Vysoké učení technické v Brne

Fakulta informačních technologií

Počítačové komunikace a sítě
2017/2018

Projekt 2 Varianta 2
DHCP Starvation Attack

Tomáš Lapšanský (xlapsa00) Brno, 09.04.2018

Obsah

Zadanie

Naštudovať problematiku DHCP útoku a relevantné informácie uviesť v projektovej dokumentácií. Naprogramovať aplikáciu DHCP Starvation útok, ktorý by vyčerpal adresný pool legitímneho DHCP serveru. Domonštrovať činnosť aplikácie v podmienkach Vašej testovacej siete.

Problematika

Našou úlohou bolo implementovať v jazyku C/C++ *DHCP Starvation útok*, ktorý v zjednodušenej forme spamuje DHCP server takzvanými DHCP DISCOVERY správami, ktorým je pridelená falošná *MAC adresa*, a taktiež falošné *transaction ID*. DHCP server pre dané správy dočasne alokuje miesto, a teda ich berie ako obsadené. Tento proces sa opakuje, čím sa vyčerpá *adresný pool*, a teda nie je možné dostať od DHCP serveru pridelenú adresu siete a získať prístup ku sieti. V kompletnom útoku by náš program mal reagovať aj na spätné DHCP OFFER správy, z ktorých potrebujeme vyčítať adresu serveru a nám pridelenú IP adresu, následne odošleme na server *DHCP REQUEST*, ktorý obsahuje *DHCP_SERVER_IDENTIFIER* a *REQUIRED_IP*, tým nám server zaalokuje po dobu lease time (často býva až do 24 hodín) pridelenú IP adresu, ktorú nebude môcť využiť iný host. Týmto opakovaným procesom vyčerpáme adresný pool a router nebude schopný prideľovať novým užívateľom adresy aj po ukončení nášho útoku. Proti tomuto rozšírenému útoku sa však na základe našich skúseností väčšina moderných zariadení dokáže efektívne brániť, a však, jednoduché zasielanie DISCOVERY správ sa nám overilo ako funkčné.

Implementácia

Vrámci našej implementácie sme sa rozhodli využívať takzvaný RAW paket, ktorý manuálne napĺňame potrebnými parametrami ktoré musí daný DHCP packet spĺňať v závislosti od typu nami posielaného paketu (DISCOVERY, REQUEST). Na pripojenie na DHCP server sme využívali knižnicu **pcap.h** pre jazyk C/C++, z nej sme použili funkcie *pcap_open_live* pre spojenie serveru s daným interface-om (uvedený v argumente programu) a pcap_inject pre odoslanie nami vytvoreného RAW paketu na server, a aj niekoľko ďalších potrebných štruktúr. Ďalej sme potrebovali vytvoriť podľa nájdeného vzoru štruktúru pre DHCP ktorú sme v pakete odosielali, štruktúru IP hlavičky, atď. Program po spustení overí argumenty a v prípade ich korektnosti sa pokúsi napojiť na daný interface, v prípade úspešného napojenia delíme program pomocou príkazu fork() na dve vlákna. Prvé vlákno vytvára pakety ktoré potom podľa našej potreby napľňa a neustále zasiela na DHCP server pomocou broadcastu vyššie spomínané DISCOVERY správy, ktorým generuje náhodnú MAC adresu a náhodné transaction ID. Druhé vlákno čaká na správy z DHCP serveru, na ktoré reaguje vypísaním IP adresy a následných zaslaním vyššie spomínanej REQUEST správy na server, ktorá by nám mala zabezpečiť dlhšie trvajúce pridelenie IP adresy. Táto implementácia sa nám však neoverila, pretože ako sme skôr spomínali, väčšina moderných zariadení sa proti takto ľahkému útoku dokáže brániť. Príkladom mechanizmu bránenia ktorý sa nám overil aj na našej domácej testovacej sieti bolo, že router je schopný overiť MAC adresu ktorá mu je doručená v pakete s MAC adresou odosielateľa. A však aj na základe zabezpečenia sa nám podarilo pomocou našej implementácie server odstaviť po dobu behu programu, čo je tiež forma Starvation útoku.

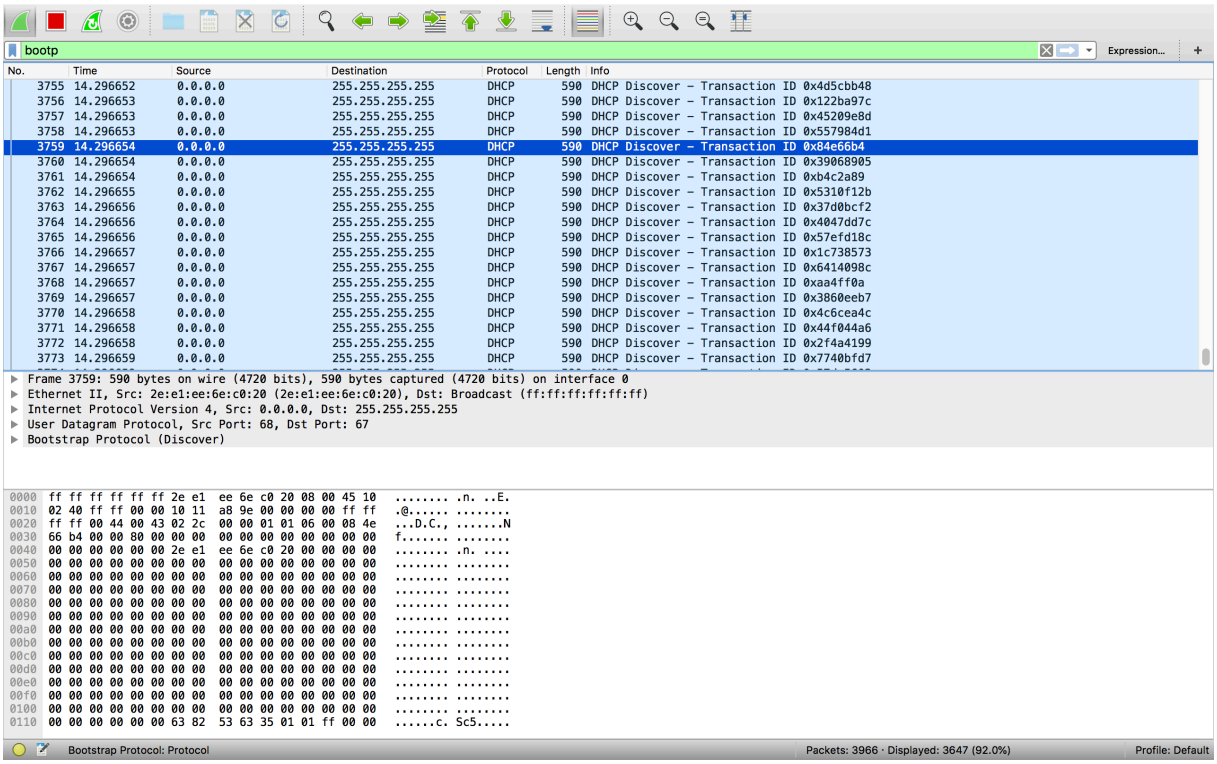
Testovanie

Vrámci testovania sme použili našu domácu sieť, v ktorej máme router ASUS RT-N14U na ktorý sme pripojili počítač ktorý realizoval útok podľa zadania. Zasielanie DISCOVERY správ sme overili pomocou nástroju Wireshark (viz. Obr. 1-2), následne sme sa pokúšali o opakované pripojenie do siete pomocou smartphonu a taktiež ďalšieho počítača, podľa očakávania neúspešne. Server však neodpovedal OFFER správami, takže sme neboli schopný overiť zasielanie REQUEST správ na server, a však projekt spĺňa zadanie aj bez tohto zasielania. V prípade že by to testovacie prvky v laboratóriu umožňovali a teda nami vytvorené REQUEST správy by boli zasielané, nezaručujeme úplnú korektnosť danej implementácie. V prípade problémov s REQUEST správami postačí v rámci súboru *ipk-dhcpstarve.cpp* zakomentovať riadok 391, a implementácia bude fungovať pomocou už spomínaného spamovania.

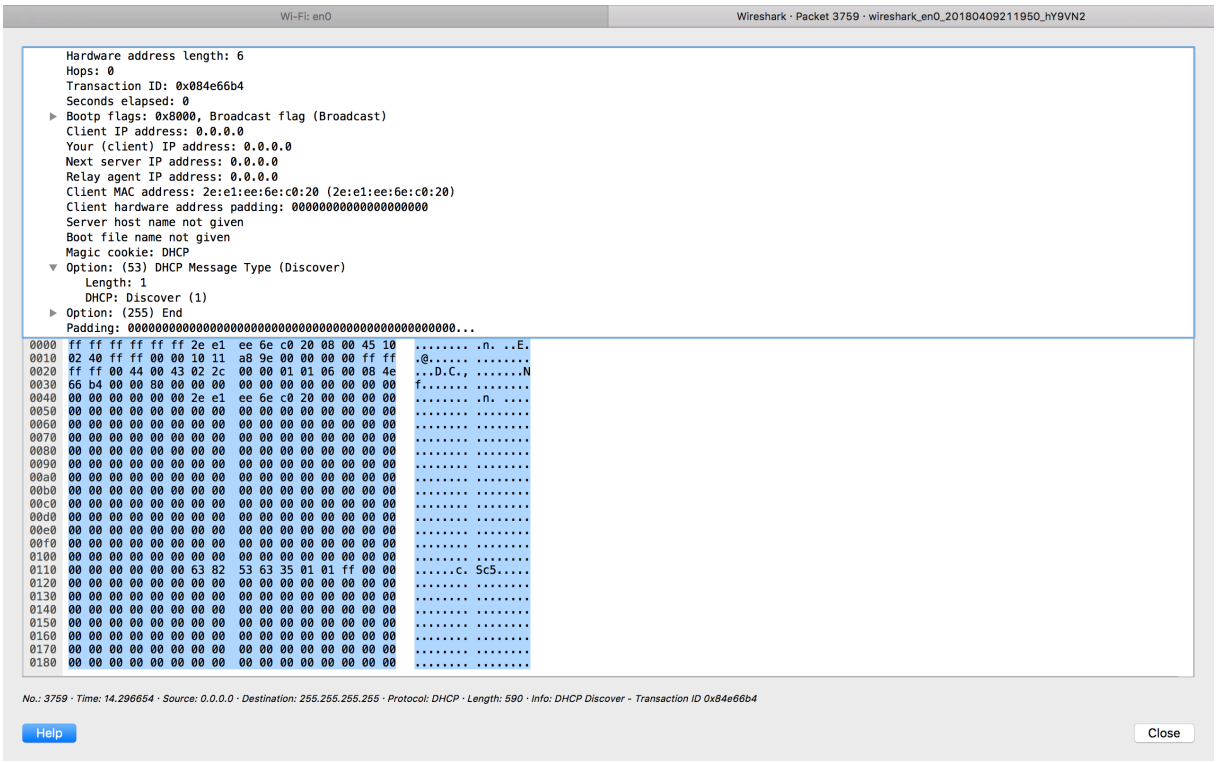
Kompilácia a spustenie

K projektu je vytvorený jednoduchý Makefile, obsahujúci cieľ all ktorý sa pomocou prekladaču gcc postará o korektné preloženie nami vytvoreného súboru. Spustiteľný súbor je terminálová aplikácia ktorá sa spúšťa pomocou zadania
./ipk-dhcpstarve -i <interface>
Kde interface udáva interface pomocou ktorého sme napojený na DHCP server, v Linuxových a Unixových distribúciách je možné zistiť dostupné rozhrania pomocou príkazu *ifconfig*, program je kompatibilný s Linuxom (funguje aj na nami testovanom macOS).

Príloha



Obr. 1



Obr. 2

Zdroje

[1] Cprograming.com forum - https://cboard.cprogramming.com/c-programming/124445-dhcp-discover.html
[2] Udphdr structure reference - http://www.cse.scu.edu/~dclark/am_256_graph_theory/linux_2_6_stack/structudphdr.html
[3] DHCP Demo project - http://ftp.icpdas.com.tw/pub/cd/8000cd/napdos/8000/843x883x/tcp/other/dhcp/client.c
[4] TCP/IP Guide, DHCP message format - http://www.tcpipguide.com/free/t_DHCPMessageFormat.htm
[5] RFC 2131 DHCP - http://www.faqs.org/rfcs/rfc2131.html
[6] RFC 2131 - https://www.ietf.org/rfc/rfc2131.txt
[7] RFC 2132 - https://tools.ietf.org/html/rfc2132
[8] DHCP plugin for Nagios - https://cs.uwaterloo.ca/twiki/pub/CF/DhcpDebug/dhcp.c
[9] Wiki/DHCP - https://en.wikipedia.org/wiki/Dynamic_Host_Configuration_Protocol
[10] Preventing DHCP Starvation attack by Cheers - http://www.revolutionwifi.net/revolutionwifi/2011/03/preventing-dhcp-starvation-attacks.html