Graph-based Detection of Anomalous Network Traffic

Do Quoc Le

Supervisor: Prof. James Won-Ki Hong

Distributed Processing & Network Management Lab
Division of IT Convergence Engineering
POSTECH, Korea
lequocdo@postech.ac.kr

2012. 06. 22

Contents

- Introduction & Motivation
- Related Work
- Graph-based Network Traffic Modeling
- Graph Metrics
- Anomaly Detection & Attack Identification
- Validation
- Conclusion

Introduction & Motivation

- **The Internet continues to grow in size and complexity**
  - Security has become a critical issue.
  - The occurrence of traffic anomalies (DDoS, flash crowds, port scans and worms).
- **Challenges:**
  - Increasingly sophisticated attacks.
  - Attacks are often hidden in existing applications, e.g. IRC, HTTP, or Peer-to-Peer: Worm scans or botnet C&C traffic.
  - Methods for detecting traffic anomalies.
    - Signature-based techniques
      - Cannot detect anomalies caused by unknown attacks.
    - Anomaly-based techniques: (Machine learning, data mining the statistical analysis, etc.)
      - Generate a huge number of false alarms.
      - Time consuming.
      - Cannot detect anomalies whose traffic is similar with normal applications (traffic volume, number of packets, number of flows and average packet size).

Introduction & Motivation

- **Goal:** Improve detection accuracy and the ability of the state of art techniques for anomaly detection.
- **Solution:**
  - Using a graph-based method to monitor network traffic and analyze the structure of communication patterns to detect anomalies and identify attacks.
- **Why we study the structure of communication patterns in network traffic?**
  - Each attack has its own structure.
  - Communication patterns' structure changes when attacks occur.
  - Can identify when attacks occur that can be difficult to detect using conventional means.

Contribution

- **One of the first works using a Traffic Dispersion Graphs (TDGs) to detect anomalies**
  - Focus on structural characteristics of networks.
  - Improve performance and ability of the state of the art techniques.
  - Support intuitive visualization of traffic patterns.
- **Introduce a new metric to analyze network traffic communication patterns overtime**
- **Implement an online anomaly detection system in an Enterprise network based on the proposed method**
- **Evaluate the approach by analyzing real attack traces**

Related Work

- **Zhou *et al.* [1] proposed a network traffic anomaly method based on graph mining**
  - Mining time-series graphs.
  - Mining edge weight.
  - Entropy of four attributes: source and destination IP address, source and destination port.
  - The drawback: Enormous size → computational complexity.

We analyze unlabeled graphs and just concentrate on their nodes

- **Godiyal *et al.* [2] used a graph matching method to identify attacks**
  - Applying isomorphism algorithm for whole traffic flow → very time consuming.

We identify attacks in abnormal network traffic only

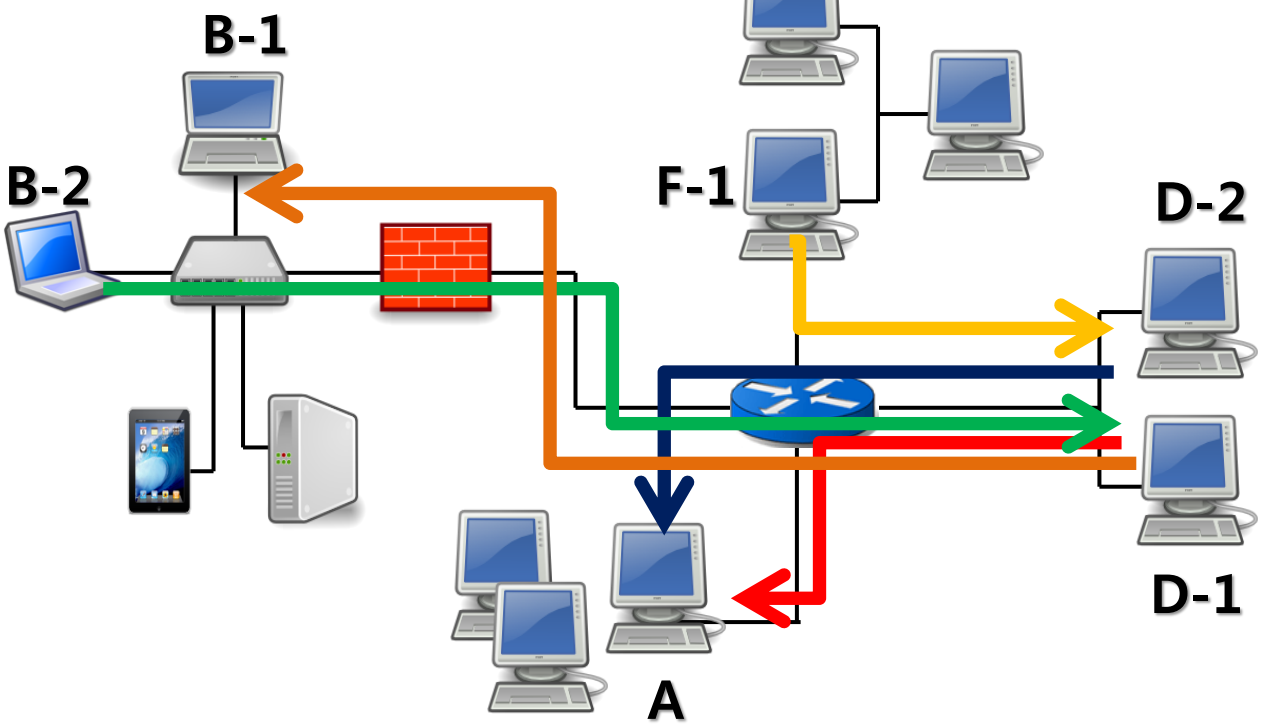
Related Work (cont.)

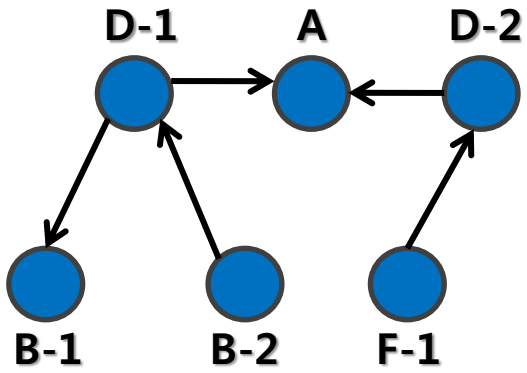
- **Iliofotou *et al.* [3] use TDG to model network traffic as series of related graphs over time**
  - Using graph metrics
    - Degree, degree distribution
    - Entropy of degree distribution
    - Graph edit distance
  - Solving problem of traffic classification, possible application to anomaly detection.

We model network traffic as TDG over time using new metrics.

❖ Traffic Dispersion Graph (TDG)

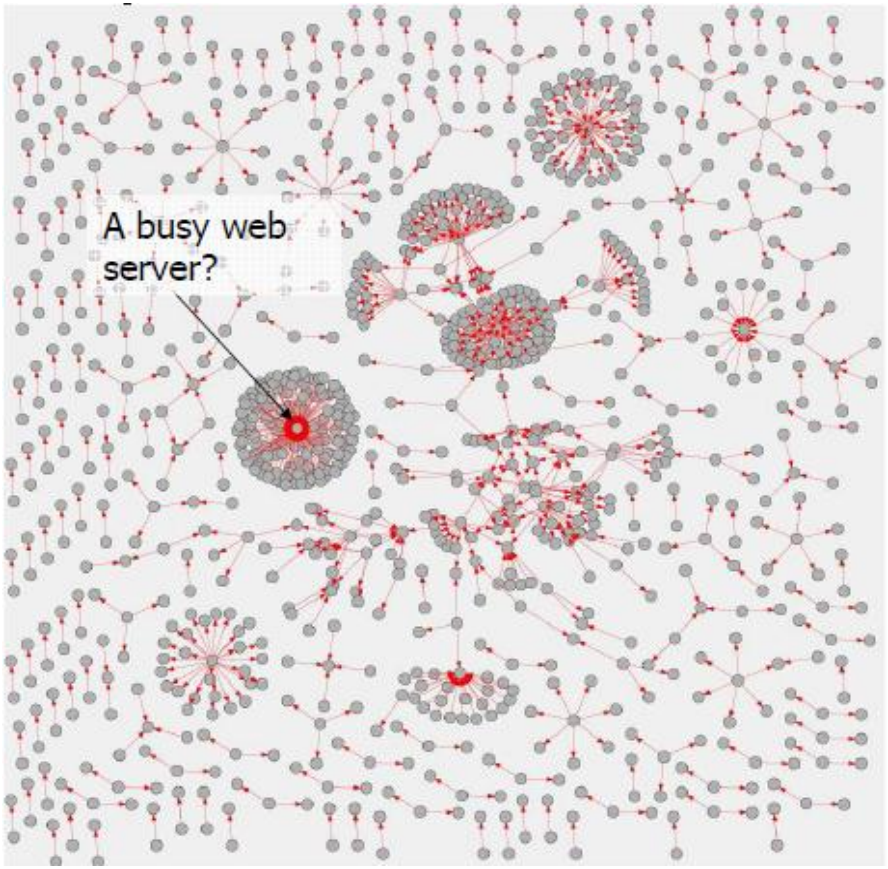
- Each node → IP address.
- Each edge → interaction (flow) between two nodes.

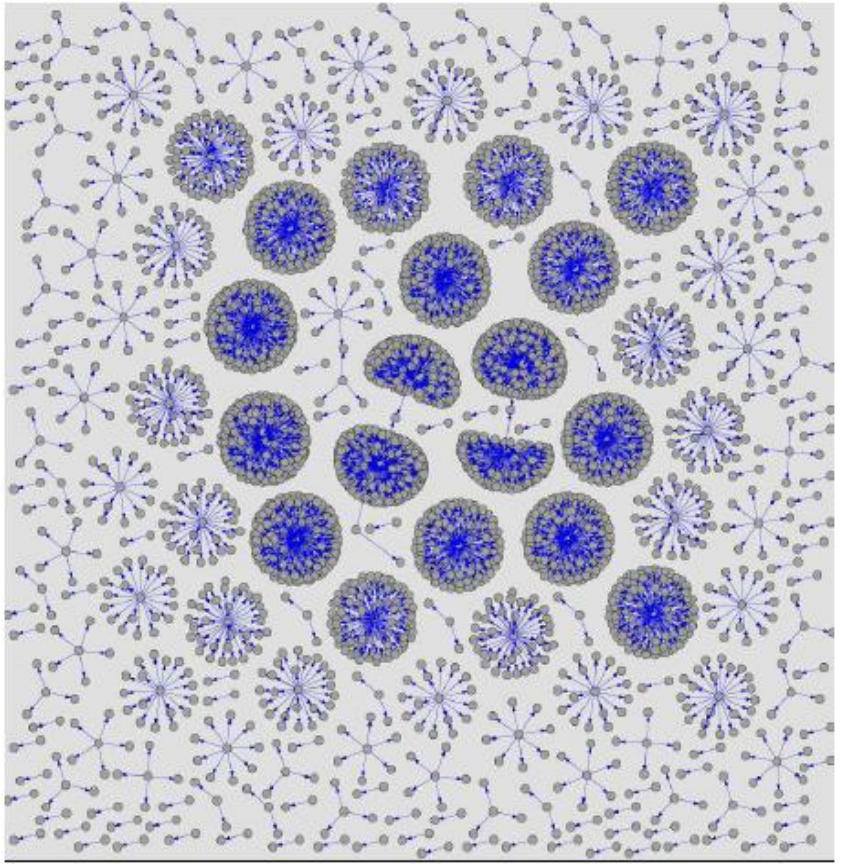




Generated TDG

TDG Visualization





❖ HTTP

- Many disconnected components
- Very few nodes with in and-out degrees
  - Web proxies?

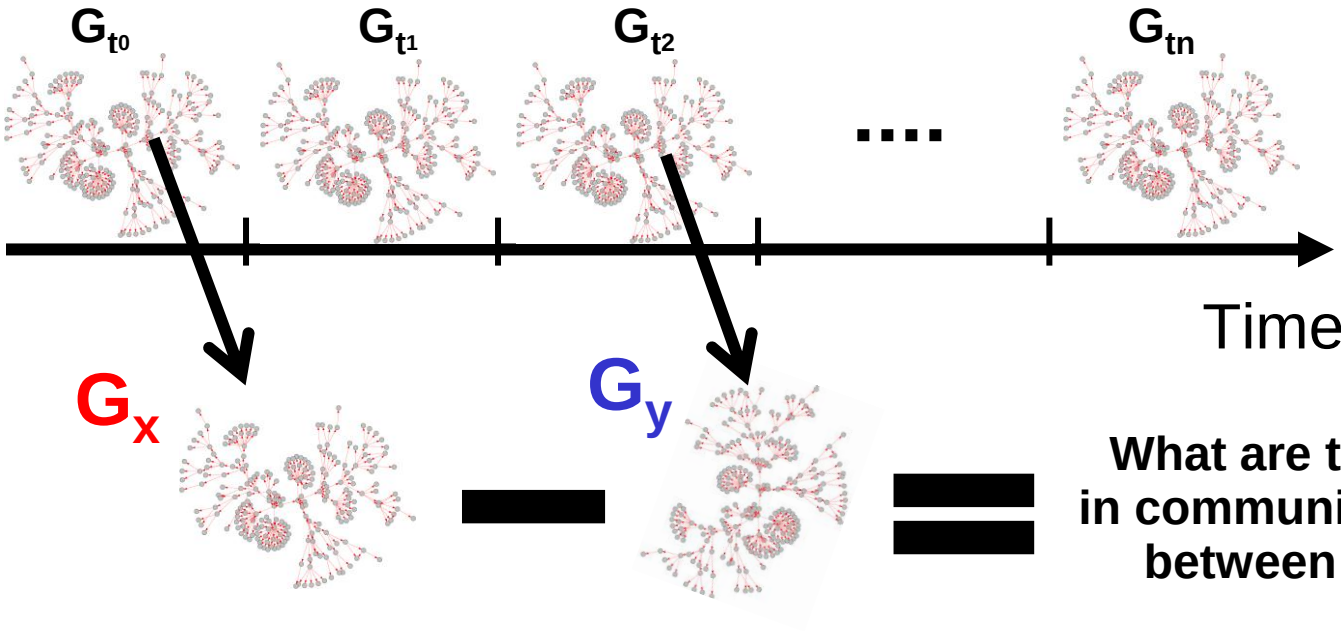
❖ Slammer Worm

- UDP Dst. port 1434
- Many high out-degree nodes
- Many disconnected components
- The majority of nodes have only in-degree
  - Nodes being scanned

Source: Iliofotou et al.

Graph Metrics on TDGs

- **What we have seen so far: "Visualization is useful by itself"**
  - However, it requires a human operator.
- **Next step?**
  - Translate **visual intuition** into quantitative measures.
- **How to quantitatively characterize properties of TDGs?**
  - **Step 1:** represent traffic as a sequence of graph snapshots.
  - **Step 2:** use metrics that quantify differences between graphs.



What are the differences in communication structure between Gx and Gy?

Graph Metrics on TDGs – Static metrics

❖ Node degree

- In-degree
- Out-degree

❖ Degree distribution

- Show an approximate power-law.

❖ Maximum degree (Kmax)

- One of metrics to detect DDoS attack.

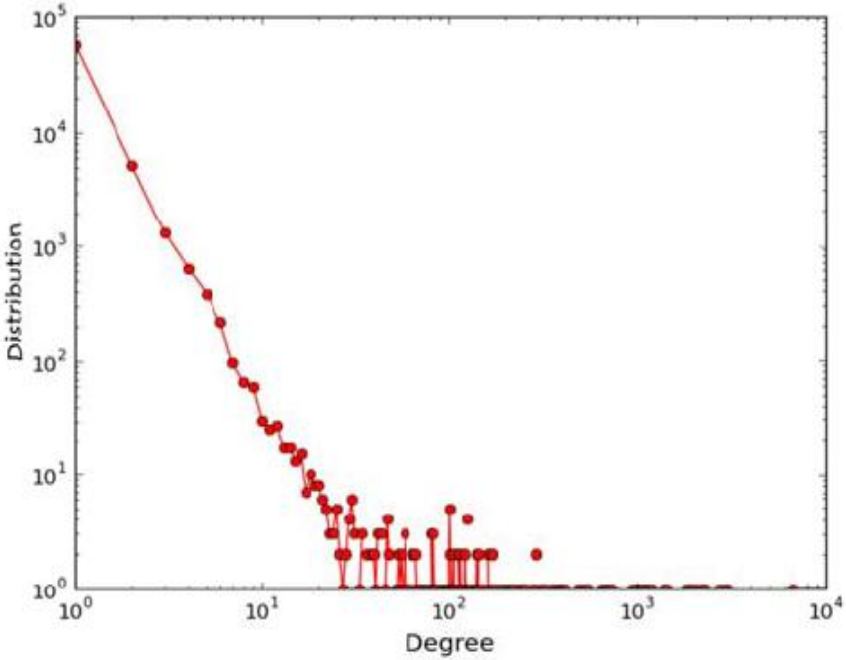


Figure 2. One-minute degree distributions of POSTECH traffic, obtained within one hour.

❖ Degree Assortativity

- Measure the tendency for nodes to be connected to similar nodes in term of their degree.

❖ Entropy of degree distribution

- Quantify heterogeneity of network : $H(X) = -\sum_{k=1,k_{max}} P(k)\log(P(k))$

Where P(k) is the probability that a node has degree k.

❖ Graph edit distance

$$d(G_i, G_j) = |V_i| + |V_j| - 2|V_i \cap V_j| + |E_i| + |E_j| - 2|E_i \cap E_j|$$

Where V_i , E_i and V_j , E_j are the numbers of nodes and edges in graph G_i and G_j , respectively.

❖ dK-2 distance metric

- Based on dK-series concept
  - Structure analysis - dK-n series: n=1,2,3,…
  - Look at inter-dependencies among topology characteristics.
  - dK-n series are degree correlations within simple connected graphs of size n.
  - dK-2 describes joint node degree distribution.
- dK-2 distance(G,G') = Euclidean distance between dK-2(G) and dK-2(G')

Anomaly Detection & Attack Identification

- Using graph metrics to detect abnormal network traffic.
- Anomalies: attacks which change communication structure in network(DDoS attacks, Internet worms and scanning)
- The overall process consist of two parts: anomaly detection and attack identification

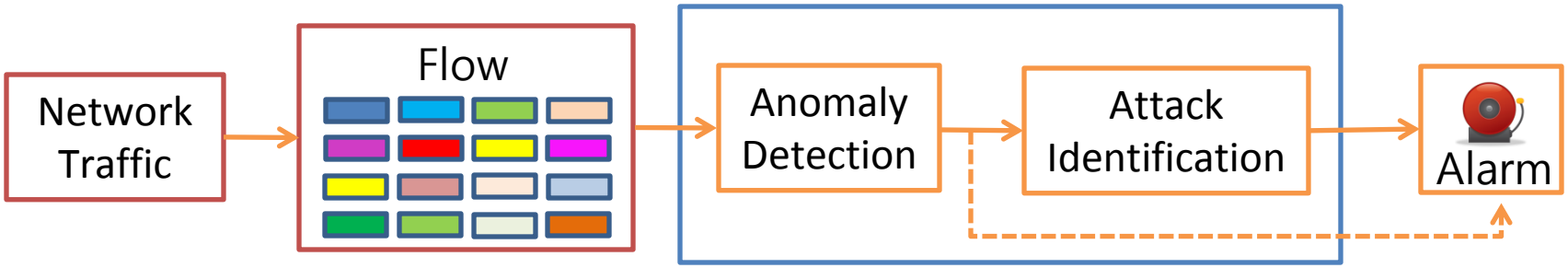


Figure 4. Overall detection process.

❖ Anomaly Detection

- **Step 1:** Sampling network traffic and generating network flows.
- **Step 2:** Creating TDG (Dot format) from network flows in time sampling intervals.
- **Step 3:** Calculating adjacency matrices of the TDG and calculating graph metrics of the TDG.
- **Step 4:** Comparing values of graph metrics of the TDG with their threshold value.
  - Graph metric value < Threshold → normal TDG.
  - Graph metric value > Threshold → abnormal TDG.

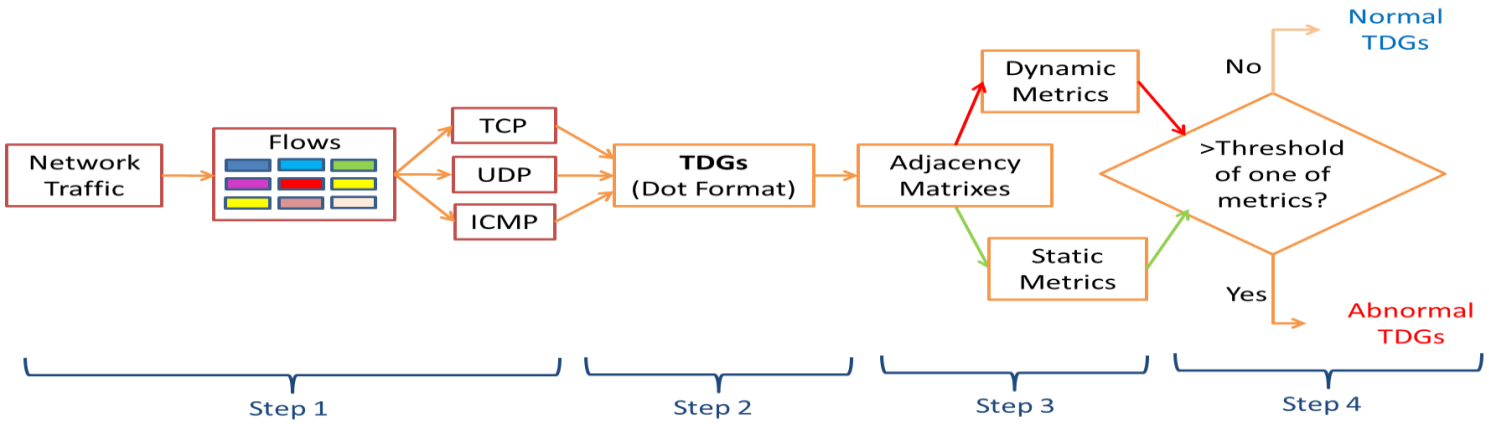


Figure 5. Detailed anomaly detection process.

❖ Attack Identification

- Attack pattern:

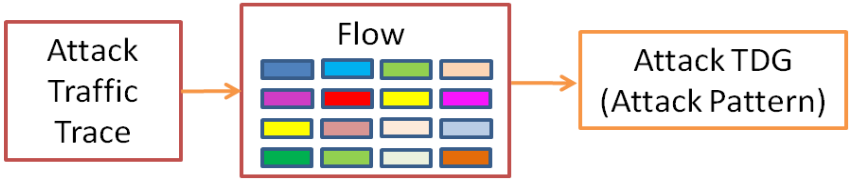
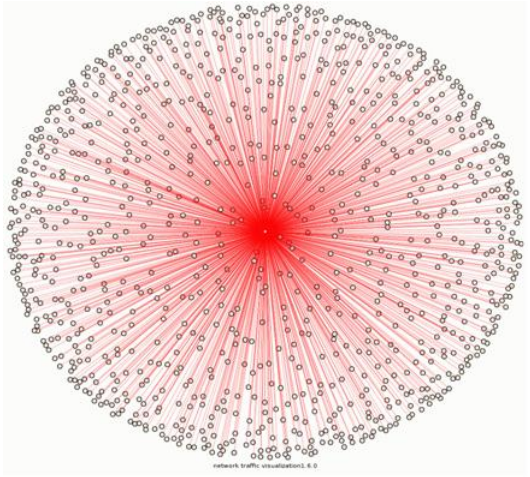


Figure 7. Attack pattern generation process.



CAIDA DDoS Trace in 2007

Figure 8. DDoS attack pattern in DDoS CAIDA trace.

- Attack identification:

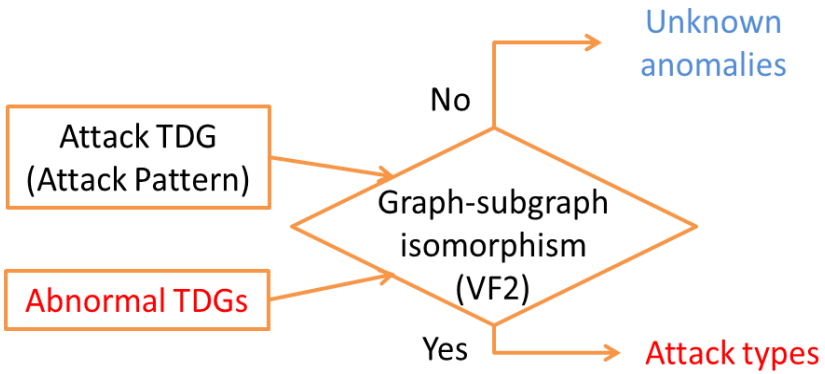


Figure 11. Attack identification process.

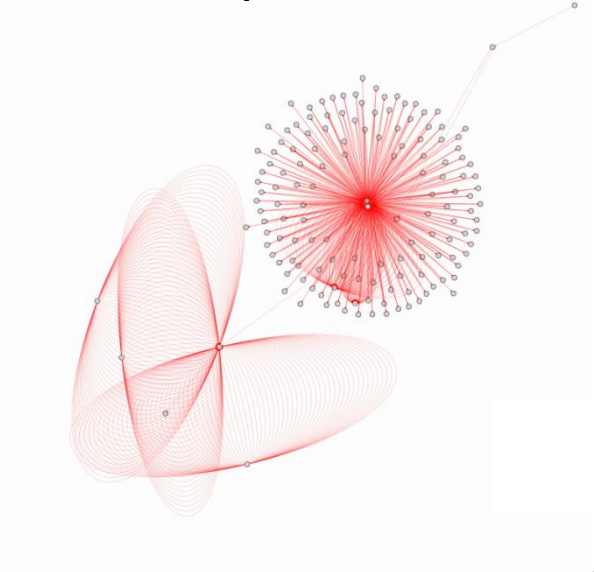


Figure 9. Peacomm P2P botnet pattern.

Validation

❖ Off-line analysis

- Trace
  - DARPA 1999 Dataset
    - Week 1 and week 3: no attack (for training data).
    - Week 2: 43 attacks belonging to 18 labeled attack types are used for system development.
    - Week 4 and week 5: 201 attacks belonging to 58 attack types (including 40 new attacks).
  - POSTECH trace in 2009. 7. 9.
    - Contain a famous DDoS attack on July 7, 2009 in South Korea.
  - CAIDA DDoS trace in 2007.
  - P2P Botnet trace (Peacomm) from a honeynet.

❖ On-line analysis

- Real-time anomaly detection
  - Testing with port scanning attack

Validation (DARPA dataset)

❖ DARPA 1999 Dataset

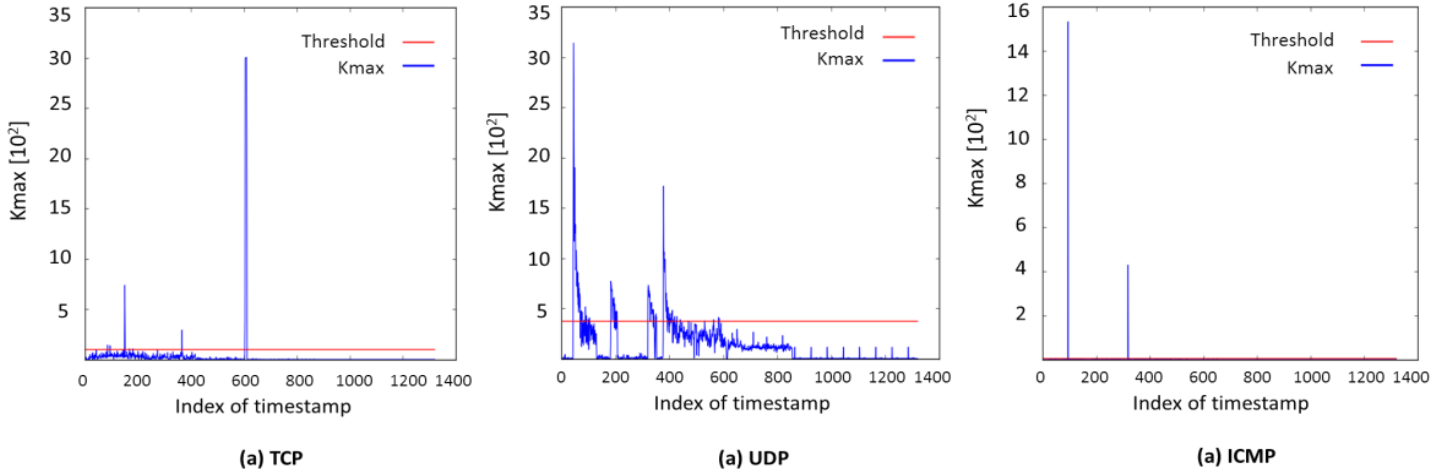


Figure 12. Kmax per minute over one day (Monday, Week 5) with normal and attacking traffic.

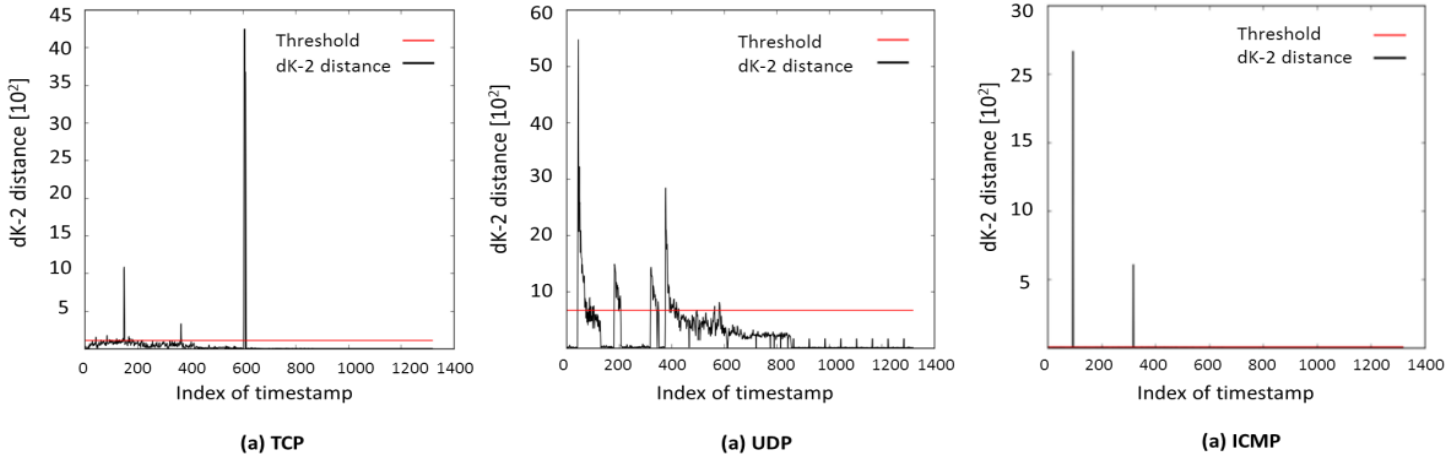


Figure 13. dK-2 distance value per minute over one day (Monday, Week 5) with normal and attacking traffic.

Validation (DARPA dataset)

❖ DARPA 1999 Dataset

Table 2. Performance of the Graph-based method using Kmax and dK-2 distance metric on Monday, Week5 traffic.

Total instances	Attacking instances	DR	FPR	CR
1320	122	100 %	1.25 %	98.86 %

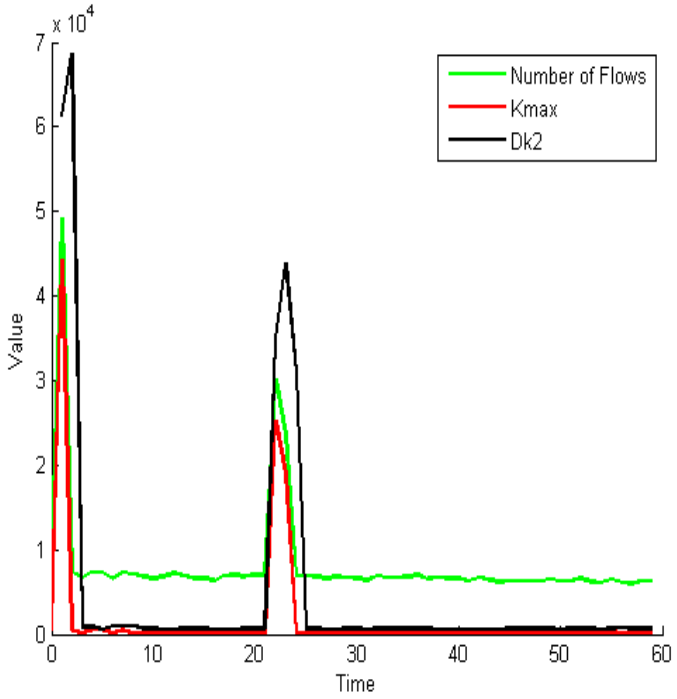
Table 3. Number of attack instances detected for each attack type on Monday, Week5 traffic.

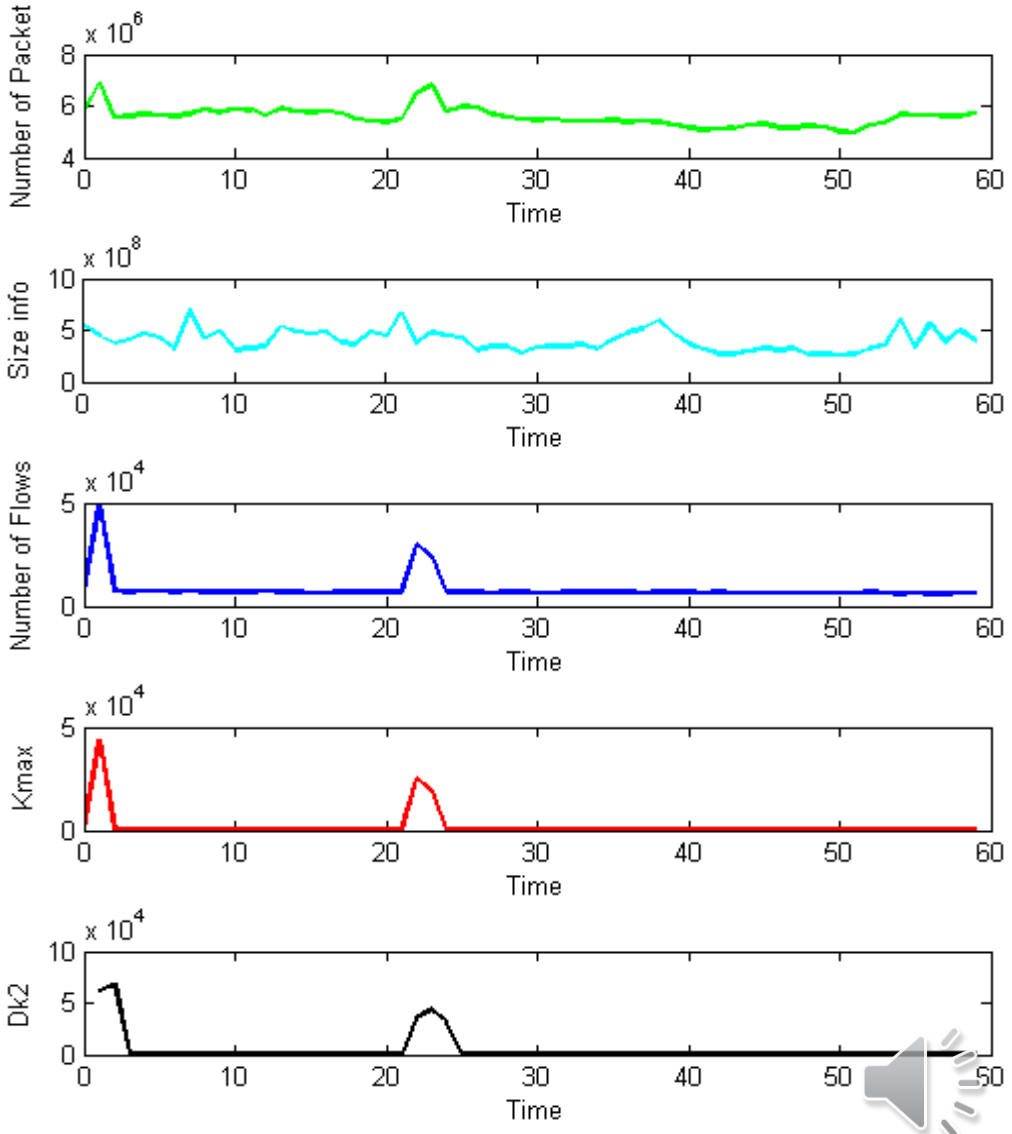
Attack Type	Number of attack instances for each attack type	Number of detected attack instances for each attack type
apache2-dos	30	30
arppoison-probe	15	15
dict-r2l	17	17
guesstelnet-r2l	4	4
ipsweep-prob	26	26
ls-probe	2	2
neptune-dos	5	5
portsweep-probe	4	4
smurf-dos	2	2
udpstorm-dos	16	16
crashiis-dos	1	1

Validation (POSTECH July, 2009)

❖ POSTECH traces on July, 2009

Date	DDoS Attack	Trace Size
03/31	No	30.7 GB
07/08	Yes	27.3 GB





Validation (POSTECH July, 2009)

❖ POSTECH traces on July, 2009

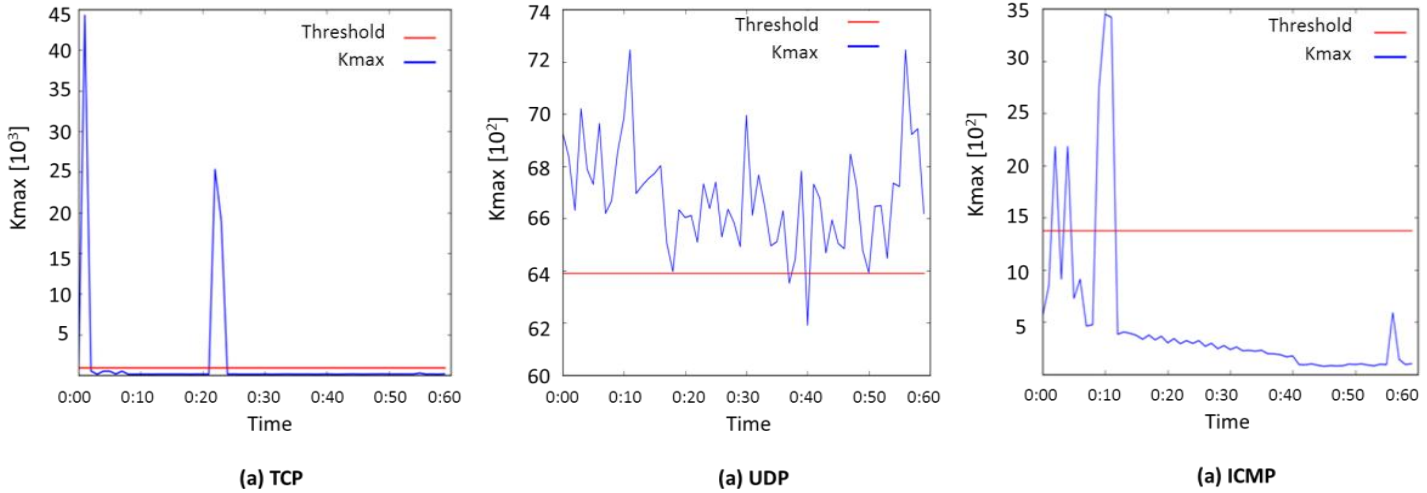


Figure 15. Kmax value over time of POSTECH's trace on July 8th 2009.

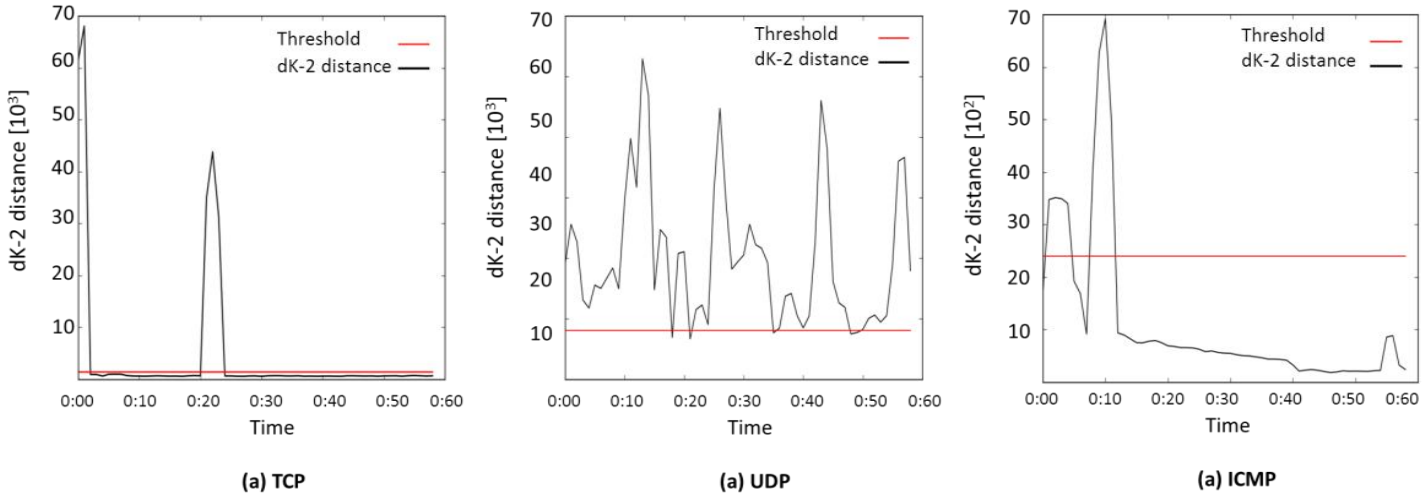
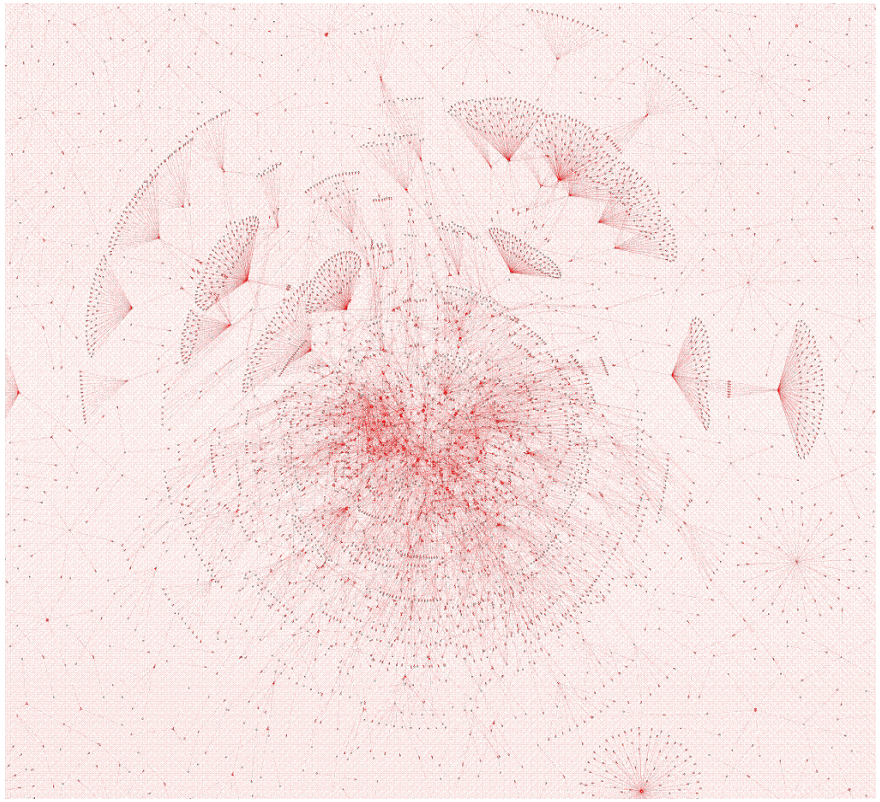


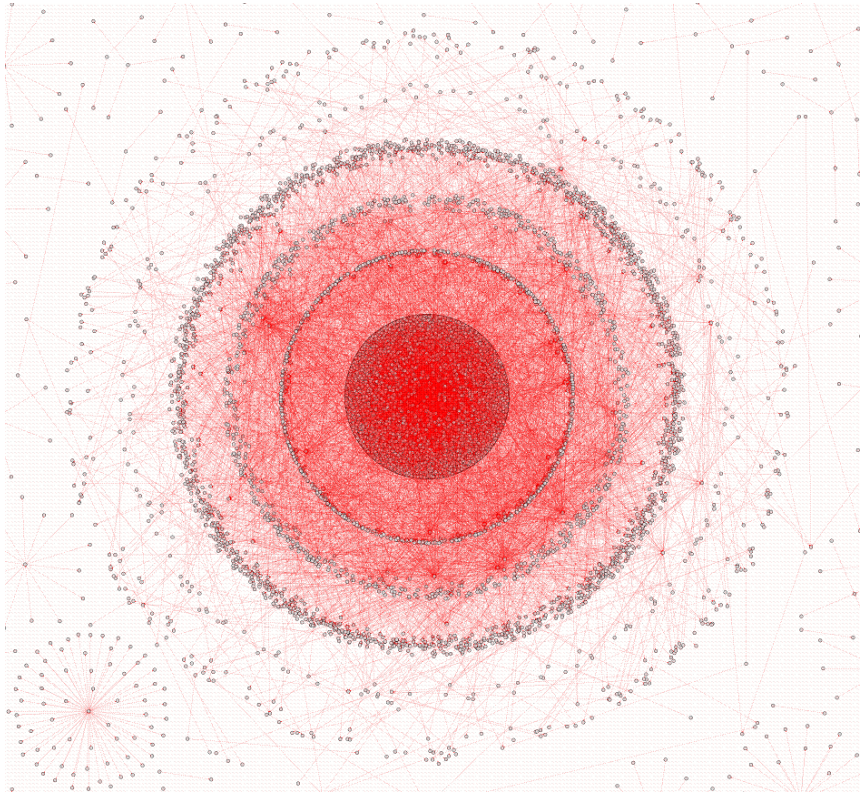
Figure 16. dK-2 distance value over time of POSTECH's trace on July 8th 2009.

Validation (POSTECH July, 2009)

❖ POSTECH traces on July, 2009



Postech Normal Trace in 2009



Postech DDoS Trace in 2009.7.9

Validation (Honeynet dataset)

❖ Real P2P botnet traffic (Peacomm) trace

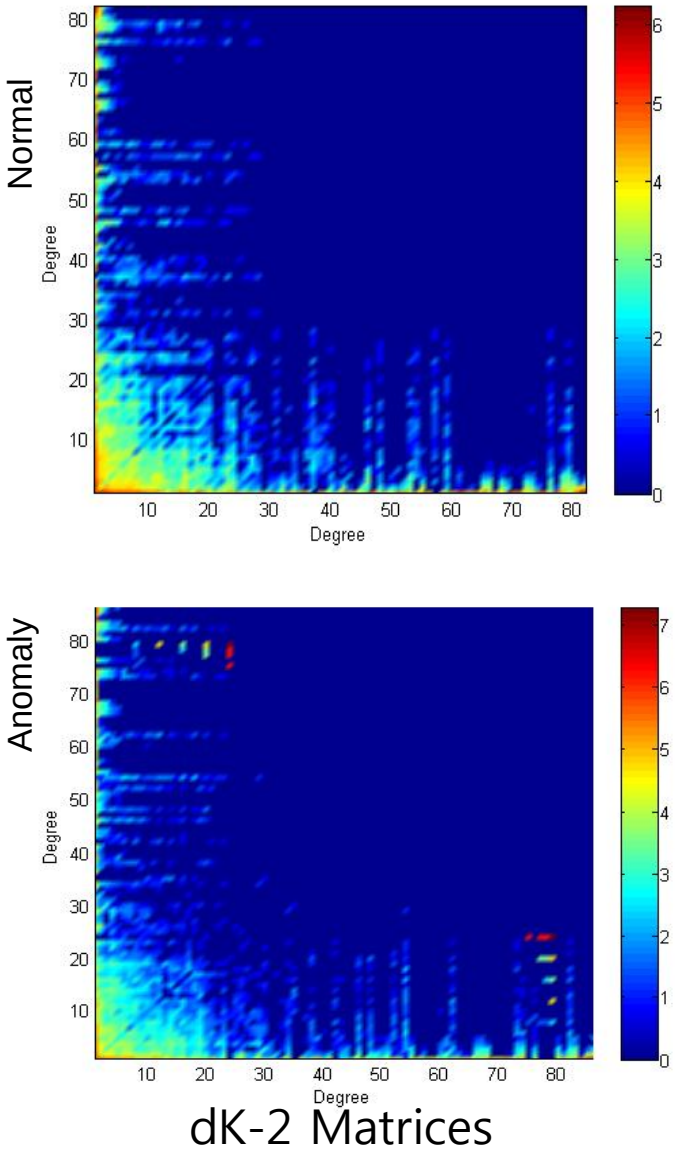
- We executed Trojan Peacomm binary files in a honeynet which consisted of 12 hosts.

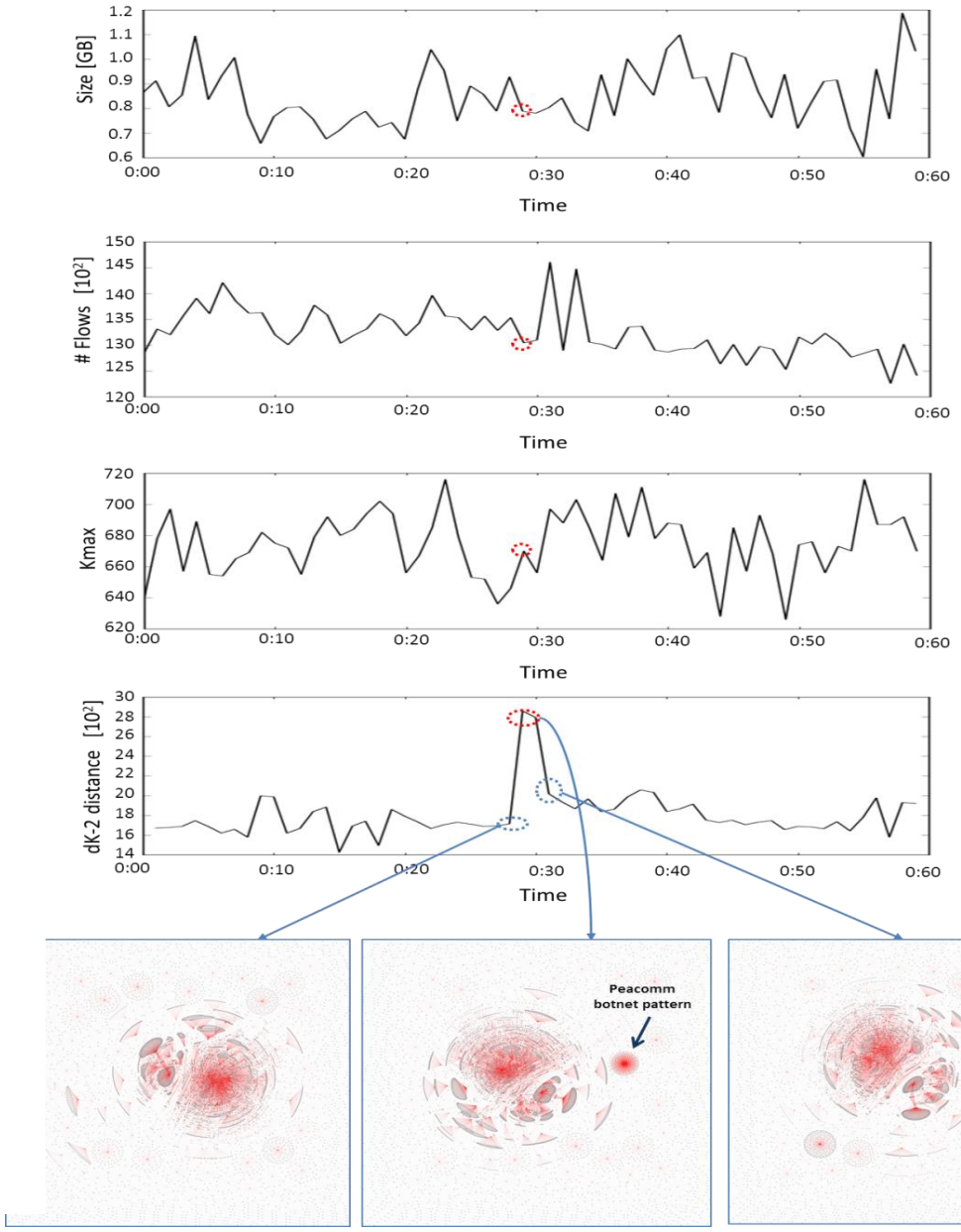
❖ Synthesized traffic dataset

- We injected P2P botnet (Peacomm) trace into normal POSTECH traffic trace.

Validation (Honeynet Dataset)

❖ Results





❖ The real-time anomaly detection system

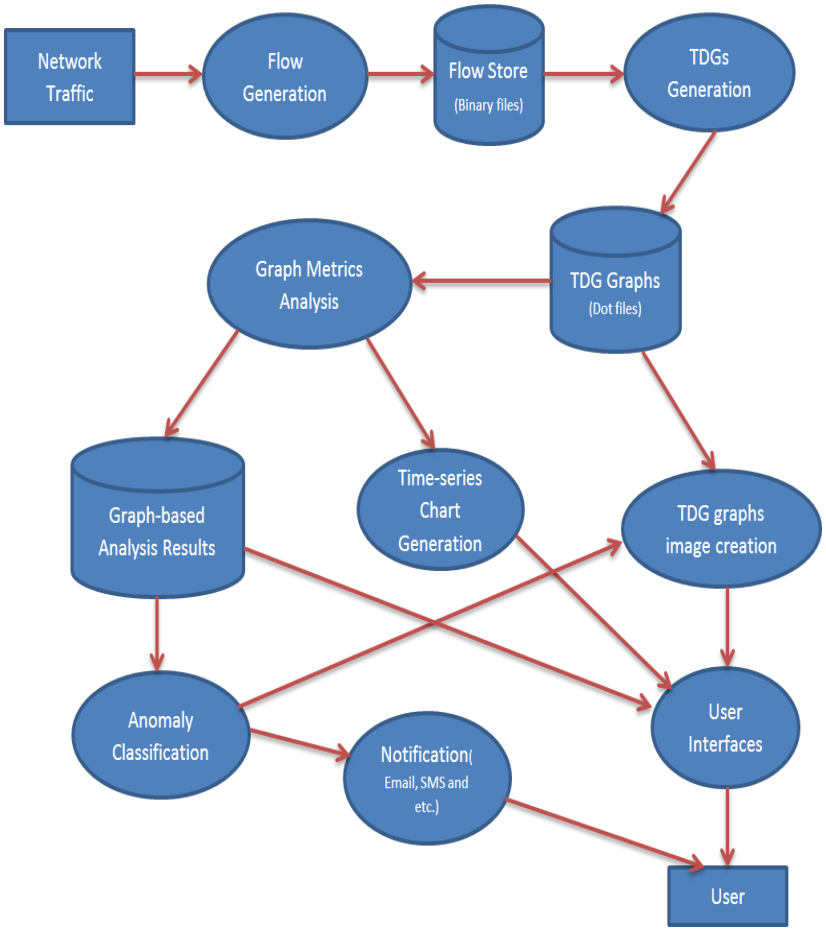
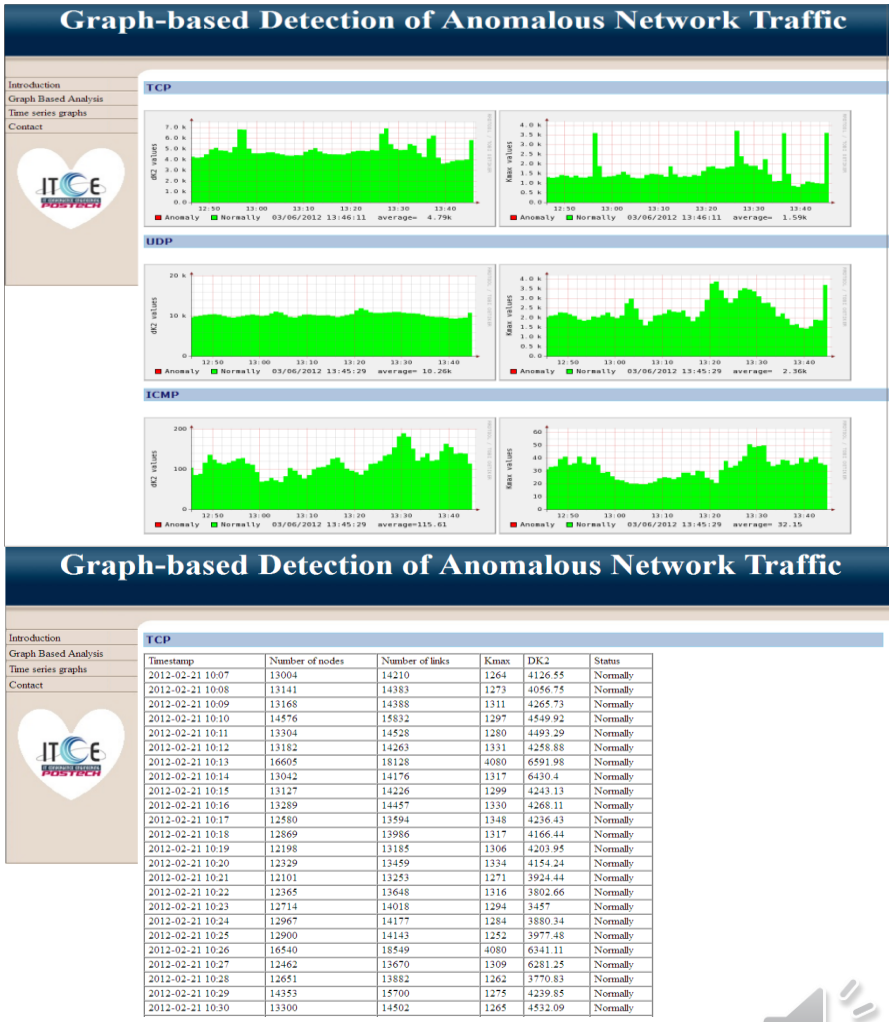


Figure 22. Real-time Anomaly Detection System: Function diagram.



Timestamp	Number of nodes	Number of links	Kmax	DK2	Status
2012-02-21 10:07	13004	14210	1264	4126.55	Normally
2012-02-21 10:08	13141	14383	1273	4056.75	Normally
2012-02-21 10:09	13168	14388	1311	4265.73	Normally
2012-02-21 10:10	14576	15832	1297	4549.92	Normally
2012-02-21 10:11	13304	14528	1280	4493.29	Normally
2012-02-21 10:12	13182	14263	1331	4258.88	Normally
2012-02-21 10:13	16605	18128	4080	6591.98	Normally
2012-02-21 10:14	13042	14176	1317	6430.4	Normally
2012-02-21 10:15	13127	14226	1299	4243.13	Normally
2012-02-21 10:16	13289	14457	1330	4268.11	Normally
2012-02-21 10:17	12580	13594	1348	4236.43	Normally
2012-02-21 10:18	12869	13986	1317	4166.44	Normally
2012-02-21 10:19	12198	13185	1306	4203.95	Normally
2012-02-21 10:20	12329	13459	1334	4154.24	Normally
2012-02-21 10:21	12101	13253	1271	3924.44	Normally
2012-02-21 10:22	12365	13648	1316	3802.66	Normally
2012-02-21 10:23	12714	14018	1294	3457	Normally
2012-02-21 10:24	12967	14177	1284	3880.34	Normally
2012-02-21 10:25	12900	14143	1252	3977.48	Normally
2012-02-21 10:26	16540	18549	4080	6341.11	Normally
2012-02-21 10:27	12462	13670	1309	6281.25	Normally
2012-02-21 10:28	12651	13882	1262	3770.83	Normally
2012-02-21 10:29	14353	15700	1275	4239.85	Normally
2012-02-21 10:30	13300	14502	1265	4532.09	Normally

Figure 23. Real-time Anomaly Detection System: User Interface.

❖ Real-time anomaly detection system testing

- We implemented a Port scanning attack from a host in the dormitory network of our campus to a host outside our campus network.
  - Using TCP Port Scanning tool to generate 100 Port scanning instances
- Result: DR = 100% and FP = 0.

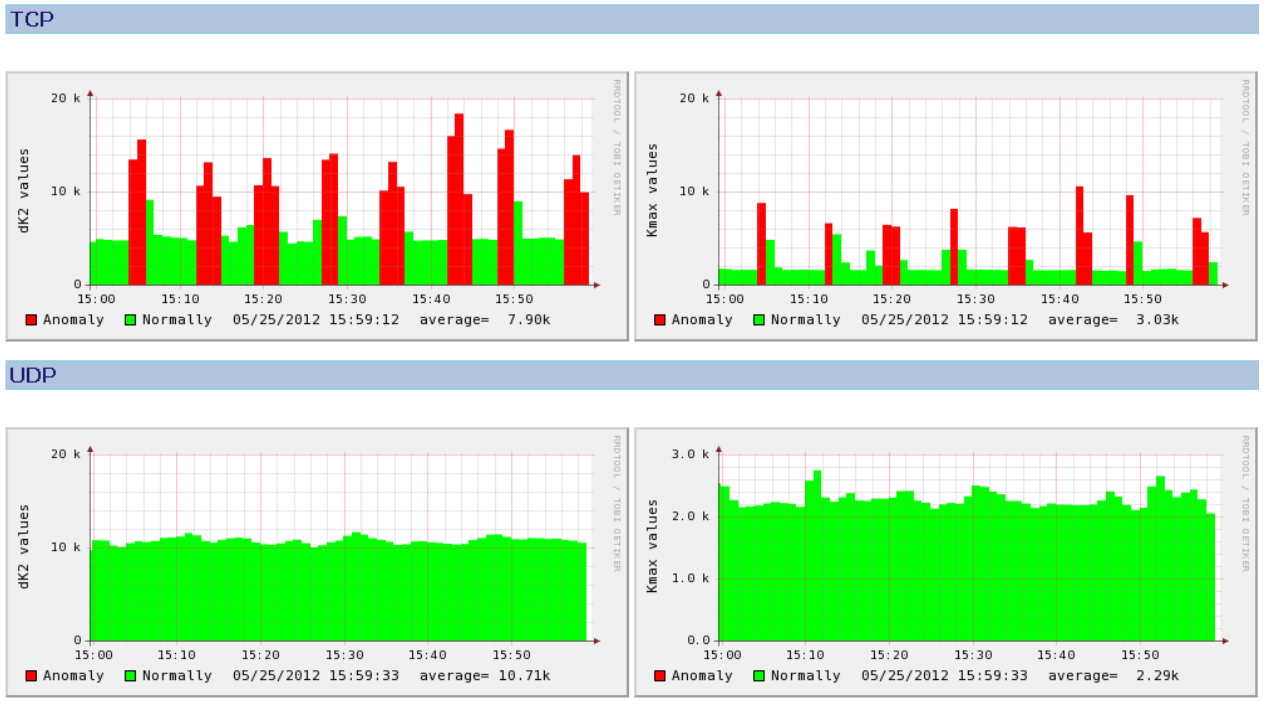


Figure 24. dK2 distance and Kmax value during TCP Port scanning attacks.

Conclusion & Future Work

❖ Conclusion

- Provide a new approach for anomaly detection.
  - Improve performance of the state of the art techniques.
- Implement a real-time anomaly detection system based on the proposed method.
- New way to analyze network traffic for anomaly detection that offers clear visualization.

❖ Future work

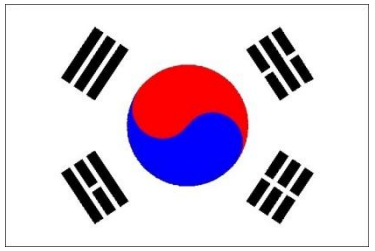
- Developing a classifier that determines the thresholds automatically and in a statistical way.
- Validating our approach with other traces.
- Using a combination of our metrics and other effective metrics to increase accuracy in terms of anomaly detection and attacks identification.

References

1. Y. Zhou, G. Hu and W. He, “Using graph to detect network traffic anomaly,” Conference on Communications Circuits and Systems, 2009.
2. A. Godiyal, M. Garland and C.H. John, “Enhancing network traffic visualization by graph pattern analysis,” 2011.
3. M. Ilifotou, P. Pappu, M. Faloutsos, M. Mitzenmacher, G. Varghese, and H. Kim, “Graption: Automated detection of P2P applications using traffic dispersion graphs (TDGs),” Tech. Rep. UCR-CS-2008-06080, Department of Computer Science and Engineering, University of California, Riverside, June 2008.
4. S. Voss and J. Subhlok, “Performance of general graph isomorphism algorithms,” Technical Report UH-CS-09-07, University of Houston, 2010.
5. J.W. Hong, “Internet traffic monitoring and analysis using NG-MON,” POSTECH, Advanced Communication Technology. The 6th International Conference, vol.1, pp. 100–120, 2004.
6. D. Whitney, “Basic Network Metrics. Lecture note,” 2008.
7. M. Iliofotou, M. Faloutsos and M. Mitzenmacher, “Exploiting dynamicity in graph-based traffic analysis: techniques and applications,” in Proceedings of the 5th international conference on Emerging networking experiments and technologies (CoNEXT '09). ACM, New York, NY, USA, 2009, pp. 241–252.
8. T.-F. Yen and M. K. Reiter, “Are your hosts trading or plotting? Telling P2P file-sharing and bots apart,” In 30th International Conference on Distributed Computing Systems, 2010.
9. D. Q. Le, T. Jeong, H. E. Roman, and J.W. Hong, “Traffic Dispersion Graph Based Anomaly Detection,” in Proc. of the Second Symposium on Information and Communication Technology (SoICT), Hanoi, Vietnam, Oct. 13-14, 2011, pp. 36–41.

Q & A

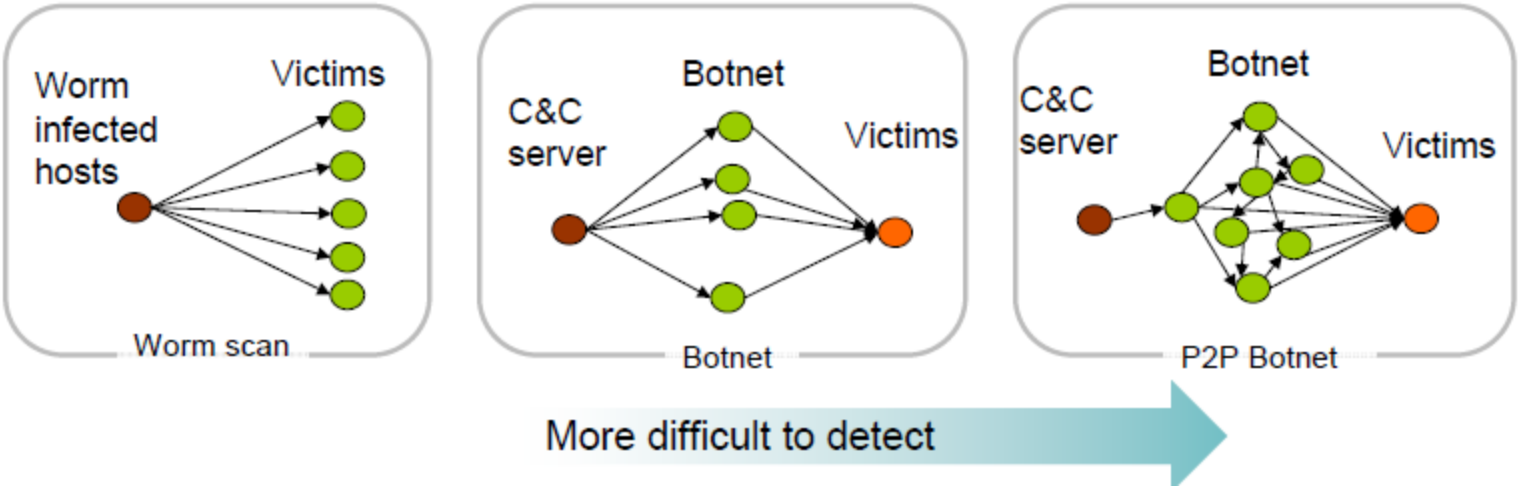
Cảm ơn



감사합니다

Communication pattern between hosts

- Can be represented as graph
- Communication graphs for anomalous traffic
  - Some of them are difficult to detect with conventional methods
    - Conventional methods: monitoring entropies in number of flows, etc



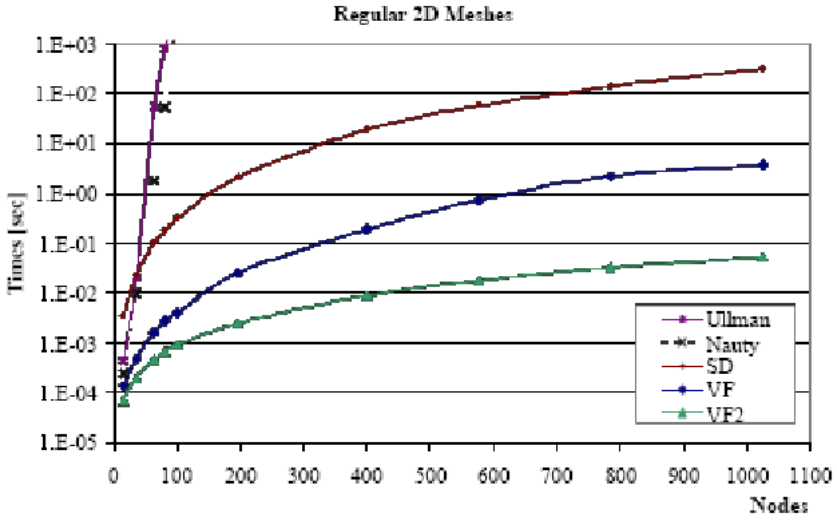
Comparison

Table 2. Performance of the Graph-based method using Kmax and dK-2 distance metric on Monday, Week5 traffic.

Method	Total instances	Attacking instances	DR	FPR	CR
Proposed method	1320	122	100 %	1.25 %	98.86 %
Wavelet-based method	1320	122	99%	56.97%	53.30%

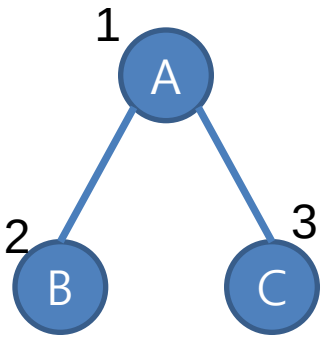
Appendix (VF2 Algorithm)

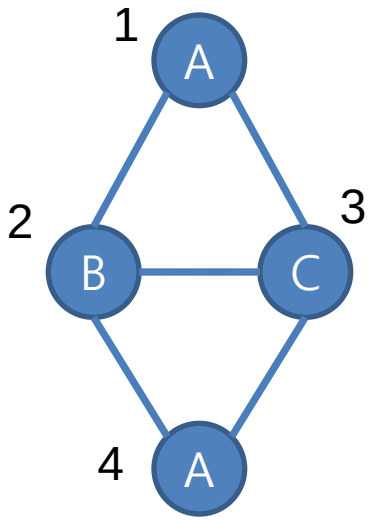
```
PROCEDURE Match(s)
    INPUT:  an intermediate state s; the initial state s0 has M(s0)=∅
    OUTPUT: the mappings between the two graphs

    IF M(s) covers all the nodes of G2 THEN
      OUTPUT M(s)
    ELSE
      Compute the set P(s) of the pairs candidate for inclusion in M(s)
      FOREACH p in P(s)
        IF the feasibility rules succeed for the inclusion of p in M(s) THEN
          Compute the state s´ obtained by adding p to M(s)
          CALL Match(s')
        END IF
      END FOREACH
      Restore data structures
    END IF
END PROCEDURE Match
```



Source: P. Figgia

Considering two graph Q and G, the (sub)graph isomorphism from Q to G is expressed as the set of pairs (n,m) (with n $\in G_1$, with m $\in G_2$)



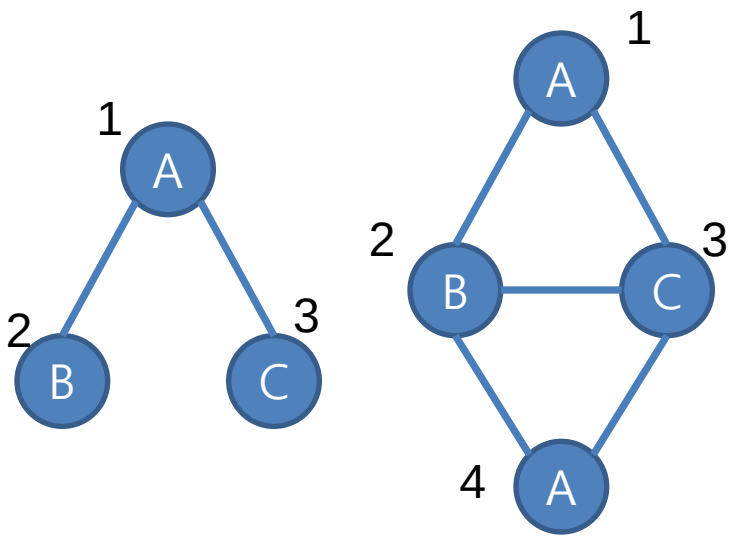


S_1	S_2
(1, 1)	(1, 4)
(2, 2)	(2, 2)
(3, 3)	(3, 3)

VF2 Algorithm

❖ **Idea:** How to find candidate pair sets for a intermediate state ?

Finding the (sub)graph isomorphism between Q and G is a sequence of state transition.



	Intermediate States
s1	(2,2)
s2	(2,2) (1,1)
s3	(2,2)(1,1)(3,3)

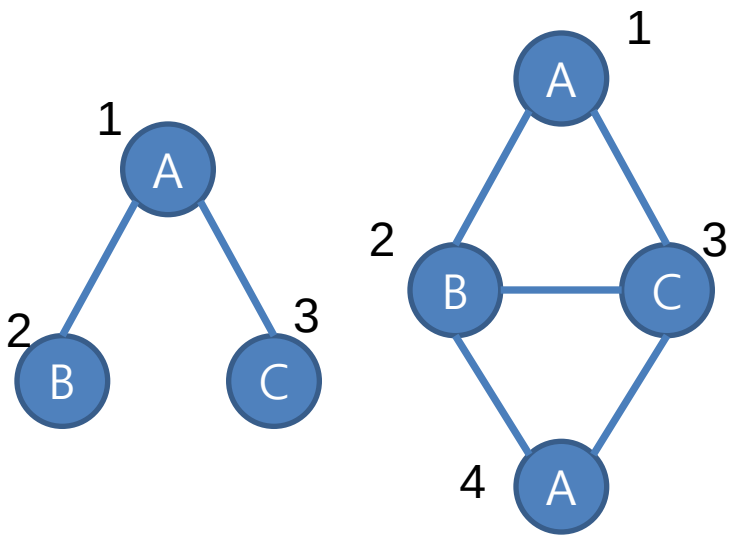
VF2 Algorithm @04

- Let s to be an intermediate state. Actually, s denotes a partial mapping from Q to G, namely, a mapping from a subgraph of Q to a subgraph of G. These two subgraphs are denoted as Q(s) and G(s).

- All neighbor vertices to Q(s) in graph Q are denoted as NQ(s), and all neighbor vertices to G(s) in graph G are denoted as NG(s). **Candidate pair sets** are a subset of NQ(s)×NG(s).

  Assume that a pair (n,m) ∈ NQ(s)×NG(s).

VF2 Algorithm



Candidate Pair Sets

(2, 2)

(1, 1) (1, 4)

(3, 3) (3,3)

VF2 Algorithm

```
PROCEDURE Match(s)
    INPUT:  an intermediate state s; the initial state s0 has M(s0)=∅
    OUTPUT: the mappings between the two graphs

    IF M(s) covers all the nodes of G2 THEN
      OUTPUT M(s)
    ELSE
      Compute the set P(s) of the pairs candidate for inclusion in M(s)
      FOREACH p in P(s)
        IF the feasibility rules succeed for the inclusion of p in M(s) THEN
          Compute the state s´ obtained by adding p to M(s)
          CALL Match(s')
        END IF
      END FOREACH
      Restore data structures
    END IF
END PROCEDURE Match
```

Drawing TDG

❖ Drawing Network Traffic Graph ?

```
18 Mar 06 17:06:39 18 Mar 06 17:06:39 udp 203.78.162.181.123 -> 203.78.171.149.123 1 0 76 0 UNK
18 Mar 06 17:06:39 18 Mar 06 17:06:39 udp 244.35.221.199.4710 -> 203.78.168.48.1026 1 0 553 0 UNK
18 Mar 06 17:06:39 18 Mar 06 17:06:39 udp 244.35.221.199.4967 -> 203.78.168.48.1027 1 0 553 0 UNK
18 Mar 06 17:06:39 18 Mar 06 17:06:49 udp 214.173.213.249.7001 -> 203.78.25.52.7003 8 0 612 0 UNK
18 Mar 06 17:06:39 18 Mar 06 17:06:49 udp 214.173.74.211.7001 -> 203.78.25.52.7003 8 0 612 0 UNK
18 Mar 06 17:06:39 18 Mar 06 17:06:48 udp 203.78.233.44.63775 -> 121.0.0.3.427 4 0 347 0 UNK
```

Generate

```
digraph SampleNetflowGraph {
    "203.78.162.181.123" -> "203.78.171.149.123";
    "244.35.221.199.4710" -> "203.78.168.48.1026";
    "244.35.221.199.4967" -> "203.78.168.48.1027";
    "214.173.213.249.7001" -> "203.78.25.52.7003";
    "214.173.74.211.7001" -> "203.78.25.52.7003";
    "203.78.233.44.63775" -> "121.0.0.3.427";
}
```

Visualize

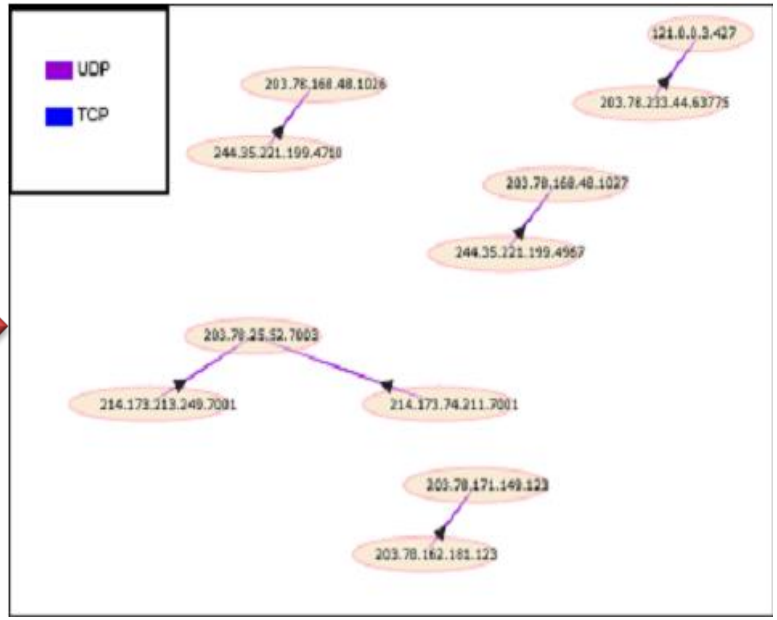
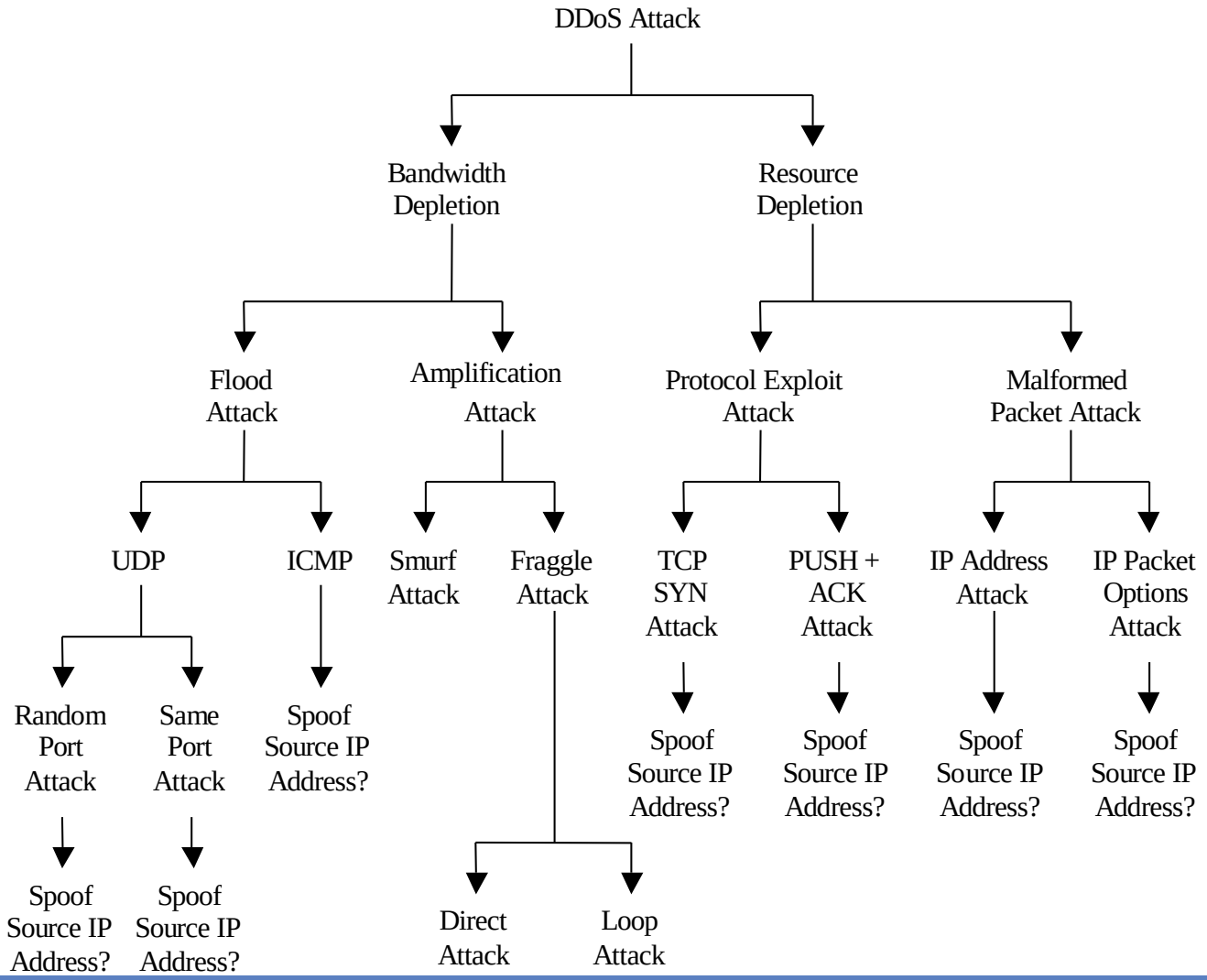


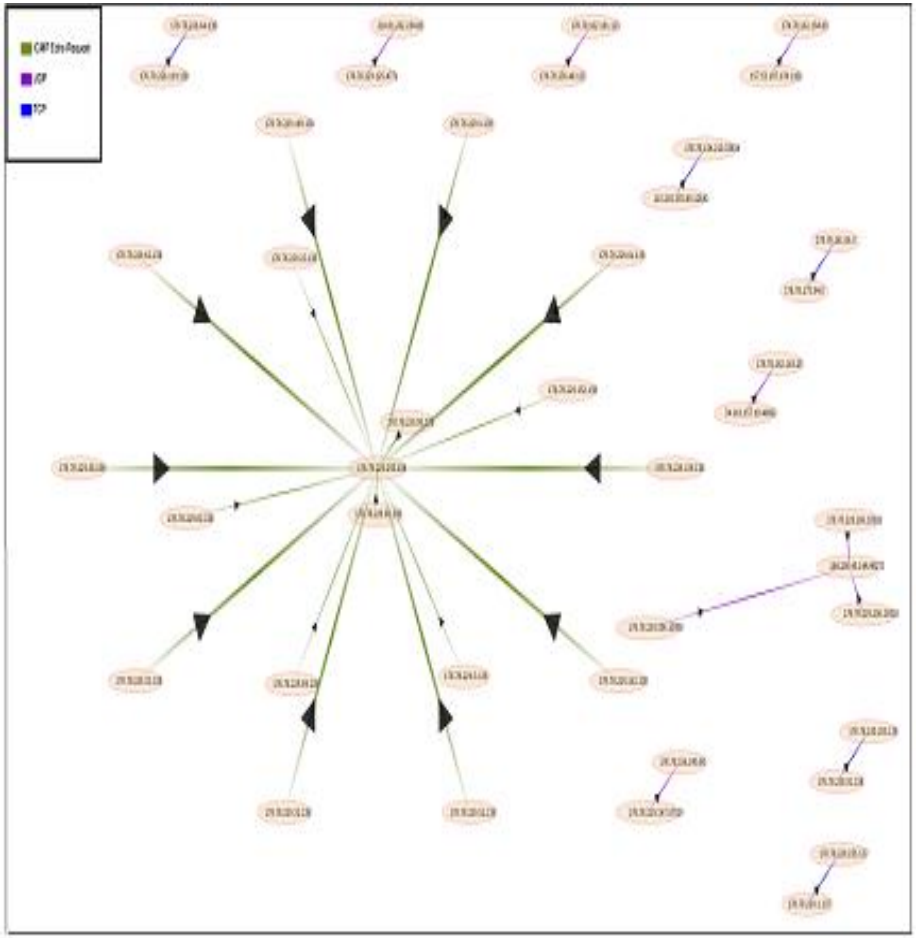
Figure 4: DDoS Attack Taxonomy



Attack Templates

❖ Pattern Specification

```
digraph DDosPattern {
        victim [pos="6.681, 3.708 "];
        attacker0 [pos="0.736, 3.708 "];    attacker1[pos="3.389, 2.514 "];
        attacker2 [pos="4.931, 0.667 "];    attacker3[pos="7.292, 0.264 "];
        attacker4 [pos="9.361, 1.458 "];    attacker5 [pos="10.181, 3.708 "];
        attacker6 [pos="9.361, 5.958 "];    attacker7 [pos=" 7.292, 7.153 "];
        attacker8 [pos=" 4.931, 6.750 "];   attacker9 [pos=" 3.389, 4.903 "];

        attacker0 -> victim [time=0, protocol="ICMP", info="Echo Request"];
        attacker1 -> victim [time=0, protocol="ICMP", info="Echo Request"];
        attacker2 -> victim [time=0, protocol="ICMP", info="Echo Request"];
        attacker3 -> victim [time=0, protocol="ICMP", info="Echo Request"];
        attacker4 -> victim [time=0, protocol="ICMP", info="Echo Request"];
        attacker5 -> victim [time=1, protocol="ICMP", info="Echo Request"];
        attacker6 -> victim [time=1, protocol="ICMP", info="Echo Request"];
        attacker7 -> victim [time=1, protocol="ICMP", info="Echo Request"];
        attacker8 -> victim [time=1, protocol="ICMP", info="Echo Request"];
        attacker9 -> victim [time=1, protocol="ICMP", info="Echo Request"];
}
```



DDoS Pattern

Attack Templates (1/3)

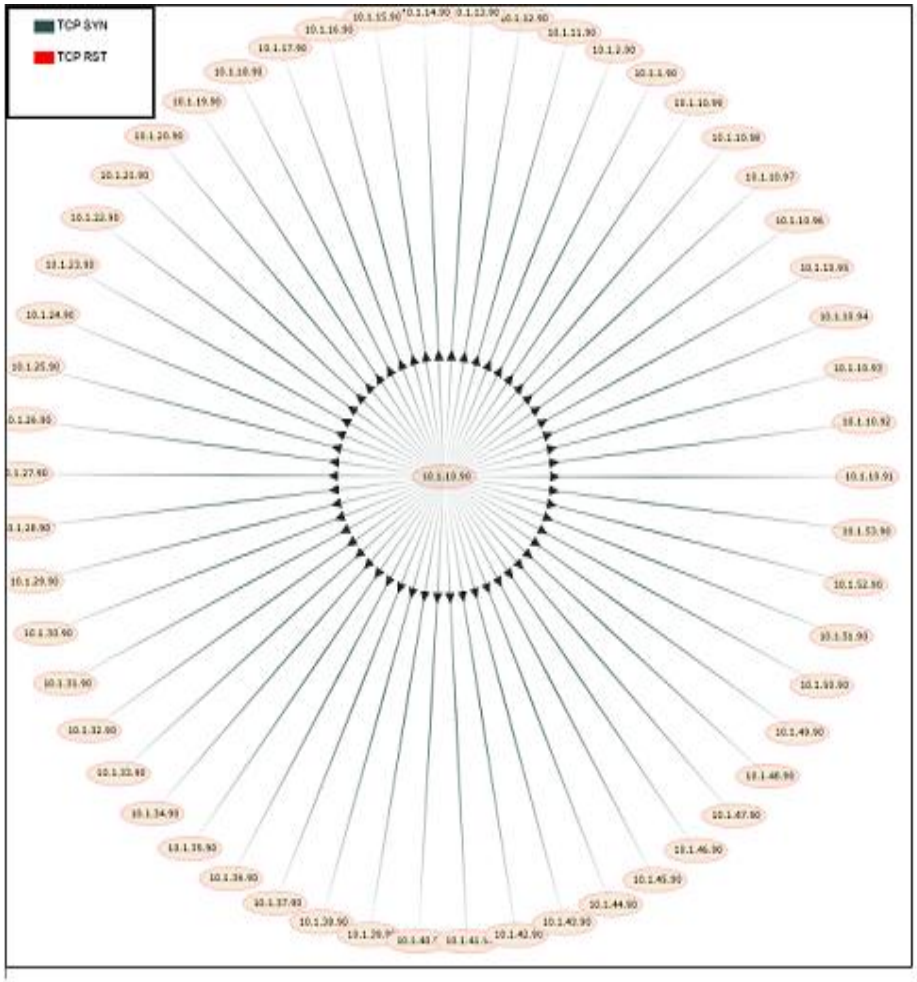


Figure 3: Host Scanning

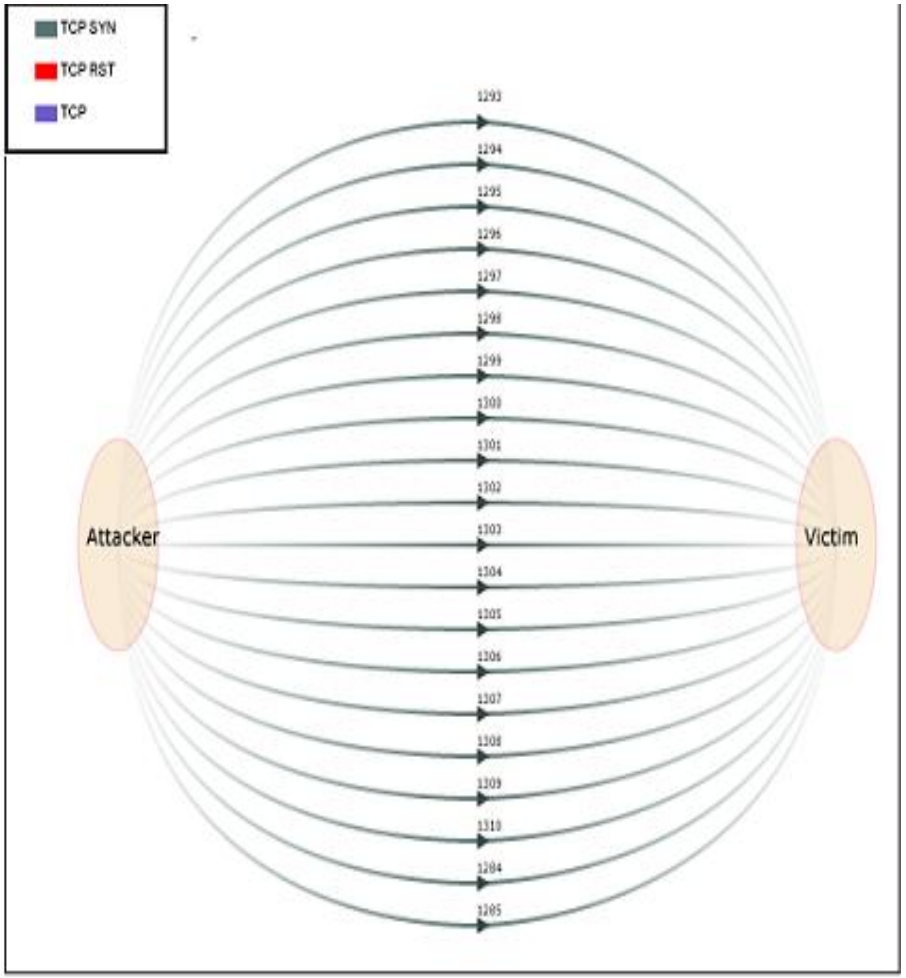


Figure 4: Port Scanning

Attack Templates (2/3)

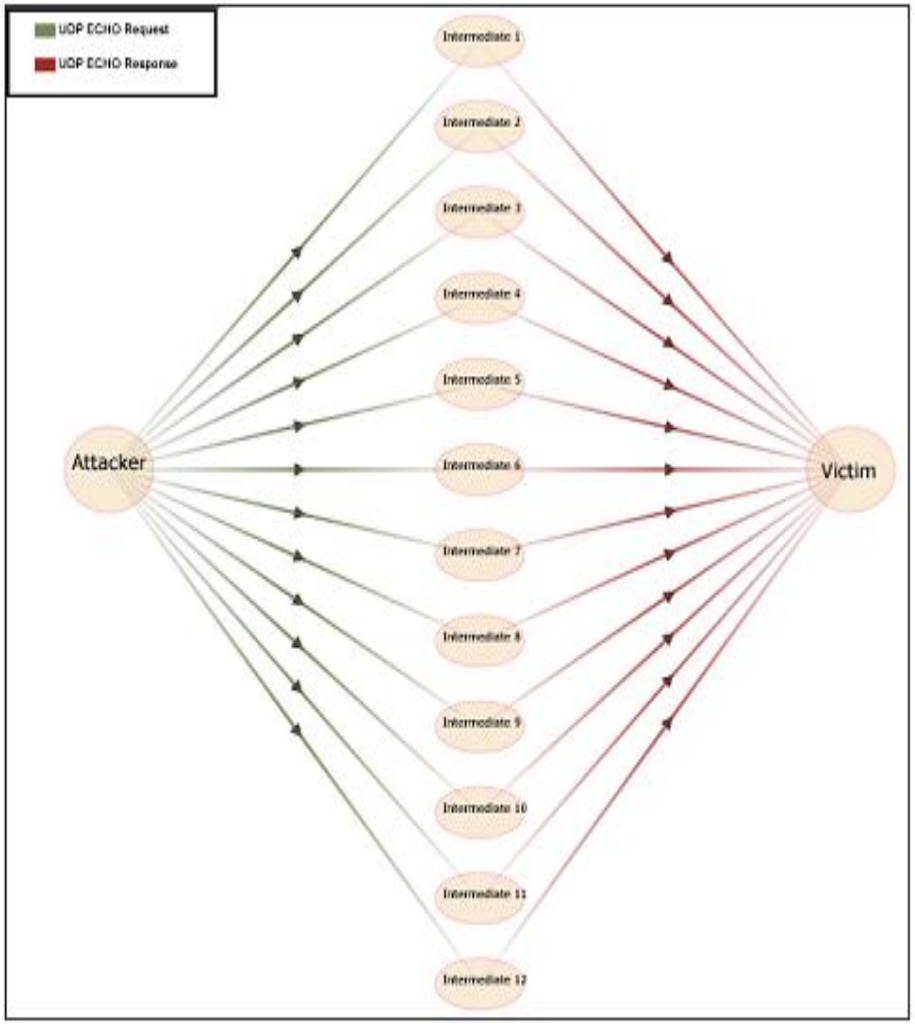


Figure 5: Smurf Attack

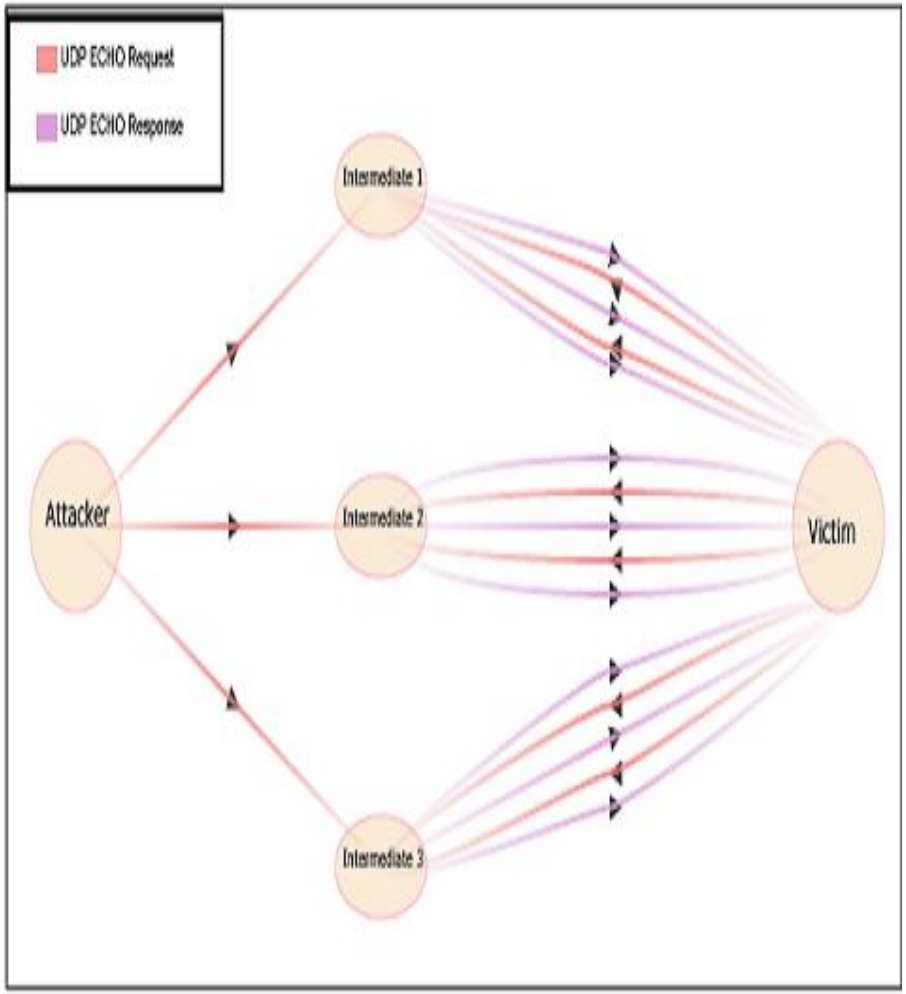


Figure 6: Fraggle Attack

Attack Templates (3/3)

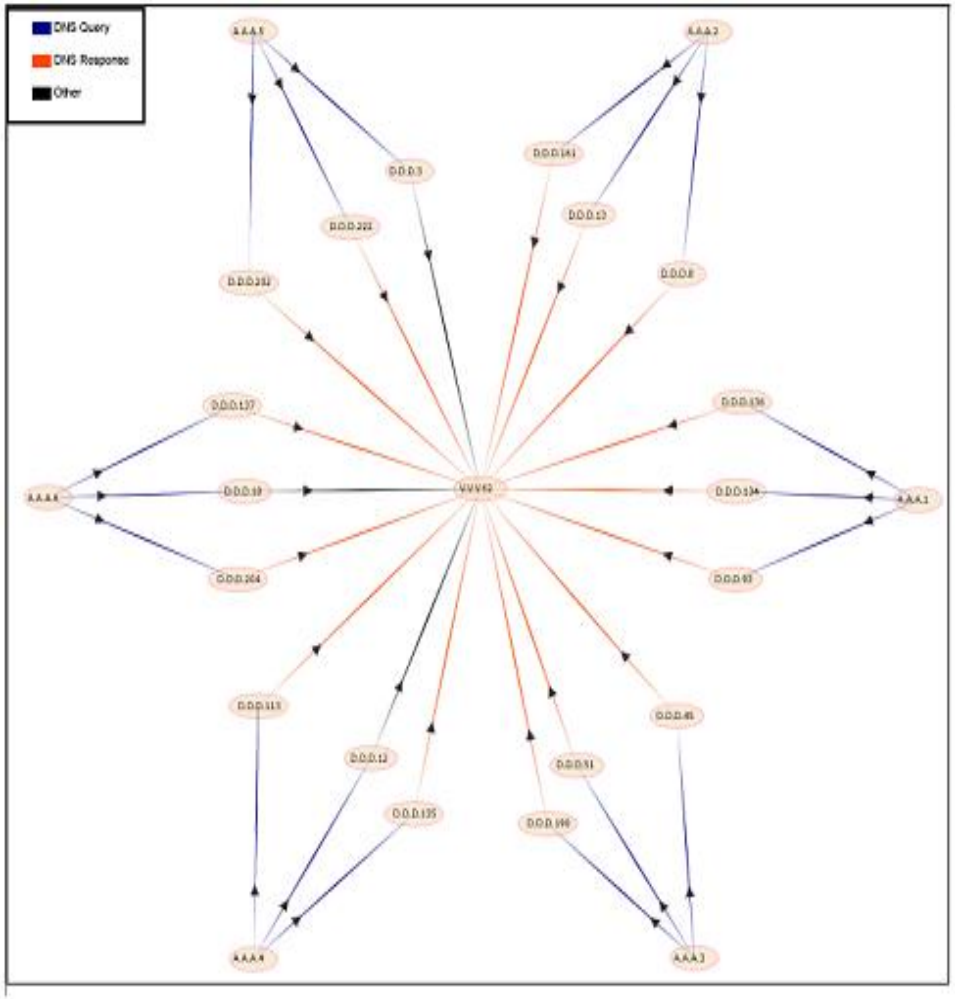


Figure 7: DNS Amplification Attack

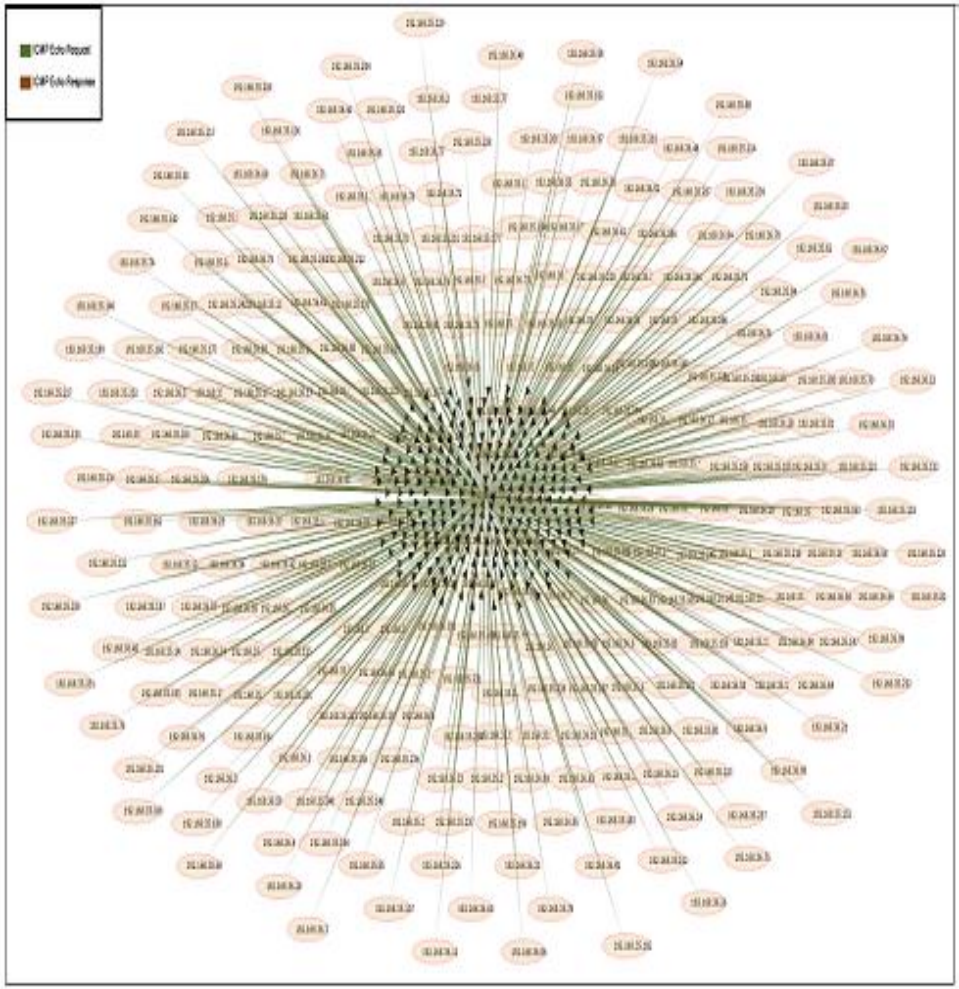
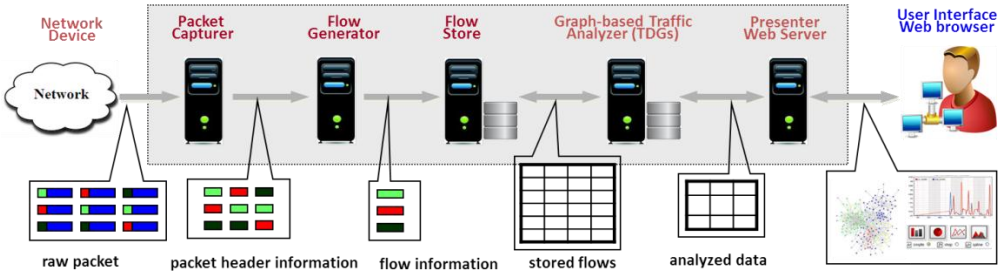
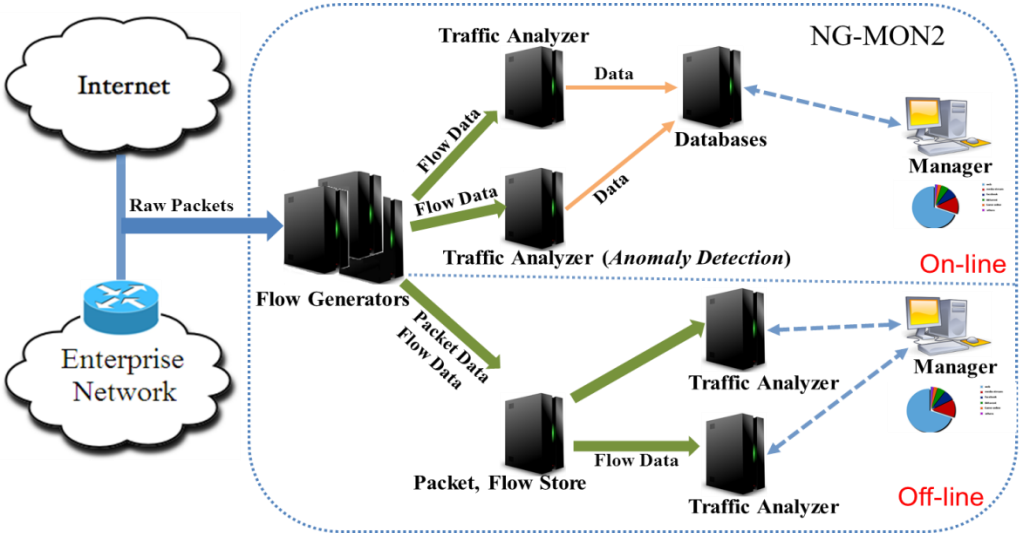


Figure 8: ICMP Flood Attack

Thresholds of POSTECH network

- **TCP**
  - Kmax: 5525
  - dK-2 distance: 11328
- **UDP**
  - Kmax: 15327
  - dK-2 distance: 23608
- **ICMP**
  - Kmax: 1425
  - dK2: 2996


Network Device
Packet Capturer
Flow Generator
Flow Store
Graph-based Traffic Analyzer (TDGs)
Presenter Web Server
User Interface Web browser
Network
raw packet
packet header information
flow information
stored flows
analyzed data


Internet
Raw Packets
Enterprise Network
Flow Generators
Traffic Analyzer
Data
Databases
Flow Data
Flow Data
Data
Traffic Analyzer (Anomaly Detection)
NG-MON2
Manager
On-line
Packet Data
Flow Data
Traffic Analyzer
Manager
Flow Data
Packet, Flow Store
Traffic Analyzer
Off-line

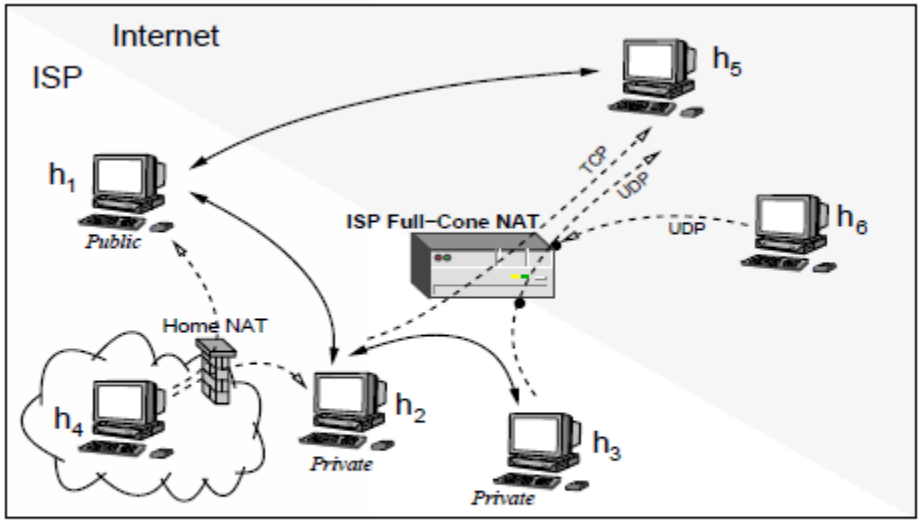


Figure 3: Schematic representation of possible connections from hosts in the MiniPop versus other hosts

Validation (DARPA dataset)

❖ DARPA 1999 Dataset

- Week 1 and week 3: no attack (for training data).
- Week 2: 43 attacks belonging to 18 labeled attack types are used for system development.
- Week 4 and week 5: 201 attacks belonging to 58 attack types (including 40 new attacks).

❖ The traffic data on Monday, Week 5 of DARPA Dataset

- Including 122 attack instances.
- Attacks that change communication structure in network graph:
  - Smurf, apache2, udpstorm, portsweep and etc.

Validation

- **We use standard measurements such as detection rate (DR), false positive rate (FPR) and overall classification rates (CR) to evaluate our approach.**
  - True Positive (TP): The number of anomalous instances that are correctly identified.
  - True Negative (TN): The number of legitimate instances that are correctly classified.
  - False Positive (FP): The number of instances that were incorrectly identified as anomalies, however in fact they are legitimate activities.
  - False Negative (FN): The number of instances that were incorrectly classified as legitimate activities however in fact they are anomalous.
  - DR = TP / (TP + FN)
  - FPR = FP / (TN + FP)
  - CR = (TP + TN) / (TP + TN + FP + FN)

Peacomm

- **Connect to Overnet**
  - The bot publishes itself on the Overnet network and connects to peers. The initial list of peers is hard coded in the bot.
- **Download Secondary Injection URL**
  - The bot uses hard coded keys to search for and download a value on the Overnet network. The value is an encrypted URL that points to the location of a secondary injection executable.
- **Decrypt Secondary Injection URL**
  - The bot uses a hard coded key to decrypt the downloaded value, which is a URL.
- **Download Secondary Injection**
  - The bot downloads the secondary injection from a web server using the decrypted URL.
- **Execute Secondary Injection**
  - The bot executes the secondary injection, possibly scheduling future upgrades on the peer-to-peer network or scheduling bot stat tracking at some other resource.

http://static.usenix.org/event/hotbots07/tech/full_papers/grizzard/grizzard_html/

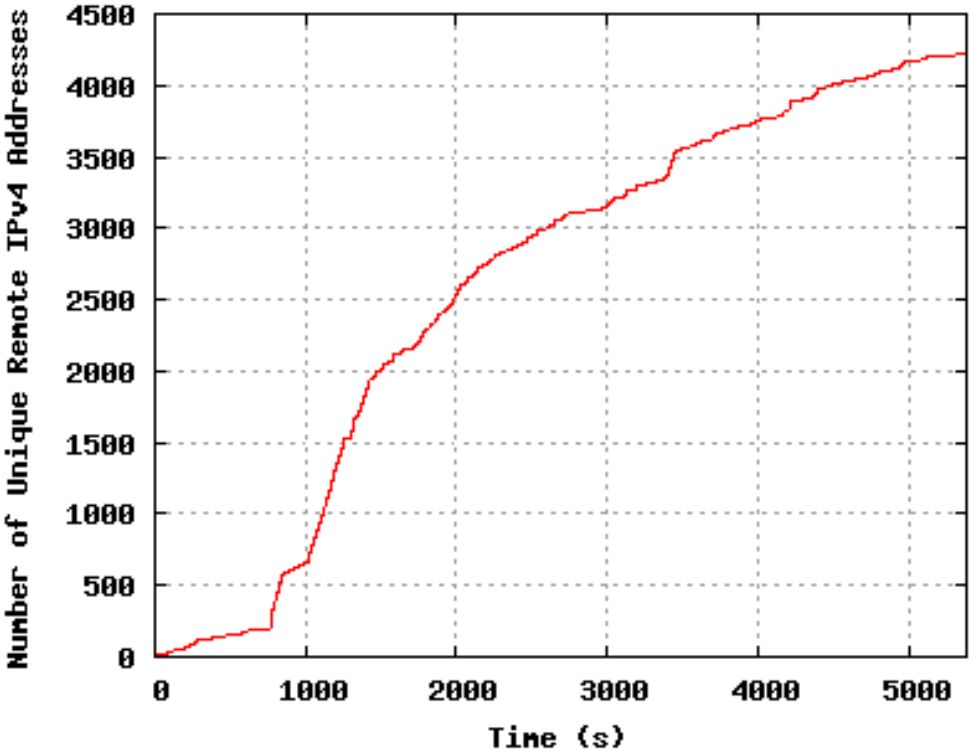


Figure 2: Number of Remote IPv4 Addresses Contacted Over Time for Duration of Infection

Inbox (2) - lequocdo@gmail.com - Gmail - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Restore Sess... | Inbox (2)... | Tổng quan ... | TOP CAREE... | Phys. Rev. E... | APS -APS M... | A SOCIAL N... | The Securit... | An Effective... | Jour

mail.google.com/mail/?shva=1#inbox

p2psim: a simulator fo... | F-Secure Weblog : Mo... | Tatsuya MORI: recent ... | Downloading - BotNet... | CIS 700/003 - Distribut... | Variant of Zeusbot/Sp...

+You Search Images Maps Play YouTube News Gmail Documents Calendar More

Google

Gmail

More

COMPOSE

Undergraduate Prep Canada - www.LSBF.ca/HND - HND - Your Pathway to Undergraduate Programs Around the World. Apply!

Inbox (2)
Important
Sent Mail
Drafts (32)
Spam
[Imap]/Drafts
DPNM

Chat

Search people...

Le Quoc Do
Carol Fung
Duc M. Le
Le Thanh Thao
Ngoc Viet

Unread

lequocdo (39)	**Anomaly Detection System 2012**	The system was being attacked by a handsome gu
lequocdo (61)	**Anomaly Detection System 2012**	The system was being attacked by a handsome gu

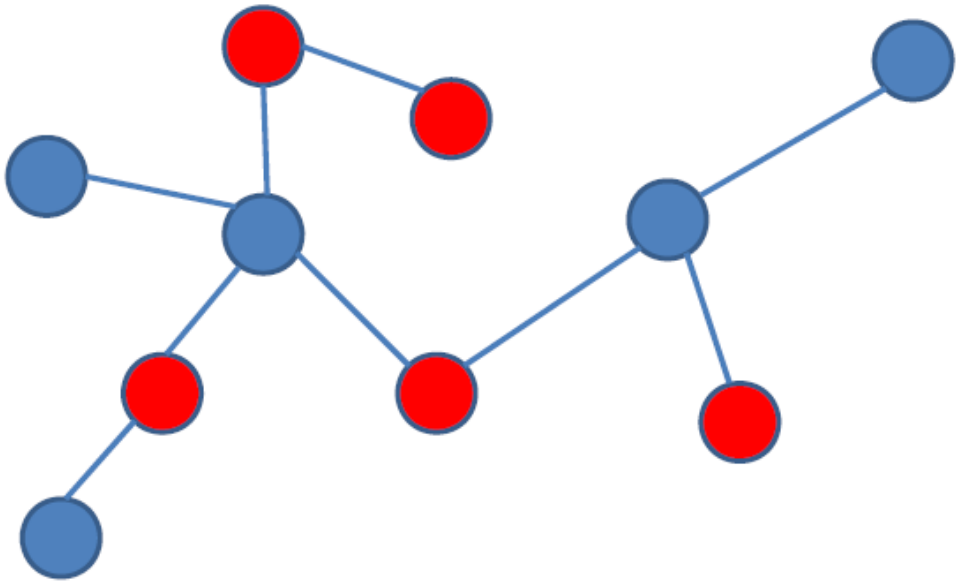
Everything else

김태영	Postech	[POSTECH Graduate Admissions and Student Affairs] Are you interested in t
Academia.edu	We found a paper that might be yours	Academia.edu Hi Do, We searched the web and
Eduardo, me (5)	Lecture tomorrow	Hi Do, this is a preliminary material for you, maybe u find it useful. S
LeThi Thanh Huyen	Fwd:[TOP CAREER] Job offer from a leading Japanese company!	Original Message Fr
서신석	Tonight, Tomorrow morining..	Dear all Let's go for a drinking party tonight for celebrating
YoonSeon, Han	오전에 조금 늦게 올라가겠습니다.	안녕하세요 한윤선입니다. 머리가 아파서 약 좀 먹고
정태열	조교 다녀오겠습니다	수업 조교 다녀오겠습니다 감사합니다
Dongwoo Kwon	은행 갔다가 가겠습니다.	오전에 기업은행에 일이 있어서 처리하고 가겠습니다. /동우
Lucidchart	Introducing custom shape libraries, an interactive Confluence viewer and more	Lucidch
IEEE Spectrum Tech Alert	Bioengineers Make a Flash Drive from DNA	To view this e-mail as a web page, go here
Jian Li	오전에 병원 다녀 오겠습니다.	내일(금요일) 오전에 A/S 받은 치아 고정 일정이 잡혀있

Find: correlation Next Previous Highlight all Match case

Assortativity vs Disassortativity

Assume nodes have some intrinsic property which separates them into different classes



Node type A

Node type B

$N = 10 \ (N_A = N_B = 5)$

$L = 9$

Construct assortativity matrix $\varepsilon_{x,y}$: In this case x={A,B}, y={A,B} representing the fraction of links between nodes of type (x,y), with the condition $\sum_{x,y} \varepsilon_{x,y} = 1$.

$$\varepsilon = \begin{pmatrix} 4/9 & 6/9 \\ 6/9 & 2/9 \end{pmatrix} \times \tfrac{1}{2} = \begin{pmatrix} 2/9 & 3/9 \\ 3/9 & 1/9 \end{pmatrix}$$

For undirected graphs count twice and divide by 2

Coefficient of assortativity

$\Delta\varepsilon$ = sum diagonal terms – sum off-diagonal terms

Example: $\Delta\varepsilon=(2/9+1/9) - (3/9+3/9) = -1/3$  What does it mean?

Consider a case in which links occur completely at random between A and B:
What is $\Delta\varepsilon$ in this case?

(Undirected graphs)

Examples: $\varepsilon = \begin{pmatrix} a & b \\ b & a \end{pmatrix}$ Then $\Delta\varepsilon = (2a) - (2b)$ (Norm: $2a+2b = 1$)

(For $a=b$ then $\Delta\varepsilon=0$)

In general: $\varepsilon = \begin{pmatrix} a & b \\ b & c \end{pmatrix}$ Then $\Delta\varepsilon = (a+c) - (2b)$ (Norm: $a+c+2b=1$)

Coefficient of assortativity

$\Delta\varepsilon$ = sum diagonal terms – sum off-diagonal terms

$\Delta\varepsilon > 0$ ⟹ Like nodes tend to be more likely linked

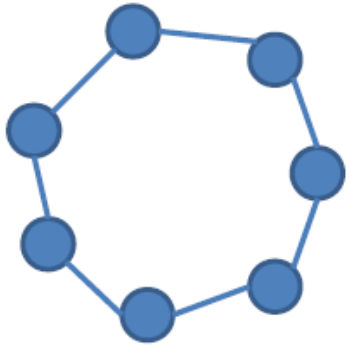
$\Delta\varepsilon < 0$ ⟹ Unlike nodes tend to be more likely linked

Values of assortativity different than zero mean possible presence of correlations (>0) or anticorrelations (<0) in the Network.

Nodes can be classified according to their degree

Examples for node-node degree assortativity

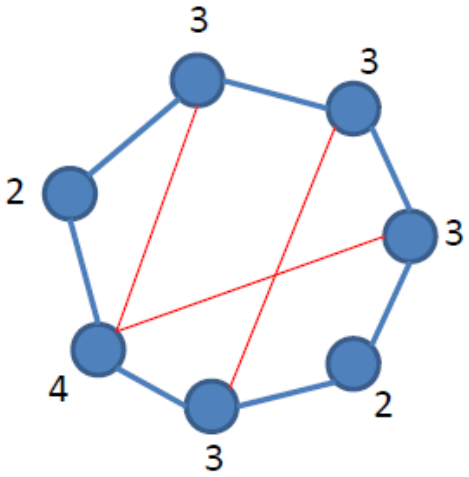
Rings:



N = 7 L = 7

$\Delta\eta = 1$ because $\eta_{2,2} = 1$ and zero otherwise

What is the effect of shortcuts:



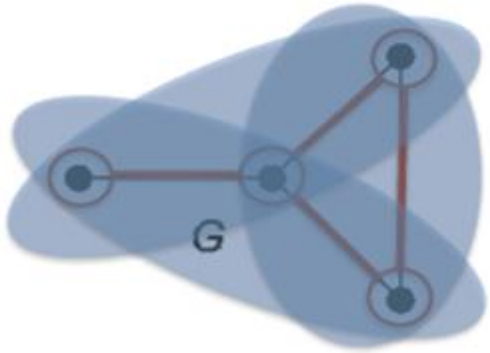
$$\eta = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 3/10 & 1/10 \\ 0 & 0 & 3/10 & 3/10 \\ 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

$\Delta\eta = (3/10) - (4/10+3/10) = -4/10 = -0.4$

Graph Metrics on TDGs – dK-2 distance

❖ Structure analysis - dK-n series: n=1,2,3,…

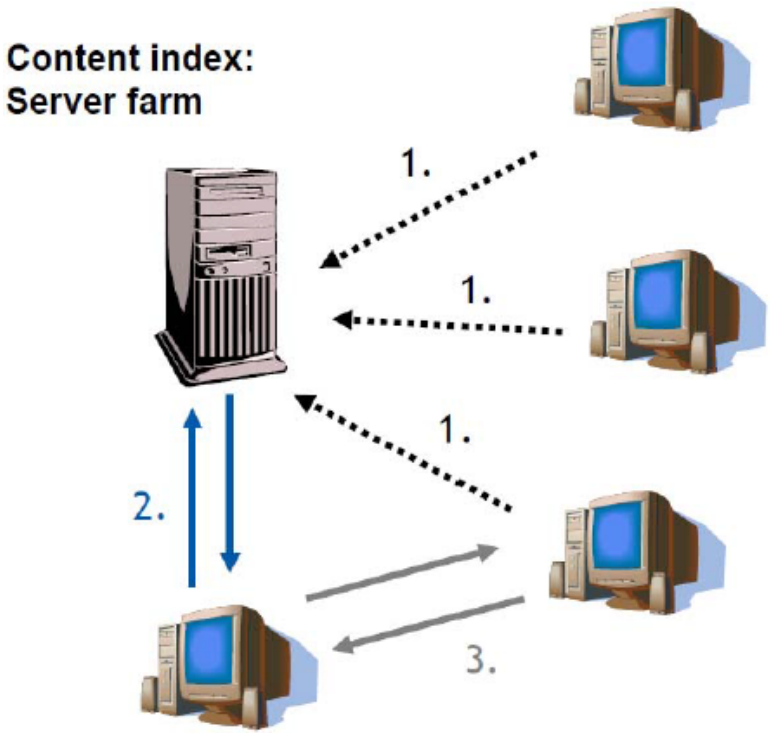
- Look at inter-dependencies among topology characteristics
- dK-n series are degree correlations within simple connected graphs of size n
  - Some examples:
    - dK-1 describes node degree distribution
    - dK-2 describes joint node degree distribution
    - dK-3 captures clustering coefficient



dk-0: average degree=2
dk-1: P(1)=1, P(2)=2, P(3)=1
dk-2: P(1,3)=1, P(2,2)=1, P(2,3)=2
dk-3: P(1,3,2)=2, P(2,2,3)=1,

Source: Ben Zhao (June 22, 2011)

Napster - Queries

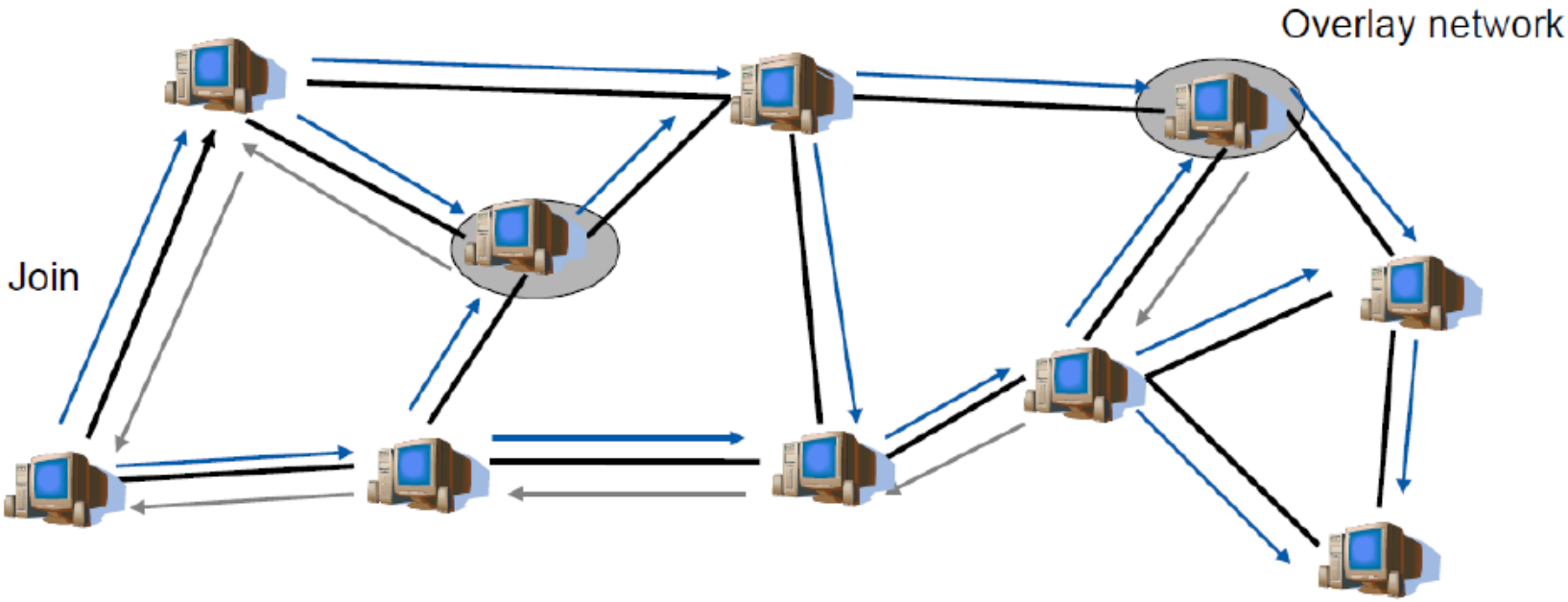


Source: Th. Strufe, VL Peer-to-Peer Networks, 2011

1. Peers register with central server, send list
2. Peers send queries to central server which has content index of all files
3. File transfers happen directly between peers

Last point is common to all P2P file sharing networks and is their main strength as it allows them to scale well!

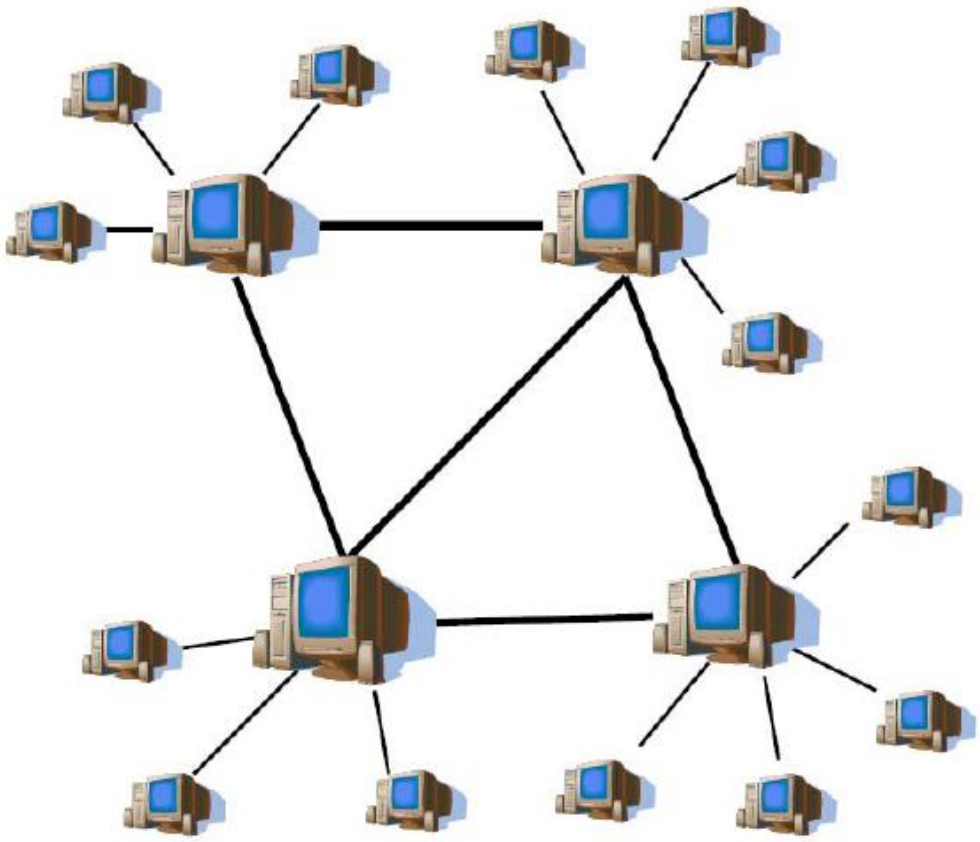
Gnutella – How it Works



Source: Th. Strufe, VL Peer-to-Peer Networks, 2011

- To join, peer needs address of one member, learn others
- Queries are sent to neighbors
- Neighbors forward queries to their neighbors (flooding)
- Replies routed back via query path to querying peer

KaZaA –Hierarchy



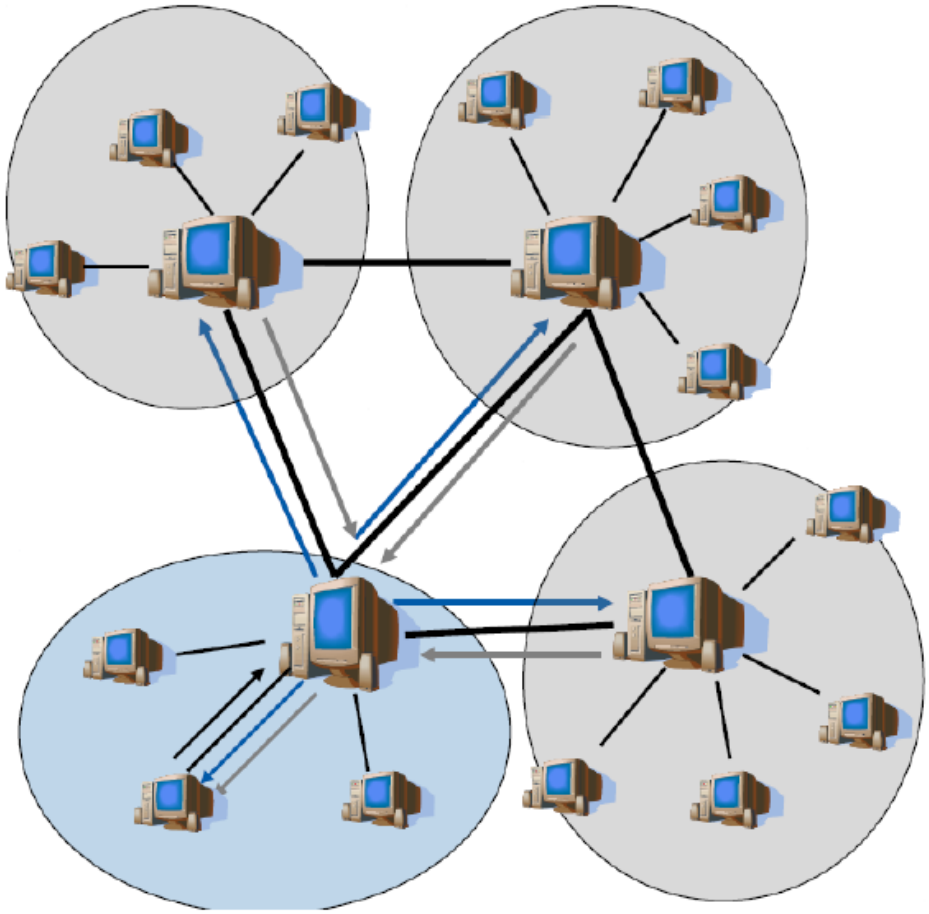
Source: Th. Strufe, VL Peer-to-Peer Networks, 2011

Ordinary Node

SuperNode

- Ordinary nodes belong to one Supernode
  - Can change SN, but not common
- Supernodes exchange information between themselves
  - Supernodes do not form a complete mesh

KaZaA – Finding Stuff



Source: Th. Strufe, VL Peer-to-Peer Networks, 2011

1. Peer sends query to its own supernode
2. Supernode answers for all of its peers and forwards query to other supernodes
3. Other supernodes reply for all of their peers

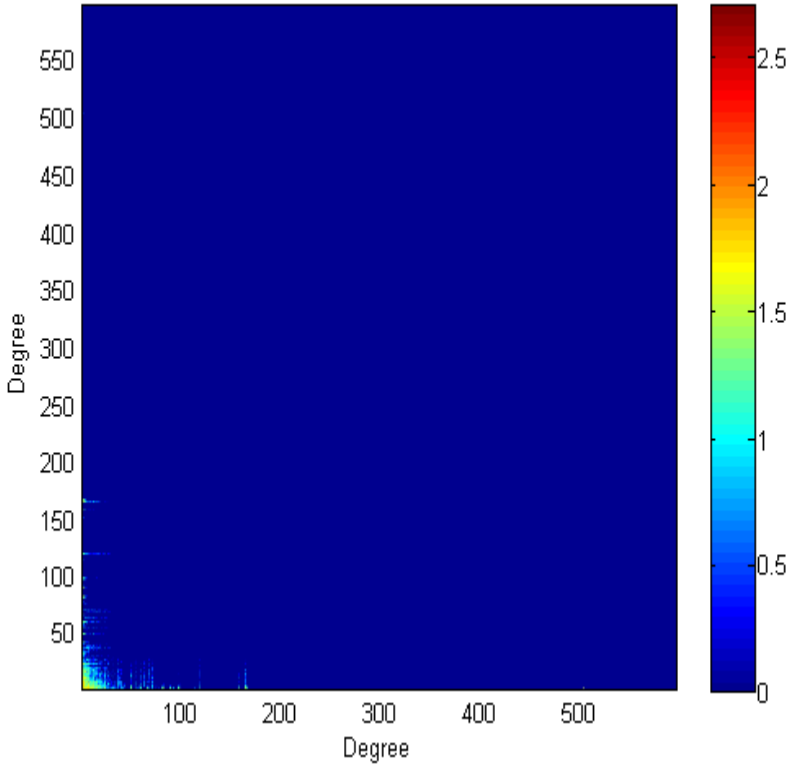
DHT: Motivation

- Why we need DHTs?
- Searching in P2P networks is not efficient
  - Either centralized system with all its problems
  - Or distributed system with all its problems
  - Hybrid systems cannot guarantee discovery either
- Actual file transfer process in P2P network is scalable
  - File transfers directly between peers
- Searching does not scale in same way
- Original motivation for DHTs: More efficient searching and object location in P2P networks
- Put another way: Use addressing instead of searching

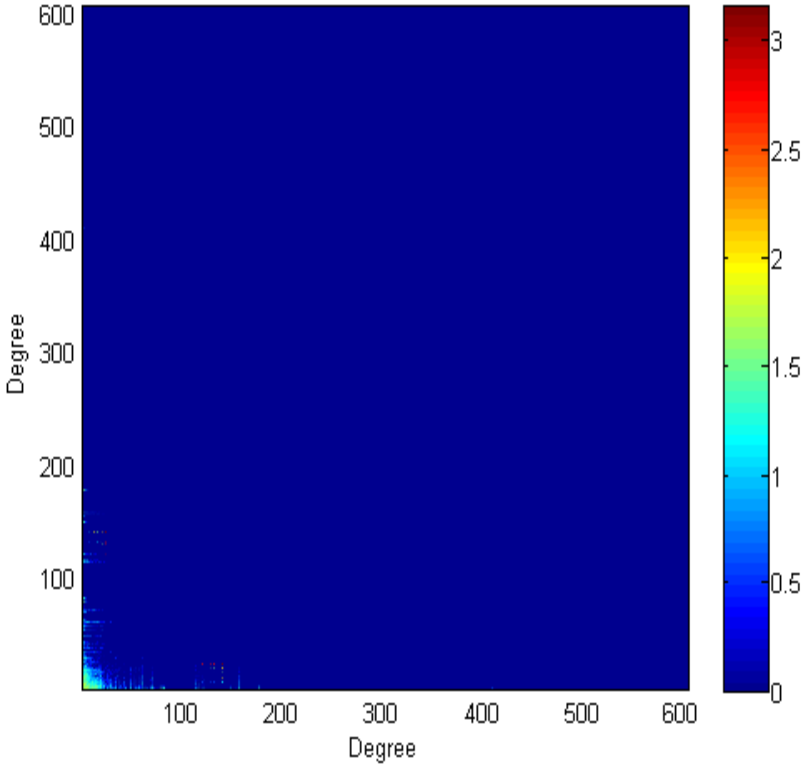
Other DHTs

- Many other DHTs exist too
  - Pastry, similar to Tapestry
  - Kademlia, uses XOR metric
  - Kelips, group nodes into *k* groups, similar to KaZaA
  - Plus some others…
- Overnet P2P network (also eDonkey) uses Kademlia
  - Wide-spread deployed DHT
- All DHTs provide same API
  - In principle, DHT-layer is interchangeable

dK-2 value matrix



Normal



Anomaly