You Cannot Escape Me: Detecting Evasions of SIEM Rules in Enterprise Networks

Rafael Uetz[*] Marco Herzog[*] Louis Hackländer[*] Simon Schwarz[†] Martin Henze[‡,*]

[*] *Fraunhofer FKIE* [†] *University of Göttingen* [‡] *RWTH Aachen University*

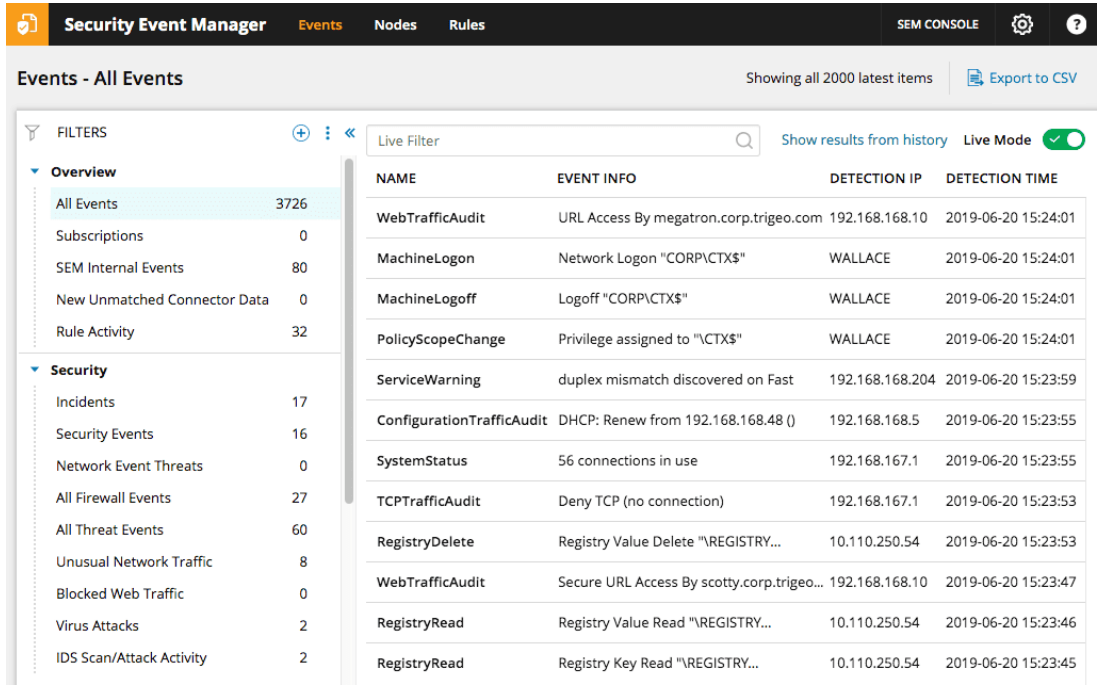
arXiv preprint arXiv:2311.10197 (2023)

Outline

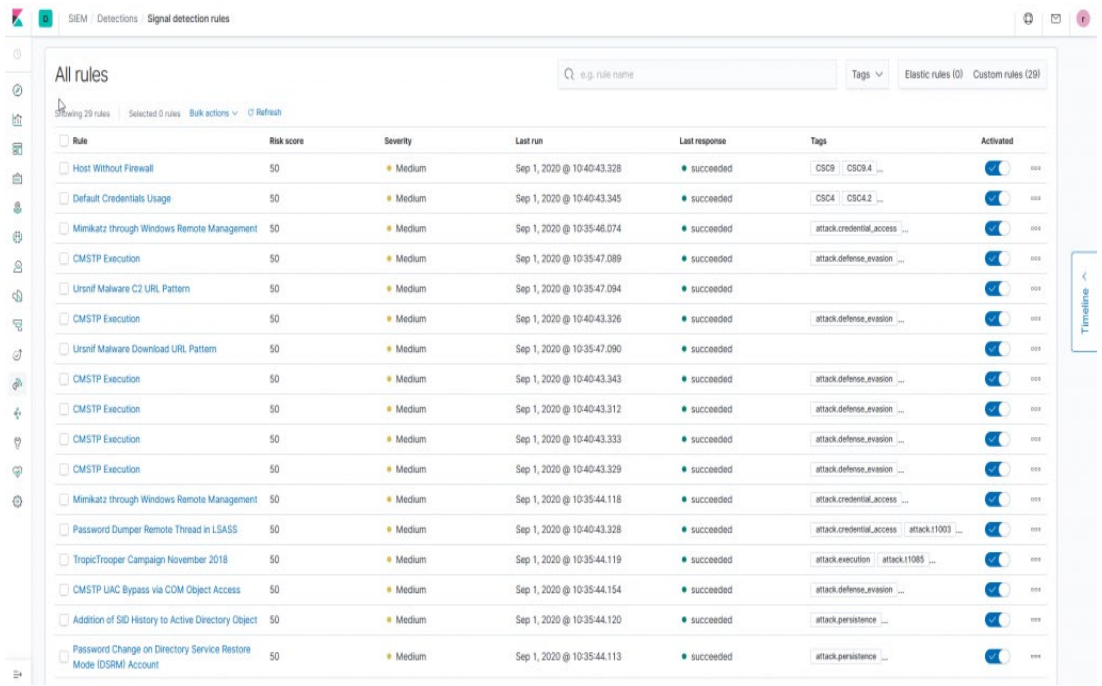
- Introduction
  - SIEM
  - Sigma
- Analysis of SIEM Rules for Evasions
- Overview
- Approach
- Evaluation
- Limitations and Discussion
- Conclusion

Introduction

- Security Information and Event Management (SIEM)
  - Organizations utilize SIEM to collect security-related events and scan them using expert-written detection rule-sets



security-related events



rule-sets

Introduction

- Sigma https://github.com/SigmaHQ/sigma
  - Sigma is an open source standardized format for describing SIEM rules
  - Solve the interoperability problem between different SIEM platforms such as Splunk, ELK, ArcSight, QRadar
  - Assumes that public Sigma rules are being used by a victims

Windows Event Log :

```
Log Name: Security
Source: Microsoft-Windows-Security-Auditing
Event ID: 4688
Task Category: Process Creation
Level: Information
Keywords: Audit Success
Description:
A new process has been created.
Subject:
    Security ID:        S-1-5-21-3623811015-3361044348-30300820-1013
    Account Name:       johndoe
    Account Domain:     CONTOSO
    Logon ID:           0x3e7
New Process:
    New Process ID:     0x1f4
    New Process Name:   C:\Windows\System32\cmd.exe
    Token Elevation Type: %%1936
    Creator Process ID: 0x1c4
    Process Command Line: C:\Windows\System32\cmd.exe /c whoami
Network Information:
    Workstation Name:   WIN-PC
    Source Network Address: 192.168.1.101
    Source Port:        12345
```

Sigma Rule for whoami :

```
title: Suspicious Process Creation - whoami Command
id: a1b2c3d4-5678-90ab-cdef-1234567890ab
description: Detects the creation of a cmd.exe process with the whoami command
author: Example Author
date: 2023/05/21
logsource:
  product: windows
  service: security
detection:
  selection:
    EventID: 4688
    NewProcessName: C:\\Windows\\System32\\cmd.exe
  condition: selection and cmdline
  cmdline:
    CommandLine|re: '.*whoami.*'
fields:
  - EventID
  - NewProcessName
  - ProcessCommandLine
falsepositives:
  - Administrative scripts
level: high
```

ALERT !

Analysis of SIEM Rules for Evasions

1. Analyzed all process creation rules that were contained in the Sigma repository
2. Re-enact the malicious process creation as described by the rules
3. If we succeeded to match the rule, then tried to find command lines that perform the exact same action, but without matching the rule

---

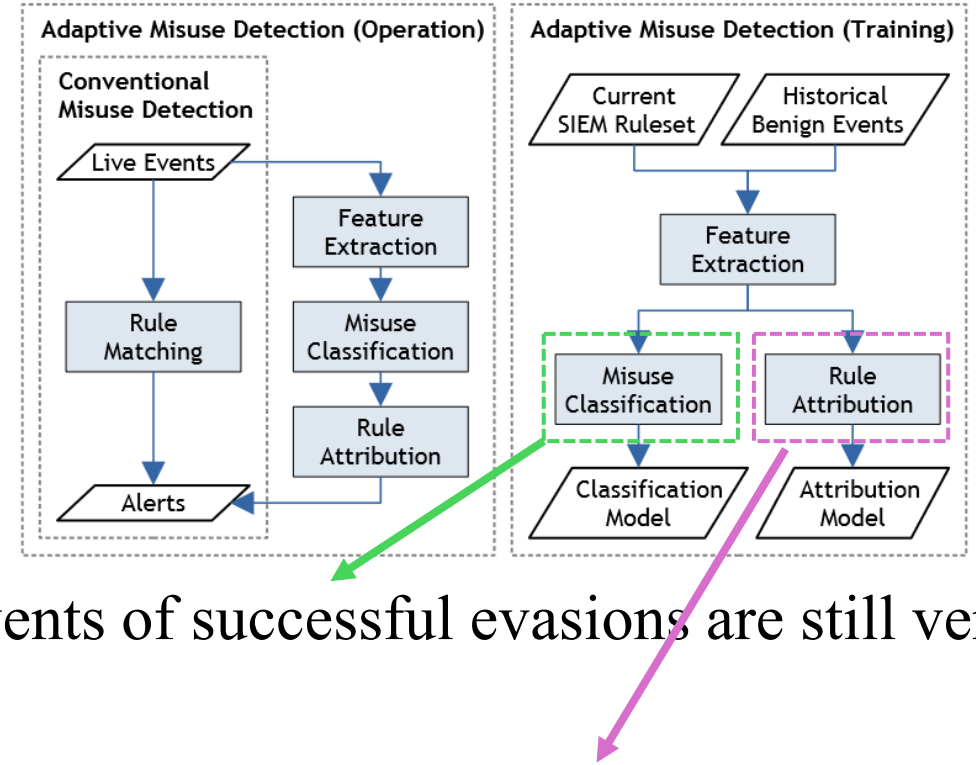
292 Sigma rules, 110 (38 %) can be fully evaded, 19 (7 %) can be partially evaded

Summarized five evasion types:

Evasion type	Sample affected rule	Affected search term	Sample match	Sample evasion
Insertion	win_susp_schtask_creation	* /create *	schtasks.exe /create ...	schtasks.exe /"create" ...
Substitution	win_susp_curl_download	␣-O␣	curl -O http://...	curl --remote-name http://...
Omission	win_mal_adwind	*cscript.exe *Retrive*.vbs *	cscript.exe ...\Retrive.vbs	cscript ...\Retrive.vbs
Reordering	win_susp_procdump	* -ma ls*	procdump -ma ls	procdump ls -ma
Recoding	win_vul_java_remote_dbg	*address=127.0.0.1*	...address=127.0.0.1,...	...address=2130706433,...

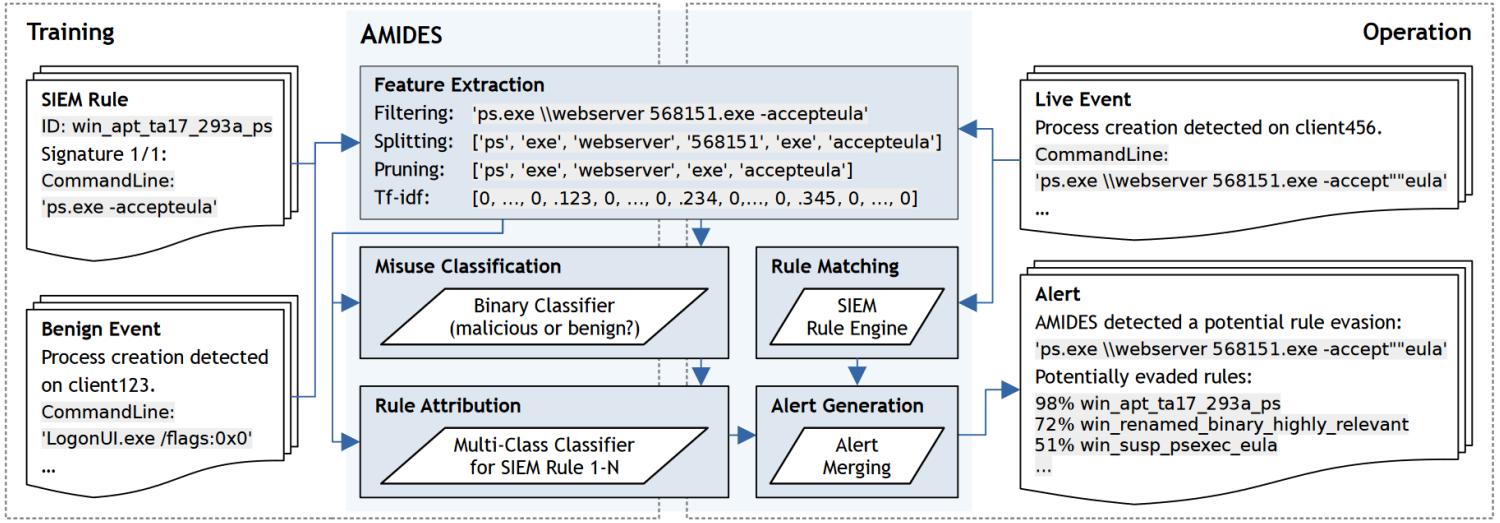
Overview

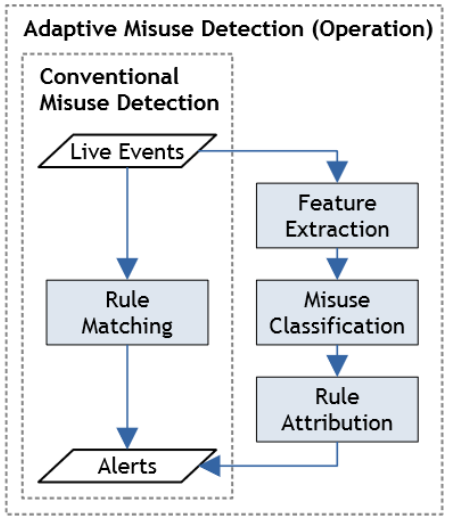
- A methodology to reduce blind spots by detecting rule evasions in addition to conventional rule matches

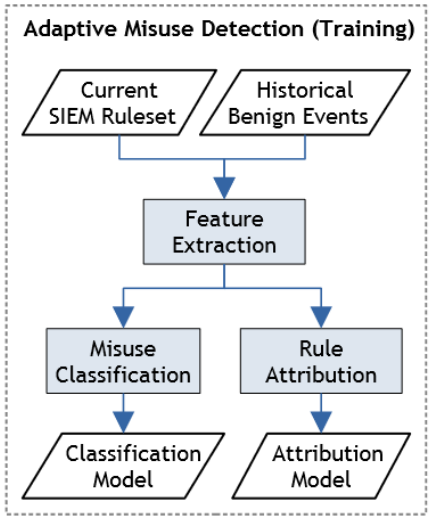


- Assume that SIEM events of successful evasions are still very similar to those of the original attack.
- When a command line is executed that is very “similar” to a signature of some rule, the respective rule should be proposed to the analyst within the alert

Approach







- Feature Extraction: Filtering + Splitting + Pruning + Tf-idf
- Misuse Classification: SVM
- Rule Attribution: train a SVM for every rule

- Performance Considerations: Implement an in-memory cache to avoid repeated classification of already-seen feature vectors

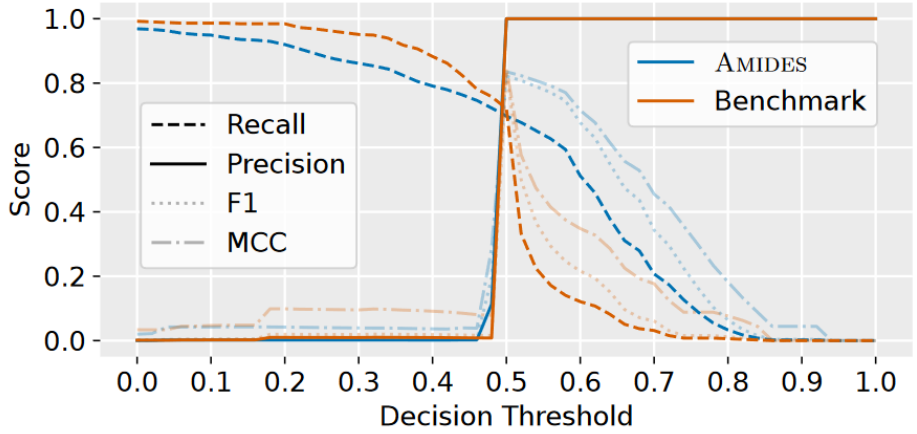
Evaluation

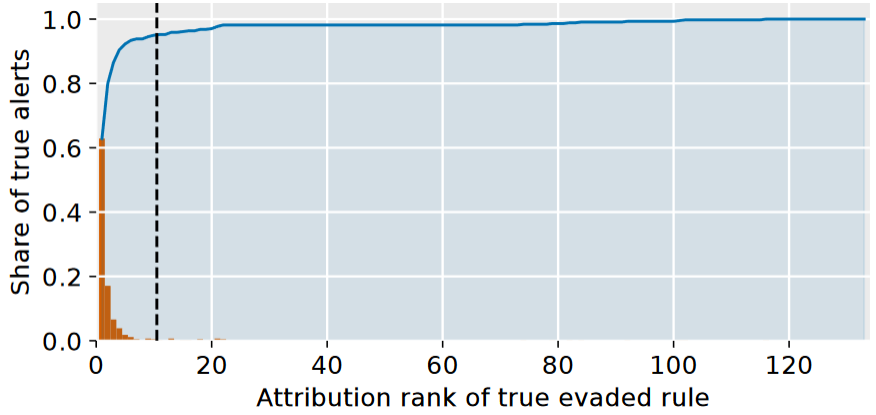
- RQ1: How well does AMIDES detect SIEM rule evasions?
- RQ2: How accurate is the rule attribution?
- RQ3: Is AMIDES suited for real-world operation?

- Dataset: collect SIEM events from a large enterprise network with more than 50000 users.

Evaluation

- RQ1: How well does AMIDES detect SIEM rule evasions?
  - Benchmark: learn from attack events and benign events

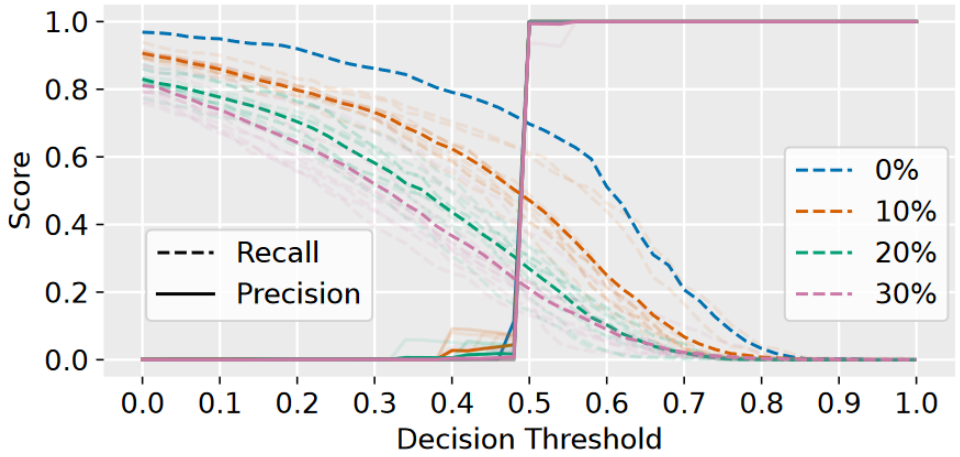


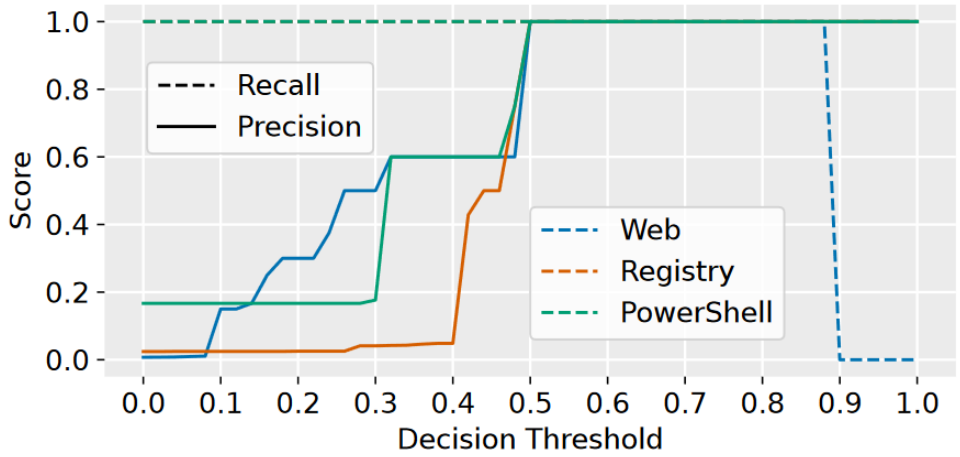


- RQ2: How accurate is the rule attribution?
  - A vast majority of the true rules (95 %) are contained within the top 10

Evaluation

- RQ3: Is AMIDES suited for real-world operation?
  - A commodity Linux server with 40 physical CPU cores (80 virtual), 384 GiB RAM, and a single SSD drive
  - AMIDES required 0.0763 ms per event on average (σ = 3.19 ms) on a single core
  - Consider the case that supposedly benign training data inadvertently contain evasions
  - Consider the case that supposedly benign training data inadvertently contain evasions





Limitations and Discussion

- Cannot detect fundamentally different classes of evasion attacks such as (undetected) code injection into benign processes
- Attackers could try to evade AMIDES itself
- Focus on Windows process creation events, particularly their command line field

Conclusion

- Analyzed 292 Windows process creation rules from the Sigma repository, finding 110 fully evadable and 19 partially evadable.
- Evaluated the open-source implementation AMIDES in a large enterprise network, detecting 70% of evasions with zero false alerts at default sensitivity.
- AMIDES processes approximately 156,000 events per second and requires 42 minutes of training, making it suitable for large enterprise networks.
- For 95% of evasions, AMIDES included the actually evaded rule within its top 10 propositions.