UNIVERSITÀ DEGLI STUDI DI MILANO
Dept. of Computer Science

Botnet

Matteo Camilli

matteo.camilli@unimi.it
http://camilli.di.unimi.it

Sicurezza dei Sistemi e delle Reti a.a. 2012/2013

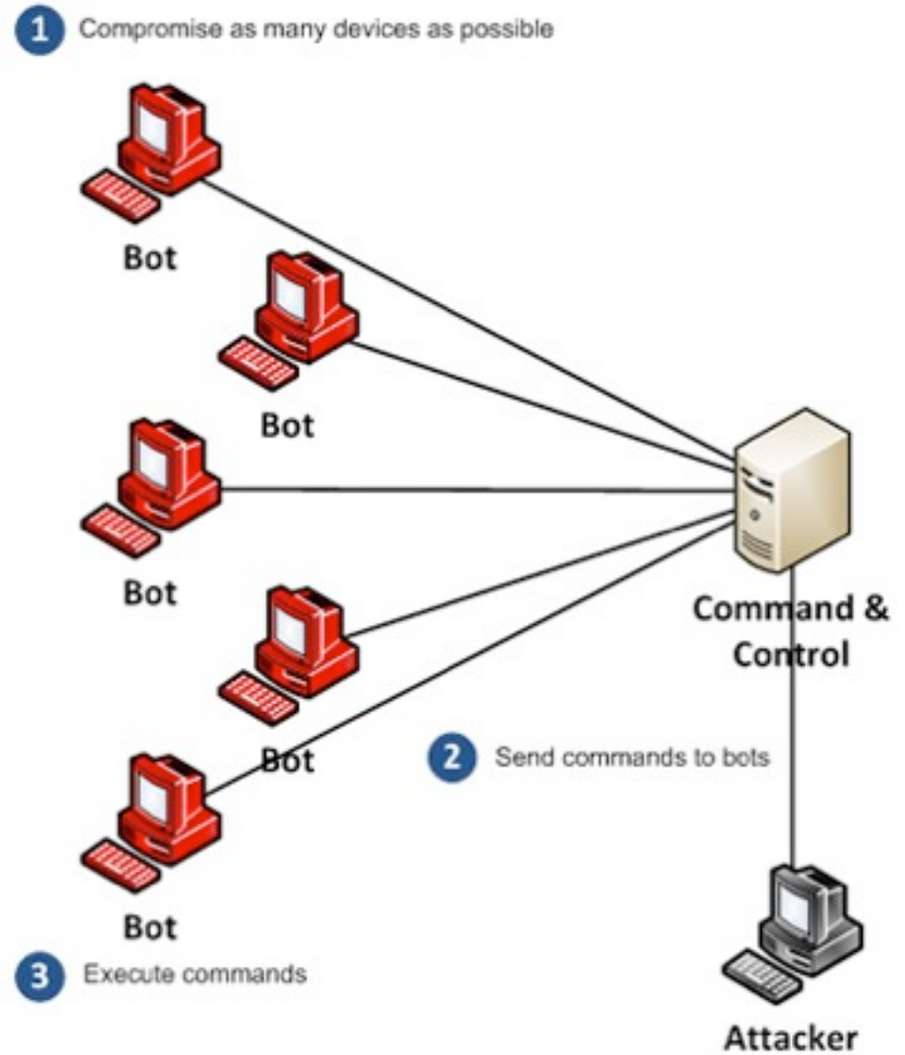
Outline

- Botnet
  - IRC (Push), HTTP (Pull)
  - P2P
  - Mobile
- Partitioning Clustering
  - K-Means
  - X-Means
- BotMiner
  - Architettura
  - Approcio non supervisionato
- Distributed Detection

Introduzione

- **Botnet**: Gruppo di istanze di malware ***coordinato*** e ***controllato*** attraverso un canale di comunicazione (C&C: Command & Control).
- Le macchine che eseguono il malware vengono chiamate **bot** e vengono utilizzate per compiere attività illegali.

- *controllato*: I bot devono contattare i C&C server/peer per ottenere i comandi da eseguire. Deve esserci comunicazione che può avvenire in modo centralizzato o distribuito.



- Port Scanning (per eseguire attacchi di tipo DDoS)
- DDoS
- Spam
- Phishing
- Identity theft

Introduzione (2)

- BBC News Technology, 26 Marzo 2012. Hot topic.

Microsoft moves to disable Zeus botnet

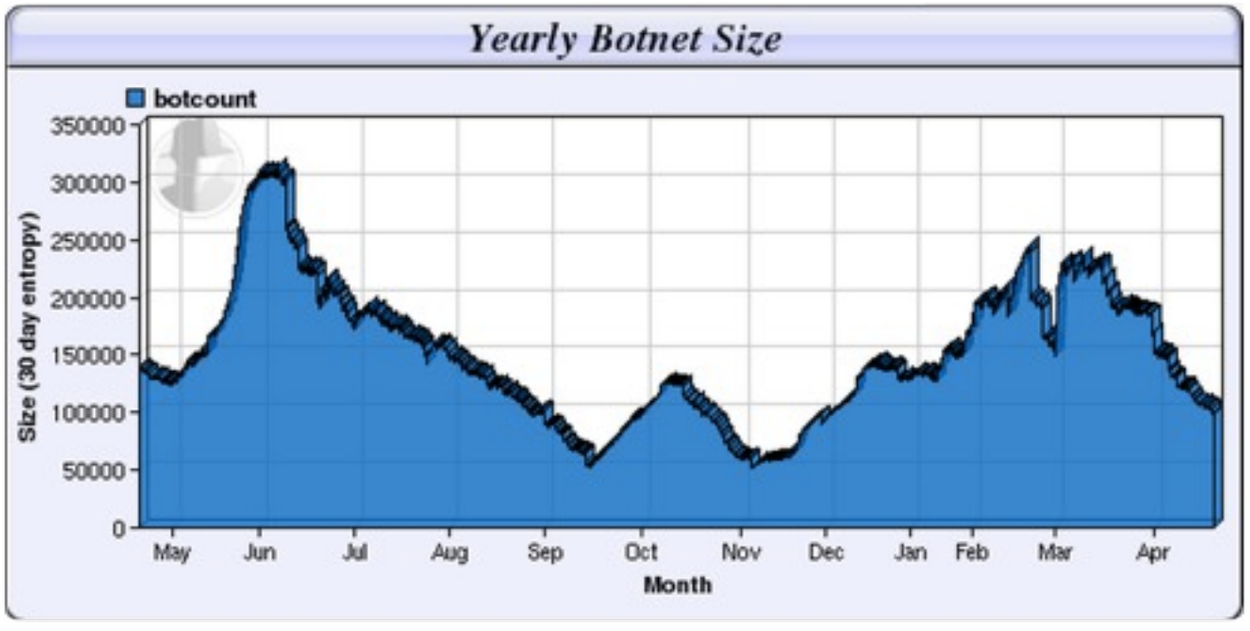
A global operation has been undertaken to disable a number of botnets believed to be responsible for the theft of millions of pounds, Microsoft has said.

The company is working in collaboration with financial services to disable botnets powered by Zeus malware.

Microsoft described the action as its "most complex effort to disrupt botnets to date".

The Zeus malware logs keyboard input when users log in to banking websites

- shadowserver.org Numero di bot attivi durante l'ultimo anno. I dati sono ottenuti monitorando botnet conosciute.



Introduzione (3)

MDK: The Largest Mobile Botnet in China

Created: 24 Jan 2013 03:05:39 GMT | Updated: 24 Jan 2013 06:28:46 GMT | Translations available: 日本語

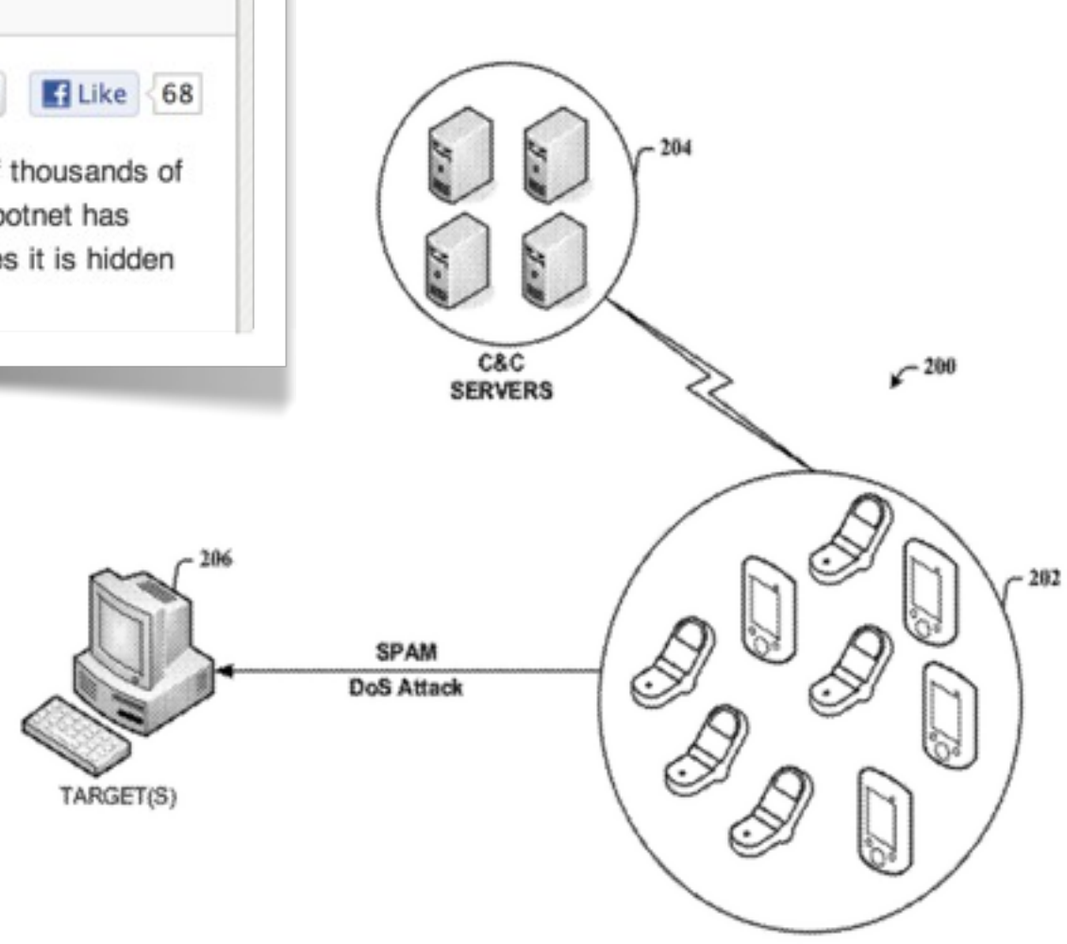
Flora Liu SYMANTEC EMPLOYEE

+2
2 Votes

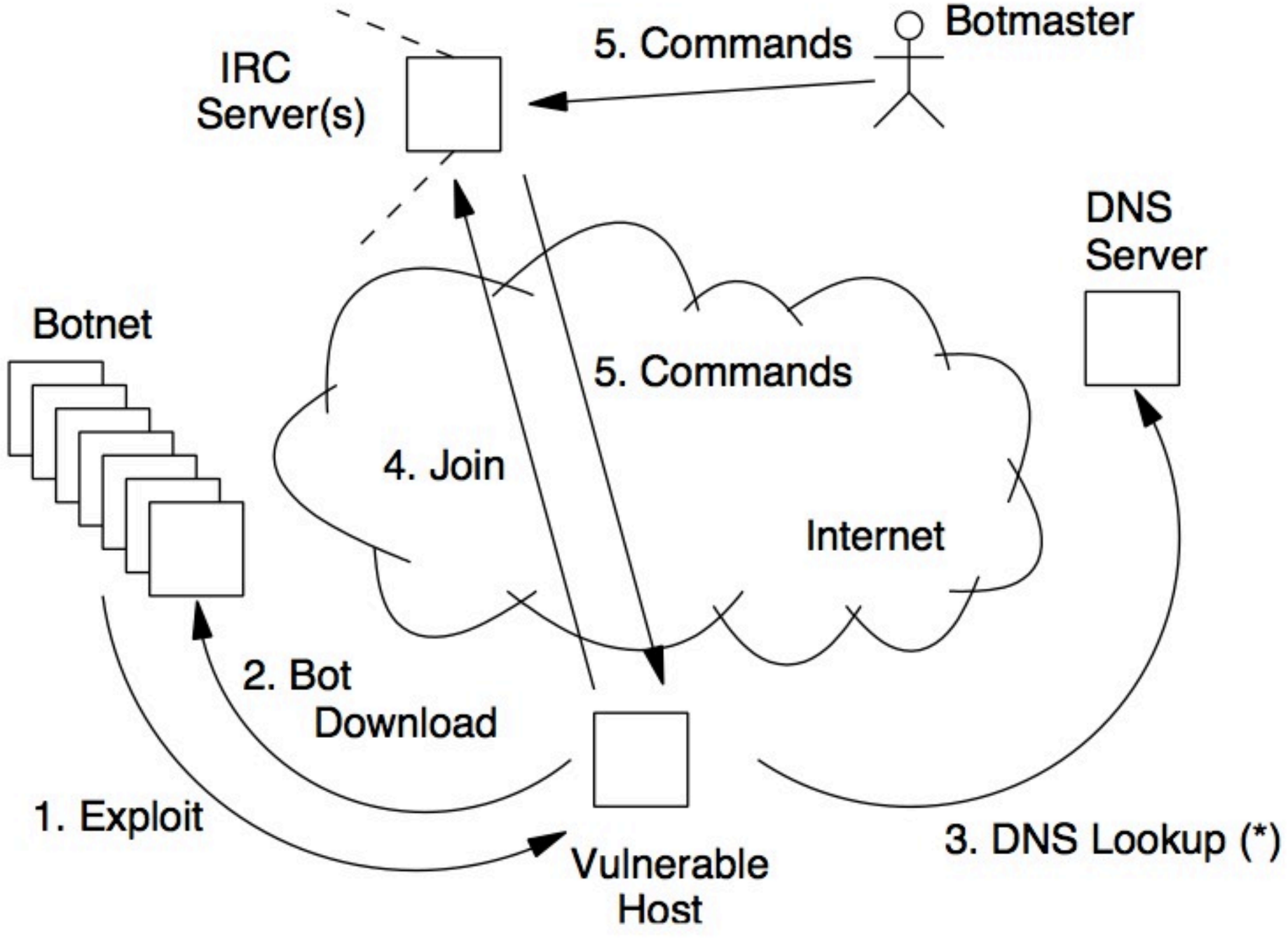
Symantec. | Official Blog

Share Share Tweet Like 68

In February 2012, we blogged about Android.Bmaster (a.k.a. Rootstrap), which infected hundreds of thousands of devices. At that time, it was the largest mobile botnet documented to date. Recently, the Bmaster botnet has been overtaken by the newly uncovered MDK botnet. Dubbed as Android.Troj.mdk, Kingsoft believes it is hidden in more than 7,000 apps and has infected up to one million devices.



Botnet basata su IRC (Push Mode)



Botnet basata su IRC (Push Mode)

1) **Exploit**: Le botnet infettano nuove vittime sfruttando vulnerabilità del software che viene eseguito. Vengono utilizzate diverse strategie: exploit remoti o più semplicemente convincendo la vittima ad eseguire del codice maligno (Es: allegato in email).

2) **Bot Download**: Una volta infettata la vittima esegue uno script (shellcode) che scarica l'eseguibile del bot da una qualche locazione. Una volta scaricato viene installato sulla macchina in modo che venga lanciato automaticamente al riavvio del sistema.

3) **DNS Lookup**: Il Bot contatta il server IRC (hard-coded) oppure risolve il suo DNS name. Nel secondo caso il botmaster mantiene il controllo della botnet anche se l'IP del server IRC viene black-listato.

4) **Join**: Generalmente esistono due tipi di autenticazione. Bot si autentica nel canale IRC tramite password. Botmaster si autentica alla popolazione di bot (non è parte del protocollo IRC) per evitare che altri botmaster prendano il controllo della botnet.

5) **Commands**: I Bot si mettono in ascolto dei messaggi scambiati nel canale ed eseguono i comandi.

Botnet basata su HTTP (Pull Mode)

- Identiche a botnet IRC.
- C&C in modalità PULL. I bot eseguono periodiche richieste HTTP a particolari indirizzi per prelevare i comandi da eseguire.

```
...

<table>
  <tr>
    <td class="asjkdha">ddos www.bank.com</td>
  </tr>
</table>

...
```

P2P Botnet

- Vengono generalmente usati due modi per effettuare il **Join** in una botnet p2p (come per una normale rete p2p per il file-sharing):

  1) Una lista iniziale di peer è hard-coded in ogni client. I nuovi peer contattano i peer presenti nella lista per aggiornare le proprie informazioni sul “vicinato”.

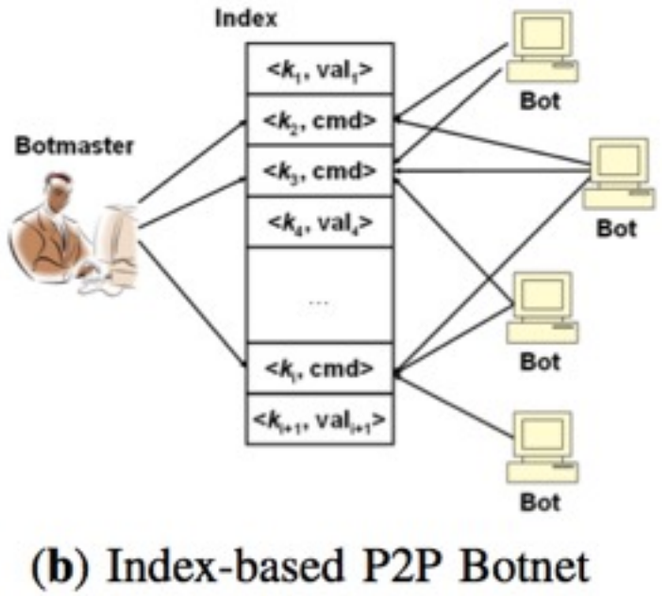
  2) Shared web cache che contiene il cui indirizzo è hard-coded nei client. I nuovi peer usano la web cache per aggiornare il “vicinato”.

- **commands**: Nelle reti p2p esiste un indice (centralizzato/distribuito, solitamente tramite DHT) che viene utilizzato dai peer per identificare la locazione dei contenuti.

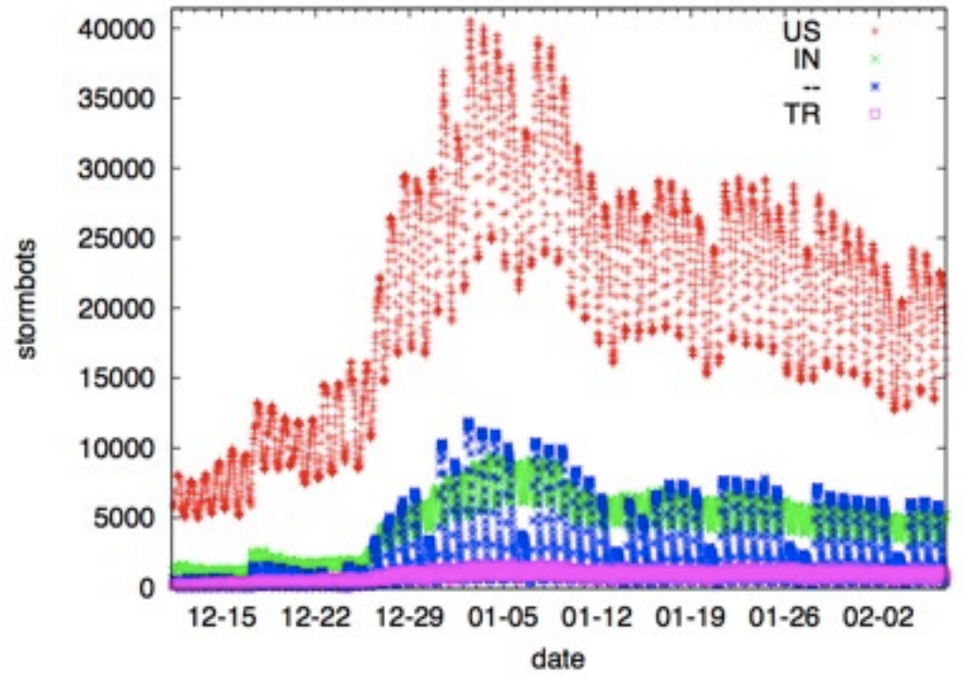
  1) Il botmaster inserisce dei record nell’indice associati a nomi di file o valori di hash predefiniti. Al posto di specificare la locazione dei file vengono inseriti dei comandi.

  2) I peer eseguono richieste periodiche per file/hash predefiniti e ricevono i comandi da eseguire.

P2P Botnet (2)



(b) Index-based P2P Botnet



- Es: **Storm** botnet.
  1) Ogni giorno ci sono 32 hash key che i bot usano per cercare i comandi. I 32 valori sono calcolati tramite una funzione che prende come input la data corrente e un numero casuale in [0, 31].
  2) Il botmaster publica 32 coppie <k_0, cmd> ... <k_{31}, cmd>
  3) Ogni peer calcola periodicamente un valore per k ed effettua una richiesta.

Caratteristiche di una botnet

1) Canale di **comunicazione** tra bot e botmaster C&C, tramite il quale i bot ricevono i comandi da eseguire.
2) **Attività malevole** coordinate (esecuzione dei comandi).

- Nota:
  - Altre forme di malware compiono attività malevole, ma non sono connessi ad un canale C&C.
  - Normali applicazioni (Es: IRC client o normali software p2p file sharing) possono mostrare una comunicazione simile a quella di un canale C&C, ma non eseguono nessuna attività maligna.

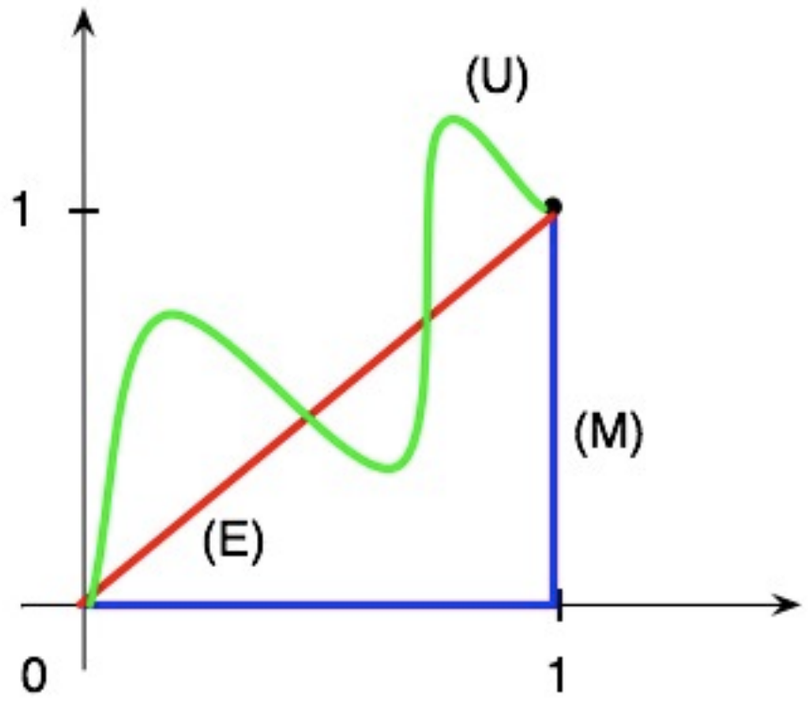
Clustering

K-Means
X-Means

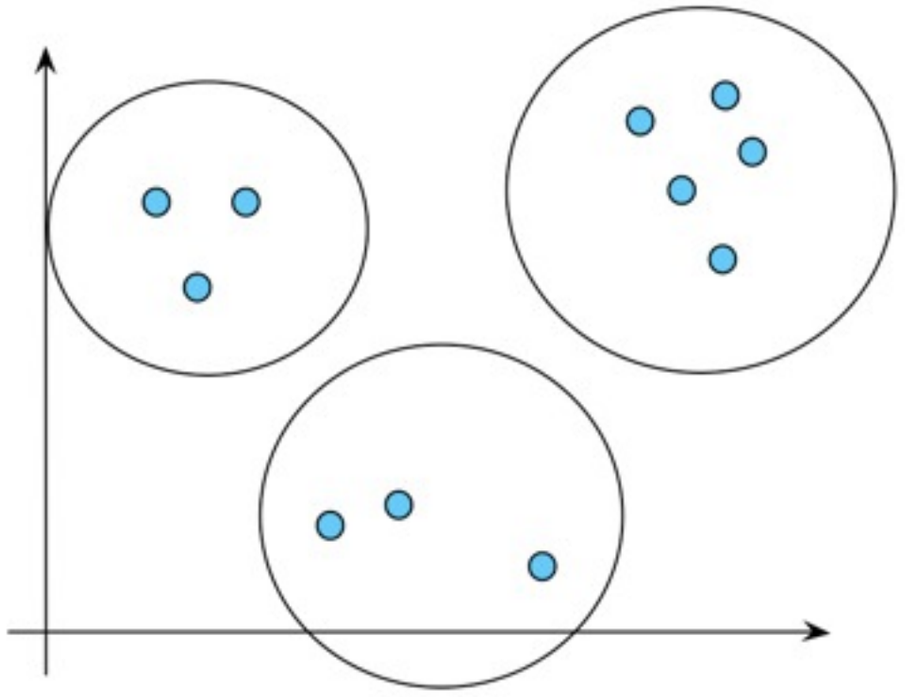
Misura della distanza

- Vogliamo raggruppare elementi di un insieme a seconda del loro grado di similarità.
- Come può essere misurata la similarità tra due elementi di un insieme:
  - Distanza Euclidea (E)
  - Distanza Manhattan (M)
  - O una qualsiasi metrica definita nello spazio di interesse (U)

- $p_1 = (x_1, y_1)\ p_2 = (x_2, y_2)$
- $E = RADQ((x_1-x_2)^2+(y_1-y_2)^2)$
- $M = |x_1-x_2|+|y_1-y_2|$



K-Means (Partitional Clustering)



- Choose k points
- Iterate{
  - Compute distance from all points to all k-centers
  - Assign each point to the nearest k-center
  - Compute the average of all points assigned to all specific k-centers
  - Replace k-centers with the new averages

}

BotMiner

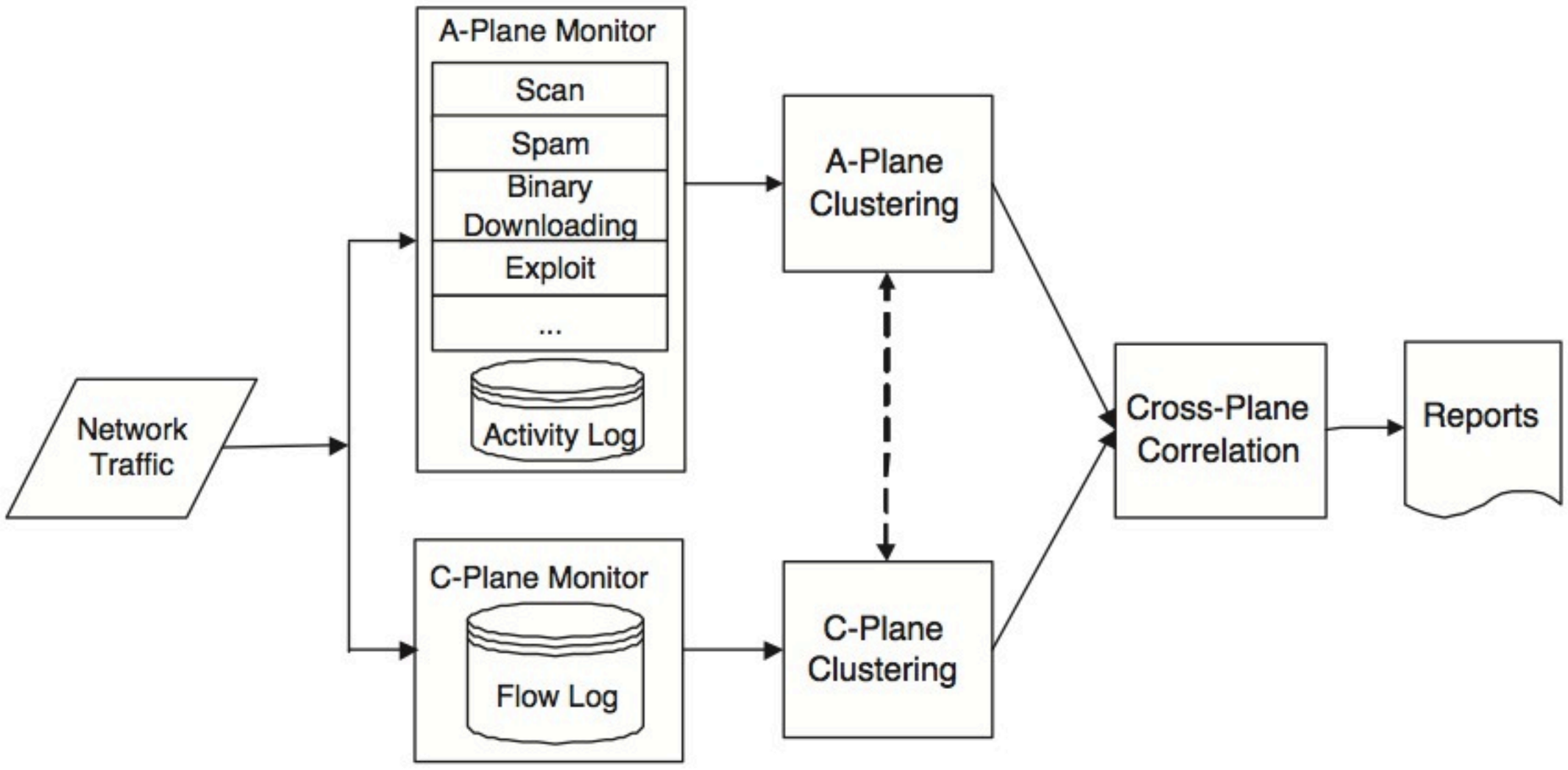
Clustering Analysis of Network Traffic for Protocol- and Structure-Independent Botnet Detection

Obiettivi

- Individuare host che fanno parte di una o più botnet analizzando (offline) il traffico di rete che generano.
- Indipendenza dalla struttura e dal protocollo usato per la comunicazione tra bot e botmaster (canale C&C) = Resistenza a cambiamenti dei C&C server.
- Indipendenza dal contenuto delle comunicazioni sul canale C&C, infatti la comunicazione potrebbe essere criptata.

Architettura

- Architettura del framework BotMiner.



C-Plane Monitor

- Cattura i pacchetti che compongono il traffico e costruisce i **flow**. I flow astraggono il concetto: **“chi sta parlando con chi”**
- Ogni record che rappresenta un flow è composto da:

  <tempo, durata, IP_{src}, $Port_{src}$, IP_{dst}, $Port_{dst}$, #packet, #byte>

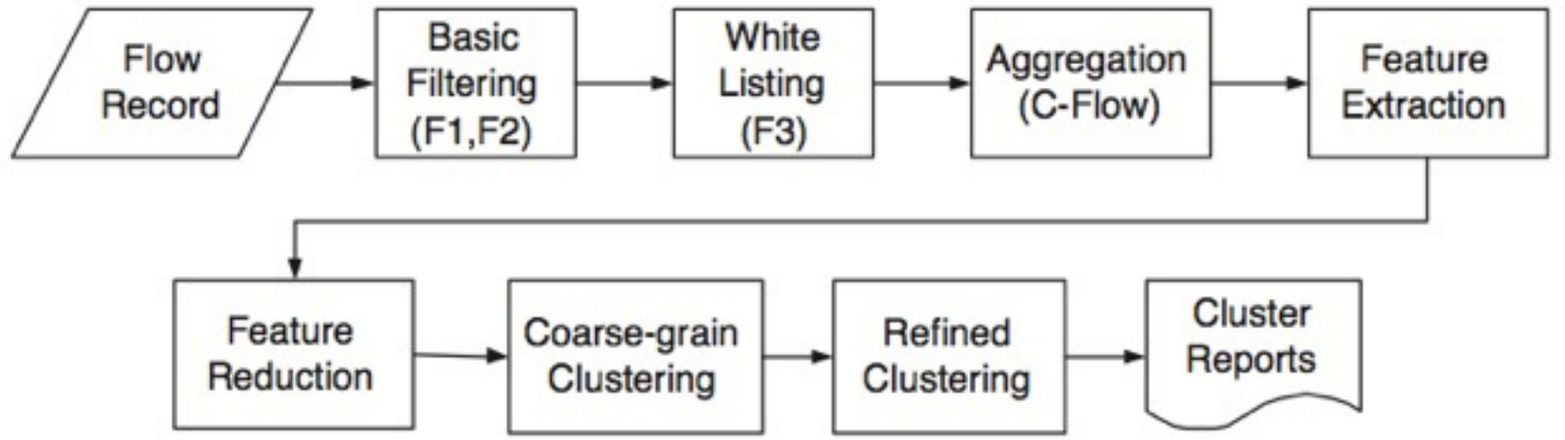
- Vengono costruiti flow TCP e UDP.

argus

- p-cap -> Flow report
- http://qosient.com/argus/

C-Plane Clustering

- Per prima cosa viene filtrato il traffico irrilevante:
  - Basic Filtering: traffico interno.
  - White Listing: traffico diretto a server conosciuti (Es: Google, Yahoo!)



- Aggregation: I flussi vengono aggregati in **C-Flow**. Data un epoca *E*, tutti i flussi m che condividono uguale protocollo, IP_{src}, IP_{dst}, $Port_{dst}$, vengono aggregati nello stesso C-Flow $c_i=\{f_j\}_{j=1...m}$. Tutti gli n C-flow dell'epoca *E* astraggono: **"chi sta parlando a quale servizio"**.

C-Plane Clustering (2)

- Per poter effettuare il clustering dei C-Flow è necessario trasformarli ed ottenere una rappresentazione vettoriale.

$$F : \text{C-plane} \rightarrow \mathbb{R}^d$$

La Funzione F, lavora nel seguente modo:

- Dato un C-Flow c_i restituisce la descrizione delle seguenti 4 variabili casuali tramite 4 vettori:

  1) **FPH** (Flows Per Hour) = numero di flow in c_i presenti per ogni ora dell'epoca *E*

  2) **PPF** (Packets Per Flow) = numero totale di pacchetti per ogni flow in c_i

  3) **BPP** (Bytes Per Packet) = numero medio di byte per pacchetto per ogni flow in c_i. Per ogni f_j in c_i, dividiamo il numero totale di byte per il numero di pacchetti.

  4) **BPS** (Bytes Per Second) = media di bytes al secondo. Per ogni f_j in c_i, dividiamo il numero totale di byte per la durata di f_j.

C-Plane Clustering (3)

- Non conosciamo la distribuzione delle 4 variabili casuali ma possiamo calcolare la ***funzione di ripartizione empirica*** $\hat{F}$ che approssima la **funzione di ripartizione**.

$$F(x) = P(X \leq x)$$

- La funzione di ripartizione empirica viene usata in statistica per descrivere fenomeni quantitativi (descritti con valori misurati). Per dimensioni del campione che tendono ad infinito, $\hat{F}$ tende in probabilità a F

$$P(\sup_x |\hat{F}(x) - F(x)| < \varepsilon) > 1 - \delta$$

- Dato un campione $\bar{X} = \{x_i\}_{i=1\ldots n}$,

$$\hat{F}(x) = \frac{1}{n} \cdot \{\#i : x_i \leq x\}$$

C-Plane Clustering (4)

- Una volta calcolata $\hat{F}$ per ognuna delle 4 variabili casuali, utilizzando i dati di tutti i C-Flow, descriviamo ogni C-Flow c_i tramite una tecnica di ***data binning***.
- Per ogni variabile casuale suddividiamo l'intervallo dei valori che può assumere in 13 intervalli: $[0,k_1]$, $(k_1,k_2]$, ..., (k_{12}, inf).
- Per ogni C-Flow c_i possiamo descrivere la distribuzione approssimata di ognuna delle 4 variabili casuali come un vettore di 13 elementi, dove ogni elemento i rappresenta il numero di volte che la variabile casuale ha assunto un valore nel corrispondente intervallo $(k_{i-1}, k_i]$.
- I 13 intervalli per ogni variabile casuale vengono costruiti calcolando i seguenti quantili: $q_{0.05}$, $q_{0.1}$, $q_{0.15}$, $q_{0.2}$, $q_{0.25}$, $q_{0.30}$, $q_{0.4}$, $q_{0.5}$, $q_{0.6}$, $q_{0.7}$, $q_{0.8}$, $q_{0.9}$, e ponendo $k_1=q_{0.05}$, $k_2=q_{0.1}$, $k_3=q_{0.15}$, ecc.

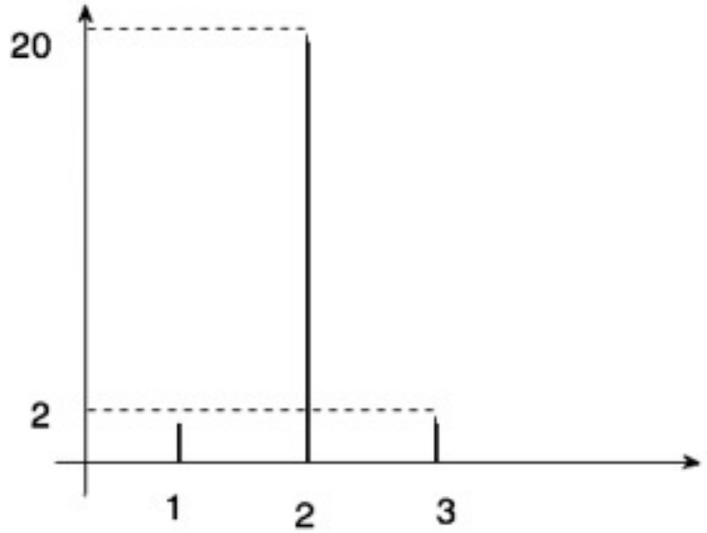
- Il quantile q_l di una variabile casuale discreta X è il valore q tale che:

$$P(X \leq q) \geq l \equiv F(q) \geq l$$

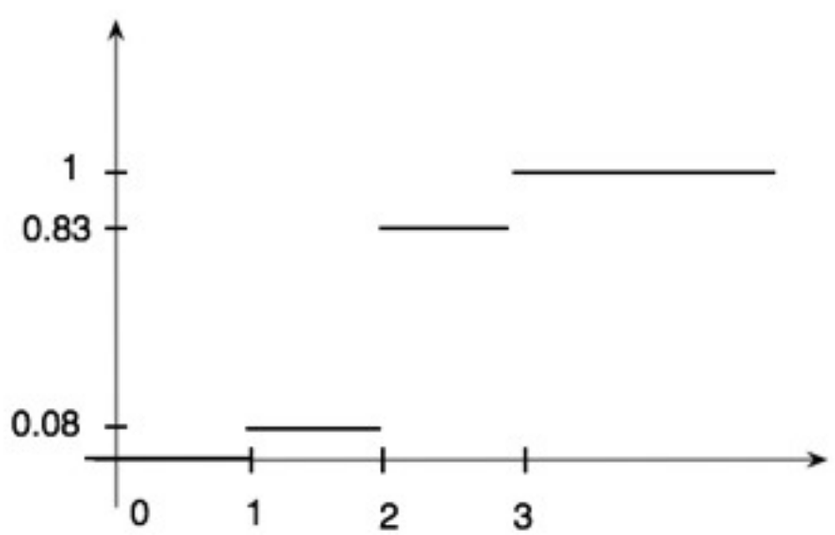
C-Plane Clustering (5)

- Esempio **FPH**
- Osservando il numero di flow per ora in un epoca di 24 ore osserviamo le seguenti misurazioni:

$$F\bar{P}H = \{1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3\}$$



$$\dots$$
$$\hat{F}(0) = \frac{1}{24} \cdot 0 = 0$$
$$\hat{F}(1) = \frac{1}{24} \cdot 2 = 0.08$$
$$\hat{F}(2) = \frac{1}{24} \cdot 20 = 0.83$$
$$\hat{F}(3) = \frac{1}{24} \cdot 24 = 1$$
$$\hat{F}(4) = \frac{1}{24} \cdot 24 = 1$$
$$\dots$$



- k_1=1, k_2=2, k_3=2, k_4=2, ... => Intervalli: [1,2], (2,2], (2,2], ...
- Per ogni C-Flow c_i, descriviamo l'FPH come un vettore di 13 elementi.
- Eseguiamo la stessa procedura per le altre variabili casuali. Alla fine per ogni c_i avremo un vettore di 52 elementi che lo descrive.

C-Plane Clustering (6)

- Una volta ottenuto il dataset:

$$\mathcal{D} = \{\vec{p}_i = F(c_i)\}_{i=1..n}$$

- Intuitivamente, i punti “vicini” in *D* rappresentano C-Flow con comportamento simile per quanto riguarda la comunicazione.
- L’obiettivo è quello di raggruppare gli host che mostrano un particolare comportamento nella comunicazione (canale C&C).

- BotMiner in questa fase usa l’algoritmo di clustering X-Means: versione ottimizzata di K-Means in cui non dobbiamo specificare il valore k, ma viene scelto in modo ottimale.

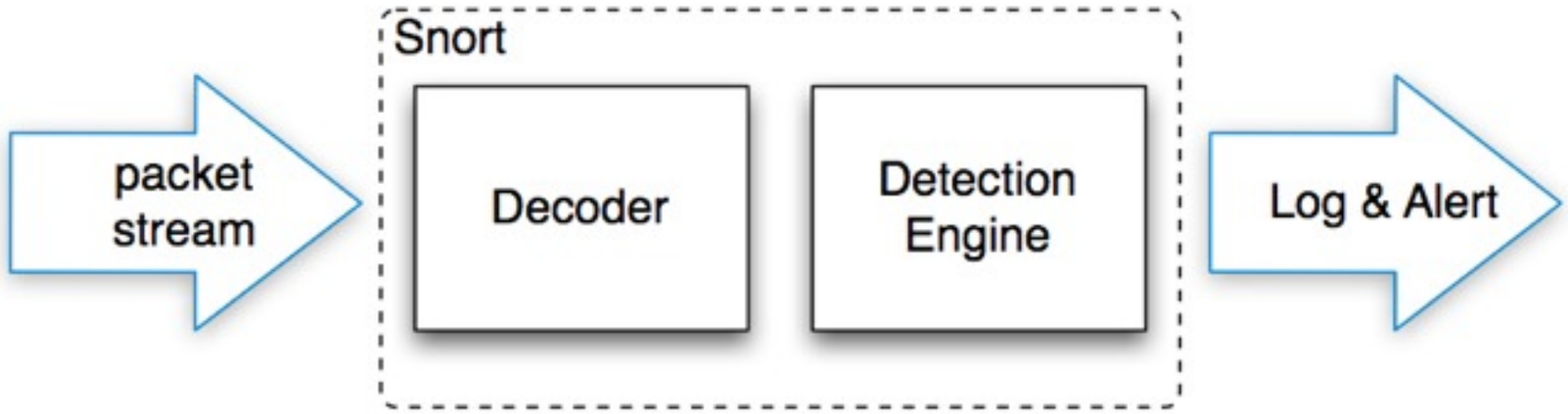
A-Plane Monitor

- Ricava informazioni su **“chi sta facendo cosa”**. Analizza il traffico per individuare attività malevole svolte dagli host (bot).
- Attività più comuni che il botmaster può comandare:
  - Scan (per propagazione di malware o DoS)
  - Spam
  - Download di binari (possibile malware update)
  - Exploit (propagazione di malware)

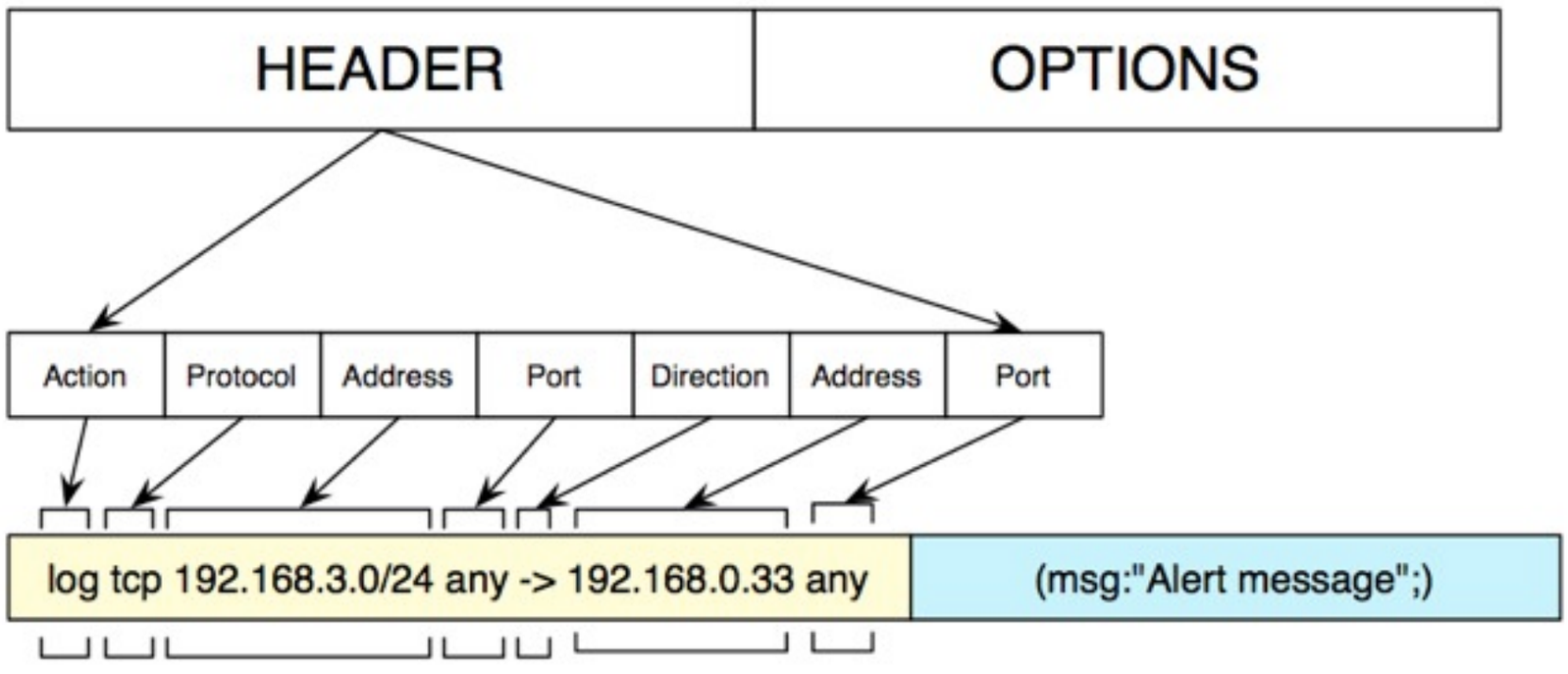
- Basato su **Snort**: Open source ***Intrusion detection System (IDS)***.
  - Usa un linguaggio basato su regole per identificare anomalie nel traffico analizzato.
  - Possiede un dataset di regole per identificare attività anomale conosciute.
  - Possibile estendere definendo nuove regole.

Snort

- **Architettura**



- **Rules**



Snort Rules

- **HEADER**

alert tcp any any -> any any (msg:”Sample alert”;)

alert tcp $EXTERNAL_NET any -> 192.168.3.0/24 80 (msg:”Sample alert”;)

- Rule Header
  - Action (log, alert)
  - Protocol (ip, tcp, udp, icmp, any)
  - Src IP & Port
  - Dst IP & Port
  - Direction (->, <>)
- Src, dst IP possono essere:
  - Variabili ($HOME_NET)
  - Indirizzi IP (192.168.3.12)
  - Liste ([192.168.3.12, 192.168.3.9])
  - Range 192.168.3.0/24
- Porte possono essere:
  - porte singole
  - range (80:85)

Snort Rules (2)

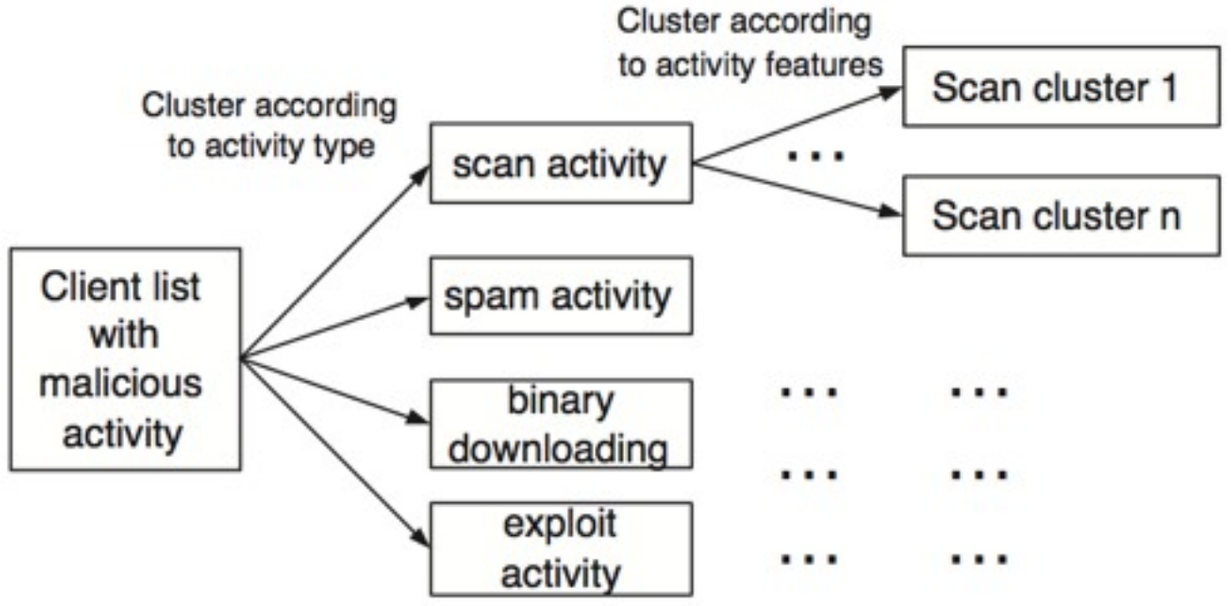
- **OPTIONS**

alert tcp $EXTERNAL_NET any -> 192.168.3.0/24 80 (msg:”Sample alert”; flow: to_server, established; pcre:”/GET.*\.htm/i”; classtype: web-application-activity; reference:url,http://www.vorant.com/advisories/20060405.html; sid:2000123; rev:1;)

- Rule Body
  - delimitato da “( )”, contiene una serie di ***opzioni: keyword + parameters***
  - Es:
    - “msg”: messaggio di allerta human-readable
    - “pcre”: ricerca nel payload tramite espressioni regolari
    - “flow”: restrizioni sul tipo di flow
    - “reference”: URL con maggiori informazioni
    - “classtype”: tipo di attacco / gravità dell’evento

A-Plane Clustering

- Una volta ottenuto il log delle attività sospette, possiamo procedere con il clustering usando come misura di similarità il tipo di attività svolta dagli host.



- ***Scan***: suddividiamo ulteriormente a seconda dell'insieme di porte target, o subnet target.

- ***Spam***: suddividiamo a seconda dei server smtp usati (possibile anche suddividere in base al contenuto dello spam, se catturiamo il traffico smtp).
- ***exploit***: suddividiamo a seconda del tipo di exploit (Snort indica diverso SID).
- ***downloading***: suddividiamo secondo il grado di similarità del binario.

Cross-Plane Correlation

- L'idea è quella di trovare grosse intersezioni tra gli host appartenenti a diversi cluster delle due tipologie (C-Plane, A-Plane).
- Per ogni host ***h***, calcoliamo un ***"botnet score" s(h)***, che rappresenta il grado di sospetto verso un certo host. Dopodiché i maggiori sospettati (*s(h)* maggiore di una certa soglia) vengono raggruppati a seconda del loro grado di similarità tenendo in considerazione il clustering di tipo C e A.

- Insieme degli host $h \in H$
- insieme degli A-cluster che contengono *h* $\mathcal{A}^{(h)} = \{A_i\}_{i=1..m_h}$
- insieme dei C-cluster che contengono *h* $\mathcal{C}^{(h)} = \{C_i\}_{i=1..n_h}$

$$s(h) = \sum_{\substack{i,j \\ j>i \\ t(A_i)\neq t(A_j)}} w(A_i)w(A_j)\frac{|A_i \cap A_j|}{|A_i \cup A_j|} + \sum_{i,k} w(A_i)\frac{|A_i \cap C_k|}{|A_i \cup C_k|}$$

Cross-Plane Correlation (2)

- Una volta filtrati gli host meno sospetti $s(h) > \theta$ raggruppiamo quelli simili.

- Insieme di bot sospetti $B \subseteq H$
- Insieme degli A-cluster che contenono almeno un bot $\mathcal{A}^{(B)} = \{A_i\}_{i=1..m_B}$
- Insieme dei C-cluster che contengono almeno un bot $\mathcal{C}^{(B)} = \{C_i\}_{i=1..n_B}$

- unione ordinata $\mathcal{K}^{(B)} = \mathcal{A}^{(B)} \cup \mathcal{C}^{(B)} = \{K_i^{(B)}\}_{i=1..(m_B+n_B)}$

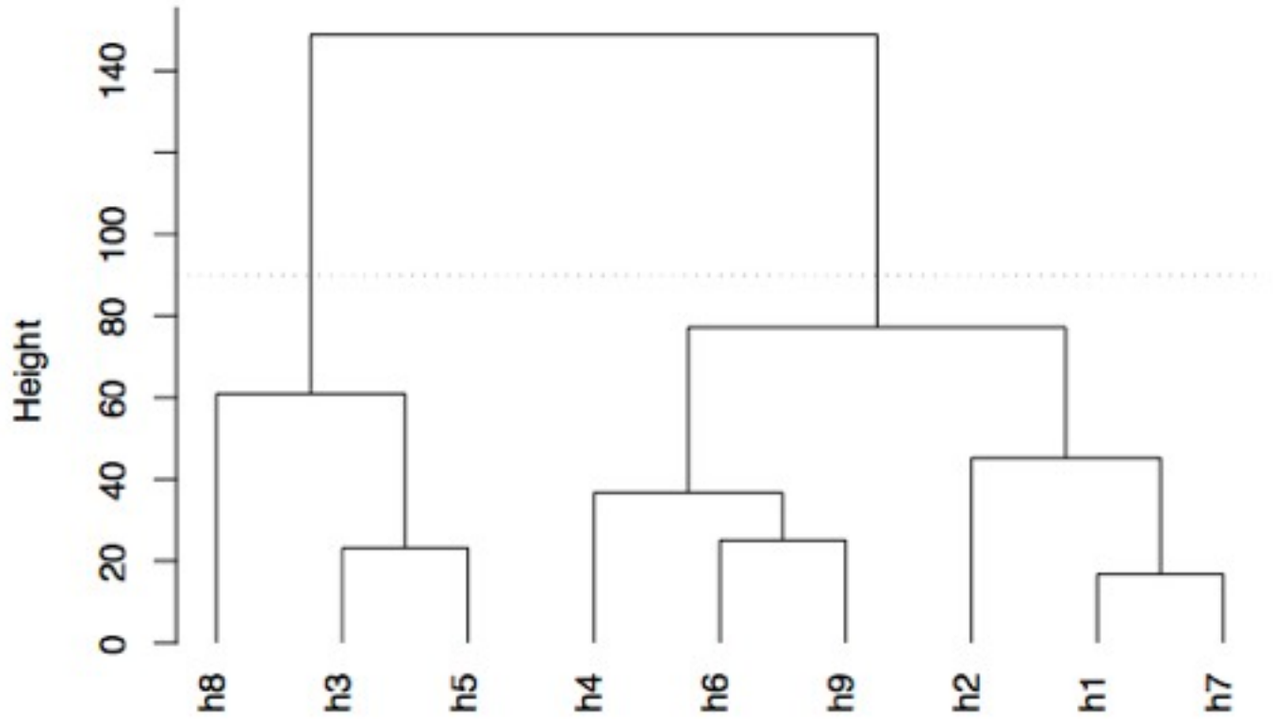
- Descrizione degli host come vettore binario $b(h) \in \{0,1\}^{|\mathcal{K}^{(B)}|}$

$$sim(h_i, h_j) = \sum_{k=1}^{m_B} I(b_k^{(i)} = b_k^{(j)}) + I(\sum_{k=m_B+1}^{m_B+n_B} I(b_k^{(i)} = b_k^{(j)}) \geq 1)$$

$$b^{(i)} = b(h_i) \text{ and } b^{(j)} = b(h_j)$$

Cross-Plane Correlation (3)

- La definizione di similarità degli host ci permette di costruire un dendrogramma:



- Davies Bouldin Index

$$R_{i,j} = \frac{S_i + S_j}{M_{i,j}}$$

$$D_i \equiv \max_{j:i \neq j} R_{i,j}$$

$$DB \equiv \frac{1}{N} \sum_{i=1}^{N} D_i$$

- Tagliando ad altezza 90 otteniamo due botnet: {h8,h3,h5}, {h4,h6,h9,h2,h1,h7}
- E’ possibile trovare il taglio ideale: ***Davis Bouldin Index*** valuta quanto è buono il clustering usando come metrica la distanza infra-cluster e intra-cluster.

Risultati BotMiner

- Rilevamento traffico “benevolo” per 24 ore.
- Rilevamento traffico di bot da alcune botnet conosciute.
- Mescolamento traffico “malevolo” con quello “benevolo” andando a sostituire alcuni IP degli host buoni (scelti random) con gli IP dei bot.

Botnet	Number of Bots	Detected?	Clustered Bots	Detection Rate	False Positive Clusters/Hosts	FP Rate
IRC-rbot	4	YES	4	100%	1/2	0.003
IRC-sdbot	4	YES	4	100%	1/2	0.003
IRC-spybot	4	YES	3	75%	1/2	0.003
IRC-N	259	YES	258	99.6%	0	0
HTTP-1	4	YES	4	100%	1/2	0.003
HTTP-2	4	YES	4	100%	1/2	0.003
P2P-Storm	13	YES	13	100%	0	0
P2P-Nugache	82	YES	82	100%	0	0

Distributed Detection

1) A-Plane / C-Plane Monitor locale

2) A-Clustering locale

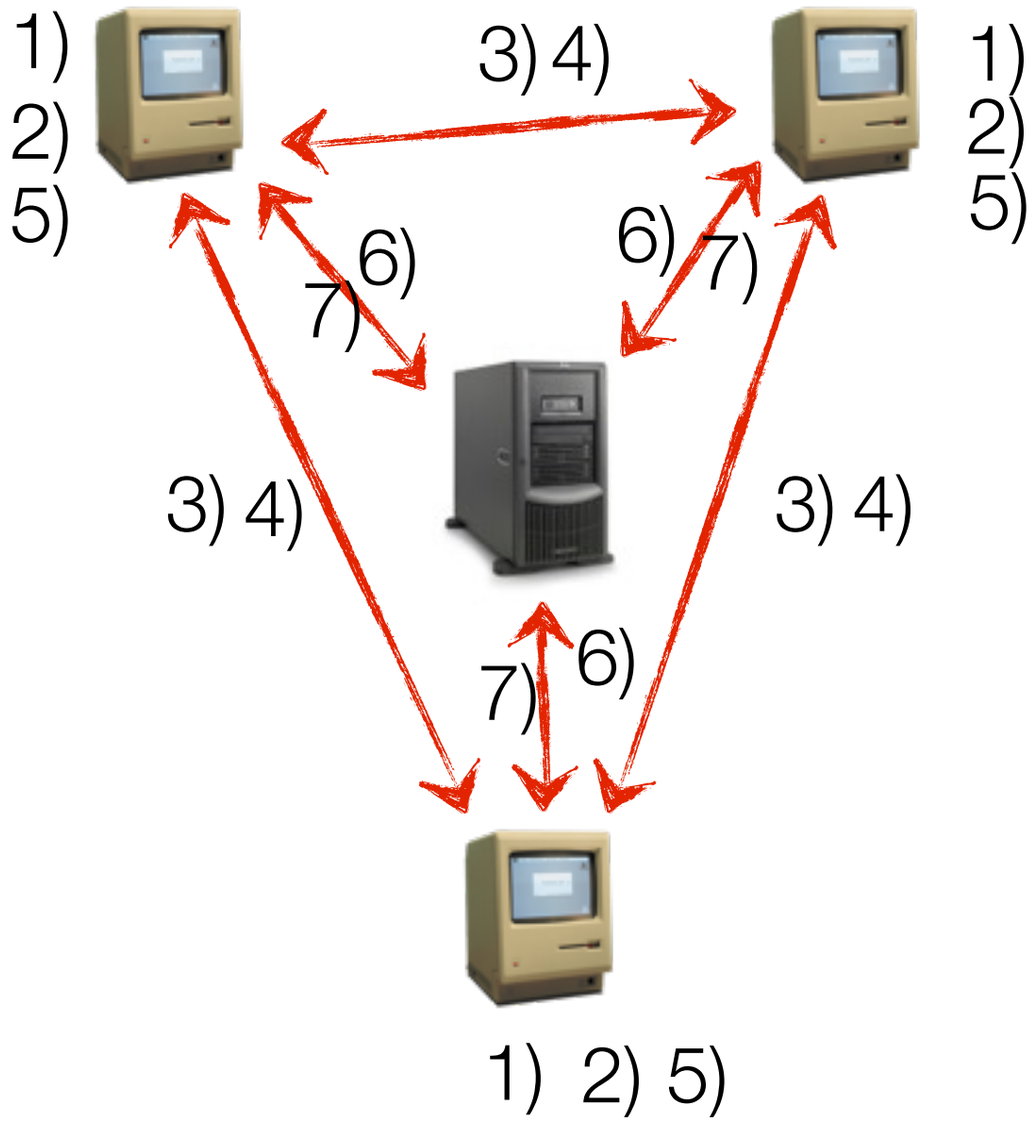
3) C-Clustering distribuito usando DS-Means

4) Scambio informazioni su dimensioni dei cluster di tipo A e C.

5) Botnetscore locale

6) Comunicazione botnetscore a coordinatore e clustering gerarchico locale

7) comunicazione dei risultati



Risultati

Traccia	Dim.	Durata	#Pkts	#TCP/UDP Flow	#Flow dopo filtraggio base	#C-Flow
sample #1	4,53Gb	24h	5.973.980	263.395	197.163	7.647
sample #2	75,3Mb	3h	408.851	14564	10.940	881
sample #3	9,89Gb	24h	14.973.046	2.153.068	33.054	7.907

Tabella 4.1. Statistiche relative al C-Plane, risultati del basic filtering e numero di C-Flow.

ISXC, University of New Brunswick

Traccia	#C-Cluster	#A-Alert	#A-Cluster	#Bot noti	#Bot raggrup-pati	Detection Rate	Falsi Posit.
sample #1	5	15	1	15	15	100%	0
sample #2	5	71	19	7(3+4)	7(3+4)	100%	0
sample #3	5	639	103	9	5	55%	0

Tabella 4.2. Risultati sulla rilevazione di botnet ottenuti con il nostro famework.

References

- Moheeb Abu Rajab, Jay Zarfoss, Fabian Monrose, and Andreas Terzis. 2006. A multifaceted approach to understanding the botnet phenomenon. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement* (IMC '06). ACM, New York, NY, USA, 41-52. DOI=10.1145/1177080.1177086 http://doi.acm.org/10.1145/1177080.1177086

- Ping Wang, Lei Wu, Baber Aslam, and Cliff C. Zou. 2009. A Systematic Study on Peer-to-Peer Botnets. In *Proceedings of the 2009 Proceedings of 18th International Conference on Computer Communications and Networks* (ICCCN '09). IEEE Computer Society, Washington, DC, USA, 1-8. DOI=10.1109/ICCCN.2009.5235360 http://dx.doi.org/10.1109/ICCCN.2009.5235360

- Thorsten Holz, Moritz Steiner, Frederic Dahl, Ernst Biersack, and Felix Freiling. 2008. Measurements and mitigation of peer-to-peer-based botnets: a case study on storm worm. In*Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats* (LEET'08), Fabian Monrose (Ed.). USENIX Association, Berkeley, CA, USA, , Article 9 , 9 pages.

- Guofei Gu, Roberto Perdisci, Junjie Zhang, and Wenke Lee. 2008. BotMiner: clustering analysis of network traffic for protocol- and structure-independent botnet detection. In *Proceedings of the 17th conference on Security symposium* (SS'08). USENIX Association, Berkeley, CA, USA, 139-154.

- Jeffrey David Ullman, Stanford University, California. Mining of Massive Datasets. ISBN: 9781107015357. December 2011

- Snort Official Documentation: http://www.snort.org/docs