

# Detecting Botnet/Bot based on Correlating Activities

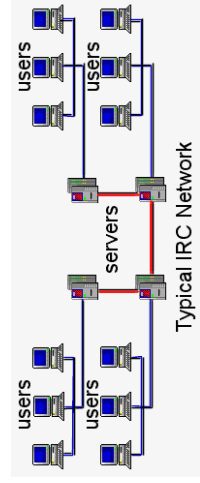
Yousof Al-Hammadi, Uwe Aickelin

Intelligent Modelling and Analysis - IMA  
School of Computer Science  
The University of Nottingham

25-11-2008

## Background - IRC Protocol (1459)

- **Server**
  - Backbone of IRC
- **Client**
- **Channel**
  - A room for chat
  - Channel Operators



## Outline

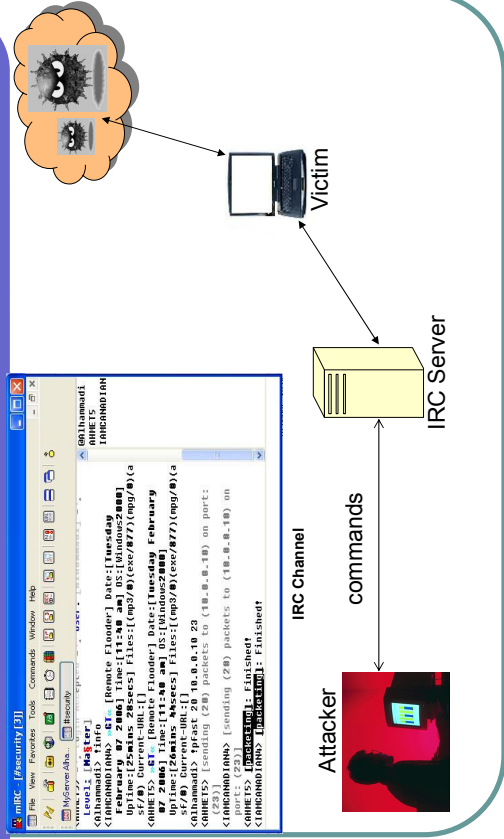
- **Background**
  - IRC – Internet Relay Chat
  - Bots
  - Methods of Infection
  - Motivation
- **Objectives**
- **Methods of Detection**
  - Botnet
  - Individual Bot
- **Results**
- **Current and Future Work**
- **Conclusion**

## Background

- **IRC Bots**
  - To mimic the channel Operators
  - Provide services to the clients
  - Respond automatically to activities
- **Malicious Bot**
  - Malicious program that is loaded in user machine without his/her knowledge
  - Responds to various instructions generated by the attacker
- **Relation between IRC and Bot**

# Methods of Infection

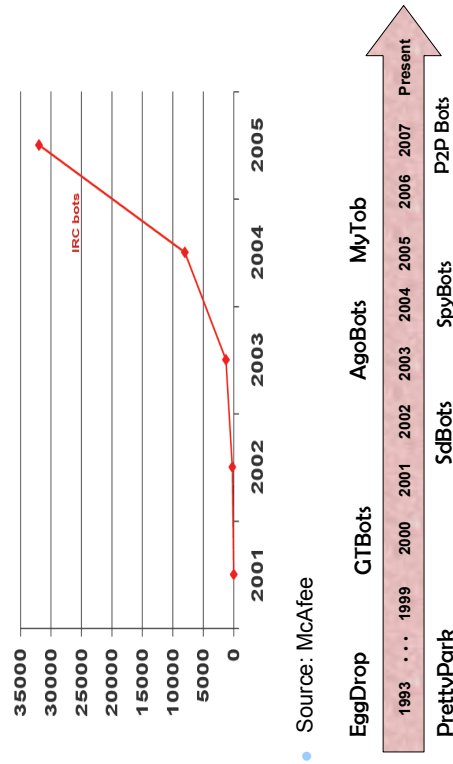
Infection Source:  
- Email virus  
- Malicious website



# Motivation

- Data regarding bots is rarely available
- Insufficient research in detecting a single bot
  - Signature-based detection
  - Network analysis
  - Problems?
- Serious threats
  - DDoS
  - Spamming
  - Keylogging
  - Spread Malwares
- large number of bots, growing exponentially

# Motivation



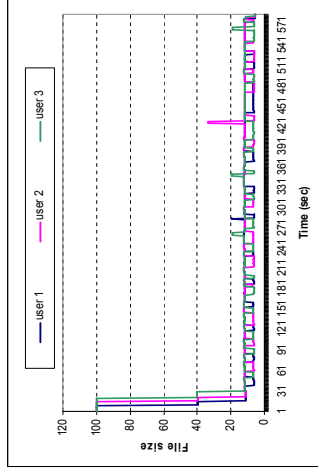
Source: McAfee

# Objectives (1)

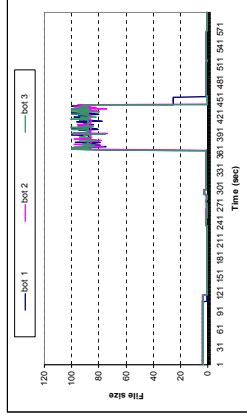
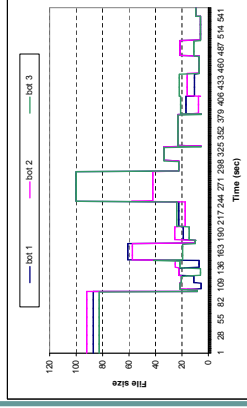
- To detect botnet
  - Monitor the change of behaviour in the system by monitoring API function calls from different sources
  - Find the correlation between these changes

## Case 1:

- Normal conversation
- Some IRC commands were used
- Result: No correlation between users



## Detecting Botnet – Attack Behaviour



### Case 2:

- Bots are idle for some time
- No flood commands (UDP, PING)
- Result:
  - Bots respond simultaneously
  - High correlation between bots

### Case 3:

- Bots are idle for some time
- UDP Flood is used
- Result:
  - Bots respond simultaneously
  - High correlation

## Detecting Bot – SRC

- **Purpose**
  - Technique to test the direction and relationship between two activities
- **Null hypothesis**
  - There is no relationship between the two activities
- **How it works**
  - Rank the two data (the higher value in each column is ranked as '1')
  - Find the difference in the ranks
  - Square the differences and sum them  $\sum d^2$
  - Calculate the coefficient (Rs)
- **Interpretation**
  - The closer Rs to (+1) or (-1) the stronger the likely correlation

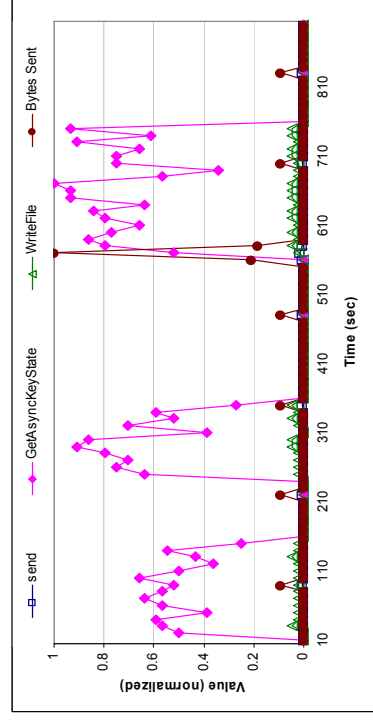
$$R_s = 1 - \frac{6 * \sum d^2}{n^3 - n}$$

## Objectives (2)

- To detect an individual bot on a machine
  - Monitoring selected API function calls executed by the bot
    - Communication functions (Send, recv, sendto, recvfrom, connect)
    - File access functions (CreateFile, OpenFile, ReadFile, WriteFile)
    - Keyboard status functions (GetAsyncKeyState, GetKeyboardState, keybd\_event)
- Each function call represents one activity
  - Idea: to correlate different activities of the process
- Correlation Methods
  - Spearman's Rank Correlation - SRC
  - Dendritic Cell Algorithm - DCA

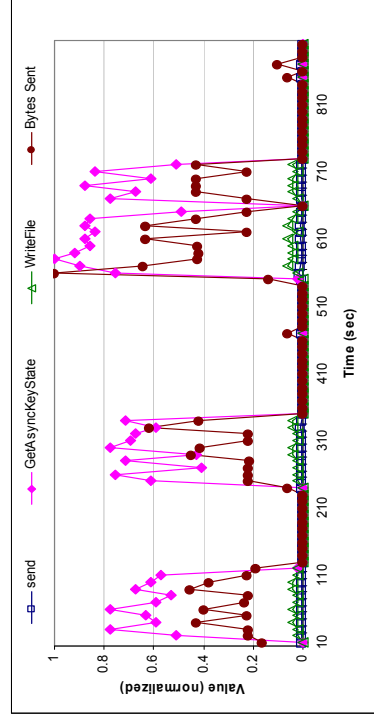
## Spearman's Rank Correlation – SRC

- Keylogger is activated and no data is transmitted



## Spearman's Rank Correlation – SRC

- Keylogger is activated and data is transmitted



## DCA – Dendritic Cell Algorithm

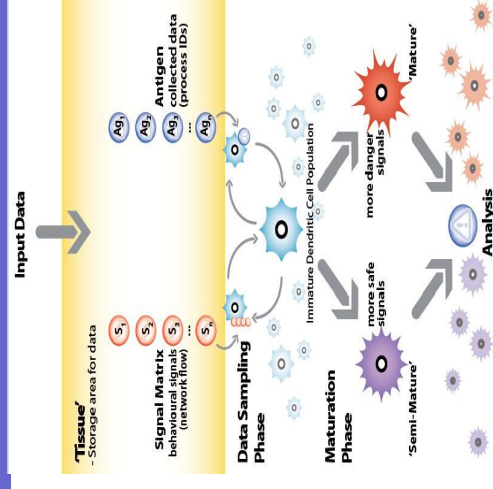
- **Three stages**
  - **Filtering**
    - Signals processing
  - **Correlation**
    - Correlating multiple input data streams (Signals with Antigens)
  - **Classification**
    - Normal/Abnormal
- **Antigen – data to be classified**
  - API function calls represented by process ID
- **Signals – data used to perform classification**
  - S1
    - Keyboard status functions
  - S2
    - Time difference between receiving and sending information
  - S3
    - Time difference between two consecutive communication functions

## Spearman's Rank Correlation – SRC

- **E1 - Inactive bot**
- **E2 - Active bot**
  - E4.1: Long sentences
  - E4.2: Short sentences
- No keylogger
- **E3 - Keylogger not activated**
  - E5: IRC
  - E3.1: Long sentences
  - E3.2: Short sentences
- Normal conversation

Experiment	SRC(Outgoing Traffic, SRC(GetAsyncKeyState, WriteFile))		SRC(GetAsyncKeyState, WriteFile)	Keylogging Activity existence	API Detection confidence
	S1	S2			
E1	0.863	0.671	1.000	No	N/A
E2	0.648	0.498	0.967	No	N/A
E3.1	0.509	0.183	0.559	Yes	Weak
E3.2	0.423	-0.003	0.928	Yes	Normal
E4.1	0.506	0.189	0.560	Yes	Weak
E4.2	0.927	0.579	0.663	Yes	Strong
E5	0.594	0.499	0.983	No	N/A

## DCA Algorithm



## DCA – MCAV/MAC

- **MCAV – Mature Context Antigen Value**
  - The probability of the process being anomalous
  - MCAV of process x is given by:
 
$$MCAV(x) = \frac{A(x)}{A(x) + N(x)}$$
  - Problem: small number of antigen leads to miss-classification

- **MAC – MCAV Antigen Coefficient**

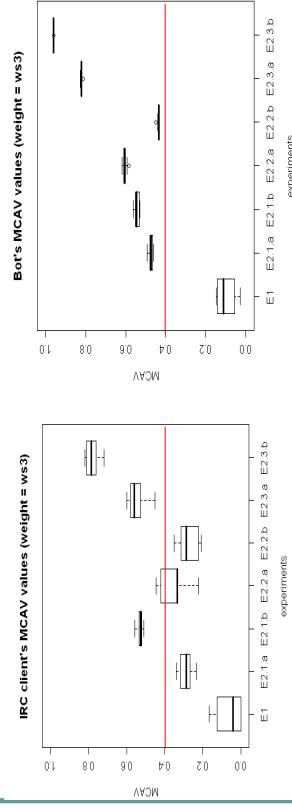
- MAC of x is given by:

$$MAC(x) = \frac{MCAV(x) * Antigen(x)}{\sum_i Antigen(i)}$$

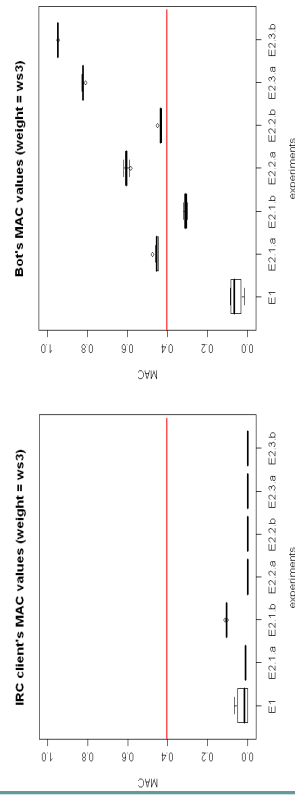
## Results

- **E1**
  - Inactive processes
- **E2.1 (keylogger)**
  - A: GetAsyncKeyState
  - B: GetKeyboardState
- **E2.2 (flood)**
  - A: SYN attack
  - B: UDP attack
- **E2.3 (keylogger + flood)**
  - A: SYN + GetAsyncKeyState
  - B: SYN + GetKeyboardState

## Results – MCAV for mIRC and Bot



## Results – MAC for mIRC and Bot



## Conclusion and Future Work

- Bots are the new threat to our systems
- Use intelligent way of correlating activities to detect the bot – Dentrific Cell Algorithm (DCA)
- P2P bot
  - C&C can be shutdown
  - P2P hard to disable
  - Apply DCA to detect P2P bots

## Thank You

- Any Questions!

## Signal Weight values

	Signal	WS1	WS2	WS3	WS4	WS5
L (csm)	S1	2	4	4	2	8
	S2	1	2	2	1	4
	S3	2	6	3	1.5	0.6
N (semi)	S1	0	0	0	0	0
	S2	0	0	0	0	0
	S3	1	1	1	1	1
A (mat)	S1	2	8	8	8	16
	S2	1	4	4	4	8
	S3	-3	-12	-6	-6	-1.2