ike
        pfs=yes
        auto=start
```

Linux3

```
/etc/ipsec.secrets
213.211.4.208 194.15.7.21: PSK "ajipI$%A*773jHU"

/etc/ipsec.conf
config setup
        interfaces=%defaultroute
        klipsdebug=none
        plutodebug=none

conn Linux3
        type=tunnel
        authby=secret
        left=192.168.17.5
        leftsubnet=192.168.17.0/24
        leftnexthop=192.168.17.1
        leftid=213.211.4.208
        right=194.15.7.21
        rightsubnet=192.168.1.0/24
        ike=aes256-md5
        esp=aes256-md5
        keyexchange=ike
        pfs=yes
        auto=start
```

PIX

```
X Version 6.3(4)
interface ethernet0 auto
interface ethernet1 auto
nameif ethernet0 outside security0
nameif ethernet1 inside security100
hostname pix
domain-name openswan.cz
clock timezone CET 1
fixup protocol dns maximum-length 512
fixup protocol ftp 21
fixup protocol h323 h225 1720
fixup protocol h323 ras 1718-1719
fixup protocol http 80
fixup protocol rsh 514
fixup protocol rtsp 554
fixup protocol sip 5060
fixup protocol sip udp 5060
fixup protocol skinny 2000
fixup protocol smtp 25
fixup protocol sqlnet 1521
access-list inside permit ip any any
access-list inside permit icmp any any
access-list nonat permit ip 192.168.2.0 255.255.255.0 192.168.17.0 255.255.255.0
access-list ven permit ip 192.168.2.0 255.255.255.0 192.168.17.0 255.255.255.0
pager lines 24
logging on
logging timestamp
logging console warnings
logging buffered debugging
```

```
icmp permit any outside
icmp permit any inside
mtu outside 1500
mtu inside 1500
ip address outside 194.15.7.25 255.255.255.128
ip address inside 192.168.2.1 255.255.255.0
ip audit info action alarm
ip audit attack action alarm
pdm history enable
arp timeout 14400
global (outside) 1 interface
nat (inside) 0 access-list nonat
nat (inside) 1 192.168.2.0 255.255.255.0 0 0
route outside 0.0.0.0 0.0.0.0 194.15.7.1 1
timeout xlate 3:00:00
timeout conn 1:00:00 half-closed 0:10:00 udp 0:02:00 rpc 0:10:00 h225 1:00:00
timeout h323 0:05:00 mgcp 0:05:00 sip 0:30:00 sip_media 0:02:00
timeout uauth 0:05:00 absolute
aaa-server TACACS+ protocol tacacs+
aaa-server TACACS+ max-failed-attempts 3
aaa-server TACACS+ deadtime 10
aaa-server RADIUS protocol radius
aaa-server RADIUS max-failed-attempts 3
aaa-server RADIUS deadtime 10
aaa-server LOCAL protocol local
ntp server 195.113.144.201 source outside
no snmp-server location
no snmp-server contact
snmp-server community public
no snmp-server enable traps
floodguard enable
sysopt connection permit-ipsec
crypto ipsec transform-set MYSET esp-3des esp-md5-hmac
crypto map mapa 1 ipsec-isakmp
crypto map mapa 1 match address ven
crypto map mapa 1 set peer 213.211.4.208
crypto map mapa 1 set transform-set MYSET
crypto map mapa interface outside
isakmp enable outside
isakmp key ******** address 213.211.4.208 netmask 255.255.255.255 no-xauth
isakmp identity address
isakmp policy 1 authentication pre-share
isakmp policy 1 encryption 3des
isakmp policy 1 hash md5
isakmp policy 1 group 5
isakmp policy 1 lifetime 86400
telnet timeout 5
ssh 213.211.4.208 255.255.255.255 outside
ssh timeout 60
console timeout 0
terminal width 80
```

Jak vidíte, je to stále variace na jedno téma. Jen si musíte dát pozor na správně napsané adresy, v tom množství se člověk snadno uklepne, a také na zapnuté routování. Sestavené tunely doporučuji testovat pingem a sledovat při tom na jednotlivých routerech, jaké protokoly z nich lezou. Ipsec můžete mít sice spojený, ale chybou v routování vám můžou data téci mimo tunel. Vím o čem mluvím, také se mi to už stalo.

X-auth

Toto je použití ipsecu v případě, že jedna ze stran nemá pevně danou IP adresu. Ipsec je na pevných IP založen a pokud se jedna strana s ip mění, nebude fungovat. Toto je běžné při připojení dial-upem, ADSL bez pevné IP nebo u mobilního připojení. U tohoto způsobu ipsecu se zapíná místo main módu agressive mód. Ten má jen tři pakety jako quick, ale používá ještě další autentizaci uživatele jménem a heslem. Pre-share klíč je nahrazen VPN skupinou s heslem. Klient má nakonfigurovanou pevnou IP VPN serveru a svou IP dynamicky přidělovanou. Sever má svou pevnou a místo klientských má nuly, které znamenají jakoukoliv IP.

X-auth jsem s openswanem nezkoušel, ani proti Ciscu. Existuje totiž mnoho VPN programů, které mi přijdou vyřešeny pro tento účel lépe. Je to především OpenVPN, které má klienty i pro ostatní OS. No a samozřejmě Cisco, které své VPN klienty má také pro všechny OS. Jejich

instalace i v Linuxu je snadná a fungování bezproblémové. Pokud někdo tento typ VPN od Cisca používá, klienta si může stáhnout zdarma, takže není co řešit. A na OpenVPN bylo již vydáno dost návodů v češtině, ale i jejich dokumentace je skutečně jednoduchá a popsaná krok za krokem. OpenVPN sice nefunguje na principu ipsecu, ale na SSL šifrování. Z hlediska fungování je to úplně jedno.

Tento způsob nasazení VPN je oproti pevném spojení dvou lokalit z bezpečnostních důvodů dost omezené, protože neumí přenášet routované sítě, které jsou za klientem. Takže jestliže budete chtít propojit dvě lokality, musíte použít způsob nastíněný výše. Cisco VPN klient prostě neumí do šifrovaného kanálu nic jiného než komunikaci od klienta a OpenVPN zase neumí u spojení peer2peer více než jeden tunel. Přestože jsem Openswan takto nezkoušel, popíšu jak se takové spojení konfiguruje podle toho, co jsem vyčetl z dokumentace:

1. **Pro autorizaci je využíván unixový PAM model, takže ho musíte mít nainstalovaný. Na webu Openswanu je to také popsáno krok za krokem. Nebo lze použít systém htpasswd.**
2. **Pak editujete Makefile.inc a dáte položku `USE_XAUTH` na true. Pro pam ještě musíte zapnout `USE_XAUTHPAM`, ale pak už normálně kompilujete a instalujete.**
3. **Po kompilaci přidáte nový modul `contrib/pam.d/pluto` do `/etc/pam.d/` a pro x-auth budou využívána unixová hesla. Nebo si vytvoříte soubor`/etc/ipsec.d/passwd` se syntaxí `username:heslo:connection_name`.**
4. **Na serverové straně vykonfigurujete:**

   ```
   /etc/ipsec.secrets
   0.0.0.0 1.2.3.4 : PSK "heslo vpn skupiny"

   /etc/ipsec.conf
   leftxauthserver=yes
   rightxauthclient=yes
   ```

   Logicky ip 1.2.3.4 je jen ukázková IP serveru, nuly v druhé IP znamenají všechny IP. Levou a pravou stranu v ipsec.conf lze samozřejmě prohodit podle vašich potřeb. Z ipsec.conf je to jen krátká ukázka, samozřejmě chybí kryptovací metody atd. Ale to jsme si už ukázali předtím.
5. **Na straně klienta je konfigurace podobná:**

   ```
   /etc/ipsec.secrets
   @groupname    1.2.3.4 : PSK "heslo vpn skupiny"

   /etc/ipsec.conf
       conn myclient
            left=%defaultroute
            leftid=@groupname
            leftxauthclient=yes
            right=1.2.3.4
            rightsubnet=192.168.1.0/24
            rightxauthserver=yes
            authby=secret
            auto=add
   ```

   Ipsec pak startujeme naprosto normálně. Je zde rozdíl v tom, že server se nepokouší aktivně spojit s klientem jako v případě peer2peer, ale čeká na něj. Při nastartování ipsecu na klientovi se klient začne spojovat se serverem a vyžádá si od uživatele jmého a heslo. Všimněte si, že v konfiguraci ipsecu nikde nefigurují uživatelská jména a hesla, to, co zde máme, je jen VPN skupina s heslem, které nahrazuje pre-share klíč. Uživatelská hesla jsou právě zcela oddělena od vlastní VPN a často bývají ověřována radiusem atd., tedy ovládána zcela vzdáleně mimo vlastní VPN.

A to je vše přátelé. Doufám, že tyto informace využijete a že vám vaše vpn sítě na Linuxu budou fungovat spolehlivě.

Související články

[IPSec na Linuxu, krok za krokem - 1]
[IPSec v kernelu 2.6 - I]

IPSec v kernelu 2.6 - II
Zabezpečení sítí Cisco
Bezpečnost v Linuxu
Velký průvodce protokoly TCP/IP: Bezpečnost
Šifrované filesystémy - I
Šifrované filesystémy - II
IPTraf - sledování sítě v reálném čase
Linuxové DMZ - I

Odkazy a zdroje

Domovská stránka Openswanu
Dokument readme k xauthu
OpenVPN na Rootu
Ipsec v kernelu 2.6 na abclinuxu

Další články z této rubriky

IPSec na Linuxu, krok za krokem - 2
IPSec na Linuxu, krok za krokem - 1
Březnové jarní Abíčko
SSL - 2 (elektronický podpis)
Ruby pro začátečníky - 3 (datové typy)

ISSN 1214-1267, (c) 1999-2006 Stickfish s.r.o.