Evaluation of an Anomaly Detector for Routers using Parameterizable Malware in an IoT Ecosystem *

John Carter[1 [0000−0003−4391−0269]] and Spiros Mancoridis[1 [0000−0001−6354−4281]]

Drexel University, Philadelphia PA 19104, USA
jmc683@drexel.edu

Abstract. This work explores the evaluation of a machine learning anomaly detector using custom-made parameterizable malware in an Internet of Things (IoT) Ecosystem. It is assumed that the malware has infected, and resides on, the Linux router that serves other devices on the network, as depicted in Figure 1. This IoT Ecosystem was developed as a testbed to evaluate the efficacy of a behavior-based anomaly detector. The malware consists of three types of custom-made malware: ransomware, cryptominer, and keylogger, which all have exfiltration capabilities to the network. The parameterization of the malware gives the malware samples multiple degrees of freedom, specifically relating to the rate and size of data exfiltration. The anomaly detector uses feature sets crafted from system calls and network traffic, and uses a Support Vector Machine (SVM) for behavioral-based anomaly detection. The custom-made malware is used to evaluate the situations where the SVM is effective, as well as the situations where it is not effective.

这项工作探讨了对机器学习异常检测器的评估，该检测器在物联网（IoT）生态系统中使用定制的可参数化恶意软件。假设该恶意软件已经感染并驻留在Linux路由器上，为网络上的其他设备提供服务。如图1所描述的那样。该物联网生态系统被开发为一个测试平台，以评估基于行为的异常检测器的功效。检测器的功效。恶意软件包括三种类型的定制恶意软件。勒索软件、密码器和键盘记录器，它们都具有向网络渗透的能力。恶意软件的参数化使恶意软件样本有多个自由度，特别是有关数据渗出的速度和规模。异常检测器使用特征集从系统调用和网络流量中精心制作的特征集，并使用支持向量机（SVM）进行行为分析。机器（SVM）进行基于行为的异常检测。定制的恶意软件被用来评估SVM有效的情况，以及它无效的情况。

Keywords: Internet of Things, malware, routers, malware detection, Linux, machine learning, anomaly detector

物联网，恶意软件，路由器，恶意软件检测。Linux，机器学习，异常检测器

1 Introduction

Malware detection on small, resource-constrained devices has emerged as an important area of research, as IoT devices have grown in popularity. Since these devices have limited resources [6], malware detection software running on them must be efficient and lightweight, yet accurate and useful. This work focuses on creating and deploying custom-made parameterizable malware on a router in an IoT ecosystem to evaluate an anomaly detector's effectiveness in detecting the presence of malware. The parameterization of the malware enables the conditions on the router to vary, which provides a variety of data with which to train and test the anomaly detector. We show that while the SVM is incredibly effective and practical as an anomaly detector on IoT devices due to its low resource consumption and high accuracy, the parameters of the malware can be adjusted to decrease the effectiveness of the SVM.

随着物联网设备的普及，小型、资源受限的设备上的恶意软件检测已经成为一个重要的研究领域。随着物联网设备的普及，小型资源受限设备的恶意软件检测已成为一个重要的研究领域。由于这些设备的资源有限[6]，在这些设备上运行的恶意软件检测必须是高效和轻量级的，但又是准确和有用的。这项工作的重点是在一个物联网生态系统中的路由器上创建和部署定制的可参数化的恶意软件物联网生态系统中的路由器，以评估异常检测器在检测恶意软件的存在。恶意软件的参数化使路由器上的条件恶意软件的参数化使路由器上的条件发生变化，这就提供了各种数据来训练和测试异常探测器。测试异常检测器。我们表明，虽然SVM作为异常检测器是非常有效和实用的由于其低资源消耗和高准确性，SVM作为物联网设备上的异常检测器非常有效和实用。恶意软件的参数可以被调整，以降低SVM的有效性。以降低SVM的有效性。

* The work was funded in part by Spiros Mancoridis' Auerbach Berger Chair in Cybersecurity.

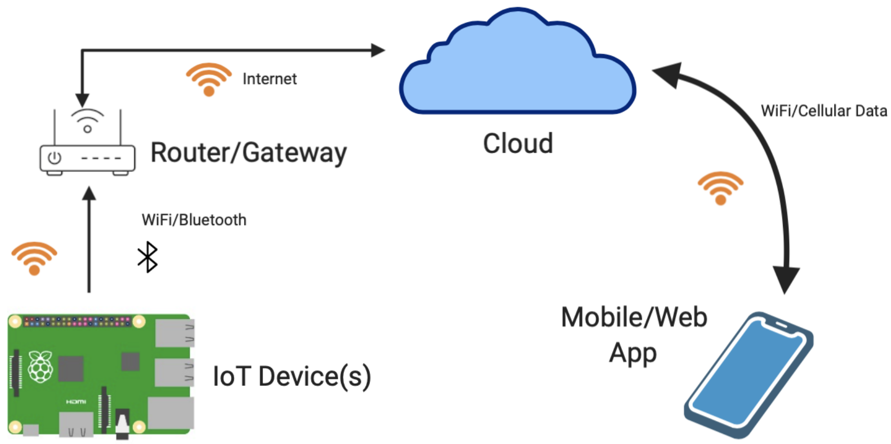

Fig. 1. The IoT Ecosystem

Routers can be described as IoT devices that do not require constant human oversight, and have limited behavioral patterns and resources. Since routers are used to connect devices on a local area network (LAN) to the Internet, their security is critical. If a router is infected with malware, there is a high probability that the malware can spread quickly to other devices on the network. This idea underpins one of the important reasons to work on the topic of securing routers. If it is possible to make a router resilient to malware, then it is possible that the router could act as a firewall to prevent malware from infecting devices connected to the router's network. In this work, the router we are working with is a Raspberry Pi 3, called Pi-Router, that has been configured to work as a router using the *hostapd* package. This package allows the Raspberry Pi to work as a wireless access point. The Raspberry Pi gives a realistic picture of working on an IoT device due to its own limited resources, and the fact that it runs on a Linux distribution. This provides a useful environment in which to develop lightweight malware detection systems that can be ported to similar Linux-based IoT devices.

路由器可以被描述为不需要人类持续监督的物联网设备。监督，而且行为模式和资源有限。由于路由器是用于将局域网（LAN）上的设备连接到互联网上，它们的安全是至关重要的。安全是至关重要的。如果一个路由器感染了恶意软件，很有可能恶意软件可以迅速传播到网络上的其他设备。这个想法是研究保障路由器安全这一课题的重要原因之一。如果有可能使路由器对恶意软件有抵抗力，那么就有可能路由器可以充当防火墙，防止恶意软件感染连接到路由器网络的设备。连接到路由器的网络上。在这项工作中，我们所使用的路由器是是一个名为Pi-Router的树莓派3，它已经被配置为一个路由器，并使用hostapd软件包。这个软件包允许Raspberry Pi作为一个无线接入点工作。作为一个无线接入点。树莓派提供了一个真实的工作画面由于其自身的资源有限，以及它运行在Linux发行版上这一事实，Raspberry Pi提供了一个真实的物联网设备工作画面。在一个Linux发行版上。这提供了一个有用的环境，在其中开发轻量级的恶意软件检测系统，可以被移植到类似的基于Linux的物联网设备.

In order to have a diverse set of fully-functional and parameterizable malware samples, we have created three types of malware for this work: ransomware, cryptominer, and keylogger, all with remote exfiltration capabilities. Some of these malware samples, such as the keylogger, may not be commonly found on routers, but including it in this research demonstrates the breadth of different malware examples that could infect any IoT device. Often, malware caught "in the wild" fails to run well, or at all, for a variety of reasons. These can include outdated code, attempts to connect to a server that no longer exists, and other reasons. This situation can make malware samples from the wild less useful to train and test a malware detector, which creates a need for custom-made malware that emulates how different malware families behave.

为了拥有一套多样化的全功能和可参数化的恶意软件样本，我们为这项工作创建了三种类型的恶意软件：勒索软件。密码器和键盘记录器，都具有远程渗透能力。其中一些这些恶意软件样本，如键盘记录器，可能在路由器上不常见到。路由器上，但在这项研究中，它显示了不同的恶意软件例子的广度。可以感染任何物联网设备的不同恶意软件实例的广度。通常情况下，在 "野外 "捕获的恶意软件通常情况下，由于各种原因，"野外 "捕获的恶意软件不能很好地运行，或根本不能运行。这些原因可能包括过时的代码，试图连接到一个不再存在的服务器，以及其他原因。原因。这种情况会使来自野外的恶意软件样本对训练和测试恶意软件检测器，这就需要定制的恶意软件来模拟不同恶意软件家族的行为方式。

The malware created all have parameterizable exfiltration rates, which means they can be tuned to attempt to elude the anomaly detector. These degrees of freedom on the malware samples are essential to emulate the adversarial relationship between the malware and the malware detection software. The parameterization of the malware is depicted on the right side of Figure 2. The specific functionality and capabilities of each of the malware samples will be discussed further in Section 3.

创建的恶意软件都有可参数化的渗出率，这意味着它们可以被调整以试图躲避异常检测器。这些自由度恶意软件样本的这些自由度对于模拟恶意软件和恶意软件检测软件之间的对抗关系至关重要。恶意软件的参数化描述在图2的右侧。每个恶意软件的具体每个恶意软件样本的具体功能和能力将在第三节进一步讨论。将在第3节进一步讨论。

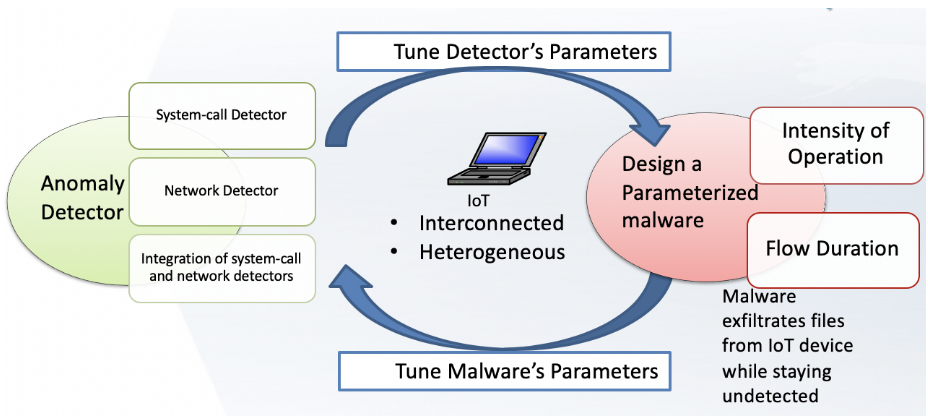

Fig. 2. Parallel Development of an Anomaly Detector and Malware Samples

A SVM is used for router behavioral anomaly detection. The SVM was chosen because it can be trained effectively with less data than other machine learning models, such as neural networks, and it is more efficient to run on a resource-constrained IoT device. The detector was trained on three types of data. The rest of the paper demonstrates how it is possible to detect malware running on the Pi-Router by first training the anomaly detector on kernel-level system call data, training the detector on network traffic data, and lastly, training the detector on a combination of the two, which is depicted on the left side of Figure 2 [8]. Each of these anomaly detectors are accurate in classifying data, but we will show that the detector trained on the combination of data generally performs the best.

一个SVM被用于路由器的行为异常检测。选择SVM的原因是因为它可以用比其他机器学习模型更少的数据进行有效的训练，如神经网络模型，而且它在资源有限的物联网设备上运行更有效。检测器在三种类型的数据上进行了训练。本文其余部分本文的其余部分展示了如何检测在Pi-Router上运行的恶意软件的可能性。Pi-Router上运行的恶意软件，首先在内核级系统调用数据上训练异常检测器。在网络流量数据上训练检测器，最后，在两者的组合上训练检测器。图2[8]的左边描述了这两者的组合。这些异常检测器中的每一个都能准确地对数据进行分类，但是我们将但我们将表明，在数据组合上训练的检测器通常表现得表现最好。

2 Related Work

This work draws on prior research in the areas of anomaly detection and malware detection on IoT devices. Previous work focused on specific IoT devices with predictable behavior, such as Amazon's Alexa running on a Raspberry Pi. This work approaches the problem more broadly in the context of routers infected with custom-made parameterizable malware [8] [7].

这项工作借鉴了先前在异常检测和恶意软件领域的研究。物联网设备上的异常检测和恶意软件检测领域的现有研究。以前的工作集中在具有可预测行为的特定物联网设备上，如运行在Raspberry Pi上的亚马逊Alexa。可预测的行为，如亚马逊的Alexa在Raspberry Pi上运行。这项这项工作从更广泛的角度来处理这个问题，即在路由器上感染了感染的路由器[8] [7]。

Sequences of system calls and network traffic data have been shown to perform well as feature sets for classifiers, as they provide insight to the programs running on a machine and can often be used to differentiate between a period of benign behavior, and a period of malware infection [8].

System call sequences have been shown to be effective as a feature set in many architectures due to the fact that they are one of the best indicators of what is happening on a machine at runtime [8] [2]. Each process running on a machine uses system calls to request resources from the OS kernel, which means the programs running normally are likely to make similar system calls each time they run. Because of this, it will be more obvious when a new program starts running because it will likely either issue a different set of system calls or issue the same system calls in a different order. It has been suggested, by results in prior work, that an anomaly detector based solely on system call traces performs well enough on its own to be effective in malware detection [8].

系统调用序列和网络流量数据已被证明可以很好地作为分类器的特征集，因为它们提供了对机器上运行的程序的深入了解。在一台机器上运行的程序，并且通常可以用来区分一个时期的良性行为和恶意软件感染的时期[8]。系统调用序列已被证明是许多架构中有效的特征集。许多架构中的有效特征集，因为它们是显示运行时机器上发生的事情的最佳指标之一。在运行时机器上发生了什么[8] [2]。在一台机器上运行的每个进程都会使用系统调用来从机器上运行的每个进程都使用系统调用来向操作系统内核请求资源，这意味着正常运行的程序在每次运行时都有可能进行类似的系统调用。它们的运行。正因为如此，当一个新的程序开始运行时就会比较明显因为它可能会发出一组不同的系统调用，或者以不同的顺序发出相同的系统调用。或者以不同的顺序发出相同的系统调用。根据先前工作的结果，有人建议之前的工作结果表明，仅基于系统调用痕迹的异常检测器的性能异常检测器本身就足以有效地进行恶意软件检测[8]。

Network traffic data have also been shown to be effective as a feature set for behavior anomaly detection [4] [10]. Network traffic provides insight into all the communication a device is having on the network, including where it is sending data and from where it is receiving data [8]. This could be useful to detect a variety of attacks, such as, for example, if a Command and Control (C&C) server is trying to recruit the host to be part of a botnet. It is likely that this is the first presence of communication between the server and the infected host, which could signal a malware infection.

This research is intended to build on the prior work discussed, and broaden its application to a more general IoT platform. Another key distinction of this work is the design and use of custom-made parameterizable malware to aid in the evaluation of behavioral-based anomaly detectors.

网络流量数据也被证明是有效的特征集，可用于行为异常检测[4] [10]。网络流量提供了对所有网络流量提供了对一个设备在网络上的所有通信的了解，包括它在哪里发送数据和从哪里接收数据。数据以及从哪里接收数据[8]。这对检测各种攻击是很有用的。各种攻击，例如，如果一个指挥和控制（C&C）服务器试图招募主机成为僵尸网络的一部分。这很可能是服务器和受感染的主机之间第一次出现通信，这可能是恶意软件感染的信号。这可能是一个恶意软件感染的信号。这项研究的目的是在先前讨论的工作的基础上，扩大其应用范围，使之成为更普遍的物联网。其应用于更普遍的物联网平台。这项工作的另一个关键区别这项工作的另一个关键区别是设计和使用定制的可参数化的恶意软件来帮助对基于行为的异常检测器的评估。

3 Malware

Three types of malware are used in the experimentation, which together provide a variety in the breadth of the malware. Some of the malware are more computationally expensive on the device and issue many system calls, while others issue fewer system calls but exfiltrate more information at varying speeds, which contributes to higher overall network traffic. All of the malware used in this research is custom-made, which provides more freedom to create interesting types of malware that are diverse in their execution behavior, yet stay true to behaviors that would be exemplified in real malware samples. In addition, the malware used in this research each have varying degrees of freedom, that include the rate of data exfiltration, as well as the size of the data exfiltration. These degrees of freedom are important because they enable the malware to adapt to changing conditions on the host prompted by malware detection software, as well as provide a better basis to show situations where the malware detection is successful, and situations where it is less successful.

实验中使用了三种类型的恶意软件，它们共同提供了恶意软件的广度的多样性。一些恶意软件在设备上的计算成本较高，并发出许多系统调用，而其他恶意软件发出较少的系统调用，但以不同的速度渗出更多的信息，这这有助于提高整体网络流量。本研究中使用的所有恶意软件都是定制的，这为创建有趣的恶意软件类型提供了更多的自由。恶意软件的执行行为是多样化的，但又与真实的恶意软件样本中的行为保持一致。此外，本研究使用的恶意软件在这项研究中使用的恶意软件都有不同的自由度，其中包括数据渗出的速度。数据渗出的速度，以及数据渗出的大小。这些自由度是很重要的，因为它们可以使恶意软件的数据渗出率提高。这些自由度是很重要的，因为它们使恶意软件能够适应主机上的变化这些自由度很重要，因为它们使恶意软件能够适应恶意软件检测软件所提示的主机上不断变化的条件，并为显示恶意软件检测成功的情况提供更好的基础。以及它不太成功的情况.

The degrees of freedom for each type of malware will be further discussed in Section 6, where the classification results vary depending on the malware parameter settings.

每种类型的恶意软件的自由度将在第6节进一步讨论。第6节将进一步讨论，其中的分类结果取决于恶意软件的参数设置

3.1 Keylogger

The first type of malware used in this research is a keylogger, which tracks all of the key presses on the device, and saves them to a buffer. The key presses can then be exfiltrated to another machine. The speed and the size of the exfiltration are set by user-defined parameters, which means an adversary can adjust the malware to attempt to avoid detection by an anomaly detector. The speed of the exfiltration refers to the rate of exfiltration, which can be defined as the interval at which the buffer of key presses, or a subset of them, are removed from the buffer and sent to a remote machine. The size of the exfiltration, in this case, refers to the number of key presses to send with each exfiltration packet to a remote machine. This allows the adversary to adapt the malware in order to evade detection and possible mitigation strategies implemented by the malware detection.

本研究中使用的第一种类型的恶意软件是键盘记录器，它跟踪设备上所有的设备上的所有按键，并将其保存到一个缓冲区。这些按键可以然后可以渗出到另一台机器上。渗出的速度和大小由用户定义的参数设定。是由用户定义的参数设置的，这意味着对手可以调整恶意软件，以试图避免被异常检测器发现。渗透的速度渗出的速度是指渗出的速度，它可以定义为按键的缓冲区或其中的一个子集被移除的时间间隔从缓冲区中取出并发送到远程机器上。渗透的大小，在这种情况下，是指按下键的数量。在这种情况下，指的是与每个渗出数据包一起发送的按键数量到远程机器。这使得对手能够调整恶意软件，以便逃避检测和恶意软件实施的可能缓解策略。检测。

3.2 Ransomware

勒索软件

The ransomware malware uses the Python cryptography library to encrypt a file system, and exfiltrate the contents to a remote host on the network. The exfiltration happens during the encryption process. An encryption key is created, and then the malware traverses the file system. For each file it finds, it first sends a copy of it to the remote host using secure copy (scp), and then encrypts it and continues to the next file. The decryption functionality is also provided. The location to save the file on the remote host is user-specified. The user can also provide an exfiltration interval, which will insert a delay in between exlfiltrating individual files. Similar to the rate of exfiltration on the keylogger, this allows an adversary to become more inconspicuous, since copying and sending files at a slower rate will likely draw less attention to the malware than performing the same process rapidly.

该恶意软件使用Python密码学库来加密一个文件系统，并将其内容渗透到网络上的远程主机。该渗出发生在加密过程中。一个加密密钥被创建。然后，恶意软件穿越文件系统。对于它发现的每个文件，它首先将它首先使用安全拷贝（scp）将它的副本发送到远程主机，然后对其进行加密并继续下一个文件。继续到下一个文件。同时提供解密功能。在远程主机上的在远程主机上保存文件的位置是用户指定的。用户还可以提供一个渗出间隔，这将在渗出单个文件之间插入一个延迟。个别文件之间插入一个延迟。与键盘记录器的渗出率相似，这使得与键盘记录器的渗出率类似，这允许对手变得更加不显眼，因为以较慢的速度复制和发送文件因为以较慢的速度复制和发送文件可能会比快速执行相同的过程更容易引起对恶意软件的注意。

3.3 Cryptominer

密码器

Lastly, the cryptominer malware is a simple coin mining script, that runs a mining simulation to emulate the computational cost of a real cryptominer. The user specifies a remote host to send the new hash that was mined on the host after the completion of the mining process. Similar to the ransomware, the user can specify an exfiltration interval, which will insert a delay between the mining process and the exfiltration of the calculated hash. While the behavior of the two previous malware discussed is often easily traceable by both system call data and network traffic data, the cryptominer malware is more easily traceable by system call data, due to the heavy computation cost of the mining process, and the relatively few packets being sent over the network as a result of its execution.

最后，cryptominer恶意软件是一个简单的硬币挖掘脚本，它运行一个采矿模拟，以模拟真正的加密器的计算成本。用户用户指定一个远程主机来发送在该主机上挖出的新哈希值。挖矿过程完成后。与勒索软件类似，用户可以指定渗出时间间隔。用户可以指定一个渗出时间间隔，这将在挖矿过程和渗出数据之间插入一个延迟。挖掘过程和计算出的哈希值的渗出之间插入一个延迟。虽然前面讨论的两个恶意软件的行为前面讨论的两个恶意软件的行为通常很容易通过系统调用数据和网络流量数据进行追踪。和网络流量数据，而密码器恶意软件则更容易被系统调用数据追踪。系统调用数据更容易追踪，因为挖掘过程的计算成本很高，而且通过网络发送的数据包相对较少。由于其执行的结果，在网络上发送的数据包相对较少。

4 Anomaly Detection Model

异常情况检测模型

The feature sets for the anomaly detection model are extracted from sequences of system calls and network traffic flows on the Pi-Router. This includes any system calls executed on the Pi-Router during data collection, as well as any packets sent to or from the Pi-Router during its execution.

异常检测模型的特征集是从Pi-Router上的系统调用和网络流量序列中提取的。Pi-Router上的系统调用和网络流量的序列中提取。这包括在数据收集期间在Pi-Router上执行的任何数据收集期间在Pi-Router上执行的任何系统调用，以及任何数据包在Pi-Router的执行过程中被送入或送出。

4.1 System Calls

System calls indicate the activity of each running process on the machine. Therefore, when a malware sample starts to execute, its process will likely make different system calls than previously seen and will be useful in detecting an anomaly present on the machine.

The pre-processing step for system calls is similar to [2], where a sequence of system calls collected in a window size of length L is treated as an observation. Then, a bag-of-n-grams approach [3], [1], [9] is used to group the system calls and create the feature vector $\mathbf{x} \in \mathbb{R}^p$. This can be described as the number of times a system call n-gram sequence was observed in an observation window of length L [8]. An n-gram length of $n = 2$ was used when the results of the classifier were compiled, although this is another parameter of the data processing code that can be changed by the user. Other values of n, such as $n = 3$, were used in the experimentation phase as well, but did not yield significantly better results.

系统调用表明机器上每个运行进程的活动。因此，当一个恶意软件样本开始执行时，它的进程可能会做出与以前不同的系统调用，这将有助于检测机器上的异常情况。异常的机器。系统调用的预处理步骤类似于[2]，即在一个窗口大小的范围内收集一连串的系统调用。在一个长度为L的窗口大小中收集的系统调用序列被视为一个观察。然后，采用包-n-grams方法[3], [1], [9]对系统调用进行分组，并且创建特征向量x 2 Rp。这可以被描述为系统调用n-gram序列在一个长度为L的观察窗口中被观察到的次数。L [8]. 当分类器的结果被编译时，使用了n=2的n-gram长度。编译时使用了n = 2的n-gram长度，尽管这是数据处理代码的另一个参数，可以由用户改变。可以由用户来改变。其他的n值，如n=3，在实验阶段也被使用过，但并没有起到作用。实验阶段也使用了n的其他值，如n=3，但并没有产生明显更好的结果。

4.2 Network Traffic

Network traffic features are extracted from network flows collected by CICFlowMeter. CICFlowMeter is a package in the Python Package Index (PyPi) that listens to network traffic on a device, generates bidirectional network flows, and then extracts features from these flows. In the network traffic data collected, one packet sent or received by the Pi-Router counts as one observation.

One issue that arises when using network traffic as a feature set is that many of the packets that are sent by normal non-malicious applications on the Pi-Router are also included in the malware dataset. This results in these benign packets being labeled as malicious data, which yields an inseparable dataset [8]. This issue is resolved by grouping the network traffic data into m-second intervals, which results in more finely-grained malware traces that distinguish them from the benign traces [8]. The key idea here is to find the optimal value of m so that the bin is large enough to collect enough packets to determine their source or destination program, but small enough so that packets from other programs are excluded, making each packet sequence more distinct. As with the bag-of-n-grams approach used in the system call processing, the value of m is a tunable parameter that can be adjusted by the user. During the experimentation phase of this work, a few different values of m were used, and it was found that smaller values of m are better for classifying our malware samples.

网络流量特征是从CICFlowMeter收集的网络流量中提取的。CICFlowMeter是Python软件包索引（PyPi）中的一个软件包，它监听设备上的网络流量，生成双向网络流量，然后从这些流量中提取特征。CICFlowMeter是Python包索引（PyPi）中的一个包，它监听设备上的网络流量，生成双向的网络流量，然后从这些流量中提取特征。在收集的网络流量数据中，一个数据包Pi-Router发送或接收的一个数据包算作一个观察值。在使用网络流量作为特征集时，出现的一个问题是，很多PiRouter上由正常的非恶意应用程序发送的数据包也包括在恶意软件数据集中。这导致这些良性的数据包被标记为恶意数据，这产生了一个不可分割的数据集。[8]. 这个问题通过将网络流量数据分组为m秒间隔来解决。间隔，从而产生更精细的恶意软件痕迹，以区别于它们与良性痕迹的区别[8]。这里的关键思想是找到最佳值m的最佳值，以便收集足够大的数据包来确定其源程序或目的地程序，但又要足够小，以便将其他程序的数据包排除在外，使每个数据包的程序的数据包被排除在外，使每个数据包的序列更加明显。如同在系统调用处理中使用的包-n-grams方法，m的值是一个是一个可调整的参数，可以由用户来调整。在这项工作的实验阶段在这项工作的实验阶段，使用了几个不同的m值，结果发现，较小的m值对分类来说更好。较小的m值更有利于对我们的恶意软件样本进行分类。

4.3 Principal Component Analysis

After the pre-processing mentioned above was applied to the datasets, approximately 2600 features were extracted from the system call dataset, and 237 features were extracted from the network traffic dataset. Combining these features brings the total number of features in the feature set to approximately 2800 features. Principal Component Analysis (PCA) was then used to reduce the dimensionality of the created feature space, similar to work by [8]. The number of components to use was determined by calculating the explained variance of the components, and selecting the number of components that explain at least 95% of the variance in the dataset. Figure 3 depicts the number of components needed to explain 95% of the variance for the Cryptominer combined feature set, which in this specific case is four components.

在对数据集进行上述预处理后，从系统调用数据集中提取了约2600个特征，从网络流量数据集中提取了237个特征。结合这些特征使得特征集中的特征总数达到约2800个特征。然后使用主成分分析（PCA）来降低创建的特征空间的维度，与[8]的工作相似。使用的成分数通过计算各成分的解释方差来确定使用的成分数。组成部分的解释方差，并选择至少能解释数据中95%方差的组成部分数量。95%的数据集方差。图3描述了解释95%的方差所需的成分数量，对于Cryptominer组合特征集，在这个特定的情况下，需要四个成分。

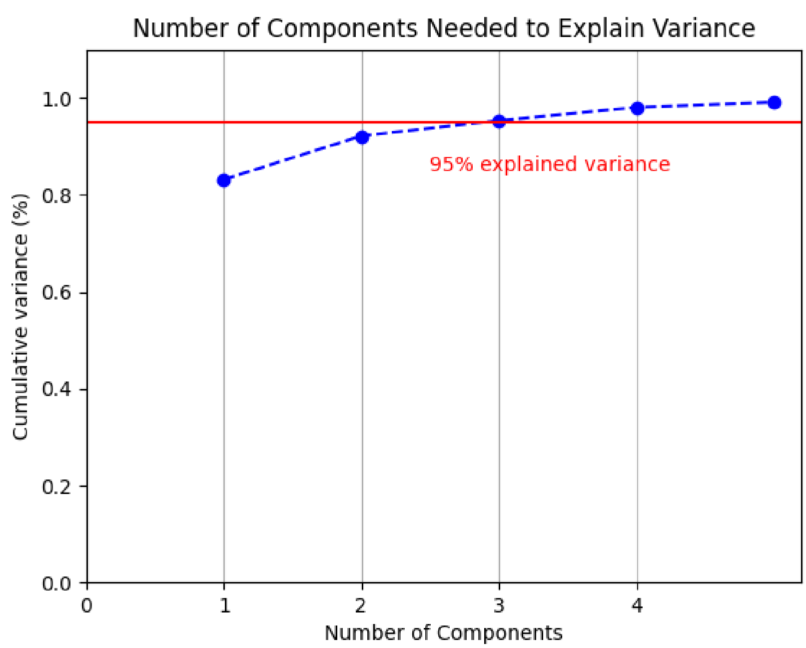

Fig. 3. Explained Variance for the Cryptominer Combined Feature Set

The anomaly detection model uses a Support Vector Machine (SVM), which finds a hyperplane that separates the training data from origin with the largest-possible margin. The objective function is thus maximizing the margin, or the distance from the origin to the hyperplane.

异常检测模型使用支持向量机（SVM），它可以找到一个超平面，以最大的余量将训练数据与原点分开。找到一个超平面，以最大的余量将训练数据与原点分开。因此，目标函数是最大限度地提高余量，或从原点到超平面的距离。从原点到超平面的距离。

5 Experimental Setup

5.1 Pi-Router

Most of the setup for the experiments conducted is focused on the Pi-Router. The Pi-Router is a Raspberry Pi 3 running Raspberry Pi OS (formerly Raspbian), which is based on the Debian Linux distribution. This Raspberry Pi has been configured to act as a wireless access point by using the software package *hostapd*, which is a user-space daemon that allows the network interface card (NIC) to act as an access point. A Flask web server running as a Linux service was created on the Pi-Router to act as a portal for users to configure the router and track the health of the network by running the malware detection software and showing the results. Figure 4 shows how the Network Health section might look to the user on the Pi-Router web server, where the graphs show the results of running the SVM on current system call and network traffic data.

所进行的实验的大部分设置都集中在Pi-Router上。该Pi-Router是一个运行Raspberry Pi OS（以前是Raspbian）的Raspberry Pi 3。它是基于Debian Linux发行版的。这个树莓派已经被配置为通过使用软件包hostapd配置成一个无线接入点。这是一个用户空间的守护程序，允许网络接口卡（NIC）充当接入点。作为一个接入点。在Pi-Router上创建了一个以Linux服务形式运行的Flask网络服务器，以作为Pi-Router的接入点。Pi-Router上创建了一个Flask网络服务器，作为用户配置路由器和跟踪网络健康状况的门户。通过运行恶意软件检测软件和显示结果来跟踪网络的健康状况。结果。图4显示了网络健康部分在Pi-Router网站上对用户来说是什么样子的图4显示了Pi-Router网络服务器上的用户可能看到的网络健康状况，其中的图表显示了在当前系统调用下运行SVM的结果，以及显示了网络健康状况。图中显示了在当前系统调用和网络流量数据上运行SVM的结果。

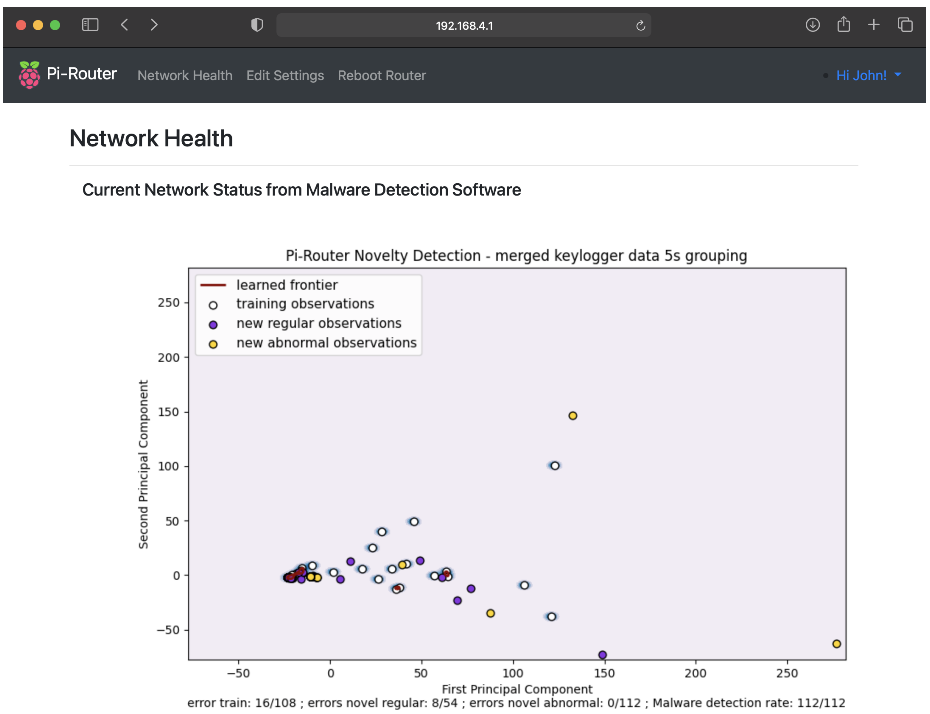

Fig. 4. Example Screenshot of the Pi-Router UI

The web server presents the essence of the usability of this system, in that it allows users to monitor the behavior and overall health of their network and track anomalous behaviors caused by malware.

网络服务器是这个系统可用性的精华所在，因为它允许用户监控其网络的行为和整体健康状况，并跟踪由恶意软件引起的异常行为。追踪由恶意软件引起的异常行为。

5.2 Pi-Network and Connected Devices

The Pi-Router's network is the Pi-Network, which has two other hosts connected to it currently: a Pi-Camera and the attacker machine. The Pi-Camera in this work is simply another host on the network that generates traffic unrelated to the router. The attacker is a laptop running Ubuntu 20.04 and is responsible for receiving all of the exfiltrated data sent by the malware. This includes key presses sent from the keylogger, files copied by the ransomware, and hashes sent by the cryptominer. As a result, the relevant network traffic would often originate from the Pi-Router or the attacker, and be received by the other. Most of the exfiltration communication was one-way, from the Pi-Router to the attacker.

Pi-Router的网络是Pi-Network，目前有两台主机与之相连：Pi-Camera和攻击者机器。目前有两台主机连接到它：一台Pi-Camera和一台攻击者的机器。在这项工作中，Pi-Camera只是网络上的另一台主机，它产生的流量与攻击者无关。工作中的Pi-Camera只是网络上的另一台主机，它产生的流量与路由器无关。路由器无关。攻击者是一台运行Ubuntu 20.04的笔记本电脑，负责接收所有的外泄数据。负责接收恶意软件发送的所有渗出数据。这包括键这包括从键盘记录器发送的按键、赎金软件复制的文件和密码器发送的哈希值。编码器发送的哈希值。因此，相关的网络流量往往来源于来自Pi-Router或攻击者，并被对方接收。大多数的渗出通信是单向的，从Pi-Router到攻击者。

6 Experimental Results

The classification experiments were conducted with the three types of malware, each with different window sizes, or values of L, used in the SVM. Each of these was carried out for the three types of features: system calls, network traffic, and a combination of the two. In Figure 5, we show a visualization of the classification process using the SVM, in which the first two principal components derived from PCA are shown. In this example, the SVM is classifying keylogger data using a five-second window size. The goal is to have as many new abnormal observations outside of the learned frontier as possible, because they will be easier to identify and isolate from the benign data and thus be better indicators of a possible malware infection.

After some experimentation, it was found that a $L = 5$ seconds window size was optimal for classification accuracy. Other window sizes yielded suboptimal results for each feature set. Figure 6 shows the comparison between F1 score and window size for the three types of malware, in which we see that for all three types of malware, a $L = 5$ seconds window size had the best classification performance of the 6 different window sizes.

In addition, we found that using a classifier trained on both the system call feature set and network traffic feature set generally outperformed classifiers trained on each type of data individually. While system call data can provide a lot of information relating to the behavior of a process, the network traffic data often augments that behavior data and makes the classification results more conclusive. Figure 7 shows an example of the merged data providing better features than system call and network traffic data individually using cryptominer data with a five second window size. In this case, both the classifier trained on system call data and the classifier trained on network traffic data each only had a mean AUC value of 0.75 or less, while the combined data had a mean AUC value of 0.98, which is a significant improvement.

In all three types of malware, the combined dataset of system calls and network traffic generally outperformed the system call and network traffic classifiers individually.

The parameters provided in the malware samples greatly affect the classification effectiveness of the SVM. Specifically, the rate of exfiltration parameter

分类实验是用三种类型的恶意软件进行的。每一种都有不同的窗口大小，或L值，用于SVM。每一个都是针对三种类型的特征进行的：系统调用、网络流量，以及两者的组合。两者的组合。在图5中，我们展示了一个使用SVM的分类过程的可视化图。在图5中，我们展示了使用SVM进行分类的可视化过程，其中显示了从PCA得出的前两个主成分。PCA得出的前两个主成分。在这个例子中，SVM使用五秒钟的窗口大小对键盘记录器数据进行分类。五秒钟的窗口大小。我们的目标是让尽可能多的新的异常观测值在学习边界之外。尽可能多地出现在学习边界之外，因为它们将更容易被识别，并从良性数据中分离出来。和从良性数据中分离出来，从而更好地指示可能的恶意软件的感染。经过一些实验，我们发现L=5秒的窗口大小是分类准确性的最佳选择。其他窗口大小产生了次优的的结果。图6显示了三种类型的恶意软件的F1得分和窗口大小之间的比较和三种类型的恶意软件的窗口大小的比较，其中我们看到，对于所有的我们看到，对于所有三种类型的恶意软件，L=5秒的窗口大小在6种不同的窗口大小中具有最佳的分类性能。在6种不同的窗口大小中，L = 5秒的分类性能最好。此外，我们发现，使用在系统调用特征集和网络流量特征集上训练的分类器调用特征集和网络流量特征集训练的分类器通常优于在每种数据上单独训练的分类器。在每种数据上单独训练的分类器。虽然系统调用数据可以提供系统调用数据可以提供很多与进程行为有关的信息，而网络流量数据往往能增强该行为数据，并使分类结果更加确凿。图7显示了一个合并数据的例子，它提供了比系统调用和网络流量数据更好的比单独使用密码器的系统调用和网络流量数据提供更好的特征。窗口大小为5秒的数据。在这种情况下，在系统呼叫数据上训练的分类器和在网络流量数据上训练的分类器都能提供更好的特征。系统呼叫数据训练的分类器和网络流量数据训练的分类器各自只有平均AUC值为0.75或更低，而综合数据的平均AUC值为0.98，这是一个重大的改进。在所有三种类型的恶意软件中，系统呼叫和网络流量的组合数据集的表现普遍优于系统呼叫和网络流量分类器个人。

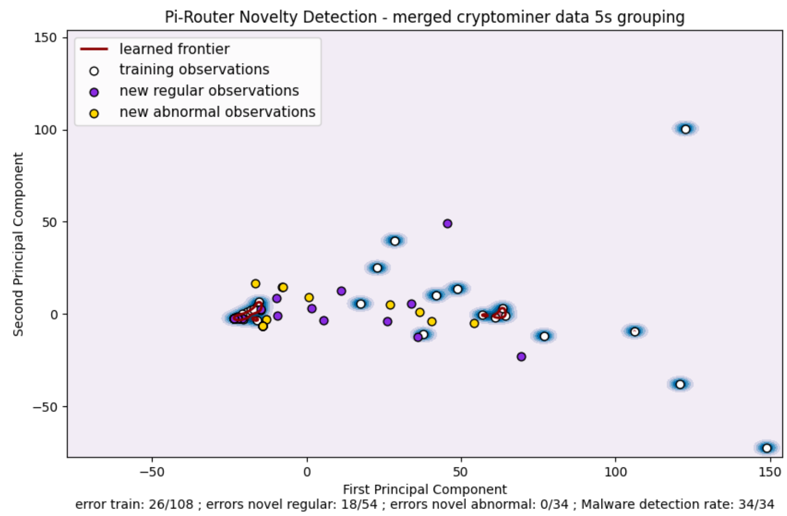

Fig. 5. Visualization of first two Principal Components on Cryptominer data using a 5s window size

in each malware sample is instrumental in its detection. In general, as one might expect, the faster the exfiltration rate, the easier the detection process becomes. Rapidly issuing system calls and sending packets results in more conspicuous malware, while slower malware that is idle in between actions for longer periods of time is much harder to detect. Figures 8-10 demonstrate this idea using ROC-AUC results for each malware with different parameter settings, again all with a window size of 5 seconds. Here, the results indicate that the keylogger exfiltration rate did play an important part in the classification results, but the number of system calls it made and the number of packets it sent still made the malware easy to detect. In contrast, the exfiltration rates of the other two malware were useful in their ability to elude the malware detection, since as the malware continued to run with slower exfiltration rates they were much harder to detect.

The differences in results shown in Figures 8-10 demonstrate the impact the degrees of freedom have on the detection of the malware. By changing one tunable parameter on the malware, they can be either easily detectable with a high AUC value, or hardly detectable, with an AUC value similar to chance classification. By using these custom-made malware samples, we can provide a larger set of data with more variations to show where the malware detection excels, and where it fails, to improve the overall classification process.

The co-design of parameterizable malware and anomaly detectors is useful to design a robust detector that can detect elusive, inconspicuous, malware.

恶意软件样本中提供的参数大大影响了SVM的分类效果。具体来说，每个恶意软件样本中的渗出率参数对其检测很有帮助。一般来说，正如人们所期望的那样预计，渗出率越快，检测过程就越容易。快速发布系统调用和发送数据包的结果是更明显的恶意软件，而速度较慢的恶意软件在较长的时间内处于闲置状态，则更难被发现。图8-10展示了这一想法，使用每个恶意软件的ROC-AUC结果与不同的参数设置，同样都是窗口大小为5秒。这里，结果表明，键盘记录器渗出率在分类结果中确实发挥了重要作用，但它所做的系统调用的数量和它所做的系统调用的数量和发送的数据包的数量仍然使恶意软件很容易被发现。相比之下，其他两个恶意软件的渗出率恶意软件在躲避恶意软件检测的能力方面是很有用的，因为随着恶意软件继续以较慢的渗出率运行，他们就更难来检测。图8-10中显示的结果差异表明自由度对恶意软件检测的影响。通过改变一个恶意软件上的一个可调整参数，它们可以很容易地被检测到，并具有高的AUC值，或几乎无法检测，AUC值类似于偶然的分类。通过使用这些定制的恶意软件样本，我们可以提供一个更大的数据集，有更多的变化。通过使用这些定制的恶意软件样本，我们可以提供更大的数据集，有更多的变化，以显示恶意软件检测的擅长的地方，以及失败的地方，以改善整个分类过程。
可参数化的恶意软件和异常检测器的共同设计是有用的设计一个强大的检测器，可以检测到难以捉摸的、不显眼的恶意软件。

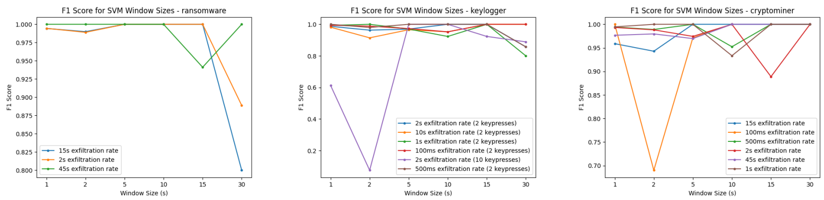

Fig. 6. A comparison of F1 Scores using different Window Size values (values of L)

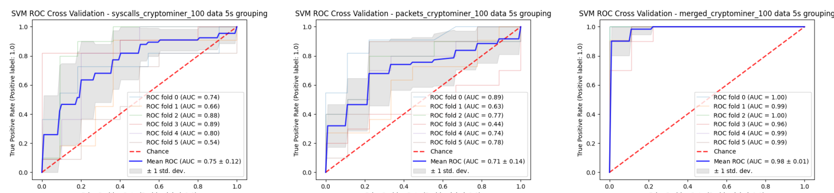

Fig. 7. As with the other types of malware, the combined feature set outperforms the feature sets comprised of system call data and network traffic data individually. Here, we demonstrate this using the Cryptominer malware with a Window Size of 5s and an exfiltration rate of 100ms. The first figure uses system call features, the second uses network traffic features, and the third uses both types of features.

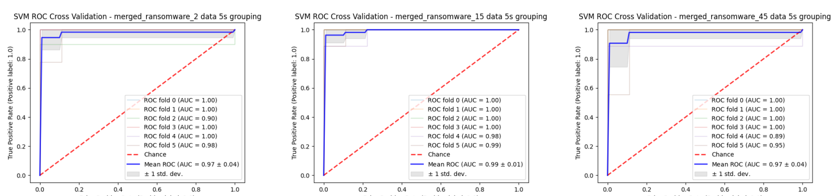

Fig. 8. ROC Curves for Ransomware with a window size of 5 seconds and a 2s exfiltration rate, a 15s exfiltration rate, and a 45s exfiltration rate.

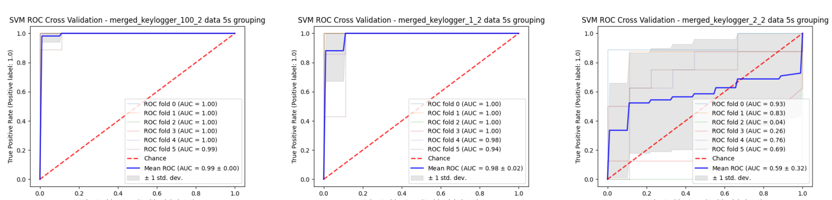

Fig. 9. ROC Curves for Keylogger with a window size of 5 seconds and a 100ms exfiltration rate, a 1s exfiltration rate, and a 2s exfiltration rate.

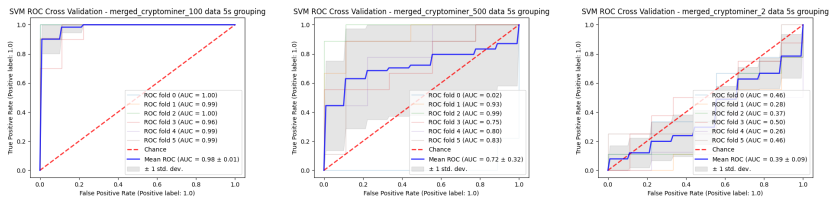

Fig. 10. ROC Curves for Cryptominer with a window size of 5 seconds and a 100ms exfiltration rate, a 500ms exfiltration rate, and a 2s exfiltration rate.

7 Conclusion & Future Work

This research focuses on creating parameterizable malware, and building an anomaly detector to detect the malware using machine learning. The malware lives on a Linux router in an IoT Ecosystem. The ecosystem was designed and created to be a testbed to enable evaluation of anomaly detectors using custom-made malware. The malware is useful because it allows for more varied data to be generated while still emulating its respective malware family. This method of malware detection is useful for IoT devices because it does not rely on prior knowledge of the environment of the device, and is also very resource efficient, which is essential if the malware detection is going to run on the IoT device itself.

In this research, we created a real IoT Ecosystem, which in this work focuses primarily on the Pi-Router. The Pi-Router is a fully-functional wireless access point that provides a network that devices can connect to locally. We also created malware that is fully-functional to run on the Pi-Router. The malware provide several degrees of freedom with which to create varied data and evaluate the anomaly detection model under different environment conditions. Lastly, we demonstrated the creation of a behavioral anomaly detection system, which is designed to detect malicious software running on the Pi-Router, and on IoT devices in general. The anomaly detection system was trained on three types of data: system calls, network traffic data, and a combined dataset comprised of system calls and network data. Our results indicated that a classifier trained on the combined data with a window size of L=5 seconds generally outperformed the other classifiers that were used. We also found that the ability of the malware samples to elude the anomaly detector was generally dependent on the exfiltration rate of the malware.

We found that this classifier is very useful for malware detection on IoT devices, and that the custom-made malware is useful for identifying situations where the SVM was successful, and situations where the SVM was less successful. The degrees of freedom of the malware were significant in providing a way to quickly generate different data from the same malware family to test the classifier's ability to adapt to changing malware behavior. We plan to continue and expand on this research by using the data obtained from the IoT Ecosystem to train a generator and a discriminator in a Generative Adversarial Network

这项研究的重点是创建可参数化的恶意软件，并建立一个异常检测器，用机器学习来检测恶意软件。该恶意软件生活在一个物联网生态系统的Linux路由器上。该生态系统被设计和该生态系统被设计和创建为一个测试平台，以便使用定制的恶意软件对异常检测器进行评估。恶意软件是有用的，因为它允许更多不同的数据生成，同时还能模拟其各自的恶意软件系列。这种方法恶意软件检测对物联网设备很有用，因为它不依赖于设备环境的事先对设备环境的了解，而且也非常节省资源。如果恶意软件检测要在物联网设备上运行，这一点至关重要。本身。在这项研究中，我们创建了一个真正的物联网生态系统，在这项工作中主要关注Pi-Router。Pi-Router是一个功能齐全的无线接入点，提供一个设备可以连接到本地的网络。我们还创建了可以在Pi-Router上运行的全功能的恶意软件。这些恶意软件提供了几个自由度，可以创建不同的数据，并在不同的环境条件下评估异常检测模型。最后。我们展示了一个行为异常检测系统的创建，该系统该系统旨在检测运行在Pi-Router上的恶意软件，以及一般物联网上的恶意软件。设备上运行的恶意软件。异常检测系统是在三种类型的数据上进行训练的数据：系统调用，网络流量数据，以及由系统调用和网络数据组成的组合数据集。系统调用和网络数据组成的组合数据集。我们的结果表明，一个在综合数据上训练的分类器窗口大小为L=5秒的综合数据上训练的分类器的表现普遍优于其他的分类器。我们还发现，恶意软件样本躲避异常检测器的能力通常取决于恶意软件的渗出率。

我们发现，这个分类器对物联网上的恶意软件检测非常有用。设备上的恶意软件，而定制的恶意软件对于识别SVM成功的情况和SVM不太成功的情况很有帮助。识别SVM成功的情况，以及SVM不太成功的情况。恶意软件的自由度在提供一种方法方面非常重要恶意软件的自由度非常重要，它提供了一种从同一恶意软件家族快速生成不同数据的方法，以测试分类器适应变化的恶意软件行为的能力。我们计划继续并通过使用从物联网生态系统获得的数据，继续并扩大这项研究。在生成对抗网络（GAN）[5]中训练一个生成器和一个鉴别器。最初向GAN提供真实数据可以节省训练时间。在这一点上，生成器将有一个有用的基准，可以开始创建真实但虚假的数据。我们希望，这将提供一个甚至更好的分类器来检测物联网设备上运行的恶意软件。异常检测代码可在GitHub上找到。仓库中的README文件仓库中的README文件有关于如何运行用于生成本文讨论的图表和结果的代码的说明。本文讨论的图表和结果。

(GAN) [5]. Feeding the real data to the GAN initially will save training time, at which point the generator will have a useful benchmark from which to start creating realistic but fake data. We hope that this will provide an even better classifier to detect malware running on IoT devices.

The anomaly detection code is available on GitHub. The README file in the repository has instructions on how to run the code used to generate the graphs and results discussed in this paper.

Bibliography

[1] An, N., Duff, A., Naik, G., Faloutsos, M., Weber, S., Mancoridis, S.: Behavioral anomaly detection of malware on home routers. In: 2017 12th International Conference on Malicious and Unwanted Software (MALWARE). pp. 47–54 (2017). https://doi.org/10.1109/MALWARE.2017.8323956

[2] An, N., Duff, A., Noorani, M., Weber, S., Mancoridis, S.: Malware anomaly detection on virtual assistants. In: 2018 13th International Conference on Malicious and Unwanted Software (MALWARE). pp. 124–131 (2018). https://doi.org/10.1109/MALWARE.2018.8659366

[3] Canzanese, R., Mancoridis, S., Kam, M.: System call-based detection of malicious processes. In: 2015 IEEE International Conference on Software Quality, Reliability and Security. pp. 119–124 (2015). https://doi.org/10.1109/QRS.2015.26

[4] Doshi, R., Apthorpe, N., Feamster, N.: Machine learning ddos detection for consumer internet of things devices. 2018 IEEE Security and Privacy Workshops (SPW) (May 2018). https://doi.org/10.1109/spw.2018.00013, http://dx.doi.org/10.1109/SPW.2018.00013

[5] García-Teodoro, P., Díaz-Verdejo, J., Maciá-Fernández, G., Vázquez, E.: Anomaly-based network intrusion detection: Techniques, systems and challenges. Computers & Security **28**, 18–28 (02 2009). https://doi.org/10.1016/j.cose.2008.08.003

[6] Hadar, N., Siboni, S., Elovici, Y.: A lightweight vulnerability mitigation framework for iot devices. In: Proceedings of the 2017 Workshop on Internet of Things Security and Privacy. p. 71–75. IoTS&P '17, Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3139937.3139944, https://doi.org/10.1145/3139937.3139944

[7] Noorani, M., Mancoridis, S., Weber, S.: Automatic malware detection on an alexa-pi iot device. In: 35th Annual Computer Security Applications Conference (ACSAC 2019) (2019)

[8] Noorani, M., Mancoridis, S., Weber, S.: On the detection of malware on virtual assistants based on behavioral anomalies (2019)

[9] Sekar, R., Bendre, M., Dhurjati, D., Bollineni, P.: A fast automaton-based method for detecting anomalous program behaviors. In: Proceedings 2001 IEEE Symposium on Security and Privacy. S P 2001. pp. 144–155 (2001). https://doi.org/10.1109/SECPRI.2001.924295

[10] Terzi, D.S., Terzi, R., Sagiroglu, S.: Big data analytics for network anomaly detection from netflow data. In: 2017 International Conference on Computer Science and Engineering (UBMK). pp. 592–597 (2017). https://doi.org/10.1109/UBMK.2017.8093473