Signature-based Intrusion Detection Using Imbalanced Datasets

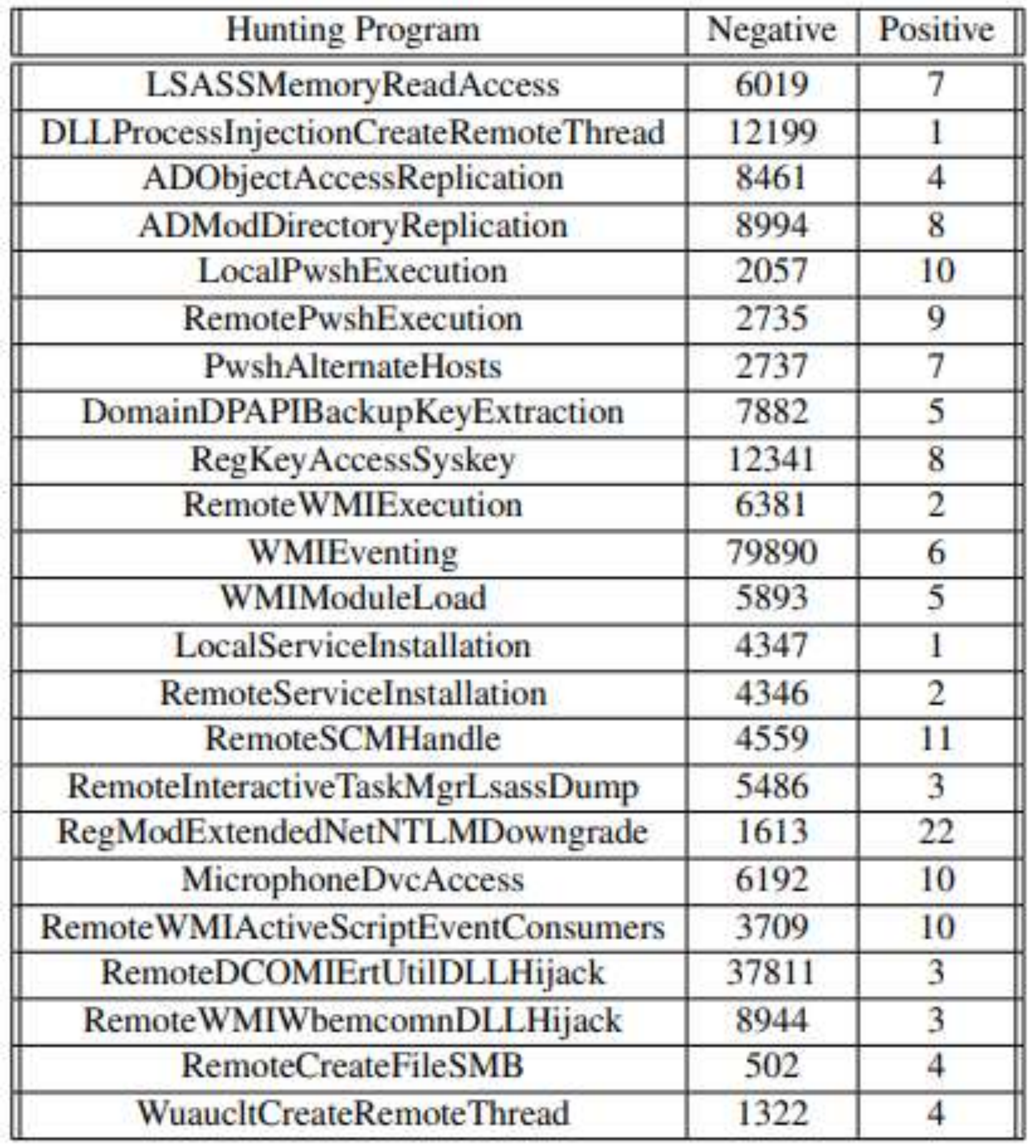
Author: Yue Leng, Supervised by: Matthew Collinson

Objectives

- To validate the usability of the Threat Hunter Playbook and its compatible Security Datasets.
- To determine the feasibility of incorporating machine learning techniques into signature-based intrusion detection systems.
- To tackle the significant challenge of handling extremely imbalanced datasets.

Threat Hunter Playbook

The datasets in Security Datasets are extremely imbalanced. With negative samples representing normal activities, and positive samples representing anomalies, we got a table for different numbers of negative and positive samples in different datasets as below:



Hunting Program	Negative	Positive
LSASSMemoryReadAccess	6019	7
DLLProcessInjectionCreateRemoteThread	12199	1
ADObjectAccessReplication	8461	4
ADModDirectoryReplication	8994	8
LocalPwshExecution	2057	10
RemotePwshExecution	2735	9
PwshAlternateHosts	2737	7
DomainDPAPIBackupKeyExtraction	7882	5
RegKeyAccessSyskey	12341	8
RemoteWMIExecution	6381	2
WMIEventing	79890	6
WMIModuleLoad	5893	5
LocalServiceInstallation	4347	1
RemoteServiceInstallation	4346	2
RemoteSCMHandle	4559	11
RemoteInteractiveTaskMgrLsassDump	5486	3
RegModExtendedNetNTLMDowngrade	1613	22
MicrophoneDvcAccess	6192	10
RemoteWMIActiveScriptEventConsumers	3709	10
RemoteDCOMIErtUtilDLLHijack	37811	3
RemoteWMIWbemcomnDLLHijack	8944	3
RemoteCreateFileSMB	502	4
WuaucltCreateRemoteThread	1322	4

Apart from that, most of our datasets in Security Datasets also have a large amount of data missing.

Contact Information

- Email: y.leng.21@abdn.ac.uk
- Phone: +44 (0)7763630611

Introduction

To ensure the security of systems, the intrusion detection system (IDS) is an essential technique in the domain of cyber security. However, developing effective IDS is a challenging task that requires precise analysis, simulation of real-life anomalies, and large, updated datasets. This study aims to address this challenge by exploring feasible methods for building signature-based intrusion detection systems (SIDS) and optimizing the effectiveness of limited data.

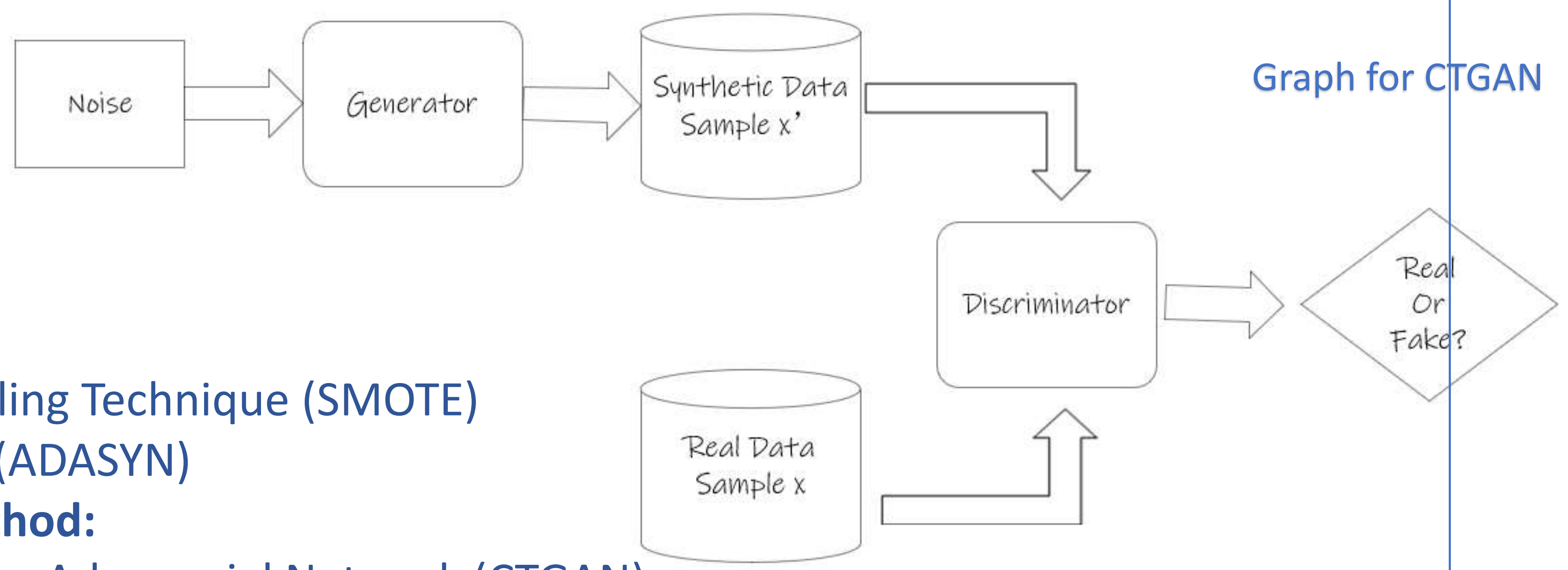
Utilizing the Threat Hunter Playbook and its compatible datasets, Security Datasets, the study constructed a SIDS. However, we also found that the datasets are extremely imbalanced and with a large amount of data missing. The study proposes a practical approach that incorporates decision trees and Deep Neural Networks for intrusion detection classification, as well as data generation techniques using both Generative Adversarial Networks and oversampling methods to handle imbalanced datasets. This study presents an effective methodology for constructing SIDS when faced with inadequate and imbalanced datasets. It offers valuable insights into developing efficient and effective IDS with limited data.

Methods

Machine Learning Methods
- Decision Tree
- Deep Neural Network (DNN)
- Oversampling Methods:

Random Oversampling (ROS)
- Synthetic Minority Oversampling Technique (SMOTE)
- Adaptive Synthetic Sampling (ADASYN)

Synthetic Data Generation Method:
- Conditional Tabular Generative Adversarial Network (CTGAN)



Graph for CTGAN

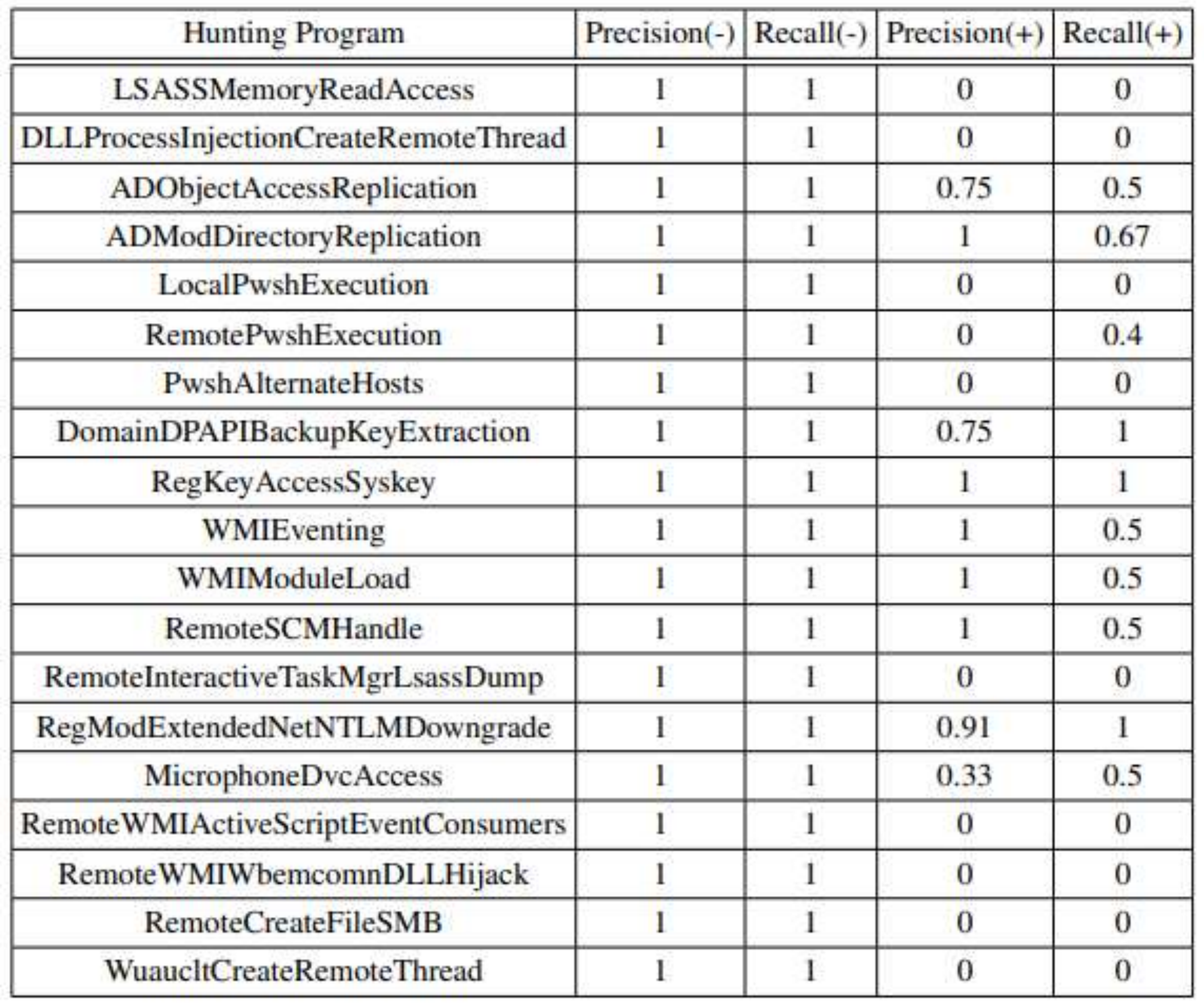
5 Scenarios

Scenario	Classification Methods	Data Handling Methods	Datasets
Scenario 1	Decision Trees	None	Original Datasets
Scenario 2	Decision Trees	ROS, SMOTE, ADASYN	Oversampled Datasets
Scenario 3	DNNs	ROS, SMOTE, ADASYN	Oversampled Datasets
Scenario 4	Decision Trees	CTGAN	Synthetic Datasets
Scenario 5	DNNs	CTGAN	Synthetic Datasets

Conclusions

- Data in Threat Hunter Playbook and Security Datasets for building strong SIDS is insufficient, more anomaly activities need to be collected and the missing data need to be added.
- For binary-classification SIDS, the decision tree is a good machine-learning method to incorporate while DNN is not.
- CTGAN can be an effective solution for handling extremely imbalanced datasets in SIDS.

Results On the Original Datasets (S1)

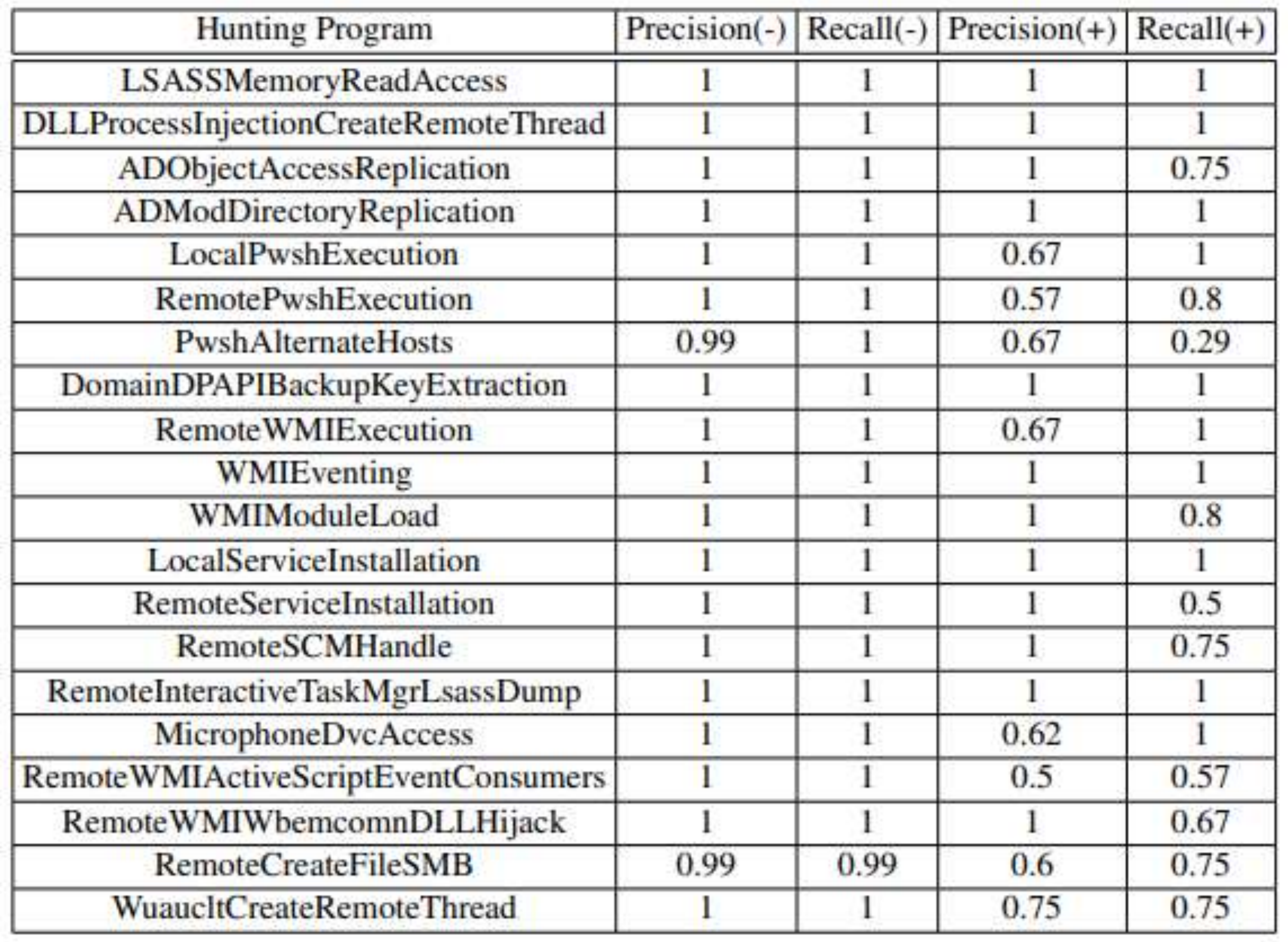


Hunting Program	Precision(-)	Recall(-)	Precision(+)	Recall(+)
LSASSMemoryReadAccess	1	1	0	0
DLLProcessInjectionCreateRemoteThread	1	1	0	0
ADObjectAccessReplication	1	1	0.75	0.5
ADModDirectoryReplication	1	1	1	0.67
LocalPwshExecution	1	1	0	0
RemotePwshExecution	1	1	0	0.4
PwshAlternateHosts	1	1	0	0
DomainDPAPIBackupKeyExtraction	1	1	0.75	1
RegKeyAccessSyskey	1	1	1	1
WMIEventing	1	1	1	0.5
WMIModuleLoad	1	1	1	0.5
RemoteSCMHandle	1	1	1	0.5
RemoteInteractiveTaskMgrLsassDump	1	1	0	0
RegModExtendedNetNTLMDowngrade	1	1	0.91	1
MicrophoneDvcAccess	1	1	0.33	0.5
RemoteWMIActiveScriptEventConsumers	1	1	0	0
RemoteWMIWbemcomnDLLHijack	1	1	0	0
RemoteCreateFileSMB	1	1	0	0
WuaucltCreateRemoteThread	1	1	0	0

Table 5.1: Performance metrics of decision trees on datasets: Negative Precision, Negative Recall, Positive Precision, and Positive Recall

Best Scenario Results (S4)

Using Decision Tree to Classify Synthetic Datasets:



Hunting Program	Precision(-)	Recall(-)	Precision(+)	Recall(+)
LSASSMemoryReadAccess	1	1	1	1
DLLProcessInjectionCreateRemoteThread	1	1	1	1
ADObjectAccessReplication	1	1	1	0.75
ADModDirectoryReplication	1	1	1	1
LocalPwshExecution	1	1	0.67	1
RemotePwshExecution	1	1	0.57	0.8
PwshAlternateHosts	0.99	1	0.67	0.29
DomainDPAPIBackupKeyExtraction	1	1	1	1
RemoteWMIExecution	1	1	0.67	1
WMIEventing	1	1	1	1
WMIModuleLoad	1	1	1	0.8
LocalServiceInstallation	1	1	1	1
RemoteServiceInstallation	1	1	1	0.5
RemoteSCMHandle	1	1	1	0.75
RemoteInteractiveTaskMgrLsassDump	1	1	1	1
MicrophoneDvcAccess	1	1	0.62	1
RemoteWMIActiveScriptEventConsumers	1	1	0.5	0.57
RemoteWMIWbemcomnDLLHijack	1	1	1	0.67
RemoteCreateFileSMB	0.99	0.99	0.6	0.75
WuaucltCreateRemoteThread	1	1	0.75	0.75

Table 5.4: Performance metrics of decision trees on datasets with synthetic data: Negative Precision, Negative Recall, Positive Precision, and Positive Recall