

# Your Botnet is My Botnet: Analysis of a Botnet Takeover

HY558 – spring 2010

Thanasis Petsas

CSD, University of Crete

# What is a botnet?

- **Botnet:** a **network of compromised machines** that are under the control of a malicious entity, typically referred to as the botmaster
- **Bot:** a type of malware that is written with the intent of taking over a large number of hosts on the Internet
- Once infected with a bot, the victim host will join a botnet

# Malware evolution

- Malware was developed for fun, to the current situation, where malware is spread for financial profit
- Botnets are the primary means for cyber-criminals to carry out their nefarious tasks, such as
  - sending spam mails ,
  - launching denial-of-service attacks
  - stealing personal data such as mail accounts or bank credentials

# How to study botnets?

- One approach to study botnets is to perform **passive analysis** of secondary effects that are caused by the activity of compromised machines
  - Collected spam mails that were likely sent by bots
  - Similar measurements focused on DNS queries or DNS blacklist queries
  - analyzed network traffic (netflow data) at the tier-1 ISP level for cues that are characteristic for certain botnets
- The detection is limited to those botnets that actually exhibit the activity targeted by the analysis

# How to study botnets?

- **Active approach** to study botnets is via **infiltration**
  - Using an actual malware sample or a client simulating a bot, researchers join a botnet to perform analysis from the inside
  - To achieve this, honeypots, honey clients, or spam traps are used to obtain a copy of a malware sample
- Observe the traffic exchanged between the bot and the C & C server(s)

# How to study botnets?

- Attackers have unfortunately adapted, and most current botnets use stripped-down IR C or HTTP servers as their centralized command and control channels
- To overcome this limitations one can attempt to **hijack** the entire botnet by taking control of the physical machines that host the C & C infrastructure

# How to hijack a C & C server?

- By collaborating with domain registrars , it is possible to change the mapping of a botnet domain to point to a machine controlled by the defender
- Several recent botnets, including Torpig, use the concept of domain flux

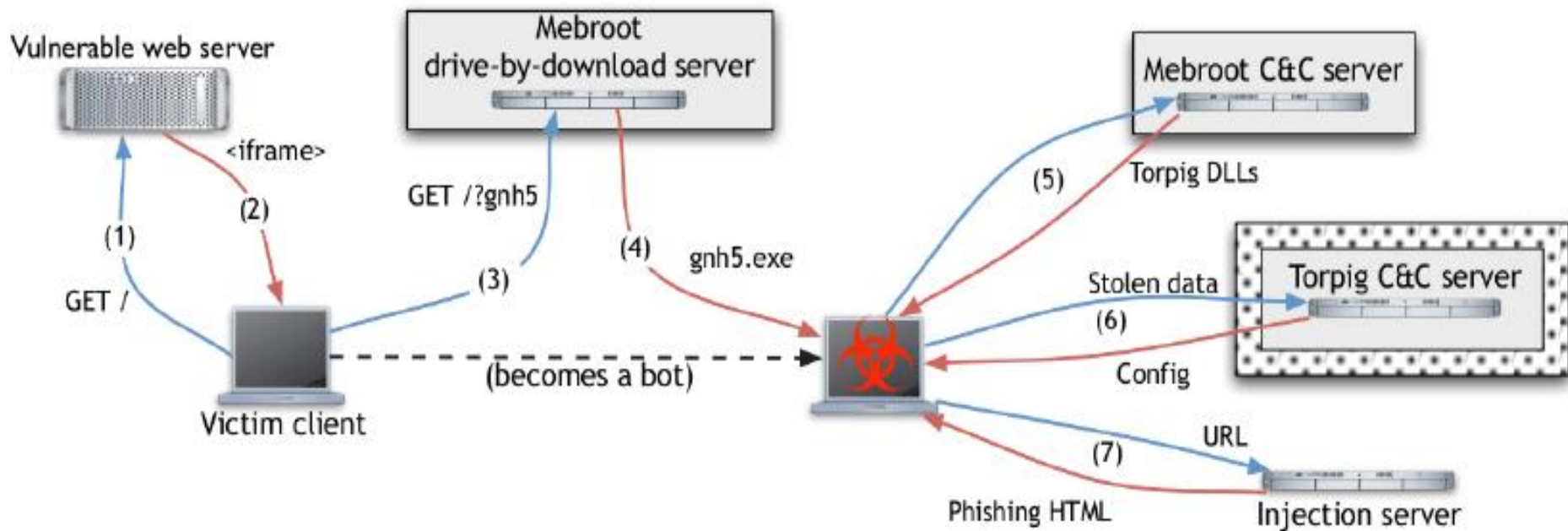
# Contributions

- Comprehensive analysis of the Torpig botnet
- For 10 days they observed information provided by more than 180.000 infected machines
- They study the size of the botnet and compare the results with alternative ways of counting the botnet population
- The analysis of the rich and diverse collection of user data provides a new understanding of the type and amount of personal information that is stolen by botnets

# Torpig

- Torpig has been distributed to its victims as part of Mebroot
  - Mebroot is a rootkit that takes control of a machine by replacing the system's Master Boot Record (MBR)
  - This allows Mebroot to be executed at boot time, before the operating system is loaded, and to remain undetected by most anti-virus tools

# Life cycle of Torbig



# Torpig's Phishing Attacks

- Torpig uses phishing attacks to actively elicit additional, sensitive information from its victims, which, otherwise, may not be observed during the passive monitoring it normally performs
  - First, whenever the infected machine visits one of the domains specified in the **configuration file** (typically, a banking web site), Torpig issues a request to an **injection server**. The server's response specifies a page where attack should be triggered (trigger page)
  - The second step occurs when the user visits the **trigger page**. At that time, Torpig requests the injection URL from the injection server and injects the returned content into the user's browser

# Man-in-the-browser phishing attack (example)

The screenshot shows a Windows Internet Explorer browser window with the title "Wells Fargo - Windows Internet Explorer". The address bar displays "https://online.wellsfargo.com/signon". The page content includes the Wells Fargo logo, a search bar, and navigation links for "Customer Service", "Locations", "Apply", and "Home". Below these are tabs for "Banking", "Loans & Credit", "Insurance", "Investing", and "Customer Service". The main content area is titled "Security Confirmation" and contains the following text: "To continue with Online Banking, please provide the information requested below." followed by input fields for "First Name:", "Last Name:", "Date of Birth (mm/dd/yyyy):", "Social Security Number:", "Mother's Maiden Name:", and "Card Number:". A note at the bottom of the form says "Enter 16-digit number printed on your ATM/Check Card." The browser's status bar at the bottom shows "Contains commands for working with the selected items." and the taskbar at the very bottom shows the Start button and a taskbar icon for "Wells Fargo - Window...".

# Coordination Schemes

- Botnet authors have identified several ways to make these schemes more flexible and robust against take-down actions, e.g., by using **IP fast-flux** techniques
- With fast-flux, the bots would query a certain domain that is mapped onto a set of IP addresses, which change frequently
- However, fast-flux uses only a single domain name, which constitutes a single point of failure

# Domain Flux

- Torpig solves this issue by using a different technique for locating its C&C servers, which we refer to as **domain flux**
- If a domain is blocked, the bot simply rolls over to the following domain in the list
- Using the generated domain name dw, a bot appends a number of TLDs: in order, dw.com, dw.net, and dw.biz

# Torpig's D G A

```
suffix = ["anj", "ebf", "arm", "pra", "aym", "unj",  
          "ulj", "uag", "esp", "kot", "onv", "edc"]  
  
def generate_daily_domain():  
    t = GetLocalTime()  
    p = 8  
    return generate_domain(t, p)  
  
def scramble_date(t, p):  
    return ((t.month ^ t.day) + t.day) * p) +  
           t.day + t.year  
  
def generate_domain(t, p):  
    if t.year < 2007:  
        t.year = 2007  
    s = scramble_date(t, p)  
    c1 = (((t.year >> 2) & 0x3fc0) + s) % 25 + 'a'  
    c2 = (t.month + s) % 10 + 'a'  
    c3 = ((t.year & 0xff) + s) % 25 + 'a'  
    if t.day * 2 < '0' || t.day * 2 > '9':  
        c4 = (t.day * 2) % 25 + 'a'  
    else:  
        c4 = t.day % 10 + '1'  
    return c1 + 'h' + c2 + c3 + 'x' + c4 +  
           suffix[t.month - 1]
```

# Botnet Protection

- 2 requirements that the botmasters must satisfy to maintain their grip on the botnet:
  1. They must control at least one of the domains that will be contacted by the bots
  2. They must use mechanisms to prevent other groups from seizing domains that will be contacted by bots before the domains under their control
- Feasibility of sinkholing attacks
- Newer variants of Conficker generate **50,000** domains per day and introduce non-determinism in their generation algorithm

# Taking Control of the Botnet

- we were able to register the .com and .net domains that were to be used by the botnet for three consecutive weeks from January 25th, 2009 to February 15th, 2009
- However, on February 4 th, 2009 , the Mebroot controllers distributed a new Torpig binary that updated the domain algorithm

# Taking Control of the Botnet

- During the ten days that we controlled the botnet, they collected over 8.7 GB of Apache log files and 69 GB of pcap data
- However, on January 19th, when we started our collection, they instantly received HTTP requests from 359 infected machines

# Protecting their victims

- **PRINCIPLE 1**

- The sinkholed botnet should be operated so that any harm and/or damage to victims and targets of attacks would be minimized

- **PRINCIPLE 2**

- The sinkholed botnet should collect enough information to enable notification and remediation of affected parties

# Botnet Analysis

- 70GB of data over a period of 10 days
- All bots communicate with Torpig C&C through HTTP POST requests:
  - URL: hex(bot id) + submission header
  - Body: data stolen from victim's machine
- The submission header and the body are encrypted using the Torpig encryption algorithm (base64 and XOR).
  - The header contains the time stamp when the configuration file was last updated (ts),
  - the IP address of the bot (ip),
  - the port numbers of the HTTP and SOCKS proxies that Torpig opens on the infected machine (hport and sport),
  - the operating system version and locale (os and cn),
  - the bot identifier (nid),
  - and the build and version number of Torpig (bld and ver)

# Torpig's submission header

*nid*

*submission header (encrypted)*

POST /A15078D49EBA4C4E/qxoT4B5uUFFqw6c35AKDYFpdZHdKLCNn...AaVpJGoSZG1at6E0AaCxQg6nIGA

*submission header (unencrypted)*

ts=1232724990&ip=192.168.0.1:&sport=8109&hport=8108&os=5.1.2600&cn=United%20States&  
nid=A15078D49EBA4C4E&bld=gnh5&ver=229

# Request body

- The request body consists of zero or more *data* types, depending on information that was stolen

Data Type	Data Items (#)
Mailbox account	54,090
Email	1,258,862
Form data	11,966,532
HTTP account	411,039
FTP account	12,307
POP account	415,206
SMTP account	100,472
Windows password	1,235,122

```
[gnh5_229]
[MSO2002-MSO2003:pop.smith.com:John Smith:
  john@smith.com]
[pop3://john:smith@pop.smith.com:110]
[smtp://:@smtp.smith.com:25]
```

a mailbox account

```
[gnh5_229]
POST /accounts/LoginAuth
Host: www.google.com
POST_FORM:
Email=test@gmail.com
Passwd=test
```

a form data item

# Botnet Size

- Counting Bots by nid
- The algorithm first queries the primary **SCSI hard disk** for its **model** and **serial numbers**. If no SCSI hard disk is present, or retrieving the disk information is unsuccessful, it will then try to extract the same information from the **primary physical hard disk** drive (i.e., IDE or SATA). The disk information is then used as input to a hashing function that produces the final nid value
- If retrieving hardware information fails, the nid value is obtained by concatenating the hard-coded value of 0xBAD1D222 with the **Windows volume serial number**

# Botnet Size

- As a reference point, between Jan 25, 2009 and February 4, 2009, **180,835** nid values were observed (2,079 cases reported different values for os, cn, bld and ver fields)
- By counting unique tuples from the Torpig headers consisting of (nid, os, cn, bld, ver), we estimate that the botnet's footprint for the ten days of our monitoring consisted of **182,914** machines
- After subtracting probers and researchers, our final estimate of the botnet's footprint is **182,800** hosts.
- 2 heuristics
  - *nid* generated by infected clients running in VM is constant (VMs provide virtual devices with fixed model and serial number) → -40 hosts
  - invalid requests that cannot be generated by Torpig bots (GET instead POST request) → -74 hosts

# Number of unique IP addresses and bots during the 10 days

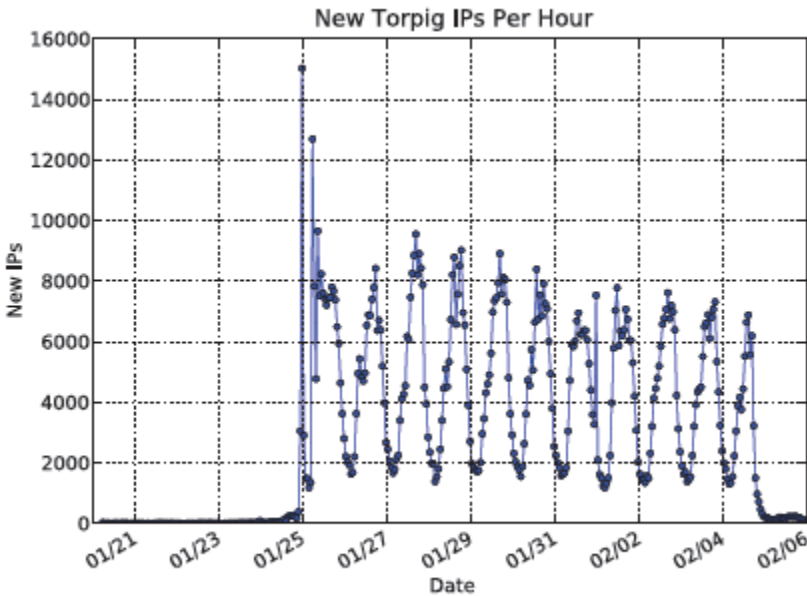


Figure 5: New unique IP addresses per hour.

Total: 1,247,642  
4,690 Ips/hour

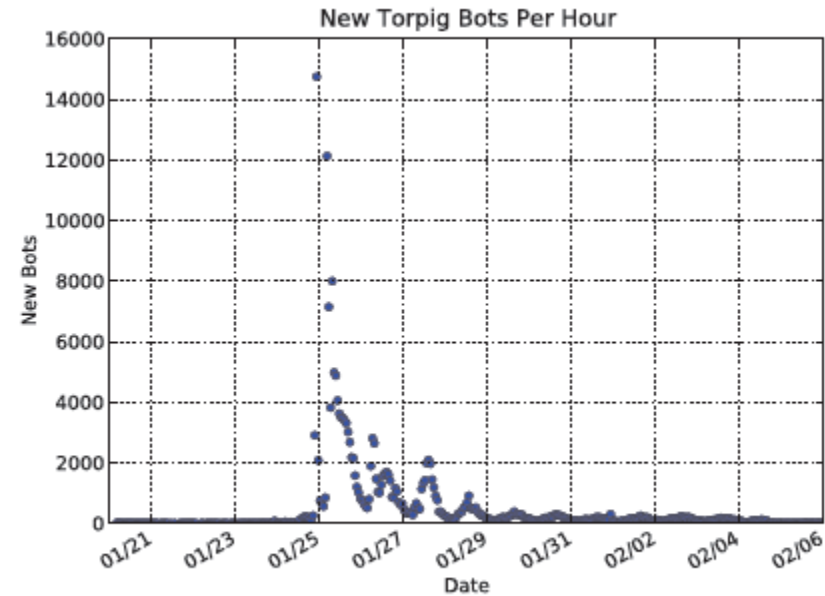


Figure 6: New bots per hour.

Total: 182,800  
705 bots/hour

# Number of unique IP addresses and bots per hour

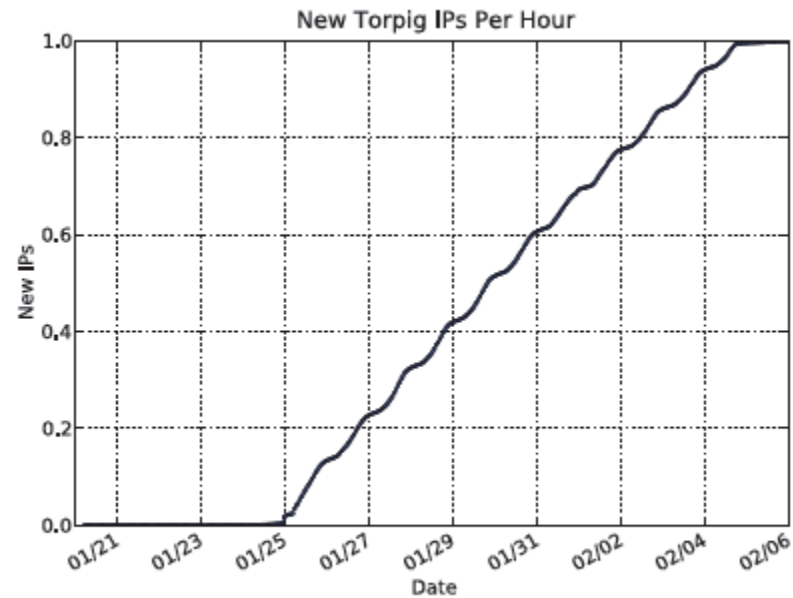


Figure 7: CDF – New unique IP addresses per hour.

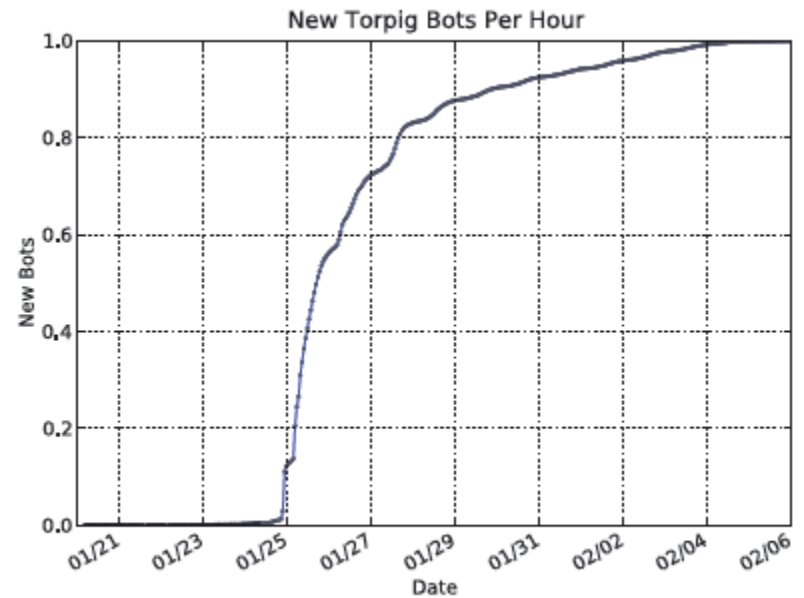


Figure 8: CDF – New bots per hour.

75 % of all new Torpig bots  
During the 10-day were  
Observed in the first 48  
hours

# Unique Bot IDs and IP addresses per hour and per day

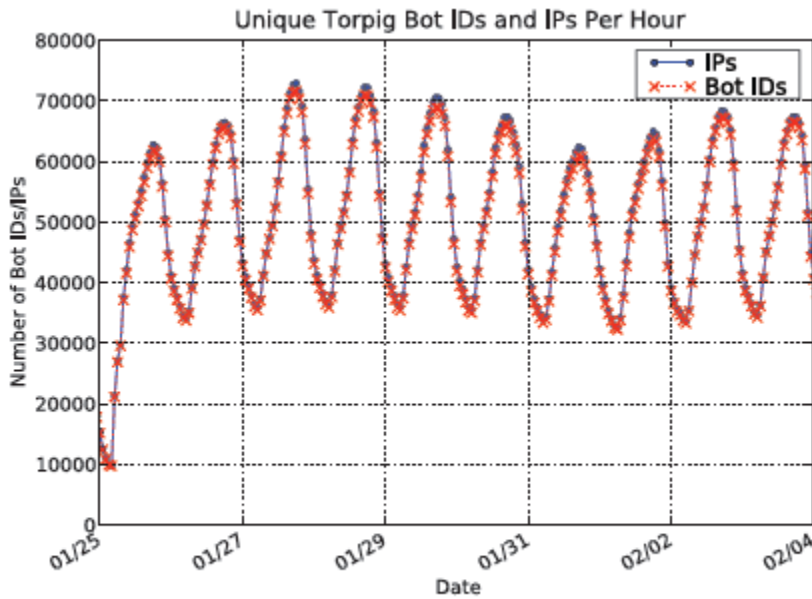


Figure 9: Unique Bot IDs and IP addresses per hour.

On average, the bot IDs were only 1.3% less than the number of IP addresses per hour. Thus, the number of unique IPs per hour provides a good estimation of the botnet's live population

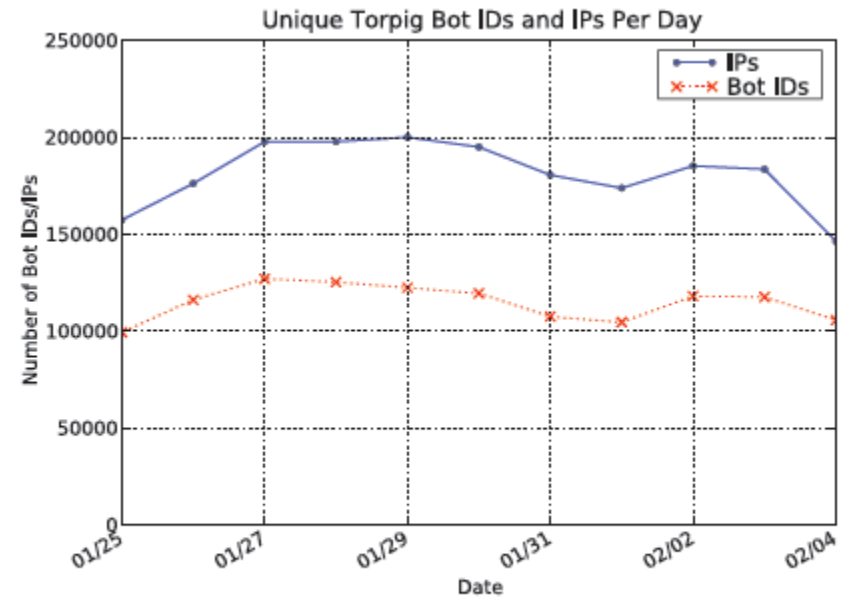


Figure 10: Unique Bot IDs and IP addresses per day.

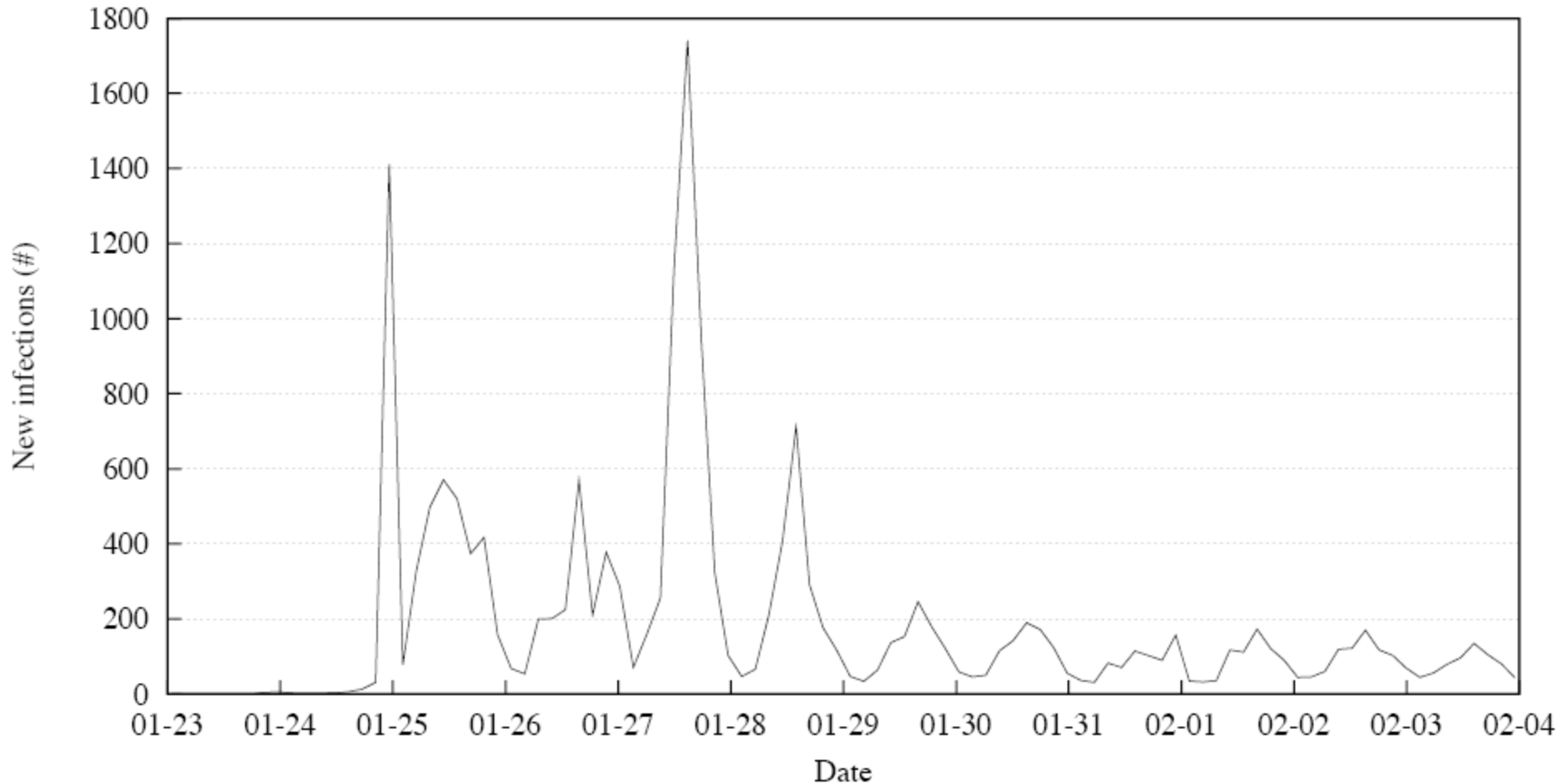
On average, the bot IDs were 36.5% less than the number of IP addresses per day. Thus, the number of IPs per day does not accurately reflect the botnet's live population

# The effect of DHCP churn at a country level

Country	IP Addresses (Raw #)	Bot IDs	DHCP Churn Factor
US	158,209	54,627	2.90
IT	383,077	46,508	8.24
DE	325,816	24,413	13.35
PL	44,117	6,365	6.93
ES	31,745	5,733	5.54
GR	45,809	5,402	8.48
CH	30,706	4,826	6.36
UK	21,465	4,792	4.48
BG	11,240	3,037	3.70
NL	4,073	2,331	1.75
Other	180,070	24,766	7.27
Totals:	1,247,642	182,800	6.83

**Table 2: Top 10 infected hosts by country.**

# New infections over time



Total: **49,294** new infections while the C&C was under their control

# Accounts at financial institutions that were stolen

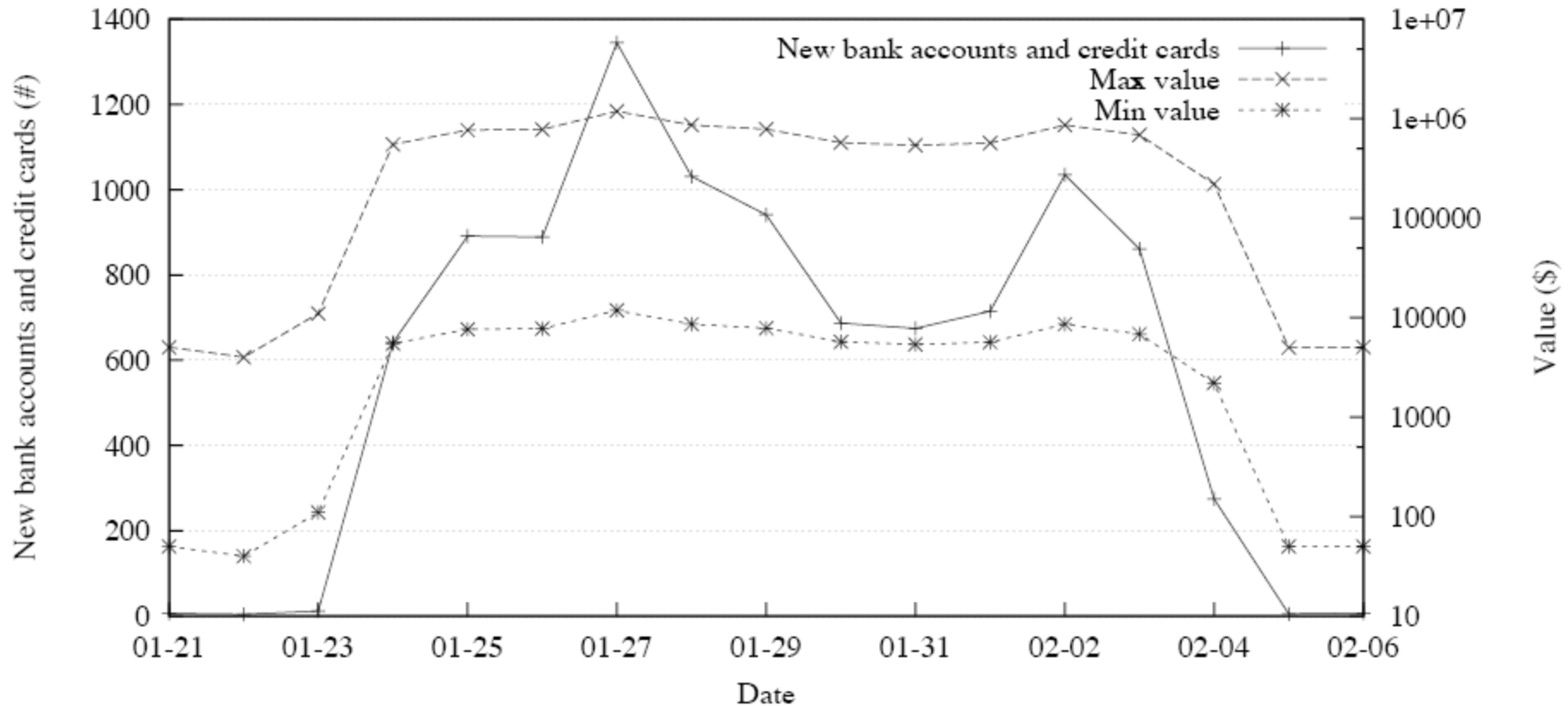
- the typical Torpig configuration file lists roughly 300 domains belonging to banks and other financial institutions that will be the target of the “man-in-the-browser” phishing attacks

## top targeted institutions:

PayPal (1,770 accounts),  
Poste Italiane (765),  
Capital One (314),  
E\*Trade (304), and  
Chase (217)

Country	Institutions (#)	Accounts (#)
US	60	4,287
IT	34	1,459
DE	122	641
ES	18	228
PL	14	102
Other	162	1,593
Total	410	8,310

# Rate at which new bank accounts & credit card numbers were obtained



**Figure 12: The arrival rate of financial data.**

# Network speed of infected hosts

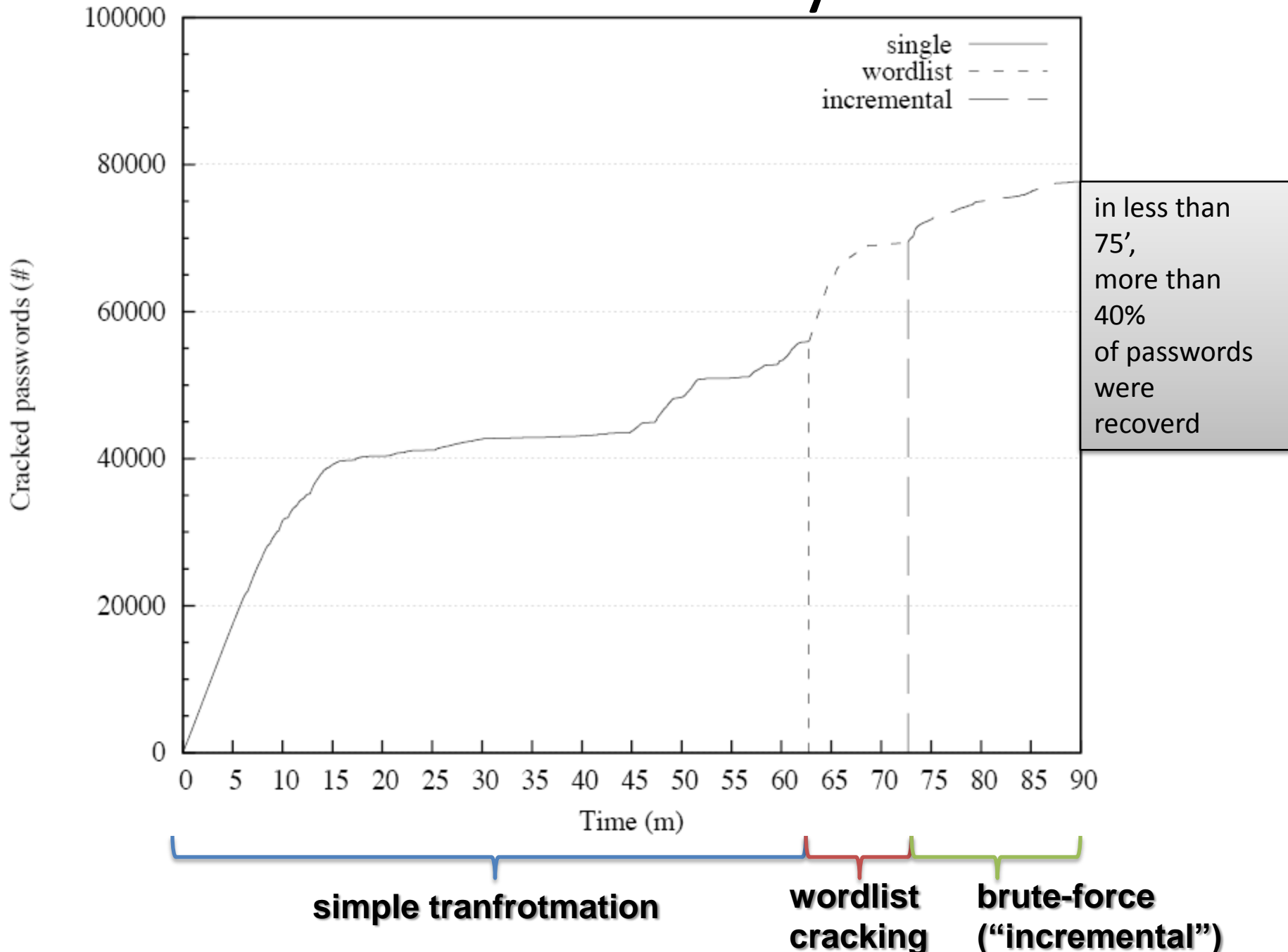
Network Speed	IP Addresses (Raw #)	Bot IDs	DHCP Churn Factor
Cable/DSL	356,428	50,535	7.05
Dial-up	129,493	9,923	13.05
Corporate	40,818	17,217	2.37
Unknown	677,434	105,125	6.44

- They mapped the IP addresses to their network speed, using the ip2location2 database (<http://www.ip2location.com>)
- **cable and DSL lines account for 65%** of the infected hosts
- a tremendous amount of bandwidth in the hands of the botmaster, considering that there were more than 70,000 active hosts at peak intervals
- a botnet of this size could cause a massive distributed denial-of-service (**DDoS**) attack

# Password Analysis

- Torpig bots stole 297,962 unique credentials sent by 52,540 different infected machines
- Our analysis found that almost **28% of the victims reused their credentials** for accessing 368,501 web sites.

# Password Analysis



# Conclusion

- They present a comprehensive analysis of the operations of the Torpig botnet
- First, they found that a naïve evaluation of botnet size based on the count of distinct IPs yields grossly overestimated results
- Second, the victims of botnets are often users with poorly maintained machines that choose easily guessable passwords to protect access to sensitive sites

Question?