Towards Reproducible Ransomware Analysis

Shozab Hussain, Musa Waseem, Turyal Neeshat, Rja Batool, Omer Ahmed, Fareed Zaffar
LUMS

Ashish Gehani, Andy Poggio, Maneesh Yadav
SRI

ABSTRACT

Ransomware attacks continue to be a prominent cybersecurity threat and the subject of considerable research activity. Despite frequent high profile public reports of ransomware attacks, we found a paucity of tangible open behavioral activity data for large collections of real world ransomware binaries. The lack of such open datasets introduces barriers to research that may otherwise lead to innovative approaches to ransomware mitigation. We have constructed a dataset of ransomware activity logs and corresponding provenance graphs. They are derived from the sandboxed execution of all ransomware-tagged binaries in the widely-known MalwareBazaar. We also provide the code for orchestrating the log collection and provenance inference steps. The aim is to enable other researchers to customize and extend it for their analyses. We hope that the dataset will facilitate the discovery of innovative and effective ransomware mitigation strategies.

CCS CONCEPTS

• **Security and privacy → Malware and its mitigation**.

KEYWORDS

ransomware, provenance, machine learning, open data

ACM Reference Format:
Shozab Hussain, Musa Waseem, Turyal Neeshat, Rja Batool, Omer Ahmed, Fareed Zaffar and Ashish Gehani, Andy Poggio, Maneesh Yadav. 2023. Towards Reproducible Ransomware Analysis. In *2023 Cyber Security Experimentation and Test Workshop (CSET 2023), August 07–08, 2023, Marina del Rey, CA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3607505.3607510

1 INTRODUCTION

It is difficult to overstate the significance of ransomware attacks in recent years. At the moment of writing, high profile ransomware news stories are easily found: the large pharmaceutical company, Merck, has won a lawsuit against its insurers to cover $1.4 billion in losses related to a NotPetya attack [20]; the Federal Bureau of Investigation (FBI) ceased negotiations with a ransomware gang that attacked municipal systems in Oakland, California [2]; recovery from a ransomware attack on critical municipal government infrastructure in Dallas, Texas is projected to take months [4].

CSET 2023, August 07–08, 2023, Marina del Rey, CA, USA

ACM ISBN 979-8-4007-0788-9/23/08…$15.00
https://doi.org/10.1145/3607505.3607510

Ransomware has achieved an unprecedented scale of prominence and, perhaps, damage. While it is quite clear that fortifying systems against these threats is difficult, researchers interested in mitigation approaches have little in the way of easily available tools and datasets for experiments and analysis.

We present the Ransomware Execution PROvenance Dataset (REPROD), a collection of 933 distinct executions of ransomware-tagged binaries available in the MalwareBazaar[3] database. Our aim is to facilitate reproducible ransomware analysis and subsequent mitigation research. The system activity logs of a Microsoft Windows sandbox virtual machine environment running the binaries were transformed (via SPADE [13]) into provenance graphs in the Open Provenance Model (OPM). These provenance graphs represent agents (e.g., the user), processes, and artifacts (e.g., files) as vertices, while events and relations between them are represented by edges. This representation allows users to craft queries about ransomware execution, including what files were created, modified, or transformed by a given process to understand dependencies and impacts.

Contribution: The primary benefit of our work is an open dataset that provides a detailed provenance representation of the execution of many ransomware-tagged binaries. Further, we describe a simple, reusable automated process to produce such datasets. The source code for this process that was used is also provided. (See Section 8.) The approach utilized for the collection of this dataset does not rely on custom triggering of ransomware execution or mitigation of virtual machine detection, making it widely usable. This type of data can help researchers gain initial insights into how the collective ransomware ecosystem is operating. It can also aid in identifying common patterns of execution that can subsequently be leveraged for attack mitigation research.

2 MOTIVATION

The significant body of earlier related research [21] should have facilitated effective mitigation systems against ransomware. The continued prominence of attacks suggests that such systems are not available yet. We could not find any mention of datasets for researchers to download and apply in their own efforts. The difficulty that researchers have in obtaining such data may contribute to the extant vulnerability of large critical systems to ransomware attacks. Further, the Cuckoo Sandbox [1] environment that was widely employed for malware analysis in the past is no longer supported or usable on modern systems.

There have been multiple efforts that broadly touch on different aspects of the dataset that we present here. Three are of particular note:

- The UNVEIL [18] system is from 2016 and built with the Cuckoo Sandbox framework. The paper claims to be able to achieve a 96.3% true positive and 0% false positive rate for ransomware binary classification across ransomware attack

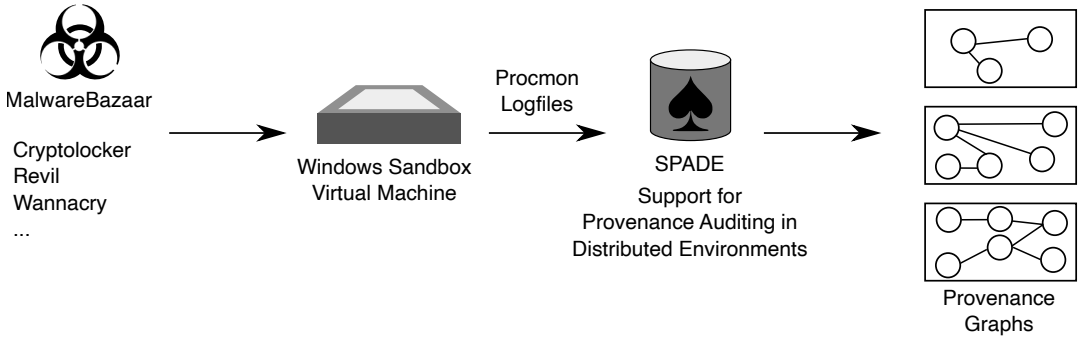

Figure 1: The REPROD workflow starts with collecting a set of ransomware binaries. These are then individually run with Windows Process Monitor active. The resulting logs are used to infere data provenance graphs. The set of logs and graphs consitute the dataset.

types (including encryption and screenlocker variants). An accompanying dataset was listed as available to researchers. However, we were unable to obtain it despite repeated requests.

- The second is RanSAP[16], which provides a dataset of storage access patterns – that is, raw read and write calls – with entropy measurements for seven well-known ransomware samples and five benign programs. While this dataset could be used in the context of low-level hardware research, it does not contain any high-level information, such as filenames or system events, that would be needed for typical security research. Though the dataset is open, no accompanying code is provided.
- PEELER [6] is a more recent system, having been published in 2021. It represents the state-of-the-art in terms of reported efficacy and efficiency. It uses a combination of detection rules and data mining. These are applied to data collected from specific system calls. The authors report detection of encryption activity within 115 milliseconds in 70% of their ransomware samples. Neither code nor resulting execution traces were published.

Consequently, a researcher interested in trying out a new detection approach cannot (i) use extant datasets, or (ii) even use any provided code to generate such data.

In contrast to earlier approaches that focused on the lower abstraction level of system calls, more recent research in malware analysis [7, 17] utilizes a richer abstraction – *data provenance* that is inferred from the combination of system call event sequences, operating system internals, and causal models that provide abstract connections between the monitored elements. This direction for ransomware analysis has previously been considered by other researchers [19]. Motivated by the need for open data, we have collected and are sharing both system activity logs and inferred data provenance graphs of a large collection of ransomware samples.

3 REPRODUCIBILITY CHALLENGE

Experimental analysis within the field of cybersecurity spans a wide set of topics, ranging from empirical evaluation of theoretical primitives (such as cryptographic ciphers or differential privacy algorithms) to the study of deployed malware detection systems. A researcher that creates a new approach to classify, detect, mitigate, or respond to a threat or class of attacks is faced with the challenge of how to evaluate the efficacy of their idea. Creating an end-to-end implementation involves significant effort:

(1) They need to consider multiple platforms to decide which one to target.
(2) A realistic benign workload may need to be deployed.
(3) Their approach may require custom system-level instrumentation as well as application-level processing of the resulting records.
(4) Metrics need to be defined and realized to be able to understand the utility of the defenses they build.

This effort is compounded further if they need to compare their system to the previous state-of-the-art, of which there may be no publicly available version.

Ransomware analysis brings with it a further complication. In the case of traditional malware analysis, scoping the side-effects is relatively tractable – that is, the environment in which the malware runs is sandboxed; its ability to communicate externally is tightly controlled, with incremental relaxation of firewall rules used to coax interesting behavior while managing risk. In contrast, in the case of ransomware the expected result is highly destructive behavior, such as locking the analyst out of the machine or disrupting subsequent forensic analysis through the encryption of numerous files.

4 PILOT ANALYSIS

We anticipated that significant development effort and computational resources would need to be invested to achieve our aim. At a high-level, the goal is the creation of a dataset that would be useful for ransomware analysis. To this end, we decided to first perform a pilot study on the type of data that we anticipated collecting at scale later. Such data would consist of execution traces and data provenance inferred from the resulting logs. If our study succeeded, we would have increased confidence that the range of information we planned to report would be useful for others. Intermediate failure would present us an opportunity to adjust the definition of our target dataset before embarking on the larger effort.

Signature:	Jigsaw	WannaCry	CryptoLocker	Troldesh	Dharma	GlobeImposter	Lockdown	Nefilim	Rapid	Hive	Phobos	REvil	Sugar	Thanos
Dataset:														
Jigsaw	1770	0	0	0	0	0	0	0	0	0	0	0	0	0
WannaCry-1	0	348	0	0	0	0	0	0	0	0	0	0	0	0
WannaCry-2	0	1811	0	0	0	0	0	0	0	0	0	0	0	0
CryptoLocker-1	0	0	256	0	0	0	4	0	0	0	0	0	0	0
CryptoLocker-2	0	0	256	0	0	0	4	0	0	0	0	0	0	0
Troldesh	0	0	0	1799	0	0	0	1094	0	0	0	0	0	0
Dharma-1	0	0	0	0	15953	0	0	0	0	0	0	0	0	0
Dharma-2	0	0	0	0	1536	0	0	0	0	0	0	0	0	0
GlobeImposter	0	0	0	0	0	60	0	0	0	0	0	0	0	0
GlobeImposter-2	0	0	0	0	0	57	0	0	0	0	0	0	0	0
Lockdown	0	0	0	0	0	0	41	0	0	0	0	0	0	0
Nefilim-1	0	0	0	0	0	0	0	4486	0	0	0	0	0	0
Nefilim-2	0	0	0	0	0	0	0	13	0	0	0	0	0	0
Rapid-1	0	0	0	0	0	0	0	0	1420	0	0	0	0	0
Rapid-2	0	0	0	0	0	0	0	0	223	0	0	0	0	0
Hive-1	0	0	0	0	0	0	0	966	0	3260	0	2	0	0
Hive-2	0	0	0	0	0	0	0	57	0	466	0	1	0	0
Phobos	0	0	0	15	0	0	0	0	0	0	3203	0	0	0
Revil-1	0	0	0	0	0	0	0	9	0	0	0	77	0	0
Revil-2	0	0	0	0	0	0	0	0	0	0	0	109	0	0
Sugar-1	0	0	0	0	0	0	0	0	0	0	0	0	1194	0
Sugar-2	0	0	0	0	0	0	0	0	0	0	0	0	1263	0
Thanos-1	0	0	0	0	0	0	0	0	0	0	0	0	0	74
Thanos-2	0	0	0	0	0	0	0	0	0	0	0	0	0	105
Benign-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Benign-2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Benign-3	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 1: Reported here are results of a pilot study performed to validate the type of data being collected. One or two executions of each of 14 ransomware binaries and three benign executions were performed. These represent the rows in the table. For each ransomware binary, a custom signature was created. These represent the columns of the table. Each binary execution gives rise to a collection of subsequences. These are matched against all the signatures. A cell entry represents the number of subsequences from a binary's execution that matched a particular signature.

Initially, we employed a manual version of the collection process that was subsequently automated (as described in Section 6.2). Specifically, a collection of 14 ransomware-tagged binaries with the following names was selected: Jigsaw, WannaCry, CryptoLocker, Troldesh, Dharma, GlobeImposter, Lockdown, Nefilim, Rapid, Hive, Phobos, REvil, Sugar, and Thanos. Additionally, a benign binary was selected. Each binary was run once or twice (except the benign, which was run thrice), with an activity log collected each time. From this log, a provenance graph was inferred. Since filesystem interaction is a key aspect of ransomware activity, we derived an abstraction of each graph that focused on the file metadata and content read and write operations.

Using the focused provenance, we then manually constructed detection signatures for each of the ransomware binaries. These signatures were articulated in a regular-expression-based language, similar to how a commercial anti-virus product would do so (to minimize the chance of false positives). Next, each signature was run over the sliding windows of the focused provenance from all ransomware and binary executions. The results are shown in Table 1. We can see that in most cases, the signature matched windows from provenance of the corresponding binary. The one exception is the Nefilim signature which resulted in a number of false positive matches with windows from provenance of other binaries – specifically, Troldesh and Hive. We note that none of the signatures matched windows from any of the benign traces.

Based on these results, we opted to proceed with the next stage – that is, constructing the larger dataset.

5 APPROACH EMPLOYED

A concrete construction of each of the elements needed to create a fielded ransomware detector has certain advantages. The obvious benefit is that it gives future researchers a basis for rapid prototyping. This is our motivation for selecting the components that we have.

Platform: Ransomware has historically been developed for and deployed on various versions of Microsoft's Windows operating system. This introduces multiple challenges:

(1) Typically, a research prototype is developed on one version. When comparing multiple detection systems, they may utilize diverse low-level instrumentation that target different APIs (as these change between operating system versions).
(2) Windows has a number of subsystems from which different types of events can be captured. Examples of these are Windows Management Instrumentation (WMI), Event Tracing for Windows (ETW), and custom filesystem filter drivers. Tools may use different subsets of such information.
(3) Recent versions of Windows require low-level instrumentation, such as filesystem drivers, to be signed with a cryptographic signing key obtained from Microsoft. This raises the bar for researchers to implement such logging (or even reuse code from other researchers, since it necessarily will not include a signing key).

To address these concerns, we utilize event logs collected by Process Monitor (also known as ProcMon). This is a utility program that was originally developed by Sysinternals. Since Microsoft acquired it in 2006, the tool has continued to be updated and freely

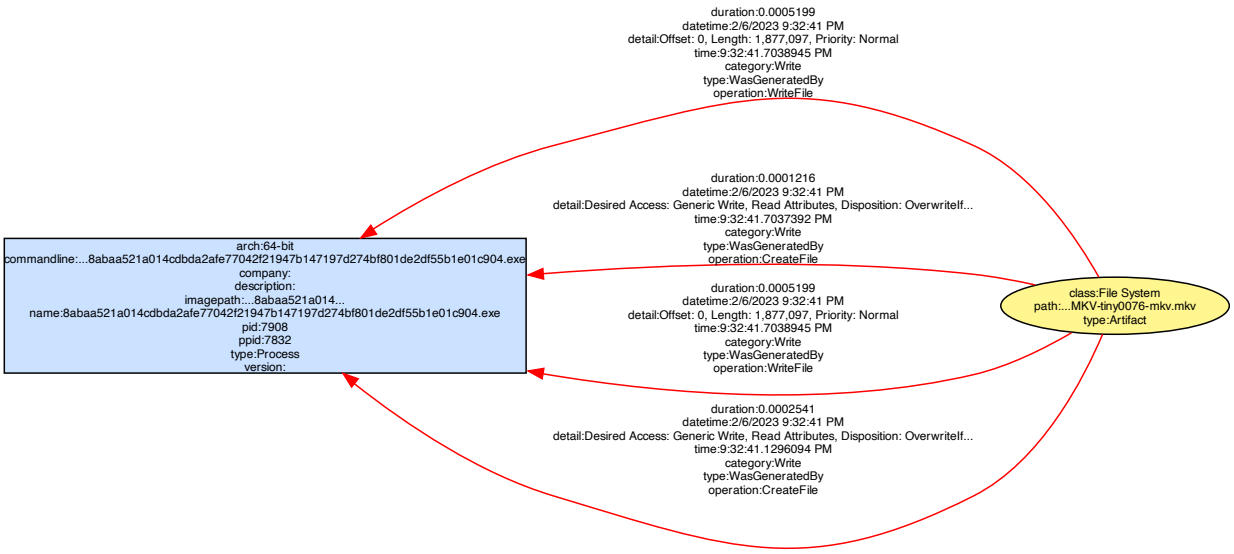

Figure 2: A small section of a DOT file in REPROD is visualized with Graphviz [11]. It is from ransomware activity of the `8abaa521a01...` binary. The blue rectangle denotes the ransomware process. The yellow ellipse depicts the video file that it is operating upon. The sequence of writes are shown with multiple red arrows. (Some annotations were dropped for brevity.)

distributed. The utility records a range of low-level events and data, abstracts it into a higher-level representation, and allows developers to configure the set of activity of interest. All necessary code signing is handled by the tool's authors, sparing developers from the burden.

Population: The REPROD workflow started with a fresh installation of the Windows operating system in a virtual machine. We opted to *populate* the filesystem with a collection of "honeypot" directories and files (drawn from the open NapierOne dataset [9]) for multiple reasons:

(1) The presence of a collection of a user files gives the ransomware a non-trivial workload to operate on. This may keep it operational for enough interesting activity to be manifested. This will in turn improve the quality of behavioral signatures that can be extracted.
(2) These inserted files can serve as *sentinels*. In particular, downstream analysis can utilize the presence of these files in unmodified form to determine whether the ransomware reached parts of the filesystem. Similarly, the state of particular types of files provides insight into what the ransomware was targeting.
(3) Malware, in general, and ransomware, in particular, may scan the environment before proceeding with its agenda. This functionality is implemented to combat defensive technologies that may disrupt its operation. The absence of user files may be interpreted as a host not worth targeting and trigger suicide logic.

Provenance: In recent years, dynamic malware analysis has been increasingly performed using data provenance graphs. The advantage of utilizing this representation is that it relates the agents, processes, and artifacts on a host, even if the connections span long periods of time. In contrast, analyses that operate on event logs directly will typically using a sliding window that limits the context utilized. For example, DARPA's 2015-2019 Transparent Computing program used streaming provenance to detecting Advanced Persistent Threats, a stealthy class of malware that is typically created by well-resourced adversaries, such as nation states. Analyses based on provenance have moved past the research stage into practical deployments now. In 2023, DistDet [10] has been deployed by 50 customers on over 20,000 hosts.

ProcMon stitches together internal Windows details into abstract events, such as the read of a file or a query of the registry for the value of a key. However, its output is a still a stream of operations emitted in the temporal order in which they occur. Dependencies between elements are not explicitly chained, precluding system-wide root-cause or impact analysis.

To compute data provenance graphs from ProcMon logs, we opted to use SPADE [13]. See Figure 2 for a minimal provenance graph and Figure 3 for a richer one. The system was selected for the following reasons:

(1) It is an open source framework, that is relatively well-documented with academic publications describing its functionality and a Wiki that provides explanations on how to use it. This simplified the development process for us and will allow future

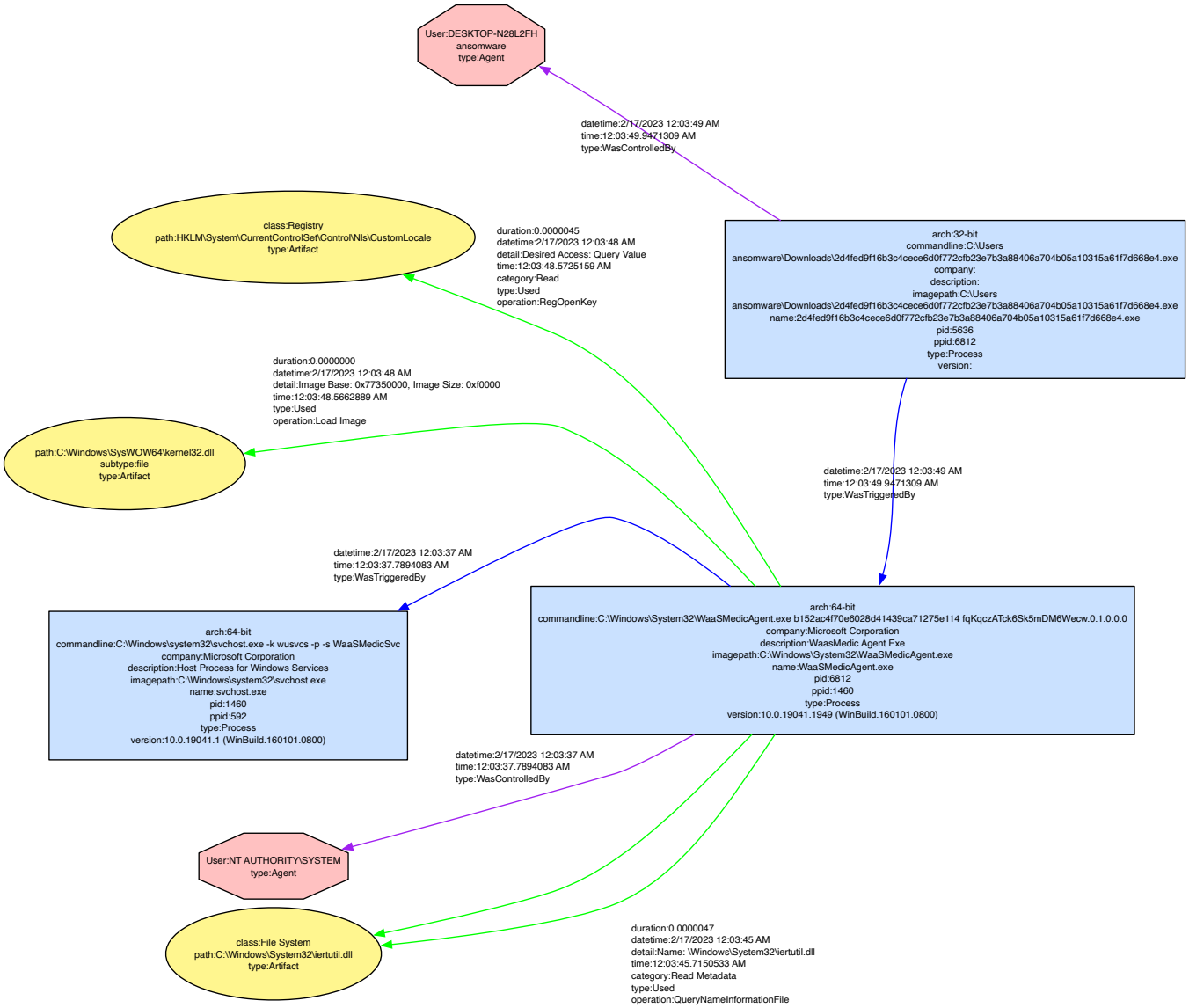

Figure 3: Data provenance provides a rich abstraction of system activity. Here two levels of ancestry of a ransomware process were extracted. Numerous artifact vertices were pruned for clarity. Accesses to single file, registry, and filesystem (yellow) vertices were kept to illustrate the type of I/O activity captured. Further, the process (blue) vertices and their relationships are seen. Finally, (red) agent vertices show the users that control the processes.

researchers to easily modify dataset construction, should they wish to do so.

(2) The modular design includes a number of backends, allowing the provenance to be stored in multiple formats. In addition to the graph representation that is natural for provenance, it also allows the data to be serialized in a tabular format (using a relational database) or the original temporal sequence. The options to generate sequence, tabular, and graph versions of the same incoming provenance data ensures that a wide range of machine learning approaches will be able to utilize it.

(3) The system has a long history. In 2022, it received ACM Middleware's 10-year Test-of-time Award [8]. We hope that this bodes well for researchers that may be interested in using it to generate comparable provenance from new ProcMon logs in the future.

6 DATA COLLECTION

6.1 Binary Selection

The overwhelming majority (i.e. 1,298) of MalwareBazaar entries with the ransomware tag were of executable (EXE) filetype. The

frequencies for the next most common types were Linux executable format ELF (87), Windows dynamic link library DLL (62), and compressed archive ZIP (59). Given the distribution, we limited the type to EXE to make the dataset uniform and internally consistent without sacrificing a significant number of samples. While our implementation targets Microsoft Windows, it can be extended to other operating systems. In particular, SPADE can infer provenance from Linux Audit and Mac OS X OpenBSM event logs as well.

A list of the 1,298 Windows ransomware binaries was constructed by downloading all entries from MalwareBazaar [3], filtering the list to select just the entries with a filetype with an "EXE" extension, and then querying MalwareBazaar though a Python API to check the tags for the presence of the string "ransomware". The MalwareBazzar API limits the query response size to a 1,000 items, necessitating this approach to construct the complete ransomware list.

6.2 Collection Workflow

An extant framework, such as Xanthus [15], could be used in principle. To accommodate Windows interactions, a custom automated process was created to perform the following steps:

(1) Download and run each binary inside a Windows virtual machine sandbox.
(2) Within the sandbox, ProcMon is also run.
(3) Each run is limited to ten minutes. At the end of a run, Density Scout is used to measure the entropy of files to determine if they were encrypted.
(4) The ProcMon log is transformed into a provenance graph using the SPADE system running in a separate virtual machine.
(5) Querying the provenance graph in SPADE [12] to produce a subgraph that corresponds to the ancestors of the process ID (PID) associated with ransomware binary.
(6) Collation of information from all the executions into a single comma-separated-value (CSV) file.

Descriptions of each component are provided below for the purpose of reproducibility. More specific details are in the REPROD repository. (See Section 8.)

6.3 Sandboxed Windows Environment

An x86-64 machine with Windows 10 installed as the operating system serves as the host. In this environment, the packages for the Python language runtime and its package manager, Pip, are installed. This is used to get the MalwareBazaar (0.1.2) Pip package, which provides programmatic access to the repository's collection of malware binary samples.

We use VirtualBox 7.0 to create an x86-64 virtual machine. It is configured with three disks:

(1) A separate local *operating system disk* is created. Microsoft Windows is installed on it – specifically, we use version 10.0.19045 Build 19045. It is located in drive `C:`. If a user wants to extend the dataset, they will need a Windows license to recreate this virtual machine with the scripts and directions we provide.
    To facilitate use of the VirtualBox's `–with_autologon` switch, its Guest Additions are also installed on this disk.
    A ProcMon configuration file is installed. This specifies the list of attributes to record for each operation that occurs on the virtual host. The resulting log can then be used for inferring data provenance (using SPADE [13]).
    Modern versions of Windows provide a number of security controls. To maximize the opportunity for the ransomware to exhibit its complete behavior, the controls are configured to be inactive. To ensure reproducibility of the results, the details of the steps taken for this are documented in the data repository.
(2) To provide the ransomware with a realistic environment in which to operate, a collection of 955 files are randomly selected from the NapierOne dataset [9]. In total, these files take about 1.2GB of disk space. This set of sentinel files are spread across various folders, including the `Desktop`, `Documents`, and `Videos` folders in the user directory on the `C:` drive, as well as in a *honeypot disk* (`E:` drive).
    A utility program DensityScout (build 45)[22] is also installed. It provides a convenient way of measuring the "density", a variant of "entropy", to help determine when a plaintext file has been transformed into ciphertext by the ransomware. (Density provides a measure of the dispersion in the histogram of bytes values in the data.)
    This Virtual Disk Image (VDI) is configured to be automatically mounted as the `E:` drive.
(3) Finally, an empty VDI is created as the `F:` drive. It serves as a *transfer zone*, where ProcMon can deposit compressed log files of activity that it collects. Subsequently, this VDI can be separately accessed from the host. This allows the logs to be (relatively) safely extracted.

To ensure the integrity of the operating system, the runtime environment, and the logs collected, pristine versions of the disks are used for each ransomware execution.

After the honeypot disk is created, the densities of files in the initial snapshot are recorded. After a run, the difference in file densities is calculated. A significant increase in density suggests the file has been encrypted. Care must be taken when selecting pre- and post-execution files to compare since the ransomware can change the name and extension of the files.

6.4 Provenance Inference Environment

To minimize the impact on the environment where ransomware instances run, we limit our changes to the ones described above. ProcMon's execution is efficient and minimally invasive, allowing logs to be collected relatively unimpeded. These are then extracted and moved to a separate virtual machine that is used for inferring data provenance graphs from the PML logs.

An x86-64 virtual machine is created and Ubuntu 18.04 is installed on it. SPADE is downloaded, configured, and built by following instructions on the project's Wiki. SPADE can be configured with a number of queryable databases. Postgres is used since it provides a SQL-queryable database.

To infer a provenance graph from a particular PML log, SPADE's state (including Postgres) is first reset. ProcMon is used to translate the log from PML to XML. The ProcMon Reporter is then configured to ingest the XML version. The Reporter module infers provenance

elements and sends them to the storage. These include Open Provenance Model `Process` vertices, including ones for the ransomware instance that is run, and `Artifact` vertices that represent elements such as files. It is worth noting that deletion events are not recorded as provenance.

The provenance graph inferred can have extraneous content for multiple reasons:

(1) Various processes may be operating in the background. Their activity is likely orthogonal to that of the ransomware.
(2) The ransomware may load standard system libraries that all other processes use as well. The provenance of such artifacts is typically not informative.
(3) As transitive dependencies are followed, they become increasingly likely to be benign.

```
%only_processes = type == 'Process'
$all_processes = $base.getVertex(%only_processes)
%rw_name = "name" like '$sha256.exe'
$rw_processes = $all_processes.getVertex(%rw_name)
$rw_activity = $base.getLineage($rw_processes, 2,
    'ancestor')
```

Figure 4: QuickGrail [12] queries for extracting two levels of ancestors of all ransomware processes originating from a specific binary (named by its hash).

Consequently, we extract provenance subgraphs rooted at the vertices of the ransomware processes and limited to two levels of ancestry. See Figure 4 for the queries used to extract such subgraphs. Once a subgraph has been extracted into a graph variable, it can be exported into Graphviz DOT format for visualization or JSON for other downstream programmatic analysis.

Alternative subgraphs can be extracted by adjusting the queries. For example, an analyst may wish to study the impact of the ransomware on the system. In this case, they can adjust the `ancestor` term in the query to `descendant`. If they wished to see three levels of descendants, they can change the parameter 2 to 3.

6.5 Results

We started with the set of all 1,298 ransomware binaries available in MalwareBazaar (as described in Section 6.1). Each was run in a fresh sandboxed Windows environment (as outlined in Section 6.3).

Ideal Execution: The best case occurs when ProcMon log generation is proceeding unimpeded at the time limit that we set for data collection (which is ten minutes). This held in 861 runs. Data provenance could be inferred in each of these instances.

Imperfect Timing: The next class of executions of interest is the set where the ransomware was triggered but log collection was disrupted. Typically, this was because the ransomware had encrypted the ProcMon log itself by time we attempted to extract it. There were 316 such instances. In each case, we were able to identify evidence of ransomware activity, such as files being renamed or the background image being altered. In the future, valid logs for this class of binaries may be retrieved by halting execution earlier.

Instrumentation Limitation: The last class consists of binaries that may not be amenable to data collection using our current methodology. It contains two subclasses. In 72 cases, there was an issue with ProcMon's termination. The side-effect is an unusable log. This issue can occur in the absence of malware. Addressing it will require alternate monitoring. In 49 instances, in addition to the log being inaccessible, inspection of the affected environment did not reveal evidence of ransomware. This may be because the environment prevented it from triggering or its effect is not detectable by our observation methodology. Further analysis will be needed to address this subset. This is facilitated by the workflow we have provided.

6.6 Potential Limitations

The approach we have adopted may have limitations, particularly in the face of the sophisticated adversaries that are presumed to be behind ransomware attacks. Techniques such as virtual machine detection or instrumentation deactivation could be used by ransomware authors to impair the workflow we have presented. There was no observable evidence of ransomware activity in 497 of the 861 "ideal execution" runs – that is, no changes in density or anomalous screenshots were observed. Multiple explanations exist – the ransomware may have detected monitoring, only affected non-sentinel files, or not run long enough. Analysis of the root cause is outside the scope of the current effort.

7 DATA OVERVIEW

Graph size	Vertex count	Edge count
Minimum	24	54
Lower-hinge	467	6,192
Median	1,042	23,125
Upper-hinge	13,477	216,729
Maximum	349,064	7,766,739

Table 2: The inferred provenance graphs vary in size. The Tukey 5-five number summary is reported here to provide an idea of the scale.

Two types of ransomware predominate. The first utilizes encryption to lock files on an infected host. The effect of this type of ransomware's execution can be detected by the change in density of sentinel files in the honeypot collection. The second locks the screen on the infected host. To detect this, first a screenshot of the target system is taken; next the ransomware instance is executed; then another screenshot is taken at the execution time limit (set to ten minutes). The two screenshots are compared to identify anomalous screen activity.

Recall that 861 executions resulted in a usable ProcMon log. In 300 runs, more than 20,000 file operations were observed. In 241 of these, the density of the sentinel files in the honeypot collection changed. This is expected since ransomware that performs encryption typically does so by operating upon files.

In total, 279 runs resulted in files with density changes in the sentinel files. In 41 runs, files with density changes could not be read after the ransomware execution, indicating they had been

encrypted. 224 executions produced anomalous screen activity. In 98 instances, both changes in file density and anomalous screenshots were observed.

Given the 861 ProcMon logs, we inferred a corresponding number of provenance graphs. There is a substantial spread in the size of these graphs, even though they all derive from ten minute executions and extraction of two levels of ancestry rooted at the ransomware process. To illustrate the spread in sizes, we report the Tukey 5-number summary [5] in Table 2. The statistic provides a concise description by reporting the minimum, first to third quartiles, and maximum number of vertices and edges in the collection of graphs.

Ransomware activity patterns are diverse. Even if the focus is on I/O-related calls, the temporal behavior differs significantly depending on the specific binary being run. The heterogeneity is exhibited with respect to multiple dimensions, including the length of time for which activity persists as well as the average and peak rate of events. This makes it challenging to design a detection system, motivating the need for a diverse enough set of samples to study. REPROD aims to provide this.

Sample Temporal Analysis

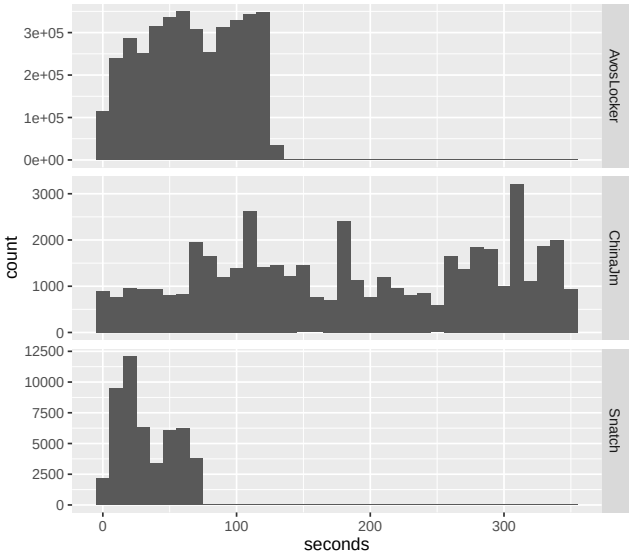

Figure 5: Three samples from the REPROD provenance graphs were randomly selected. Each is distilled into a set of `Write` events. These are binned temporally to obtain a histogram. The three resulting graphs illustrate the diversity of behavior across samples.

To illustrate the issue, we selected three logs at random. These were generated from the AvosLocker, ChinaJm, and Snatch samples. For each, we distilled the corresponding provenance graph to the set of `Write` events performed by the ransomware process. These events were then bucketed into 10 second bins. The result is a temporal histogram that provides insight into the activity rate as well as the periods of time over which the writes occurred. Figure 5 shows the resulting histograms.

AvosLocker generated tens of thousands of `Write` events in each 10 second bin. The activity stopped after two minutes had elapsed (out of the ten minute execution). In contrast, ChinaJm operated at a lower rate (closer to a thousand writes in each 10 second bin) but continued through most of the execution. The screenshot taken at the end was anomalous. No sentinel file encryption was observed for either of these ransomware samples. The temporal pattern of Snatch also peaked at over ten thousand writes (like AvosLocker) in a ten second bin but ended much earlier. In contrast to the other two, the writes did result in observed file density changes.

These observations provide tangible evidence of the range of activity across diverse samples that must be considered by any mitigation approach.

8 ONLINE RESOURCES

The code used to produce REPROD has been shared on GitHub [14] and the dataset is on Zenodo [23].

The dataset contains provenance graphs (in DOT format) for all 861 ideal executions. In addition, it contains 405 ProcMon compressed logs (in PML format). This includes 98 logs where both density changes of honeypot files and anomalous screen activity was observed, 181 logs where only density changes were seen, and 117 logs where only anomalous screen activity occurred.

9 CONCLUSION

REPROD represents a broad collection of activity logs and provenance graphs for all ransomware binaries in MalwareBazaar. REPROD provides a dataset that can help analysts understand how ransomware can be expected to operate. Many binaries exhibit encryption activity within the first ten minutes of execution, despite no special actions taken to trigger this. Further, clear file access patterns are visible in the provenance graphs. These motifs can be leveraged in future work on ransomware detection and mitigation (using machine learning, for example). Open datasets like REPROD can help improve the efficiency of mitigation, which may ultimately reduce ransoms.

ACKNOWLEDGMENTS

This work was supported in part by the Office of Naval Research under Grant N00014-21-1-2754. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of ONR.

REFERENCES

[1] [n. d.]. Cuckoo Sandbox. https://cuckoosandbox.org/
[2] [n. d.]. FBI No Longer Negotiating with Ransomware Group That Leaked Oakland Data. https://abc7news.com/oakland-ransomware-hacked-data-leaked-fbi-dark-web/13225220/
[3] [n. d.]. MalwareBazaar. https://bazaar.abuse.ch/browse/
[4] [n. d.]. Ransomware Full Recovery Could Take Months, Dallas Officials Say. https://www.dallasnews.com/news/politics/2023/05/11/ransomware-full-recovery-could-take-months-dallas-officials-say/
[5] [n. d.]. Tukey five-number summary. https://en.wikipedia.org/wiki/Five-number_summary
[6] Muhammad Ejaz Ahmed, Hyoungshick Kim, Seyit Camtepe, and Surya Nepal. 2021. Peeler: Profiling Kernel-Level Events to Detect Ransomware. *26th European Symposium on Research in Computer Security* (2021).

[7] Mathieu Barre, Ashish Gehani, and Vinod Yegneswaran. 2019. Mining Data Provenance to Detect Advanced Persistent Threats. *11th USENIX Workshop on the Theory and Practice of Provenance (TaPP)* (2019).
[8] Gordon Blair. 2022. Test of Time Award. *ACM Middleware* (2022). https://middleware-conf.github.io/2022/awards/#testOfTime
[9] Simon Davies, Richard Macfarlane, and William J Buchanan. 2022. NapierOne: A Modern Mixed File Data Set Alternative to Govdocs1. *Forensic Science International: Digital Investigation* 40 (2022).
[10] Feng Dong, Liu Wang, Xu Nie, Fei Shao, Haoyu Wang, Ding Li, Xiapu Luo, and Xusheng Xiao. 2023. DISTDET: A Cost-Effective Distributed Cyber Threat Detection System. *30th USENIX Security Symposium* (2023).
[11] John Ellson, Emden Gansner, Lefteris Koutsofios, Stephen C North, and Gordon Woodhull. 2002. Graphviz — Open Source Graph Drawing Tools. *9th International Symposium on Graph Drawing* (2002).
[12] Ashish Gehani, Raza Ahmad, Hassaan Irshad, Jianqiao Zhu, and Jignesh Patel. 2021. Digging Into "Big Provenance" (With SPADE). *Commun. ACM* 64(12) (2021).
[13] Ashish Gehani and Dawood Tariq. 2012. SPADE: Support for Provenance Auditing in Distributed Environments. *13th ACM/IFIP/USENIX International Middleware Conference* (2012).
[14] REPROD GitHub. [n. d.]. Code for orchestrating ransomware execution log and provenance collection. https://github.com/REPROD-prov
[15] Xueyuan Han, James Mickens, Ashish Gehani, Margo Seltzer, and Thomas Pasquier. 2020. Xanthus: Push-button Orchestration of Host Provenance Data Collection. *3rd ACM Workshop on Practical Reproducible Evaluation of Computer Systems (P-RECS)* (2020).
[16] Manabu Hirano, Ryo Hodota, and Ryotaro Kobayashi. 2022. RanSAP: An Open Dataset of Ransomware Storage Access Patterns for Training Machine Learning Models. *Forensic Science International: Digital Investigation* 40 (2022).
[17] Hassaan Irshad, Gabriela Ciocarlie, Ashish Gehani, Vinod Yegneswaran, Kyu Hyung Lee, Jignesh Patel, Somesh Jha, Yonghwi Kwon, Dongyan Xu, and Xiangyu Zhang. 2021. TRACE: Enterprise-Wide Provenance Tracking For Real-Time APT Detection. *IEEE Transactions on Information Forensics and Security (TIFS)* 16 (2021).
[18] Amin Kharaz, Sajjad Arshad, Collin Mulliner, William Robertson, and Engin Kirda. 2016. UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware. *25th USENIX Security Symposium* (2016).
[19] Rui Mei, Han-Bing Yan, and Zhi-Hui Han. 2021. RansomLens: Understanding Ransomware via Causality Analysis on System Provenance Graph. *Science of Cyber Security* (2021).
[20] Richard Vanderford. 2023. Merck's Insurers On the Hook in $1.4 Billion NotPetya Attack, Court Says. *Wall Street Journal* (2023).
[21] Aldin Vehabovic, Nasir Ghani, Elias Bou-Harb, Jorge Crichigno, and Aysegul Yayimli. 2022. Ransomware Detection and Classification Strategies. *IEEE International Black Sea Conference on Communications and Networking* (2022).
[22] Christian Wojner. [n. d.]. DensityScout. https://cert.at/en/downloads/software/software-densityscout
[23] REPROD Zenodo. [n. d.]. Ransomware execution trace and provenance data. https://doi.org/10.5281/zenodo.7933806