BareCloud

Bare-metal Analysis-based Evasive Malware Detection

Referred slides on: https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/kirat

Presented by:
Pallavi Kasula
why
Automatic dynamic malware analysis?

- Static-Analysis and Fingerprinting
  - Suffer from techniques to evade like Polymorphism, obfuscation and encryption
- With increasing volume and sophistication of malware, “Automatic dynamic malware analysis” is widely adopted to detect malware.
Dynamic Malware

Dynamic Malware Analysis

Virtualization/Emulation
Execute

Behavior Profiles
Reports
Evasive Malware

Dynamic Malware Analysis

Virtualization/Emulation - Execute

Behavior Profiles

Reports
Detect Analysis

- Detection based on Fingerprinting of the runtime environment of the analysis system. Checking for specific artifacts such as
  - registry keys
  - background processes
  - function hooks
  - IP addresses

- Most analysis systems use emulated or virtualized environment. Such platforms can be detected by checking Platform-specific characteristics different from baseline environment
  - Timing properties of execution
  - small variation in the CPU execution semantics
Solution

- Look for evasive malware characteristics—similar to signature based detection

- Weak and easily evaded

- Detecting deviation in malware behavior in different analysis environments.
Transparent Analysis

Execution Environment

Monitoring Components
Dynamic Malware Analysis

Transparency

Visibility
Can we automatically identify evasive malware under reduced visibility?
BareCloud

Dynamic Malware Analysis

Bare-metal system

Execute

Behavior Profiles

Reports

No in-guest monitoring component
Monitoring Environments

- **Virtualization**
  - Cuckoo Sandbox - Type-2 hypervisor
  - Function-hooking-based in-guest monitoring components

- **Emulation**
  - Anubis Platform - Qemu based emulator
  - Monitors by observing the execution of pre-computed memory addresses
  - Implements Product-ID randomization etc to prevent direct detection

- **Hypervisor**
  - Ether - Xen based transparent malware analysis framework which utilizes intel hardware virtualization extensions
  - Monitoring incurs substantial overhead
BareCloud

- Cluster of hardware-based modular worker units
- Leverage copy-on-write techniques to perform disk restoration
- IPMI - To automate the execution of malware
  - Allows to control power cycle of the analysis worker units
- iSCSI - (Internet Small Computer System Interface) - To attach remote disks to worker units
- LVM - Logical Volume Manager - based copy-on-write snapshots to host remote disks
- Network based malware initiator
BareCloud
Behavior Deviation

- Evasive Behavior

- Intrinsic non-determinism - hierarchical similarity based comparison

- Internal environment - Providing similar operating environments

- External environment - Synchronized execution, Identical local network, Network service filters
Behavior Comparison

Jaccard Similarity = \( \frac{A \cap B}{A \cup B} \)
Behavior Similarity

Profile A
- Create File X
- Create File Y
- Create File Z

Profile B
- Create File X
- Create File Y
- Modify File Z

Profile C
- Create File X
- Create File Y
- Connect to C&C
Behavior Similarity

Profile A
Create File X
Create File Y
Create File Z

Profile B
Create File X
Create File Y
Modify File Z

Profile C
Create File X
Create File Y
Connect to C&C

Jaccard Similarity(A,B) = 2/4 = Jaccard Similarity(A,C)
Behavior Comparison

**What type of Events?**

**Are events related to the same object?**

**What type of operations?**

- Filesystem?
  - Network?
- Same file?
  - Same network endpoint?
- Create?
  - Delete?
  - HTTP?
Behavior Similarity Hierarchy

Root

Object Type

Object Name

Operation Name

Operation Attribute

Object Name

Operation Name

Operation Attribute

Object Name

Operation Name

Operation Attribute
Behavior Similarity Hierarchy

Profile B

Create File X

Create File Y

Modify File Z
Hierarchical Similarity

Profile A

- Root
- File
  - C:\X
    - Create
      - Size
  - C:\Y
    - Create
      - Size
  - C:\Z
    - Create
      - Size

Profile C

- Root
- File
  - C:\X
  - C:\Y
  - C\&C address
- Network
  - Http
  - Size

Candidate Sets
- Sim_1 = 1/2
- Sim_2 = 2/3
- Sim_3 = 1
- Sim_4 = 1

Sim(A,C) = AVG (Sim_1 - Sim_4) = 0.79
Hierarchical Similarity

Profile A

Profile B

Candidate Sets

Sim_1 = 1

Sim_2 = 1

Sim_3 = 1/2

Sim_4 = 1

Sim(A,B) = AVG (Sim_1 - Sim_4) = 0.87
Behavior Similarity

Profile A
- Create File X
- Create File Y
- Create File Z

Profile B
- Create File X
- Create File Y
- Modify File Z

Profile C
- Create File X
- Create File Y
- Connect to C&C

Jaccard Similarity (A, B) = 2/4 = Jaccard Similarity (A, C)
Hierarchical Similarity (A, B) > Hierarchical Similarity (A, C)

.87 > .79
Deviation Score

- Behavior Distance
  - Distance(A, B) = 1 - Sim(A, B)

- Deviation Score D
  - Quadratic mean of the behavior distances with respect to the bare-metal analysis

- Deviation threshold t
  - Evasive is D > t
Evaluation

- Ground truth
  - 111 evasive samples (29 families)
  - 119 non-evasive samples (49 families)

- Calculated behavior Deviation score

- Calculated Jaccard distance-based deviation JD
  - Maximum Jaccard distance among different behavior profiles of a malware

- Precision recall analysis by varying the deviation threshold $t$
Evaluation

![Graph showing precision-recall characteristics](image)

- **Precision** vs **Recall**
- **Hierarchical similarity** vs **Jaccard similarity**
- The graph compares the performance of hierarchical similarity and Jaccard similarity in detecting evasive malware samples.

**Figure 3:** Precision-Recall analysis of the hierarchical similarity-based deviation score. The hierarchical similarity-based deviation score gives better results. It is able to produce higher precision and recall compared to the Jaccard similarity-based deviation score.
Evaluation

Figure 4: Precision-recall analysis of the behavior deviation threshold value $t$. Threshold value $t = 0.84$ gives the highest recall rate with 100% detection precision.

We can see that with the threshold value $t = 0.70$, more than half of the evasive samples can be detected with above 98% precision. While with threshold $t = 0.84$, we get 100% precision with the recall rate of 40.20%. In the next large-scale experiment, we used this threshold value $t = 0.84$ to have a high confidence on the detection results.

5.2 Experiment II

In this experiment, we applied our behavior comparison approach to the incoming malware feed received by Anubis. We first filtered the incoming samples based on the size and the type of the behavioral profile extracted in the Anubis analysis environment. This is required to select interesting samples from the large volume of the incoming malware without introducing any strong bias. This pre-selection process randomly selects samples from the following four groups.

- Samples with minimal activity: These are the samples that show minimal to no activity in the Anubis analysis environment (less than 1000 events). This group may contain evasive malware that successfully evade Anubis analysis environment.
- Samples with high system and network activity: These are the samples that show a substantial amount of system-related activity (more than 1000 events) and network activity in the Anubis analysis environment. We include this group to see if similar behavior can be observed in all other analysis environments.
- Samples with high network activity: These are the samples that show minimal system activity but high network activity (more than 10 packets) in the Anubis analysis environment.
- Samples with high system activity: These are the samples that show no network activity but high system activity in the Anubis analysis environment.

We selected 110,005 samples from the above groups observed during a four months period, starting from July 2013. We extracted behavioral profiles of these samples from all four analysis environments and computed the deviation scores. We used the behavior deviation threshold of $t = 0.84$ that was selected in the previous experiment. With this threshold, we were able to detected 5,835 samples as evasive. That is, these evasive samples evade one or more analysis environments. Since we do not have the ground truth for this dataset, we cannot provide the true precision of this detection.

We also categorized these evasive samples into environment-specific evasions by using the number of events observed within the various analysis environments. That is, if no activity that results in persistent change is observed in an analysis environment $a_x$ while the same sample shows persistent change activities in at least one of the other analysis environments $[a_1, ... , a_n]$, we considered that as an evasion of the analysis environment $a_x$. The results are presented in Table 2. We found that 2,530 samples showed no activity that resulted in persistent change in all three common malware analysis environments, while such persistent change activities were observed in the bare-metal analysis environment. We see that many samples were able to evade the Ether environment, which is considered as a transparent environment. Manual analysis of those samples revealed that many samples were able to fingerprint Ether by leveraging the dependency of the Xen-hypervisor on Qemu-based device models to emulate required virtual devices, such as hard disks. Since Ether is based on Xen-hypervisor, it also requires the Qemu-based device models to run the unmodified guest, which acts as the transparent analysis host. Therefore, if a malware sample inside Ether tries to fingerprint its devices, it might identify Ether as a Qemu-based analysis system. In addition, system call tracing in Ether introduces additional overhead to the system, and, in some of the cases, regular timing-based attacks were successful in detecting the presence of the analysis system.
Large Scale Evaluation

- Recent real-world malware feed observed by Anubis

- Randomly select samples with
  - Low system and low network activity
  - High system and high network activity
  - High system and low network activity
  - Low system and high network activity

- 110,005 samples in a four months period from July 2013
Large Scale Evaluation

<table>
<thead>
<tr>
<th>Environment</th>
<th>Detection count</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anubis</td>
<td>4,947</td>
<td>84.78</td>
</tr>
<tr>
<td>Ether</td>
<td>4,562</td>
<td>78.18</td>
</tr>
<tr>
<td>VirtualBox</td>
<td>3,576</td>
<td>61.28</td>
</tr>
<tr>
<td>All</td>
<td>2,530</td>
<td>43.35</td>
</tr>
<tr>
<td>Total</td>
<td>5,835</td>
<td></td>
</tr>
</tbody>
</table>
Limitations

• Stalling codes cannot be detected in BareCloud

• Heuristic checks like waiting for user input or trying to resolve non-existent domain can detect the presence of bare-metal analysis system

• High levels of randomization in persistent changes will result in erroneous detection of evasion.
Conclusion

- Evasive malware is a real threat to the dynamic analysis based malware detection systems
- A novel approach to detect these evasive malware automatically is presented
Question?