NODLINK: An Online System for Fine-Grained APT Attack Detection and Investigation

Shaofei Li[1], Feng Dong[2], Xusheng Xiao[3], Haoyu Wang[2], Fei Shao[4], Jiedong Chen[5],
Yao Guo[1], Xiangqun Chen[1], Ding Li[1]

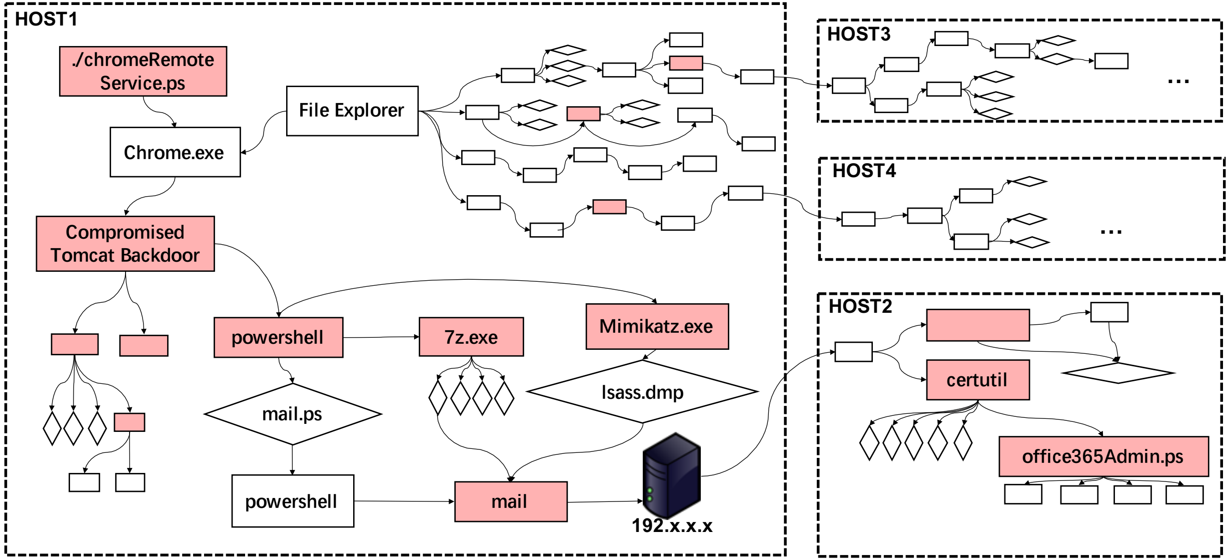
[1]School of Computer Science, Peking University, [2]Huazhong University of Science and Technology
[3]Arizona State University, [4]Case Western Reserve University, [5]Sangfor Technologies Inc.

Advanced Persistent Threats

- **APT attacks** have become a major threat to modern enterprises
  - **Advanced:** Attackers have diverse attack vectors → Zero-day exploits
  - **Persistent:** Long duration → Low-and-slow attack patterns
- Existing **Endpoint Detection and Response (EDR) systems** often fail

Provenance-Based Detection Systems

- Provenance-based detection systems are based on **provenance log**
  - **Node:** system entities (process, file, network)
  - **Edge:** system events (read, write, fork, execve, sendto, recvfrom, …)
- **Threat detection is to search for a needle in a haystack**



Provenance-Based Approaches

- Workflow of provenance-based APT detection:

Attack Candidate(IOC) Detection → Provenance Graph Construction → Comprehensive Detection

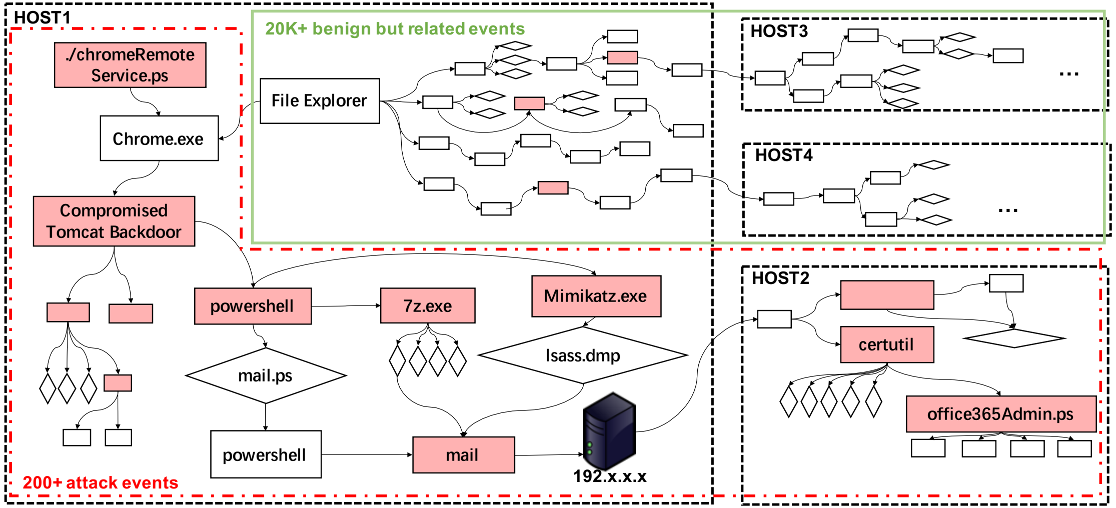
- **Provenance graph construction**

	Efficiency	**Conciseness**	**Generalizability**
Rule-based Systems	Efficient	Fine-grained	Manual rules
Anomaly-based Systems	Heavy	Coarse-grained	Generalized

- **Goal:** Achieve **efficiency**, **conciseness** and **generalizability** altogether

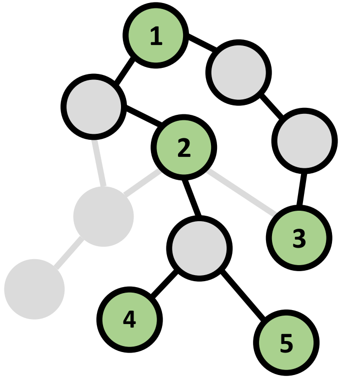
Key Insight: Properties

- **Attack Affinity:**
  - Attacks are more likely to generate suspicious processes
- **Attack Polymerism:**
  - Attack actions are topologically close



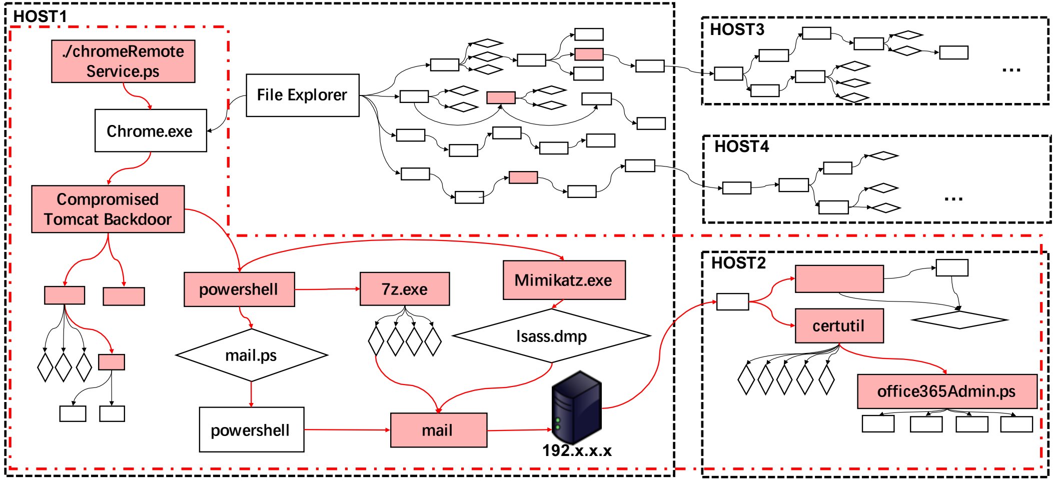
Key Insight

- To utilize these two properties, we model provenance graph construction as an **online Steiner Tree Problem**
- A **Steiner tree** is a subgraph that spans the given node set **(terminal set)**
- **Online Steiner Tree Problem (OSTP)**
  - Undirected graph with non-negative weight for each edge
  - New terminals are online revealed
  - Keep the Steiner tree that has the minimal weight
- **Optimal greedy algorithm**
  - **Bounded approximation algorithm**
  - When new terminal arrives:
    - Find the **shortest path** from new terminal to the existing Steiner tree
    - Extend the existing Steiner tree with the new path



Key Insight

- To utilize these two properties, we model provenance graph construction as an **online Steiner Tree Problem**
  - **Terminals**: IOCs or anomaly events
  - **Edge weight**: the same non-negative weight
  - Search for a subgraph that links all the anomalies with minimal number of edges

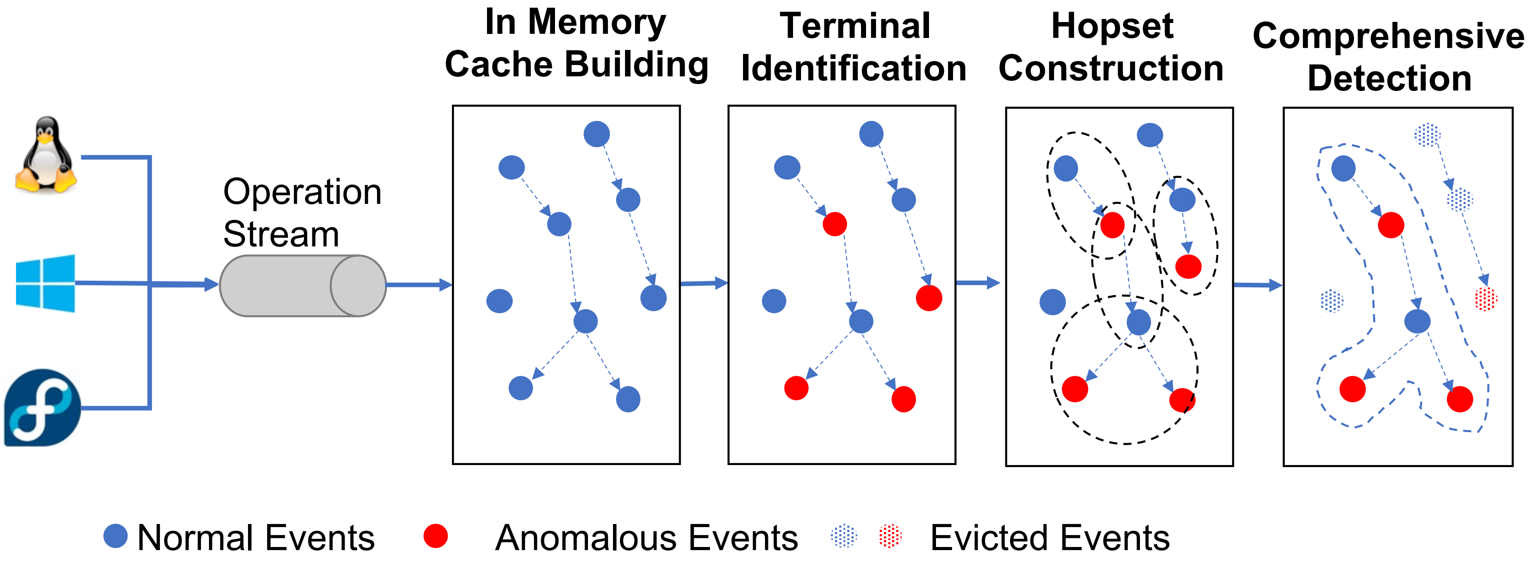


Key Challenges

- **C1: How to detect long-term attacks with limited resources**
  - Requires knowing the whole provenance graph.
  - **Solution1:** In-memory cache design prioritizing suspicious events
- **C2: How to identify terminals in STP with constraint of timeliness.**
  - Existing methods are time-consuming and not suitable for online system.
  - **Solution2:** Inverse Document Frequency-weighted Variational AutoEncoder
- **C3: Current algorithms for OSTP are not efficient enough for APT attack detection.**
  - Find the shortest path from the new terminal to all previous terminals.
  - **Solution3**: Importance-Score-Guided greedy algorithm for OSTP optimization

Our Solution: NODLINK

- **Detect anomalies through four phases periodically:**



C1: How to detect long-term attacks

- **Solution1:** In-memory cache design prioritizing suspicious events
  - Cache **more suspicious** and **actively evolving** graphs
- **In every time window:**
  - **In-Memory Cache Construction**
    - Find the solution on the current provenance graph(**Hopset**)
  - **Cache Update**
    - Update cache with the hopset we just constructed
    - Preserve Top-K hopsets in the cache and evict others to disk
    - **Prioritize metric**: Energy of hopset

$$E = \epsilon^{\mathrm{Age}} * \mathrm{HopsetAnomalyScore}(H) \qquad \epsilon < 1$$

$$\mathrm{HopsetAnomalyScore}(H) = \sum_{v \in H} \mathrm{AnomalyScore}(v)$$

C2: How to identify terminals in STP

- **Solution2:** Inverse Document Frequency(IDF)-weighted Variational AutoEncoder (VAE)
  - **Embed** process nodes and **classify** them
- **Process-centric Embedding**
  - **Node-level feature embedding**: pretrained FastText model using historical data

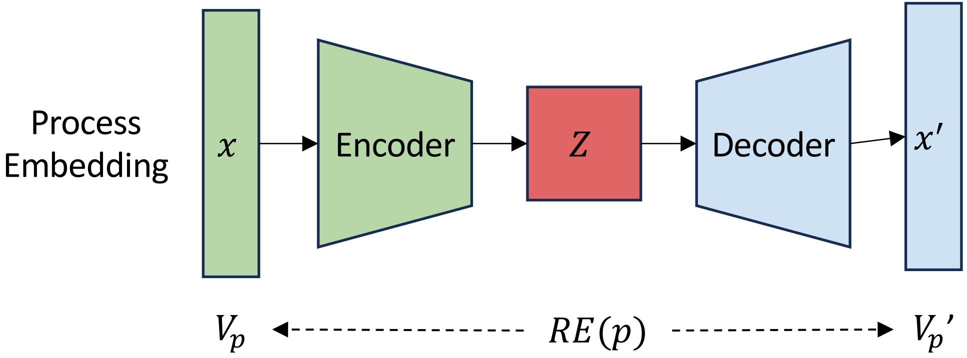
Type	Node-level Feature	Sentence
Command Line	date -d 4857 second ago +%s	[date, second, ago]
Files	/etc/tmp/log.txt	[etc, tmp, log, txt]
IP Addresses	<126.7.8.7, 80, 162.0.0.1, 8080>	[126, 7, 8, 7, 80, 162, 0, 0, 1, 8080]

  - **IDF-Weighted combination**:

$$V_p = w_c * V_c + \sum w_{f_i} * V_{f_i} + \sum w_{n_i} * V_{n_i} \quad w_c = \log(\frac{P}{P_c}) \quad w_{f_i} = \log(\frac{P}{P_{f_i}}) \quad w_{n_i} = \log(\frac{P}{P_{n_i}})$$

C2: How to identify terminals in STP

- **Anomaly Detection:** classify unusual processes as anomaly
  - **VAE-based detection**: based on reconstruction error (RE)



  - **Anomaly score**: balance the RE for unstable processes

$$\mathrm{AnomalyScore}(p) = \log\left(\frac{\mathrm{RE}(p)}{\mathrm{StableValue}(p)}\right)$$

  - **Anomaly:** processes with anomaly score over 90th percentile

C3: More Efficient Algorithm

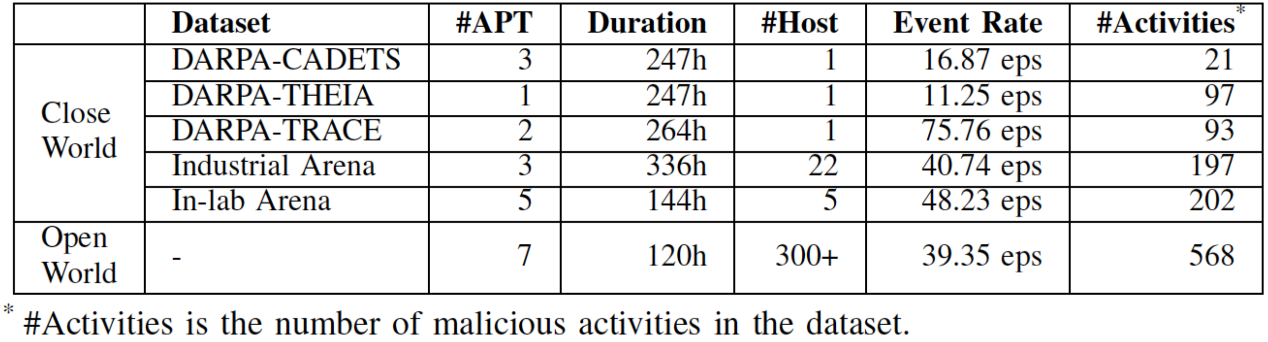
- **Solution3:** Importance-oriented greedy algorithm for OSTP optimization
  - **Search for hopset locally** and takes advantage of **attack polymerism**
- **Hopset Construction**: Importance-Score-Guided Search (ISG)
  - Start local searching procedures for each terminal
  - Hopset keeps θ nodes prioritized by **Importance Value(IV)**

$$IV(v) = \alpha^{\text{Distance}}\left(\beta * \text{AnomalyScore}(v) + \gamma * \frac{\text{OutDegree}(v)}{\text{InDegree}(v) + 1}\right) \qquad \begin{array}{l} \alpha < 1 \\ \beta \gg \gamma \end{array}$$

- **Complexity:** $O(E + \theta N)$
- **Competive Ratio**: $2\theta O(logk) \approx O(logk)$

Evaluation: Datasets

- **Close-World Datasets:**
  - DARPA TC dataset
  - Industrial Arena dataset
  - In-lab Arena dataset[1]
- **Open-World Datasets:**
  - Deploy NODLINK to monitor **10 realistic customers**



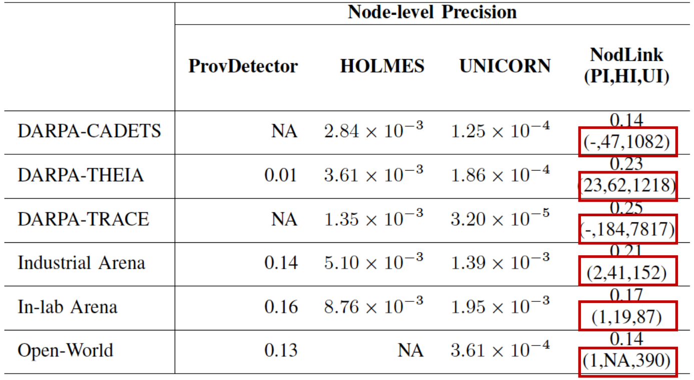
	Dataset	**#APT**	**Duration**	**#Host**	**Event Rate**	**#Activities***
Close World	DARPA-CADETS	3	247h	1	16.87 eps	21
	DARPA-THEIA	1	247h	1	11.25 eps	97
	DARPA-TRACE	2	264h	1	75.76 eps	93
	Industrial Arena	3	336h	22	40.74 eps	197
	In-lab Arena	5	144h	5	48.23 eps	202
Open World	-	7	120h	300+	39.35 eps	568

* #Activities is the number of malicious activities in the dataset.

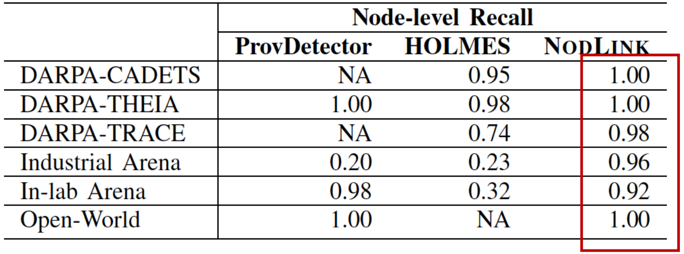
[1] https://github.com/PKU-ASAL/Simulated-Data

Evaluation: Effectiveness

- **Graph-level accuracy:**
  - Detects **all the attacks** and only reports **14 false positives**.
  - ProvDetector and HOLMES report 783 and 416 false positives
- **Node-level accuracy**:
  - **Node-level precision:** comparable with ProvDetector; **one to three orders of magnitude higher** than two online baselines
  - **Node-level recall:** covers **98%** of the attacks on average



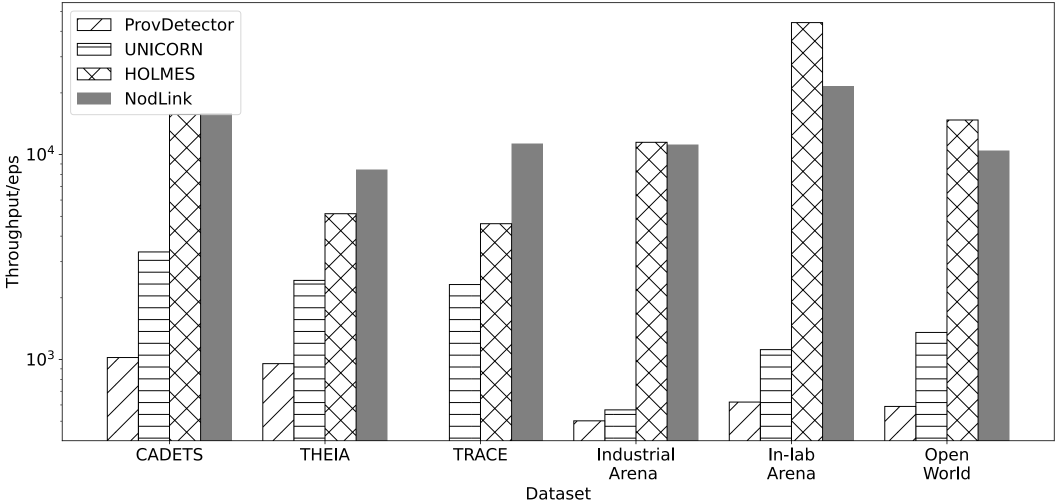
	Node-level Precision			
	ProvDetector	**HOLMES**	**UNICORN**	**NodLink (PI,HI,UI)**
DARPA-CADETS	NA	2.84×10^{-3}	1.25×10^{-4}	0.14 (-,47,1082)
DARPA-THEIA	0.01	3.61×10^{-3}	1.86×10^{-4}	0.23 (23,62,1218)
DARPA-TRACE	NA	1.35×10^{-3}	3.20×10^{-5}	0.25 (-,184,7817)
Industrial Arena	0.14	5.10×10^{-3}	1.39×10^{-3}	0.21 (2,41,152)
In-lab Arena	0.16	8.76×10^{-3}	1.95×10^{-3}	0.17 (1,19,87)
Open-World	0.13	NA	3.61×10^{-4}	0.14 (1,NA,390)



	Node-level Recall		
	ProvDetector	**HOLMES**	**NodLink**
DARPA-CADETS	NA	0.95	1.00
DARPA-THEIA	1.00	0.98	1.00
DARPA-TRACE	NA	0.74	0.98
Industrial Arena	0.20	0.23	0.96
In-lab Arena	0.98	0.32	0.92
Open-World	1.00	NA	1.00

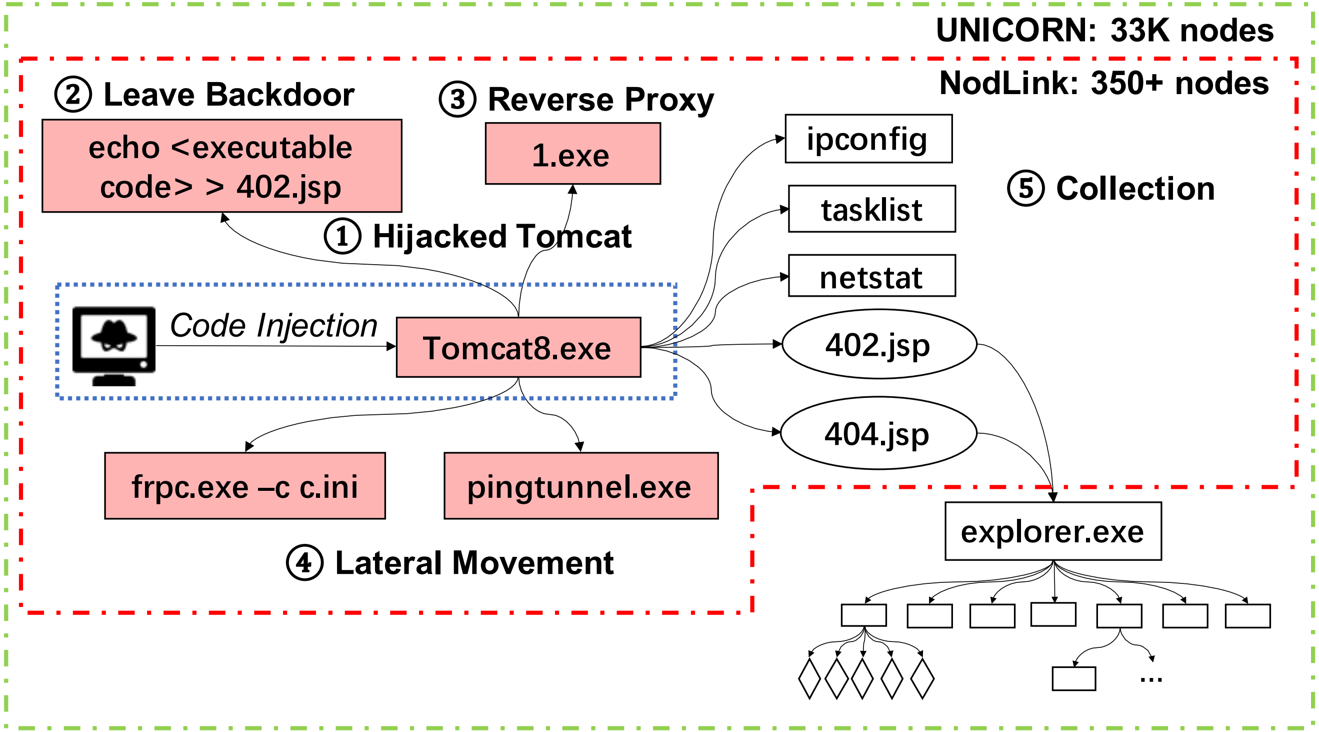
Evaluation: Efficiency

- **Throughput:** how many system events can be processed per second (eps)
  - Comparable with rule-based HOLMES; **21x higher** than ProvDetector; **7x higher** than UNICORN
  - Capable of monitoring **329** hosts, considering that an open-world host generates an average of **40** events per second



Evaluation: Attacks Detected In Production

- NODLINK detects **7 real attacks** in the open-world experiment



Conclusion

- Online provenance-based detection systems are preferred over post-mortem ones in APT attack detection.
- We propose **NODLINK**, an online provenance-based detection system that can achieve **efficiency**, **conciseness** and **generalizability** altogether. The key idea is to model the APT attack detection problem as an online STP.
- Our experiments show that NODLINK can achieve higher accuracy with the same or higher throughput.

https://github.com/PKU-ASAL/Simulated-Data

lishaofei@pku.edu.cn