Distribuovaný SSH honeypot

přepad časoprostorovou dírou 10 let zpět

Ondrej Mikle • ondrej.mikle@nic.cz • 14. 11. 2015

CZ.NIC | SPRÁVCE DOMÉNY CZ

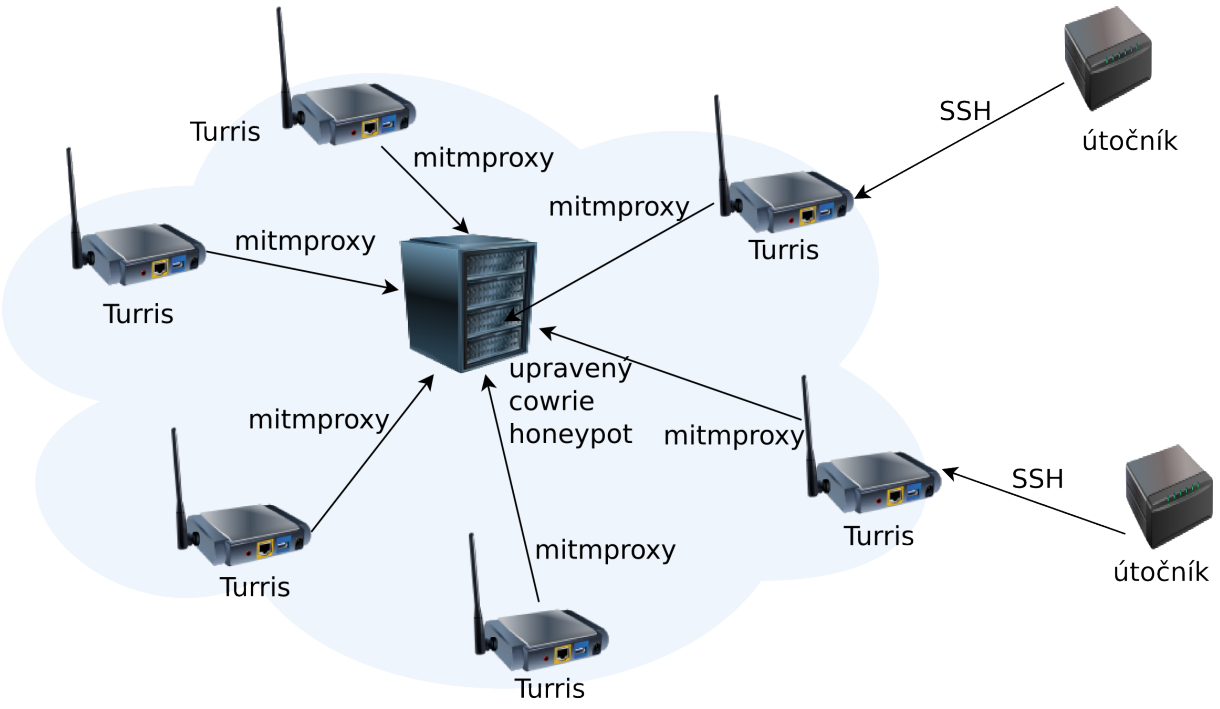
Co je Turris SSH honeypot

- uživatel si dobrovolně v nastavení může zapnout SSH port 22 zvenku
- tento bude přesměrován k nám do SSH honeypotu
- tím pádem i kdyby se v honeypotu nalezla exploitovatelná chyba, neohrozí uživatele routeru

Rychlé statistiky SSH honeypotů

- cca 200 zapojených Turrisů
- 2 000 - 32 000 SSH sessions denně
- 1 000 - 5 000 SSH sessions denně přihlášeno úspěšně s alespoň jedním příkazem
- museli jsme zakázat „root:root“ a „admin:admin“, protože ty čísla jinak začínala bý astronomická
  - jiné heslo pro uživatele „root“ a „admin“ funguje

Architektura SSH honeypotu



Cowrie

- fork kippo SSH honeypotu
  - původně náš honeypot běžel na kippo, změny do cowrie se musely mergovat manuálně
- napsané pomocí Python Twisted frameworku
  - Python Twisted slogan: *„pokud chcete i v pythonu psát jako v Java EE“*
  - umí sice spoustu věci, ale špatně dokumentované
  - „radost“ debugovat („reactive framework“)

Cowrie

- zároveň asi jediný honeypot, který můžete upravit tak, že stovky instancí sežerou jen pár GB RAM
- filesystem je v každé instanci „čistá kopie“
  - šílená implementace (list listů listů listů...jak LISP)
  - kopírují se jen metadata, obsah souborů v „paralelním filesystému“
- implementace má dost chlupatých částí, ale současný maintainer je dost aktivní
  - přijíma velmi rychle i pull requesty

Mitmproxy

- založená hádejte na čem – PythonTwisted
- vezme příchozí SSH spojení, začne odchozí spojení do SSH honeypotu
- autentizaci předává z jednoho „endpointu“ do druhého
  - tj. heslo z původního spojení útočníka se objeví na honeypotu
  - teoreticky by měla umět autentizaci klíčem, ale má to race conditiony

Rozeznávání jednotlivých Turrisů

- jednotlivé Turrisy jsou mapované podle portů
  - není to úplně ideální
  - mitmproxy každý Turris podle seriového čísla nastaví na jiný port
  - navíc se přenáší sériové číslo pro dvojitou kontrolu
- proč mitmproxy?
  - DNAT by byl jednodušší, ale ztratí se původní adresa útočníka
  - v IPv4 není místo, kam tuto informaci zakódovat
  - IPv6 zas většina Turrisů nemá (a neprošlo by to middle-boxemi)

Kdo útočí

- botnety
  - často jen průzkum
    - ifconfig, /proc/meminfo, /proc/version...
  - mnoho binárních variant z builderů, ale jen několik málo rodin
- script kiddies
- pochybuju, že nějaká útočící IP je „reálná“ IP útočníka (možná u pár script kiddies)

Botnety

Botnety

- v zásadě všechno DDoS
  - Bill Gates, PNScan.2, Tsunami, Mayday.f, Ganiw, Xor.DDoS...
  - virustotal mnoho vzorků zná
- zhruba každých pár dní se obměňují varianty
- dost pokusů o port forwarding (zakázáno)
- dělal jsem si klasifikátor
  - jenže skripty jsou Turingovsky úplné

Botnety

- způsob instalace:
  - wget, chmod, spustit
  - shell skripty
    - pozabíjet konkurenci, wget, spustit
  - přes stdin
    - cat >file; <stdin data>; chmod +x file; ./file
  - sftp
- vzorky instalovaných botnetích souborů se ukládájí pro pozdější analýzu

Skrývání

- opakuje se situace z před 10 let u Windows
- nejprve jednoduché viry/červy/trojany snažící se obsadit router, vyhodit konkurenci, později polymorfismus
- zkoušejí defaultní hesla (ekvivalent spustění exe z přílohy)
- mění argv[0], aby nebyli jednoduše vidět v „ps“
- rootkit viděn jen u jedné rodiny
  - musí matchovat běžící kernel ⇒komplikace navíc

Rozpoznávání honeypotů botnety

- honeypot je triviální na rozeznání, ale málokterý botnet se vůbec snaží o detekci
- echo -e '\x67\x61\x79\x66\x67\x74'
  - lehké odsimulovat
- cat /proc/version
  - hledá specifickou verzi kernelu
- různé složitější „shelloviny“ - true && cat <(echo $(echo foo))
  - umíme rozparsovat lexerem, ale vykonání ⇒ implementace reálného shellu

DDoS-ování od botnetů

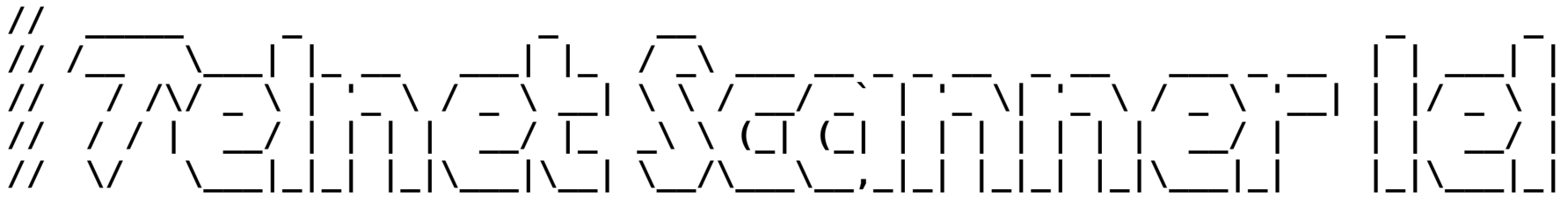
- všechno jde DoS-nout
- někdy se vyrojí kampaně, kde se najednou vyrojí za den 20x víc útoků než průměr
- nejdražší zdroj je RAM
  - při tisíckách spojení je RAM klony filesystému znát, i když image filesystému je celkem miniaturní (2.5 MB)
  - bez klonování by zase nebyla separace
  - pythoní memory leaky („stale references“)

Script kiddies

- instalují perlové DDoS boty
- IRC DDoS boty
- backdoornuté SSH daemony
- dost často se pokouší instalovat herní servery ze Steamu (CounterStrike apod.), TeamSpeak
- občas někdo instaluje něco bizarního, jako internetové rádio nebo se pokouší skompilovat GUI aplikaci

Pár perliček ze zdrojáků botů

- občas stáhnou zdrojáky a snaží se je skompilovat
- ASCII art, botnet příkazy v „LOLcat“ speaku
  - if(!strcmp(argv[0], "LOLNOGTFO"))
  - void petarda(...)



```
void StartTheLelz(int nothing)
```

Koho my vidíme jako přímé utočníky

- téměř zásadně již napadené stroje
- mnoho mnoho domácích routerů
  - poznat podle Shodanu nebo HTTP GET na port 80
- mnoho adres je dynamických
  - jedna útočící adresa dostane druhý den jinou IP od DHCP a na původní adrese je jiné zařízení
  - omezuje užitečnost reputace IP adres nebo blacklistů

Děkuji za pozornost

Ondrej Mikle • ondrej.mikle@nic.cz

CZ.NIC | SPRÁVCE DOMÉNY CZ