Malware

Malware poses a significant threat to many computers and devices and can be very tricky and difficult at detecting and defending against. There is a plethora of ways to detect malware but when it comes to malware detection, there are just as many ways to subvert those detection mechanisms meant to provide security. The majority of malware detection comes in the form of other software and programs that itself can be penetrated and avoided by malware attacks which then renders the malware untraceable on certain machines and/or there is no defense that the malware can’t subvert when it comes to certain types of machines, primarily the average computer used by the average citizen who is not educated on properly securing their machine. There are some ways though in which malware can be detected without using machine specific software or new techniques that are being developed to more effectively and efficiently detect malware.

The first method used to detect malware doesn’t utilize machine specific software that be easily subverted. This method utilizes a device that monitors the machine power usage for anomalies in power consumption that would allow it to tell if there is malware on the machine [1]. The device used is Data Acquisition system (DAQ) which was connected to the computer power supply. The computer used in this experiment was a Dell OptiPlex 755 with a clean installation of 32-bit windows 7. The team used software on a separate secure machine that received data from the DAQ (a separate machine was used to ensure the integrity of the data monitoring software). The team developed and used their own data monitoring software because it provided many advantages to the generic software they originally wanted to use.

The architecture of the neural network used is very complex but this is a simple overview. The first section is PE Metdata and PE Import which are side by side, these feed into a single layer, the input payer (PE format is used to structure windows executable files). The input layer feeds into the hidden layer, and then yet another hidden layer. On the other side up next to the PE layers, there are Opcodes which feed into an input layer. This feeds into a convolutional layer and then a pooling layer. The pooling layer and second hidden layer from the first part feed into a single layer, the dropout layer, and finally feed into the last layer, the output layer. The power data from the
computer was collected in two different scenarios, the first being the machine running in normal conditions with no malware on the machine. The second scenario where data was collected was when the machine was running with malware demonstrating anomalous behavior. They ran many experiments and collected data for 3 events (idle, opening Internet Explorer, and booting/rebooting) and with the two rootkits. The results were based on what voltage components within the machine run on. The +3.3V Rails did not demonstrate any significant change in power usage that showed there was malware on the machine. Same as the +3.3V Rails, the +5V Rails demonstrated the same behavior when the machine was and wasn’t infected. When comparing the +12V Rails on the other hand, they were able to notice significant difference in power usage from the motherboard when using Internet Explorer but these were not noticeable on the +12V Rails on the CPU. The only effective way of determining if there was a virus or not was by analyzing the power usage data of the +12V Rails of the motherboard when running Internet Explorer, but not during boot/reboot, idle, or on any of the other Rails on the machine. The results did show that there is a trace left by malware on all system data that was collected, but the only one noticeable by the naked eye was the +12V motherboard Rails. This method, with more research and experimentation, could prove to be an extremely useful way to monitor for malware on general-purpose computers.

The next method being used to detect malware uses machine learning and although neural networks are not really used yet in the detection of malware, there are many trying to get neural networks there [2]. Neural networks are machine learning constructs that are biologically motivated and uses non-linear feature transformation. These parameters are trained using gradient decent procedures. Many of these layers can be connected together by using the output of any given layer as the input to another layer. In practice, increasing the number of layers can actually improve the performance of the neural network. Their experiment used four ways to classify the malware they collected, hybrid neural network, feedforward, convolutional, and a support vector machine. Out of these four methods, the hybrid neural network was the most accurate and effective at detecting and classifying malware, scoring an F1-score value of 0.92 and scoring 0.93 for precision and recall values. One big way attackers avoid detection is through reordering instructions but with the hybrid neural network with values of up to 0.90. The neural network was very robust and resilient to bogus code and reordering and was able to outperform the other methods of machine learning when it came to malware detection but there are still some flaws/obstacles.
The last form of malware detection also utilizes a form of deep neural networks. Deep Neural Networks (DNNs) can be very good at detecting malware but there are still ways in which it can be subverted and one of those more non-conventional ways is using the networks learning algorithms against itself: by figuring out the algorithm the network is using, the attacker can put any malware they want through the network to avoid detection. To avoid such scenarios from happening, teams can increase the effectiveness by adding in potential blind spots to the training and known samples which is called adversarial training [3]. To do this, they can enhance the model complexity which is already inherently complex by adding a defensive distillation mechanism which gives the DNN two samples of data and trains it to be able to tell which one is distilled from the other. This allows the DNN to become less sensitive to adversarial samples and allows it to be more accurate. The next way is through random feature nullification which adds a layer between the input and the hidden layer. This acts as a source of randomness for both training and testing randomly nullifying features within the input. These methods prove to be very effective at defending against adversaries and current defense methods still prove to be quite vulnerable to adversarial samples. Using these methods and the proposed Random Feature Nullification, it has proven that it would be impossible for an attacker to create any specific adversarial sample that the DNN would misclassify showing that it correctly classified all samples that went through it. This specific DNN had negligible inaccuracy compared to other defenses that rely on generating adversarial samples that are model specific. The proposed method improved model resistance as well.

These 3 forms of malware protection and resistance each have their advantages and disadvantages. The power consumption method can detect malware in a device but it cannot detect the exact kind of malware, only that there is malware within the machine. It also is only capable of detecting it if it is monitoring +12V Rails on the motherboard using Internet Explorer or another web application. Although it can be detected on other voltages and in different sectors of the computer, it is very difficult to do. This can prove very effective though in the sense that it could avoid malware that is programmed to subvert malware detection software because this method is very unique and vastly different from any other generic form of malware detection. The next method proposed using neural networks for malware detection and this one proved to be extremely promising. This method was able to detect and classify malware with up to 93% accuracy but was still susceptible to certain adversarial attacks that could render it relatively useless. The last method was the most promising. It used a special kind of deep neural network that also had blank spots added to it that created nullifications, allowing it to detect and classify malware with 100% accuracy.
when given malware that was picked out for it with negligible problems. All three of these methods could be used in the future for far more advanced and far more effective malware detection but for right now, these methods need more testing and research to determine how truly effective they would be in the real world and how fast they can work when it comes to protecting the public and organizations vast networks of servers and machines.
