ProvNet-IoT: Provenance based Network Layer Forensics in Internet of Things

By:
Lakshminarayana Sadineni, Emmanuel S. Pilli and Ramesh Babu Battula

From the proceedings of
The Digital Forensic Research Conference
DFRWS APAC 2022
Sept 28-30, 2022

Forensic Science International: Digital Investigation 43 (2022) 301441

Contents lists available at ScienceDirect

Forensic Science International: Digital Investigation

journal homepage: www.elsevier.com/locate/fsidi

DFRWS 2022 APAC - Proceedings of the Second Annual DFRWS APAC

ProvNet-IoT: Provenance based network layer forensics in Internet of Things

Lakshminarayana Sadineni, Emmanuel S. Pilli*, Ramesh Babu Battula

Department of Computer Science and Engineering, Malaviya National Institute of Technology, Jaipur, Rajasthan, 302017, India

ARTICLE INFO

Article history:

Keywords:
Internet of things forensics
Packet analysis
Network provenance
Provenance modeling

ABSTRACT

Internet of Things is rapidly changing the human lives to bring convenience in domestic, public and industrial environments spanning across multiple application domains. At the same time, increasing security attacks on these networks raised alarms for timely response by forensic investigators to avoid severe consequences of the attacks. Major network forensic approaches proposed so far for IoT are based on recording and analyzing the network traffic to produce suitable evidences. One of the greatest challenges in this process is the identification and correlation of suitable artifacts among volumes of network packets to reconstruct the attack scenarios during forensic investigation. To address this challenge, we propose ProvNet-IoT, a novel provenance based forensic model for investigating network level attacks in IoT environment. The interactions between different nodes at network layer are depicted using information, functional, and event modeling techniques. We use progressive network provenance to explain different events pertaining to various attack scenarios and to provide forensically sound evidences. ProvNet-IoT is validated using two publicly available labeled IoT datasets with a corpus of different attacks. Experimental results showed the benchmark performance of ProvNet-IoT in identifying selective artifacts to produce reliable evidences during forensic investigation.

1. Introduction

The presence of Internet of Things (IoT) is growing every day and it is estimated that the world will be overwhelmed with number of IoT devices reaching the mark of 75 billion by the year 2025 Nguyen et al. (2022). Advancement in IoT opened doors for several applications like Smart Homes, Smart e-Healthcare, Smart Agriculture, Smart Transportation, Smart Grids and Smart Cities. As the traditional network technologies become backbone of many IoT networks, challenges in those networks persist in IoT which make them vulnerable to common attacks Mohamad Noor and Hassan (2019) and hinder their wider adoption. In recent times, many cybercrime incidents are witnessed in IoT networks and are likely to increase in an unprecedented way. The risks may include the combination of both digital and physical threats that bring the concern to design and develop suitable forensic investigation models to tackle the criminal incidents.

Constrained nature of IoT devices pose many challenges in identifying suitable evidences for forensic analysis Khan and Salah (2018). The logs and network traffic collected from IoT devices are used during forensic investigation to produce the evidences for security incidents. It needs special expertise related to the hardware or software platforms used in devices to extract those logs. Moreover, application logs may not always reveal network level threats as they can only point to historical activities at device level and platform or application level. In this case, network traffic collected from IoT devices and other nodes in the network becomes primary source of evidence for attack attribution and reconstruction.

Network provenance can be used to represent the interactions among various nodes of IoT environment in a structured manner. It records the relationships between series of activities caused certain malicious behaviour within the system. It is being instrumental in several domains such as scientific work-flows, databases, cloud computing, wireless sensor networks Xie et al. (2016), other domains including intrusion detection and digital forensic investigation Islam et al. (2020). Packet analysis is one of the important step in any network forensic investigation including Internet of Things.

Majority of the network forensic solutions proposed for IoT environments are designed to collect various network level

* Corresponding author.
E-mail addresses: 2018rcp9083@mnit.ac.in (L. Sadineni), espilli.cse@mnit.ac.in (E.S. Pilli), rbbattula.cse@mnit.ac.in (R.B. Battula).

https://doi.org/10.1016/j.fsidi.2022.301441

features from the attack traffic and apply machine learning (ML) techniques on them to trace the abnormal events. Some recent works aimed to leverage the graph based approach to classify the node behaviour in computer networks Zola et al. (2022). Our work aims to add provenance as a new dimension to the network graphs to facilitate reliable forensic analysis in IoT networks with following contributions.

- ProvNet-IoT, a novel provenance model is proposed to facilitate reliable forensic analysis of network-layer attacks in Internet of Things.
- ProvNet-IoT is implemented using various opensoure tools and tested against two publicly available datasets Edge-IIoT Dataset Ferrag et al. (2022) and IoT Network Intrusion Dataset Kang et al. (2019).
- Evaluation of ProvNet-IoT is done against both the datasets for which provenance is generated. Also, provenance graphs are statistically analyzed to identify the patterns and list of features. Finally, provenance graph is queried to extract the attack graph and correlated with the artifacts to produce reliable forensic evidences.

The remainder of the paper is organized as follows. Section 2 discusses some recent works on IoT forensics leveraging the application of provenance and graph technologies. Section 3 presents the architecture and theoretical details of the proposed ProvNet-IoT model. Section 4 discusses the experimental setup and description of two publicly available IoT datasets used in this study followed by a proof-of-concept implementation of ProvNet-IoT. Section 5 presents the experimental results. Section 6 outlines the merits and limitations of ProvNet-IoT. Last section 7 gives the conclusion and future work.

2. Related work

This section reviews the state-of-the-art literature on IoT network forensics and draws an incremental line to connect the developments in their approaches. Internet of Things Forensics has seen wide interest from the research community in recent years. Most of the early works in this field are confined to theoretical models Zawoad and Hasan (2015); Perumal et al. (2015); Kebande and Ray (2016); Meffert et al. (2017); Sadineni et al. (2019); Akatyev and James (2019) and very limited practical works are proposed for specific IoT environments Oriwoh and Sant (2013); Zia et al. (2017); Babun et al. (2018).

Wang et al. (2018) proposed ProvThings, a platform-centric approach to audit the application level activities in IoT networks using a centralized approach. Provenance is used as a means to attribute malicious behaviors to respective sources and to provide active lineage-based authorization of activities to prevent attacks. They used source code instrumentation mechanism to build the system.

Using an approach similar to provenance, Nieto (2020) proposed JUDAS,[1] a methodology and tool to correlate JSON logs and network traffic collected from IoT environment for defining the context of digital investigation. In a similar approach proposed by Tok et al. (2020) a tool, STITCHER,[2] is developed aiming to assist investigators in mitigating challenges they face in IoT digital forensics. STITCHER performs classification, processing and correlation of IoT forensic evidences from different sources like, firmware image, network captures and system processes.

Koroniotis et al. (2020) proposed Particle Deep Framework (PDF), a novel network forensic framework for Internet of Things based on based on Particle Swarm Optimization (PSO) and Multilayer Perception (MLP) deep learning algorithms. PDF is evaluated against Bot-IoT Koroniotis et al. (2019) and UNSW-NB15 Moustafa and Slay (2015) datasets to show it's better performance in terms of 0.999 detection accuracy and processing speed of 14,762 records per second.

Pluskal et al. (2020) developed Netfox Detective,[3] a comprehensive open-source network forensic analysis tool (NFAT) for computer network forensics. The tool is designed using modular approach which enables it to be extended to include new features an future enhancements. It offers many basic and advanced features that include support for a large number of protocols including GSE, SIP fraud analysis and web page reconstruction. Authors also presented an expected list of properties for network forensic tools.

Wu et al. (2021) examined the network traffic of 32 consumer IoT devices to derive some conclusions about the vulnerabilities still exist in IoT devices and their communications. They provided IoT Network Analyzer tool,[4] a prototype implementation that can be used by investigators to analyze network captures.

Nguyen et al. (2022) presented a novel method for detecting botnets in IoT networks using Printable String Information Graph (PSI-graph) based approach with statically and dynamically generated features. Authors evaluated their approach against IoT botnet samples collected through IoTPOT and VirusShare. Benign samples were collected from OpenWRT. Proposed method achieved 98.1% accuracy of malware detection and 91.99% accuracy for classification.

Zola et al. (2022) proposed a threefold graph-based approach for node behavior classification in a network with temporal dissection and data-level preprocessing. Authors have used UNSW-NB15 dataset to evaluate their approach which uses two novel graph data-level preprocessing techniques such as R-hybrid and SM-hybrid to exploit the most relevant graph substructures. Comparison is made between Neural Network (NN) and two Graph Convolutional Network (GCN) approaches among which NN performed well while detecting malicious actor nodes in network traffic data.

Some of the existing solutions address security and forensics challenges in IoT with different approaches employing network provenance and graph based technologies. This paper aims to propose a holistic solution ProvNet-IoT, inspired from above approaches while introducing novel methodologies focusing on the reliable forensic analysis of IoT network traffic.

3. ProvNet-IoT

While there are multiple approaches possible as discussed from the literature to address the problem of forensic analysis in IoT networks, this article focus on following research questions to formulate the problem and derive the proposed methodology.

- Does provenance help investigators to understand the network traffic better and how to model it?
- How can we unify the network provenance and graph technologies to process and derive reliable conclusions?
- How can we derive some novel features from the network provenance graphs to detect the presence of attacks?
- How those identified features help investigators to query the provenance graph to extract the attack subgraph?

[1] https://github.com/cadirneca/judas.
[2] https://github.com/poppopretn/Stitcher.
[3] https://github.com/nesfit/NetfoxDetective.
[4] https://github.com/Dyvels/IoTAnalyzer.

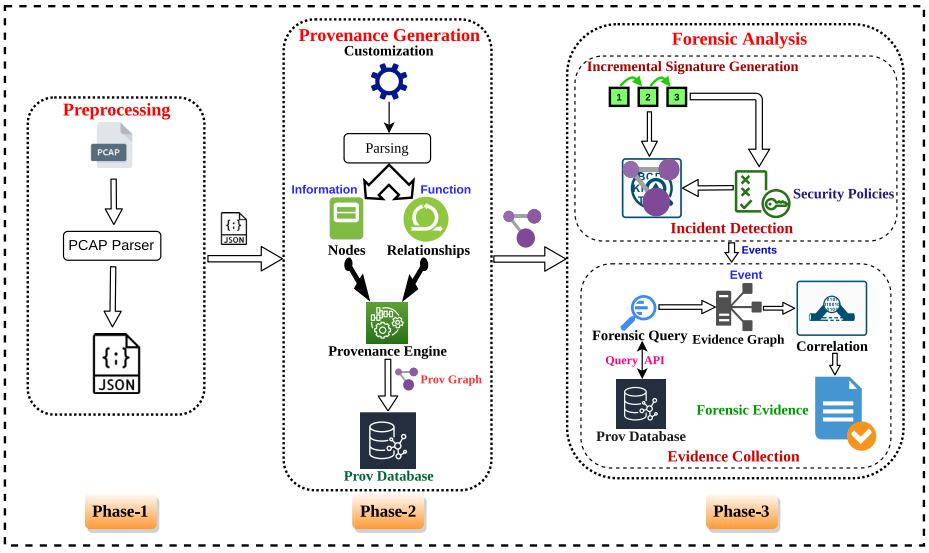

Fig. 1. ProvNet-IoT architecture.

- How the attack graph queried from database can be correlated with actual artifacts collected form network to produce reliable forensic evidences admissible in a court of law?

This section presents a novel model ProvNet-IoT, to answer above research questions and Fig. 1 shows it's architecture. ProvNet-IoT is designed to be dynamic and configurable to offer flexible forensic solution for IoT networks. It processes the network traffic and transforms that into provenance graph. As shown in Table 1, ProvNet-IoT attempts to model the network level provenance based on three techniques, namely information modeling, functional modeling and event modeling. Table also lists the standard PROV[5] concepts which can be mapped to network concepts in third column. Later, generated provenance is processed to perform incident detection and evidence collection as part of forensic investigation. The methodology for ProvNet-IoT is divided into three phases as shown in Fig. 1, namely preprocessing, provenance generation, and forensic analysis. Following sections 3.1, 3.2 and 3.3 explain these phases in detail.

3.1. Preprocessing

During this phase, the artifacts which are furnished to forensic investigators are processed and they are assumed to be collected from the target IoT network in a forensically sound manner. How these artifacts are collected from the network, is not in the scope of this study. As part of this study, we consider the artifacts to be network captures recorded in pcap format. A module PCAP parser, converts the input pcap files into equivalent JSON files which represent the arrays of packet objects.

3.2. Provenance generation

In this phase, JSON artifacts are parsed to identify various components of the provenance model. Fig. 2 shows a sample information and functional model for a typical IoT network. Address and protocol properties data is omitted from the image for simplicity. All nodes in the graph depict either a source or destination and edges represent the type of communication between them. JSON data is parsed and all the unique nodes and set of communications among them are identified separately.

At information level, the attributes of the network nodes and their state changes are represented. It also includes the application data generated and consumed by the nodes. At functional level, interactions between the nodes done by transferring single or streams of packets are represented. Customization is made at protocol level in order to specify what properties of nodes and communications should be included as part of network provenance. Later, this information is passed on to the provenance engine module which converts them into a graph structure and store in a secure provenance database.

3.2.1. Provenance engine

It offers various functionalities to manage the provenance information such as storage, modification, querying, and validation of provenance graphs. It validates the provenance against certain policies to identify various network events. Provenance engine also provides necessary functionalities to transform and store the provenance information in a secure provenance database and runs queries on it to extract different features. It offers a high level API which is built on top of native database query language to develop various functionalities for forensic analysis on network graph.

3.2.2. Prov database

It is a system where provenance information can be stored securely. It is implemented using a graph database as it offers a natural way to store and query provenance graphs.

3.3. Forensic analysis

In this phase, provenance information is processed and forensically analyzed through different steps as described below to detect the security incidents and produce forensically sound evidences.

3.3.1. Incident detection

Accuracy of intrusion detection systems play crucial role in timely mitigation of security risks or first response in post incident forensic investigation. This study proposes a passive incident

[5] https://www.w3.org/TR/2013/REC-prov-dm-20130430/.

Table 1
Map of network concepts to PROV and modeling concepts.

Network Concept	Description	PROV Concept	Model
Node	Any node in the network with a resolved name, physical address and network address along with any other relevant properties to be added while processing the provenance to generate signatures	Agent	INFORMATION
Communication	Any network packet that represents an interaction between a source and destination nodes. *e.g.*, ARP, TCP, UDP and Streams	Activity, Entity	FUNCTION
Attack	Set of network nodes and communications that represent a malicious activity along with relevant features used to identify that behavior. *e.g.*, ARP spoofing, DoS attack	Event, Entity	EVENT

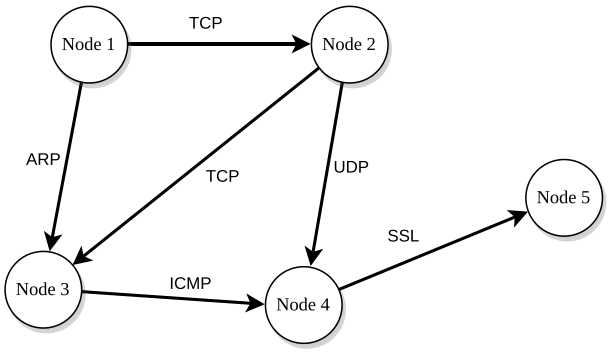

Fig. 2. Sample modeling of network communication.

detection mechanism which is triggered during post attack investigation process. Following steps define the methodology for signatures and policy based incident detection.

Algorithm 1. Incremental Signature Generation

Input: ProvGraph, Signatures
foreach $signature \in Signatures$ **do**
  foreach $timeWindow \in signature.timeWindows$ **do**
    $result \leftarrow ProvGraph.match(signature, timeWindow)$
    $id \leftarrow result.id$
    $keys \leftarrow signature.properties$
    foreach $key \in keys$ **do**
      $value \leftarrow result.getValue(key)$
      $ProvGraph.updateProperty(id, key, value)$
return;

Incremental Signature Generation step starts with the generation of various attack signatures from provenance graph by applying relevant statistical and machine learning techniques. Later these signatures are transformed into node level properties in an incremental process for different time windows using following algorithm 1. This algorithm is called whenever there is a change in either signatures or time windows. To implement this functionality, we propose a simple Signature Definition Language (SDL) based on Cypher patterns as specified in listing 1. In SDL, signature-type specifies two levels either node or relationship. Patterns are represented as cypher queries which are executed against the network graph to generate aggregate information. Signatures corresponding to all possible attacks on the network are defined in a simple JSON like format using SDL. Later, these signatures are executed on provenance graph to extract corresponding node or relationship level properties. Query is modified on every iteration to embed time constraint to filter the graph data. Apart from these signature based features, different other network level features are also collected from the provenance graph and those details are given in section 5.3.

Listing 1. Signature Definition Language Syntax

```
{
"signature" : {
"type" : "signature-type",
"patterns" : ["pattern-1", "pattern-2",
......, "pattern-n"],
"properties" : ["property-1", "property-2"
......, "property-n"],
"timeWindows" : [w1, w2, w3, ...., wn]
}
}
```

Security Policies are a set of rules identified by the domain experts for different security attacks against which provenance graph is validated. Apart from original attack signatures, these policies may also be designed based on node level properties generated incrementally in previous step. A typical policy sets a threshold for different properties of network graph to identify malicious nodes and respective properties.

Once all the signatures are processed and policies are defined, following algorithm 2 is run to detect the presence of attacks in the network graph. Incident detection algorithm classifies the nodes based on security rules and returns the list of malicious events and their respective properties. At event level, an abstract view of malicious activities identified by enforcing the security policies are represented. Every event includes its own details such as name, event type, risk level and other relevant information along with a set of activities that caused the event.

Algorithm 2. Incident Detection

Input: ProvGraph, SecurityPolicies
$events \leftarrow []$
foreach $policy \in SecurityPolicies$ **do**
  $result \leftarrow ProvGraph.enforce(policy)$
  $events.append(result.events)$
return events;

3.3.2. Investigation

As part of this step, investigators receive the outcome of incident detection i. e a set of malicious events detected from the network and perform forensic investigation on them.

Forensic Queries are prepared for each event based on the information contained in event object. That information includes the details about both malicious and victim nodes that are part of the attack and series of communications among them.

Query API is a functionality offered by provenance engine and it translates the forensic queries into native graph database commands to retrieve the attack graph which represents a security event.

Evidence Graph models a security attack in a graphical form which include the lineage of activities in the network that lead to a specific attack state and also point to corresponding adversary and victim nodes.

Correlation is the process of matching the investigation outcome with original artifacts to ensure the reliability and integrity of both evidences and investigation process.

Forensic Evidence includes the final report of the investigation and set of artifacts linked with the event during correlation which are admissible in a court of law.

4. Implementation

This section discusses the proof-of-concept implementation of ProvNet-IoT to validate the proposed methodology. Following sections present the experimental setup, two publicly available datasets to benchmark the performance of ProvNet-IoT and list of various open-source tools used during implementation.

4.1. Experimental setup

Implementation and experiments are conducted on a desktop which runs Ubuntu 20.04 operating system and with Intel(R) Core™i9-10900K CPU @ 3.70 GHz × 20, 62.5 GiB RAM, and 1 TB SSD. All functional implementation is done using Spring Boot[6] framework developed for Java platform and other open-source technologies such as Neo4j,[7] Cypher[8] and Spring for GraphQL.[9]

4.2. Datasets overview and attack selection

To validate the proposed model, we used two publicly available datasets, Edge-IIoT Dataset and IoT Network Intrusion Dataset. The list of attacks performed in each dataset and selected attacks for evaluation are listed in Table 2. Both datasets include 15 and 8 different attacks on real IoT devices used in a typical IIoT and smart home scenarios. Among them, only those 14 attacks are included in this study which are targeted on network layer.

4.3. Proof-of-concept

PCAP parser is developed on top of tshark tool which is invoked using Java Native Interface (JNI). JSON parsing is done using Jackson library to identify the nodes and relationships in the network traffic. A node is uniquely identified with the combination of both it's physical and network addresses. Any node that changes it's IP addresses is represented multiple times in the network. Later, information related to these changes are aggregated and added as node properties.

All the set of communications between a pair of nodes are recorded as relationships identified uniquely with the combinations of source, destination addresses and time epoch of network packet representing that communication. Both nodes and relationships have a label and properties list. Fig. 3 present the transformed json into nodes and relationships (properties are omitted for simplicity). Customization is made using configuration files which represent the list of allowed properties to be persisted into provenance graph database for each communication protocol. Neo4J graph database is used to store provenance information.

Provenance Engine is implemented using GraphQL library on top of repository service provided as part of Spring Data Neo4j (SDN)[10] project. We also defined custom repositories to facilitate the execution of queries built for signature generation, policy enforcement and evidence graph extraction.

Initial signatures are generated from provenance graph after performing statistical analysis using by graph data science library offered by Neo4j.[11] Later, those signatures are defined using the simple SDL proposed in section 3 for all the attacks being investigated. Sample signature for ARP spoofing attack is given in listing 1. Different security policies and their relevancy with various features to identify the attacks are presented in section 5. Listing 3 shows a sample query to fetch DDoS attack graph based on event information gathered during incident detection. Forensic query returns the list of victim and attacking nodes along with the communications between respective pair of nodes. Communication properties include all date and time information of attack traffic which reveals the duration of attack as well.

Listing 2. Signature to Identify ARP Spoofing Attack

```
{
"signature" : {
"type" : "relationship",
"patterns" : [" MATCH (src:Node)-[com:arp]->()
WHERE com.properties.layers.arp.opcode = 2"
AND com.properties.layers.arp.src.ip !=
src.ip.addr
RETURN src, count(*) as spoofing_attempts],
"properties" : ["src", "spoofing_attempts"],
"timeWindows" : [5, 10, 30, 60] }
}
```

Listing 3. Forensic Query to fetch DDoS attack graph

```
query fetchDDoSAttackGraph($event:Event!) {
  victims(ids:$event.victim_ids!) {
    name,
    mac,
    ip,
    hitRate
  }

  attackers(ids:$event.attacker_ids!) {
    name,
    mac,
    ip,
    location
  }

  traffic(ids:$event.traffic_ids!) {
    label,
    properties
  }
}
```

5. Experimental results

In this section, we present various experimental results to benchmark the performance of ProvNet-IoT.

5.1. Dataset size

Datasets used in experimental works include various attacks as listed in Table 2. Following Fig. 4 shows the size of datasets in different formats like original Pcap files, converted Json files and provenance stored in Json format. It is observed that the size of provenance is smaller corresponding original dataset, because through customization only selected features are included to be part of provenance.

5.2. Provenance metrics

Following Fig. 5 presents node and relationship metrics related to network provenance generated from datasets. Metrics are given on logarithmic scale as number (in thousands) of nodes and relationships under different attack scenarios. Scenarios are defined based on the categories of attacks listed in Table 2 where scenarios 1 to 4 are for Dos/DDoS/Mirai attacks, information gathering

[6] https://spring.io/projects/spring-boot.
[7] https://neo4j.com/.
[8] https://neo4j.com/developer/cypher/.
[9] https://spring.io/projects/spring-graphql.
[10] https://spring.io/projects/spring-data-neo4j.
[11] https://neo4j.com/product/graph-data-science/.