INSTITUTE of INFORMATION SECURITY

Benign Activity Extraction for Dependency Reduction in Data Provenance-based Attack Analysis

Taishin Saito [a)] Masaki Hashimoto [b)] Kuniyasu Suzaki [a)]

a) Institute of Information Security

b) Kagawa University

Contents

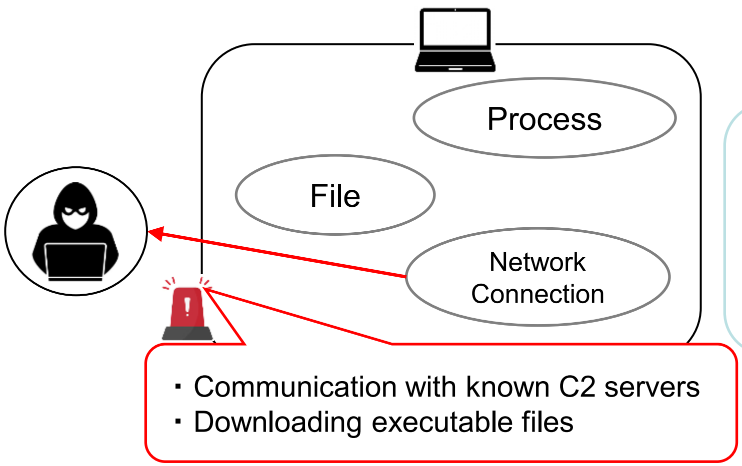
- Introduction
- Objective and Contribution
- Related Work
- Proposed Method
- Experimental Evaluation
- Results and Discussion
- Conclusion

Attack Detection Solutions

INSTITUTE of INFORMATION SECURITY

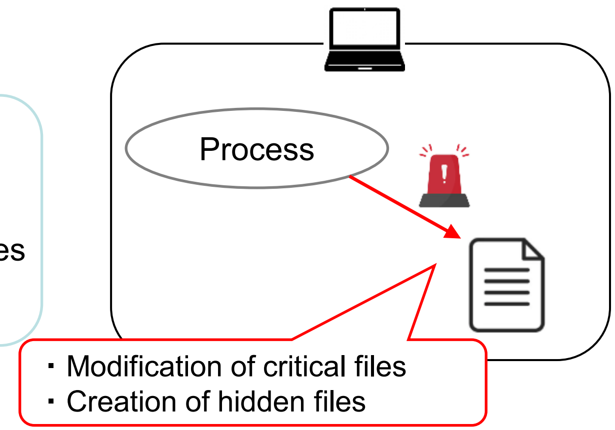
Conventional Attack Detection Solutions(Rule-based)

- Detection rules based on a single event



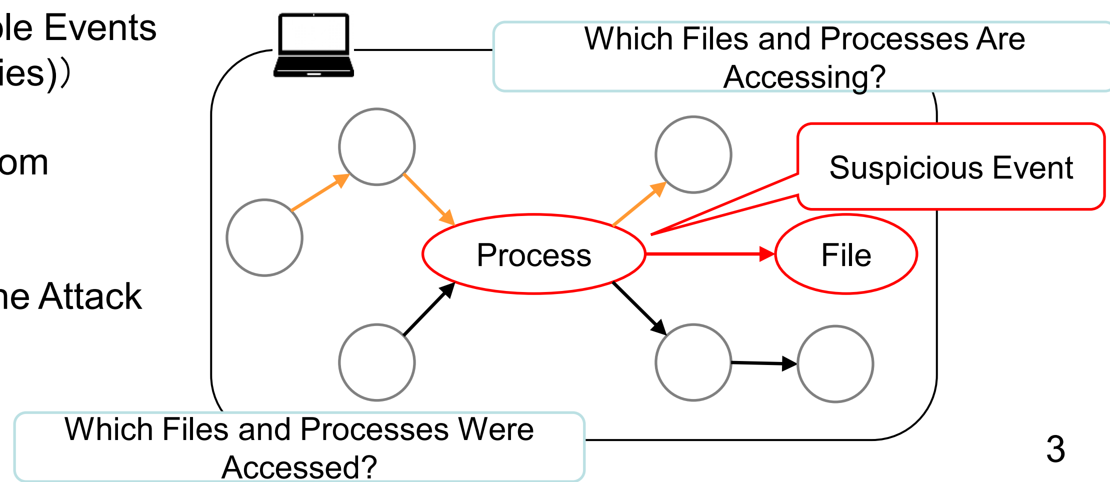
Difficult to Determine Whether It Is Truly an Attack

・Contains Legitimate User Activities
・Outdated IoC information



Recent Systems (Provenance-based IDS/EDR)

- Tracking Dependencies Among Multiple Events ((Identifying Activities))
- Determining Whether It Is an Attack from Activities Around Suspicious Events
- Identifying the Source and Scope of the Attack

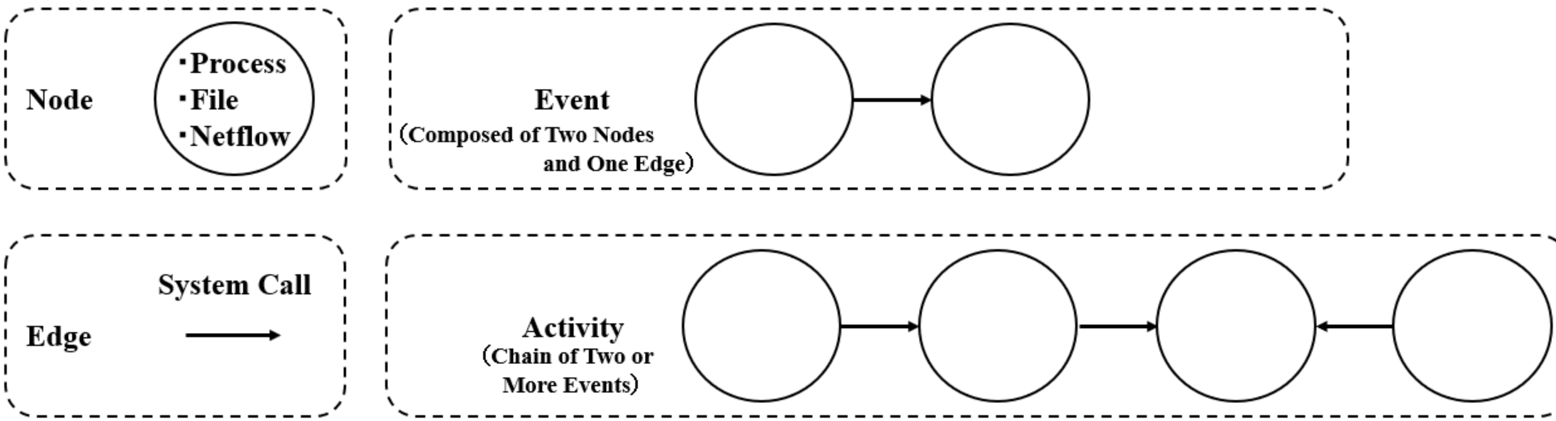


INSTITUTE of INFORMATION SECURITY

Data Provenance[1]

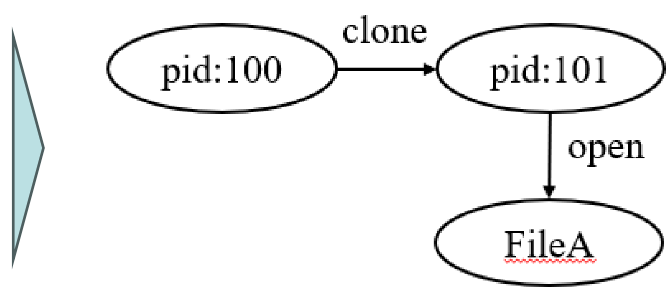
From where was a process or file generated or moved?

→ Trace the origin of the data, link related entities, and visualize them as a graph.



Log data

- Process {uuid: ”0”, pid: ”100”, src: ”ppid:”99”}
- Process {uuid: ”1”, pid: ”101”, src: ”ppid:100”}
- File {uuid: ”2”, filename: ”FileA”}
- Event {type:”clone”, src:”uuid:0”, dst:”uuid:1”}
- Event {type:”open”, src:”uuid:1”, dst:”uuid:2”}

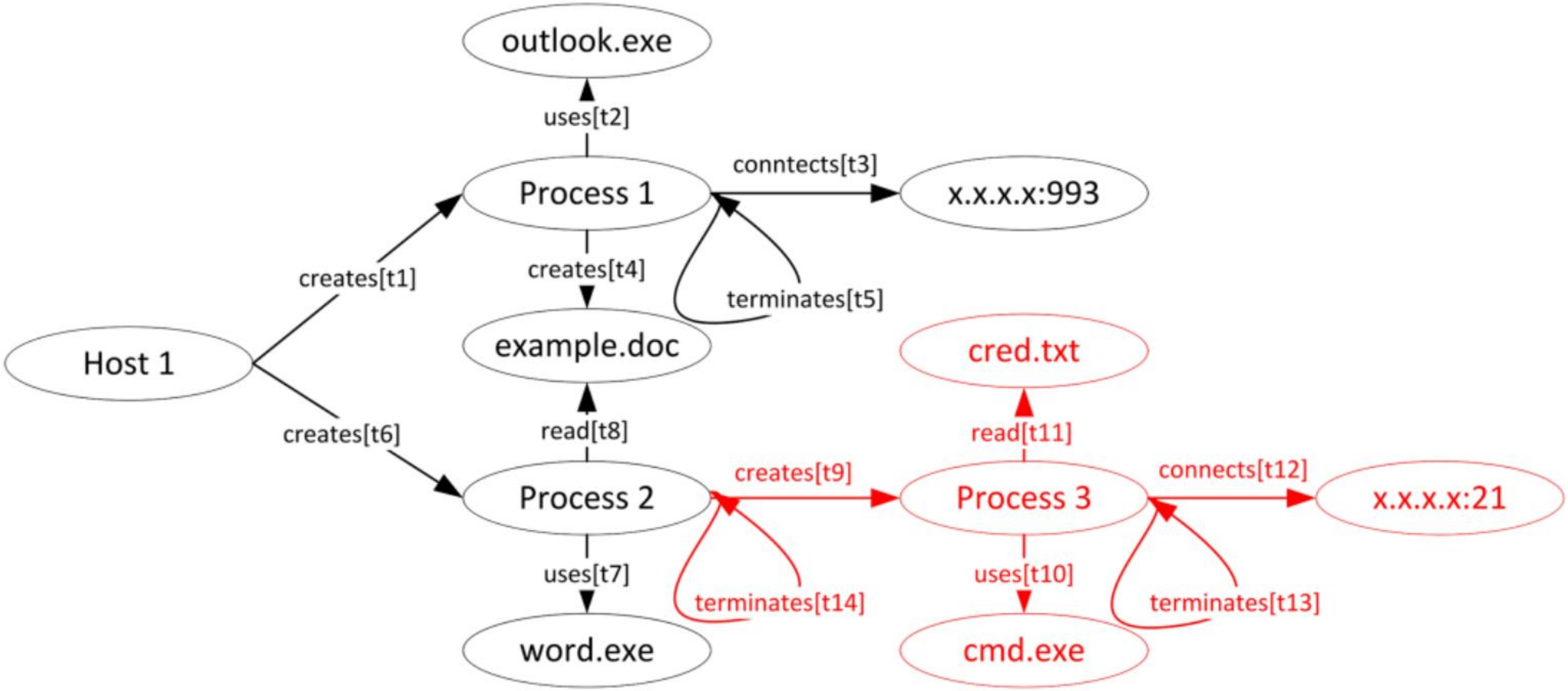


Log Data and Provenance Graph

[1] NIST. NIST SP 800-53 Rev. 5, Security and Privacy Controlsfor Information Systems and Organizations

Provenance Graph — Use Case

INSTITUTE of INFORMATION SECURITY



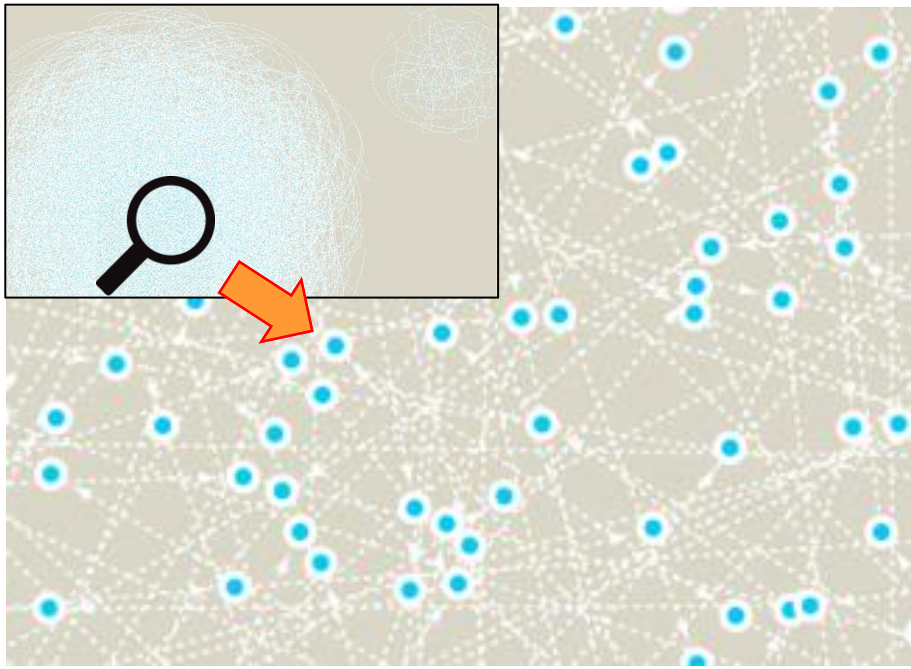
Example of a Provenance Graph exhibiting malicious activity [2]

- User reads an email using Outlook.
- User downloads an email attachment (example.doc) containing malicious code.
- User opens example.doc.
- Host spawns a process to run Word (Process 2).
- Host uses Word to open example.doc.
- Malicious code executes via Word macros.
- The malicious code leverages the existing process (Process 2) to spawn malware (Process 3).
- Malware uses the command prompt to read sensitive information (cred.txt) and exfiltrate it.

[2] Zipperle, Michael, et al. ”Provenance-based Intrusion Detection Systems: A Survey”ACM Comput. Surv. 2023

Challenge: Dependency Explosion

INSTITUTE of INFORMATION SECURITY



Dependency Explosion

Only attack-relevant activities

- Huge audit log volumes (e.g., several TB).
- Graphs include many dependencies unrelated to attacks.
  （Normal user actions and benign system behavior ）

→ **Result:** graphs become massive, making analysis difficult.

Ideal: Consist only of attack-relevant dependencies.

Contents

- Introduction
- Objective and Contribution
- Related Work
- Proposed Method
- Experimental Evaluation
- Results and Discussion
- Conclusion

Objective and Contribution

INSTITUTE of INFORMATION SECURITY

Objective

Extract benign activities within computer systems and reduce dependencies.

Contribution

- Propose a method to extract benign activities from log data and remove them from dependency graphs.
  ※ To our knowledge, this is the first attempt at dependency reduction itself.
  ※ Previous studies focused on removing benign events (single events) or extracting malicious activities.
- Demonstrate that approximately **6.8%–39%** of system activities can be defined as benign activity patterns.
- Show that using benign activities extracted from about **1～3%** of the log data can reduce dependencies by up to **52.3%**, indicating that a small amount of data can be used to shrink the search space in large-scale datasets.
- Analyze the **DARPA public dataset** and estimate the characteristics of each dataset.

Contents

INSTITUTE of INFORMATION SECURITY

- Introduction
- Objective and Contribution
- Related Work
- Proposed Method
- Experimental Evaluation
- Results and Discussion
- Conclusion

Related Work

INSTITUTE of INFORMATION SECURITY

Research proposing a method to reduce dependency explosion.

Name(Conference)	Year	Method	Dataset
LogGC (CCS)	2013	●Exclusion of temporary file deletion events	Original
CPR (CCS)	2016	●Merging of duplicate events	Original
NODOZE (NDSS)	2019	■Weighting based on anomaly scores	Original
DEPIMPACT (USENIX)	2022	■Weighting based on data flow volume and timing	DARPA TC + Original
NODLINK (NDSS)	2024	■Calculation of anomaly scores using NLP and VAE	Original

● Removal of Benign Events

■ Malicious Activity Identification via Weighting

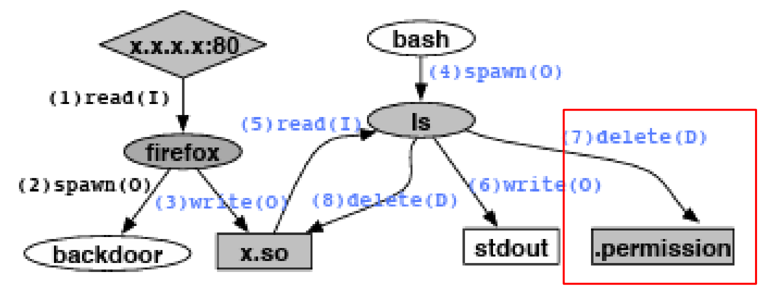
Removal of Benign Events

INSTITUTE of INFORMATION SECURITY

Remove benign events that do not affect analysis from the graph.

<u>LogGC[3]: Removal of Benign Events</u>

- About 23.8% of all log data consists of **temporary file deletion events**.
- These events are excluded from the dependency graph in advance.
- Temporary files are those handled by only one process during their lifetime.



(a)

<u>CPR[4]: Deduplicate Removal</u>

- **Integrates duplicate events**
- Reducing the number of edges

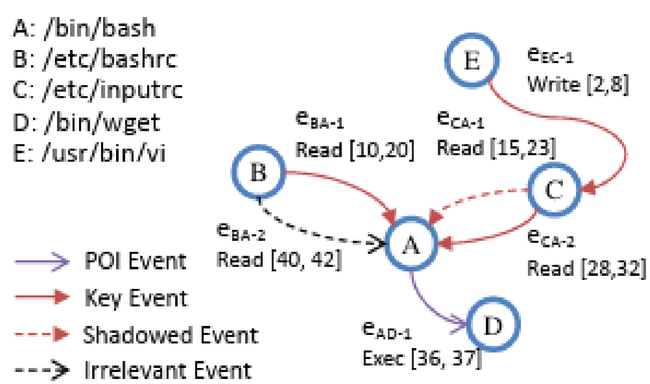


Figure 2: Unequal Dependencies in Backtracking

[3] Kyu Hyung Lee et al., LogGC: garbage collecting audit log, CCS '13: Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security

[4] Zhang Xu et al., High Fidelity Data Reduction for Big Data Security Dependency Analyses, CCS '16: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security

Malicious Activity Identification via Weighting

INSTITUTE of INFORMATION SECURITY

Assign weights to events based on their deviation from normal behavior and propagation from known malicious nodes or edges to extract malicious activities.

- Weighting factors include distance from detection points and data flow volume.
  - Events with characteristics similar to those detected by other IDSs.
  - Events that deviate from the features of normal behavior.
  - Events with known malicious attributes (e.g., file names, IP addresses).

- Only nodes and edges judged to be related to malicious activities are retained in the graph

<u>NODLINK[5]</u>:Converts natural language information in log data into numerical vectors.

Cmmand Line	"cat /etc/tmp/log.txt"
File Path	"/etc/tmp/log.txt"
Dst IP address : Port	"126.7.8.7 : 80"

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$

Uses a <u>V</u>ariational <u>A</u>uto <u>E</u>ncoder (VAE) to calculate anomaly scores.

[5] Shaofei Li et al. "NODLINK: An Online System forFine-Grained APT Attack Detection and Investiga-tion". In: Proceedings 2024 Network and DistributedSystem Security Symposium.

Related Work and Our Approach

INSTITUTE of INFORMATION SECURITY

Removal of benign events (e.g., temporary file deletions, duplicate events)

- Limited effectiveness in reducing dependencies since only part of the events are removed
- Assumes the existence of general benign events (events unnecessary for analysis).
  → **But are they truly unnecessary? This depends on the environment and analyst**.

Malicious activity identification via weighting

- Requires retraining to adapt to evolving attack behaviors.

A method is needed that can adapt to different environments and analysts
without requiring frequent retraining.

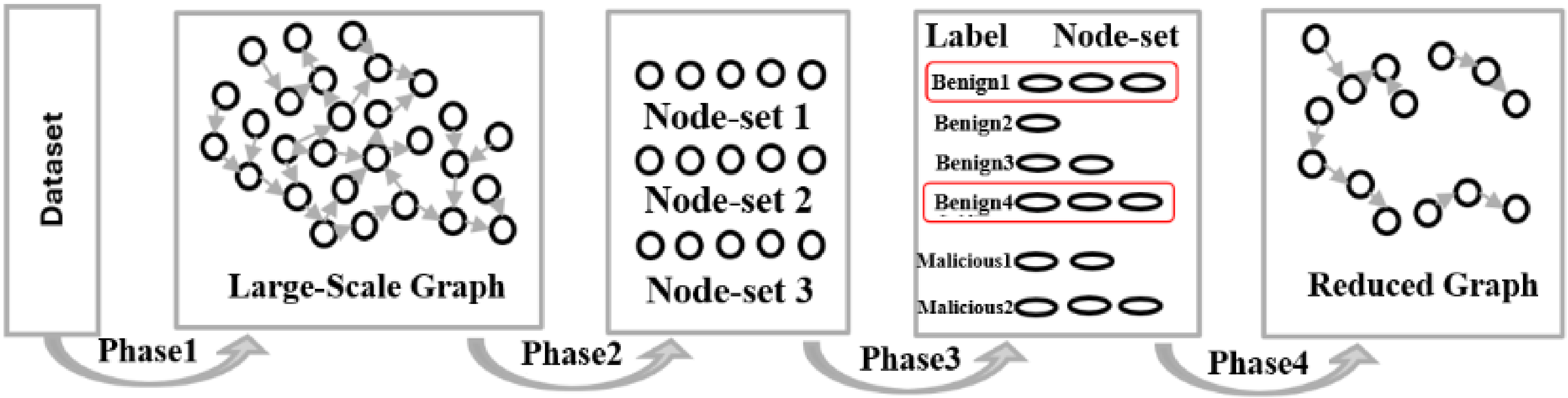
- Related work focused on **removing benign events** or **extracting malicious activities**.
- Dependency reduction through the **removal of benign activities** has not been explored.

Can we extract **system-specific benign activity patterns from log data** to reduce dependencies?

Contents

- Introduction
- Objective and Contribution
- Related Work
- Proposed Method
- Experimental Evaluation
- Results and Discussion
- Conclusion

Proposed Method



Overview of the proposed method

Phase 1: Data Preprocessing
Phase 2: Node-set Construction
Phase 3: Node-set Labeling
Phase 4: Ranking Labels and Reducing Dependencies

Phase 1: Data Preprocessing

INSTITUTE of INFORMATION SECURITY

Data Preprocessing

- Extraction of Target Edges and Nodes

Target nodes: processes, files, and network flows
Target edges: system calls used in related work[6][7] that were included in the dataset

Analysis targets

Events	System Call
Process/File	open, read, write, chmod, pipe
Process/Process	execve, clone
Process/NetFlow	recvfrom, sendto, recvmsg, sendmsg

- Extracted node information (command line, file name, IP address, and port number) for use in the next phase

- Constructed dependency graphs using the extracted data

[6] Shaofei Li et al. "NODLINK: An Online System forFine-Grained APT Attack Detection and Investiga-tion". In: Proceedings 2024 Network and DistributedSystem Security Symposium.
[7] Pengcheng Fang et al. , Back-Propagating System Dependency Impact for Attack Investigation, 31st USENIX Security Symposium (USENIX Security 22), 2022

Phase 2: Node-set Construction

INSTITUTE of INFORMATION SECURITY

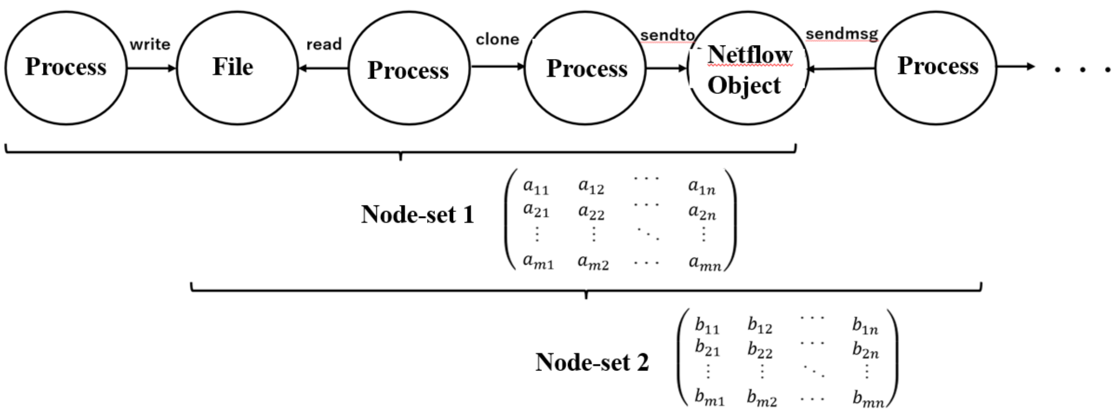
<u>Vectorization of Log Data and Construction of Node Sets</u>

- Vectorization

  Used **NODLINK** to convert command lines, file names, IP addresses, and port numbers into numerical vectors using **FastText [8]**.

- **Node-Set** Construction

  Extracted subgraphs consisting of five nodes to form each node set.



- Calculated feature values based on NODLINK's method:

$$V = w_c * V_c + \sum w_{f_i} * V_{f_i} + \sum w_{n_i} * V_{n_i} \qquad (1)$$

V_c, V_{f_i}, V_{n_i} : Features of command lines, files, and network flows

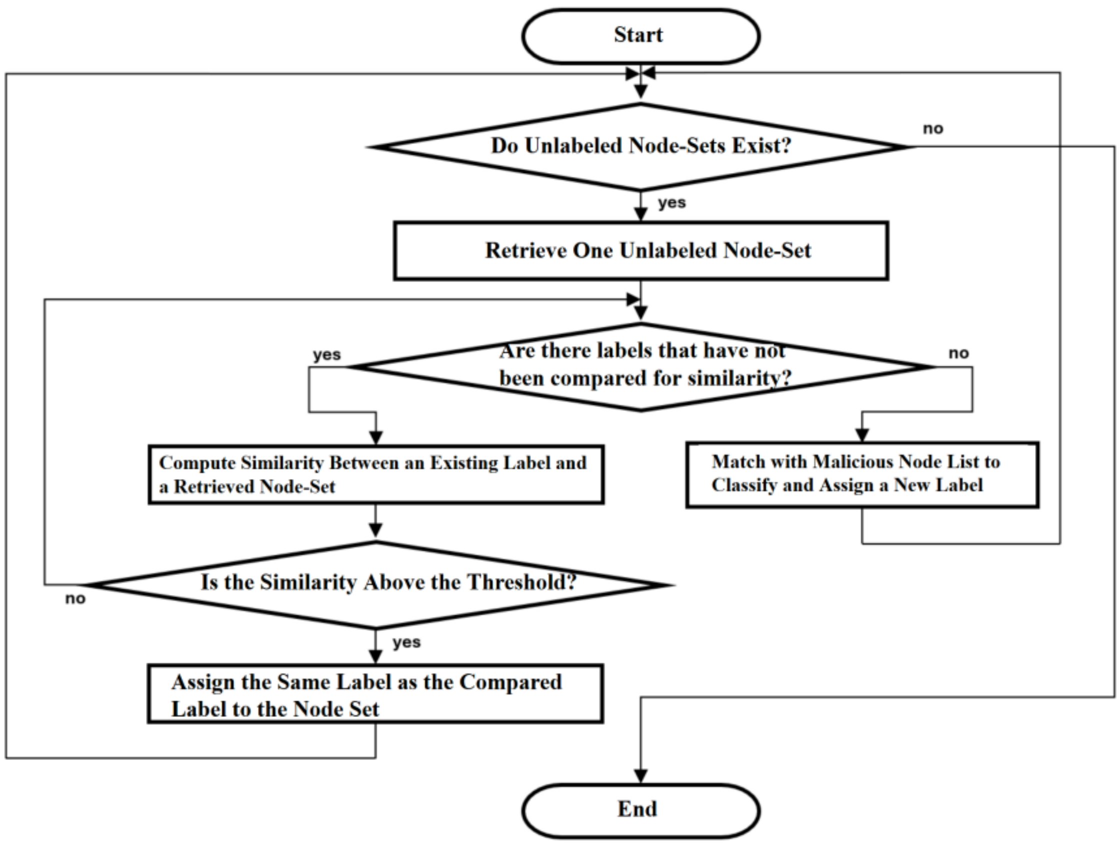
w_c, w_{f_i}, w_{n_i} : Weights for command lines, files, and network flows

[8] Bojanowski Piotr et al. "Enriching Word Vectors withSubword Information". In: Transactions of the Asso-ciation for Computational Linguistics 5 (Dec. 2017)

Phase 3: Node-set Labeling

INSTITUTE of INFORMATION SECURITY

Labeling of Node Sets



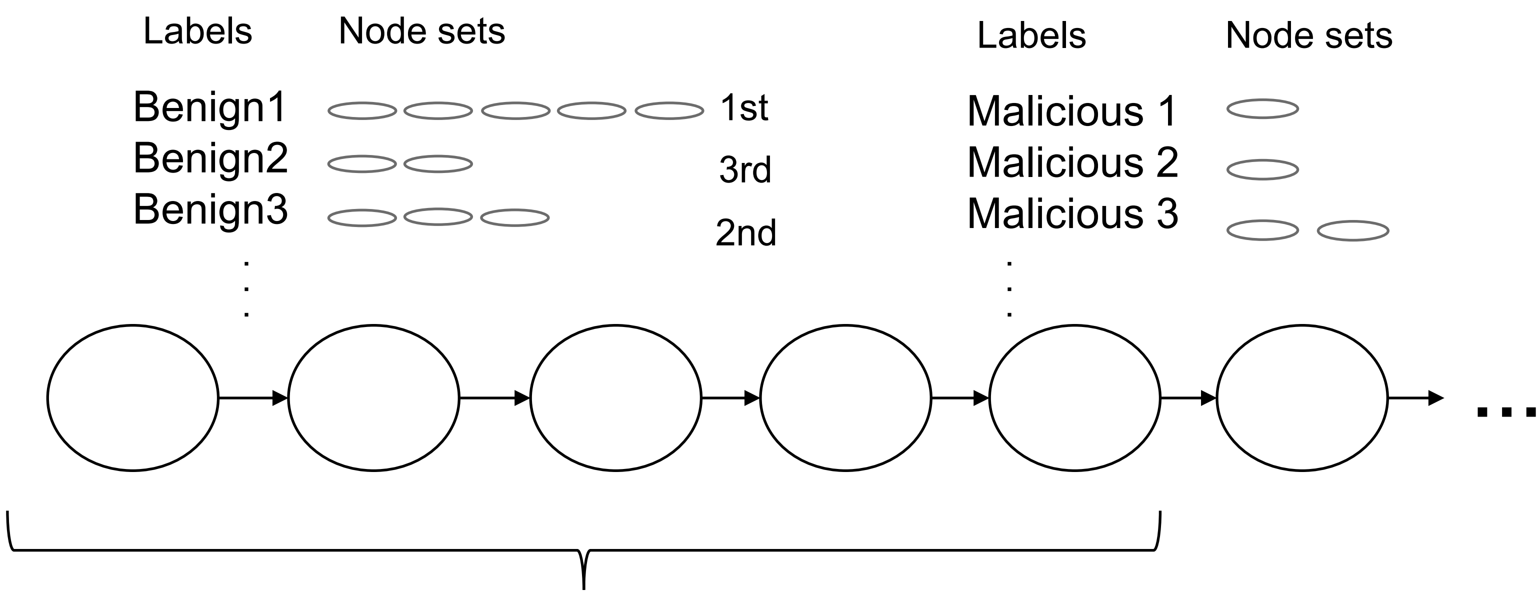
Labeling flow

- Prepared a predefined list of malicious nodes.
- If a node set contains at least one malicious node, a malicious label is assigned to that node set.
- Compare the feature vectors between node sets using cosine similarity.
- Repeat this process for all node sets.

Phase 4: Ranking Labels and Reducing Dependencies

Ranking and Dependency Graph Reduction

Remove activities similar to the top-ranked labels from the dependency graph constructed in Phase 1.



Check similarity with the **top-ranked label**.
→ If the similarity exceeds the threshold, the corresponding activity is removed from the graph.

Apply the Same Process to the Entire Graph

Set how many of the top-ranked labels are used for removal, and repeat the process for that number of labels.

Contents

- Introduction
- Objective and Contribution
- Related Work
- Proposed Method
- Experimental Evaluation
- Results and Discussion
- Conclusion

Experimental Setup

INSTITUTE of INFORMATION SECURITY

Environment

CPU : Intel(R) Xeon(R) Silver 4314(16core/2.4GHz)

Memory: 256GB

OS : Ubuntu22.04 64-bit

Parameters

Node Set Size: 5 Nodes

Cosine Similarity Threshold: 1.0

Experiment by varying the number of top-ranked benign label types selected for removal

- From **top 3 to top 1500 label types**

Metrics

False Negative (FN): The number of malicious nodes removed from the graph

False Positive (FP): The number of good nodes left in the graph

Node Reduction Rate: $(1-\frac{\text{Number of Nodes in the Graph Before Dependency Reduction}}{\text{Number of Nodes in the Graph After Dependency Reduction}})\times 100$

Dataset

INSTITUTE of INFORMATION SECURITY

DARPA Transparent Computing(TC) Data[9]

- Only public dataset used in previous studies
- Engagement 3 (E3) released in 2018
- Engagement 5 (E5) released in 2019
- Three datasets used: E3 Theia, E5 Theia, and E5 Marple
- Each dataset was partially extracted for training data (**three subsets: A, B, and C**)

Dataset	Data size	Proportion to Evaluation Data
E3 Theia-A	3.8GB	13.4%
E3 Theia-B	3.8GB	13.4%
E3 Theia-C	3.8GB	13.4%
E5 Theia-A	4.0GB	1.35%
E5 Theia-B	4.0GB	1.35%
E5 Theia-C	4.0GB	1.35%
E5 Marple-A	3.6GB	2.98%
E5 Marple-B	3.6GB	2.98%
E5 Marple-C	3.8GB	3.15%

E3 Theia
Log data from one Ubuntu 12.04 host (28.3 GB used for evaluation).
Includes backdoor installation exploiting a Firefox vulnerability and records of phishing emails.
(81 malicious nodes)

E5 Theia
Log data from three Ubuntu 12.04 hosts (295.8 GB used for evaluation).
Includes backdoor installation exploiting a Firefox vulnerability and communications to C2 servers.
(4 malicious nodes)

E5 Marple
Log data from one Windows 7 host (120.7 GB used for evaluation).
Includes backdoor installation exploiting a Firefox vulnerability and communications to C2 servers.
(10 malicious nodes)

[9] DARPA. TC: Transparent Computing [Online].https://www.darpa.mil/research/programs/transparent-computing. (visited on2025-01-06)

Contents

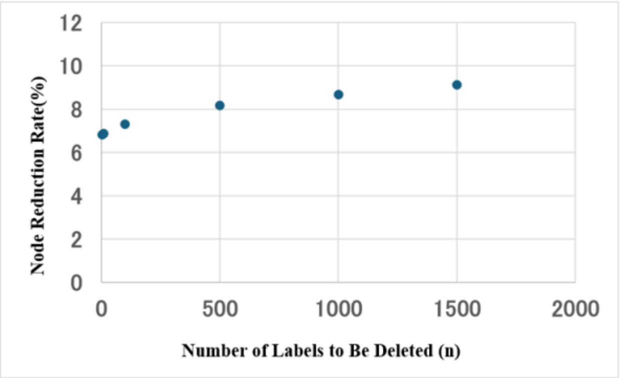
- Introduction
- Objective and Contribution
- Related Work
- Proposed Method
- Experimental Evaluation
- Results and Discussion
- Conclusion

Experimental Results: E3 Theia

INSTITUTE of INFORMATION SECURITY

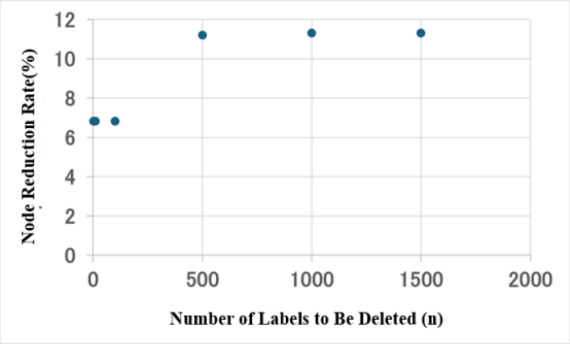
Experimental Results(E3 Theia-A)

Number of Labels to Be Deleted (n)	Node Count	FN	FP	Node Reduction Rate(%)	Execution Time(sec)
3	50,802	0	50,721	6.82	7.264
10	50,778	0	50,697	6.87	8.701
100	50,541	0	50,460	7.30	16.09
500	50,064	0	49,983	8.17	44.08
1000	49,796	0	49,715	8.67	77.24
1500	49,549	0	49,468	9.12	103.1



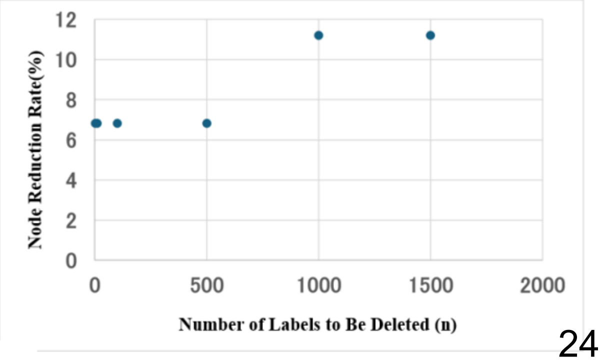
Experimental Results(E3 Theia-B)

Number of Labels to Be Deleted (n)	Node Count	FN	FP	Node Reduction Rate(%)	Execution Time(sec)
3	50,802	0	50,721	6.82	7.396
10	50,802	0	50,721	6.82	8.682
100	50,798	0	50,717	6.83	15.96
500	48,393	0	48,312	11.2	42.88
1000	48,383	0	48,302	11.3	68.49
1500	48,380	0	48,299	11.3	90.37



Experimental Results(E3 Theia-C)

Number of Labels to Be Deleted (n)	Node Count	FN	FP	Node Reduction Rate(%)	Execution Time(sec)
3	50,802	0	50,721	6.82	6.83
10	50,802	0	50,721	6.82	7.779
100	50,801	0	50,720	6.82	15.98
500	50,800	0	50,719	6.83	43.89
1000	48,398	0	48,317	11.2	69.63
1500	48,398	0	48,317	11.2	91.43

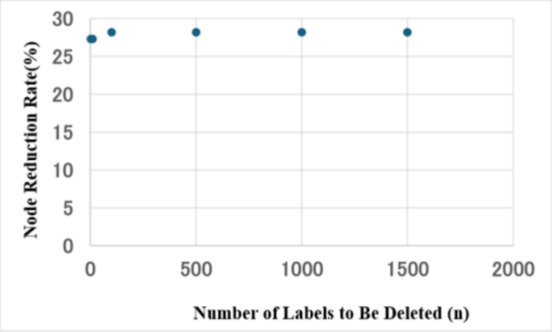


Experimental Results: E5 Theia

INSTITUTE of INFORMATION SECURITY

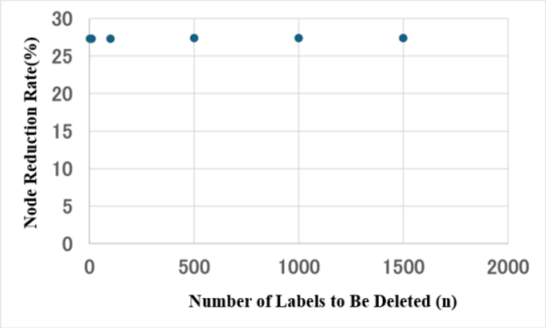
Experimental Results(E5 Theia-A)

Number of Labels to Be Deleted (n)	Node Count	FN	FP	Node Reduction Rate(%)	Execution Time(sec)
3	2,206,930	0	2,206,926	27.3	795.4
10	2,206,813	0	2,206,809	27.3	829.1
100	2,178,684	0	2,178,680	28.2	1100
500	2,177,642	0	2,177,638	28.2	1906
1000	2,177,392	0	2,177,388	28.2	2638
1500	2,177,329	0	2,177,325	28.2	3136



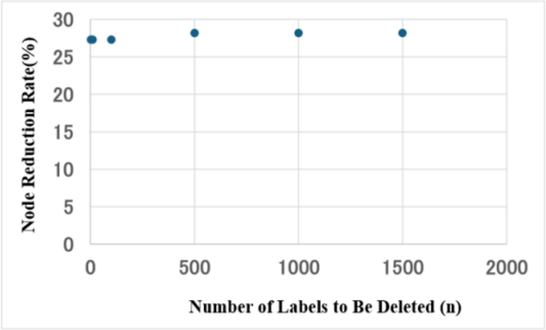
Experimental Results(E5 Theia-B)

Number of Labels to Be Deleted (n)	Node Count	FN	FP	Node Reduction Rate(%)	Execution Time(sec)
3	2,206,909	0	2,206,905	27.3	783.9
10	2,206,769	0	2,206,765	27.3	810.7
100	2,205,545	0	2,205,541	27.3	1051
500	2,204,039	0	2,204,035	27.4	1794
1000	2,203,720	0	2,203,716	27.4	2435
1500	2,203,584	0	2,203,580	27.4	2879



Experimental Results(E5 Theia-C)

Number of Labels to Be Deleted (n)	Node Count	FN	FP	Node Reduction Rate(%)	Execution Time(sec)
3	2,206,900	0	2,206,896	27.3	797.8
10	2,206,790	0	2,206,786	27.3	822.7
100	2,205,921	0	2,205,917	27.3	1,085
500	2,177,493	0	2,177,489	28.2	1,932
1000	2,177,258	0	2,177,254	28.2	2,654
1500	2,177,248	0	2,177,244	28.2	3,169

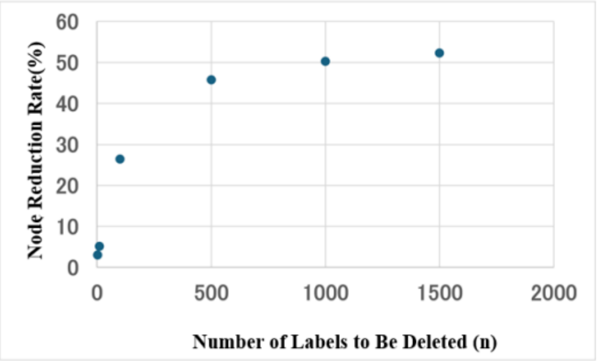


Experimental Results: E5 Marple

INSTITUTE of INFORMATION SECURITY

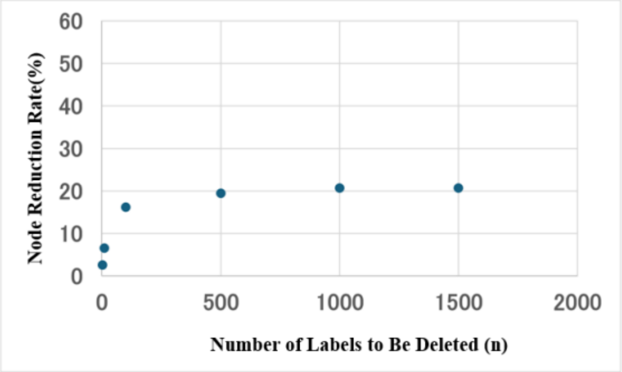
Experimental Results(E5 Marple-A)

Number of Labels to Be Deleted (n)	Node Count	FN	FP	Node Reduction Rate(%)	Execution Time(sec)
3	11,803,667	0	11,803,657	3.03	558.7
10	11,545,745	0	11,545,735	5.15	717.1
100	8,956,381	0	8,956,371	26.4	2,091
500	6,600,627	0	6,600,617	45.8	5,768
1000	6,048,300	0	6,048,290	50.3	9,130
1500	5,800,663	0	5,800,653	52.3	11,690



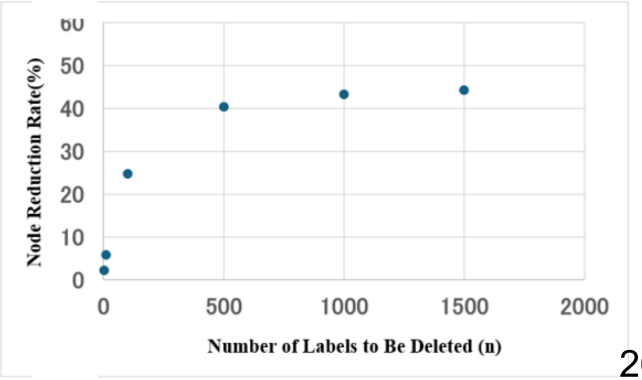
Experimental Results(E5 Marple-B)

Number of Labels to Be Deleted (n)	Node Count	FN	FP	Node Reduction Rate(%)	Execution Time(sec)
3	11,855,130	0	11,855,120	2.61	527.5
10	11,373,838	0	11,373,828	6.56	627.8
100	9,648,817	0	9,648,807	16.2	1,516
500	9,795,840	0	9,795,830	19.5	3,008
1000	9,648,817	0	9,648,807	20.7	3,603
1500	9,648,817	0	9,648,807	20.7	3,631



Experimental Results(E5 Marple-C)

Number of Labels to Be Deleted (n)	Node Count	FN	FP	Node Reduction Rate(%)	Execution Time(sec)
3	11,903,443	0	11,903,433	2.21	538.9
10	11,459,815	0	11,459,805	5.85	678.7
100	9,154,142	0	9,154,132	24.8	1886
500	7,258,716	0	7,258,706	40.4	5265
1000	6,900,864	0	6,900,854	43.3	8375
1500	6,779,474	0	6,779,464	44.3	10370



Effectiveness of Dependency Reduction

Is it possible to reduce dependencies by **extracting benign activities?**

- No false negatives were observed within the scope of this experiment
  → Node sets containing malicious nodes showed distinct features from benign labels.
- The graph reduction rate increased as the number of labels selected for removal increased.

Dependency reduction based on benign activities is feasible.

How much can dependencies be reduced?

- The maximum reduction rate in this experiment was **52.3%**.
- Based on the average reduction rates across datasets, approximately **6.8%–39%** of dependencies were identified as frequent benign activities within computer systems.
- Using about **13%** of the total log data for benign activity extraction (E3 Theia) resulted in a lower average reduction rate than using only **1.4–3.2%** of the data (E5 Theia, E5 Marple).
  → A small portion of the log data is sufficient for extracting benign activities.

Experimental results using **E5 Marple-A**.

Labels	Node Count	FN	FP	Node Reduction (%)	Exec. Time (sec)
3	11,803,667	0	11,803,657	3.03	558.7
10	11,545,745	0	11,545,735	5.15	717.1
100	8,956,381	0	8,956,371	26.4	2.091
500	6,600,627	0	6,600,617	45.8	5.768
1000	6,048,300	0	6,048,290	50.3	9.130
1500	5,800,663	0	5,800,653	52.3	11.690

Average reduction rate and execution time for each dataset.

Dataset	Avg. Reduction Rate (%)		Avg. Execution Time (sec)	
	Min ($n = 3$)	Max ($n = 1500$)	Min ($n = 3$)	Max ($n = 1500$)
E3 Theia	6.82	10.5	7.16	95.0
E5 Theia	27.3	27.9	792	3061
E5 Marple	2.62	39.1	542	8563

Dataset Features and Their Impact

INSTITUTE of INFORMATION SECURITY

Highest Average Reduction Rate

- 3 labels → **E5 Theia**
- 1500 labels → **E5 Marple**
- **E3 Theia** showed a lower rate than the other two datasets

Dataset	Avg. Reduction Rate (%)		Avg. Execution Time (sec)	
	Min ($n=3$)	Max ($n=1500$)	Min ($n=3$)	Max ($n=1500$)
E3 Theia	6.82	10.5	7.16	95.0
E5 Theia	27.3	27.9	792	3061
E5 Marple	2.62	39.1	542	8563

Characteristics of Each Dataset

E3 Theia
- Frequent use of general-purpose applications (e.g., *Firefox*, *Thunderbird*)
  → Various command lines are used.

E5 Theia
- Many system administration and update-related processes
  → Usage is concentrated on specific command lines.

E5 Marple
- Many command lines for analyzing document files
  → Usage is concentrated on similar command lines performing the same operations.

E3 Theia: General-purpose system

E5 Theia / Marple: Repetitive, task-specific systems

→ High potential effectiveness for systems that perform repeated, specific operations (e.g., dedicated servers)

Limitations and Future Work

Limitations

- It requires at least partially analyzed data for training.
  → Difficult to apply to environments that have not been analyzed at all.
- Difficult to apply when logs from multiple OSs are mixed, since natural-language features such as command lines and file paths differ by OS.
- The natural language processing component depends on related work(NODLINK).

Future Work

- Optimization of node set size and cosine similarity threshold.
- Verification using a wider variety of datasets.
- Application to intrusion detection systems:
  Extracted benign activities could be utilized in whitelist-based intrusion detection.

Contents

INSTITUTE of INFORMATION SECURITY

- Introduction
- Objective and Contribution
- Related Work
- Proposed Method
- Experimental Evaluation
- Results and Discussion
- Conclusion

Conclusion

INSTITUTE of INFORMATION SECURITY

- Recent studies link and visualize malicious activities for attack analysis.

- The dependency explosion problem remains unsolved.

- We propose a method to reduce dependencies by extracting benign activities using NLP and cosine similarity.

- **Our method demonstrated that:**
  - Dependency reduction through benign activities is feasible.
  - About 10% of system activities can potentially be defined as patterned benign activities.
  - Benign activities can be extracted even from small-scale data (1~3%).

<u>Future Work</u>
- Investigate the effects of parameter variation (node set size, similarity threshold).
- Validate the method using a wider variety of datasets.
- Explore applications to intrusion detection systems.