Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 6.017

IJCSMC, Vol. 6, Issue. 5, May 2017, pg.175 – 182

An Effective Approach to Detect Malware that Exploit Information Hiding using Artificial Intelligence in Android Devices

Aswini S[1], D.Palanikkumar[2], T.Palani Raja[3]

[1]PG Student, Computer Science and Engineering, Nehru Institute of Technology, Coimbatore, India
[2]Professor, Computer Science and Engineering, Nehru Institute of Technology, Coimbatore, India
[3]Assistant Professor, Computer Science and Engineering, Nehru Institute of Technology, Coimbatore, India
[1] saswini38@gmail.com; [2] palanikkumard@gmail.com; [3] palaniraja1987@gmail.com

Abstract— Malware is a found to be a big threat in computing world. It continues to grow and evolve in complexity. Modern malware hide from static and dynamic analysis tools using advanced techniques. The existing system uses classification based and regression based approach for detection. The proposed system utilizes the classification based approach and regression based approach for detection for the malware. In addition to that correlation analysis is performed to improve the accuracy of the detection. In order to verify the effectiveness of proposed approach, eleven covert channels have been utilized. Implementation of eleven covert channels improves the accuracy of the detection up to 95%. The experimental result shows the feasibility and effectiveness of the proposed approach to detect the presence of malware and analysis of detection.

Keywords— Malware; Classification; Regression; Covert channel; Correlation analysis; Android

I. INTRODUCTION

Android OS is one of the widely used mobile Operating Systems. The number of malicious applications and adware's are increasing constantly on par with the number of mobile devices. A number of signature based tools are available to prevent the penetration and distribution of malicious applications. Most of the detection system work well only up to certain level and malware authors uses numerous techniques to evade these tools.

A. Types of Malwares

a. Information Extraction
b. Automatic Calls and SMS
c. Root Exploits

d. Search Engine Optimizations
e. Dynamically Downloaded code
f. Covert channel
g. Botnets

II. RELATED WORKS

Various detection of malware is studied in the literature. A Study of Android Malware Detection Technology Evolution-Hsieh et al.[19] reviews the evolution of malware which makes detection of malware more difficult, as well as the development of malware detection software makes the smart phones safer. It discusses Android malware survival techniques which are used by the malware developers to evade the detection of anti-virus technologies.

The investigations on general indicators like energy based and power based approach are the most widely used technique to detect covert channel that exploit information hiding. Other approaches such as Anomaly based approach and signature based approach are also used for detection. The most recent work on these two approaches is MADAM[1], which monitors the device actions, its interaction with the user and the running apps, by retrieving five groups of features at four different levels of abstraction, namely the kernel level, Application-level, user-level and package-level.

Wade Gasior et.al[24] implemented a network covert channels on mobile devices that allow data to be leaked from the device via its network connection in a manner that is very difficult to detect. They also implemented both timing-based and storage-based network covert channels to show how those channels covertly transfer information from the phone to a server.

Most of the method captures the communications between mobile applications and Android system, but it ignores the communication among components within or cross application boundaries. As a result, most of the existing methods are less effective in identifying malwares which require few or no suspicious resources. Lalande et.al[13] introduces a novel convert channel linked to a minimized footprint to achieve a high covertness. They developed a malware that slowly leaks all the collected information and sends it synchronously based on four covert channel techniques which easily escapes from the detection system**.**

III. DETECTION USING ENERGY MEASUREMENTS

The proposed system uses two main techniques for detection namely regression-based detection (RBD) and classification-based detection (CBD) approaches for detection. The RBD method is composed of two steps: the first consists in modelling the power consumed in a "clean" system. The past values of the consumption are used to predict the future ones. The second step is based on a comparison between the forecast consumption and the actual one. The CBD method consists in solving a classification problem starting from a set of measurements both in the presence and in the absence of colluding applications. It requires the definition of a set of "features" representing the power consumption of the device in a concise, effective manner.

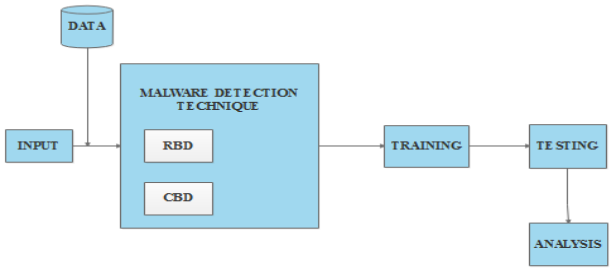

Fig.1 Architecture for malware detection using energy measurements

Input to the malware detection technique is the data set of energy values consumed by the system. Regression based detection and classification based detection can performed via either neural network or decision tree by training and testing the dataset. Finally the presence of malware can be detected and analysis of predicted value is performed.

IV. ARTIFICIAL INTELLIGENCE TOOLS

Two well-known artificial intelligence tools are used for detection they are

1. Neural networks
2. decision trees

They are able to learn from a set of past collected energy measurements whether hidden communication is present, and then reveal threats. Two detection methods such as regression and classification can be developed each one using both neural networks and decision trees. This artificial intelligence tools are used to reveal malicious code or to prevent the execution of hazardous software on Android devices.

V. MEASUREMENT METHODOLOGY

Accuracy of detection can be estimated using the parameters like true positive, false positive, true negative, false negative.

Fig 3 shows the test result of 20 test samples randomly taken to perform detection using RBD and CBD

In machine learning true positive (tp), true negative (tn), false positive (fp), false negative (fn) are used in classification task to compare the result of classifier under test.

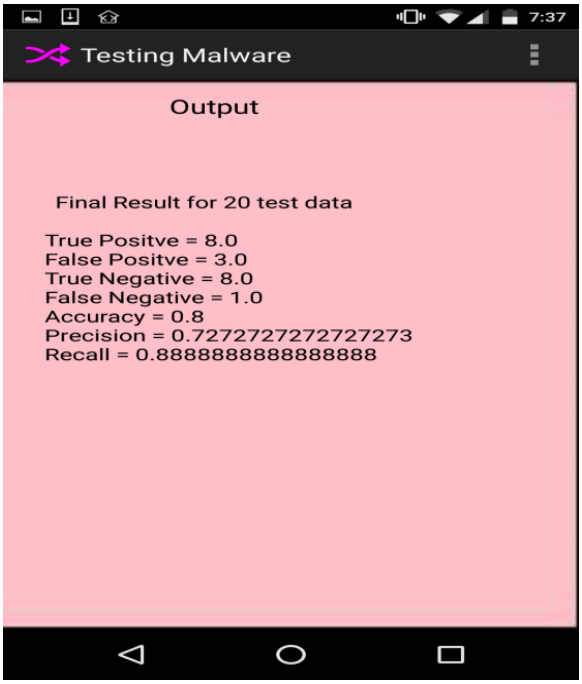

Fig 3.Test result of 20 test sample

Precision, recall and accuracy is estimated as follows

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

VI. REGRESSION BASED DETECTION

The RBD method is composed of two steps: the first consists in modeling the power consumed in a "clean" system, i.e., when no colluding applications performing hidden data exchanges are present in the device. To this end, a collection of past values of the consumption are used to predict the future ones. The second step is based on a comparison between the forecast consumption and the actual one. Hidden communication is spotted if the expected energy footprint deviates too much from the real one.

We assume that a covert communication among two colluding applications is present if the prediction error is greater than a certain positive threshold. Otherwise, it is considered absent. The rationale is that the approximate model, if properly trained, is able to predict the future behavior of the energy consumption of the "clean" system with a good level of accuracy. Hence, severe deviations reveal the presence of two processes covertly exchanging data. Clearly, the performance of the RBD depends on the quality of the approximating model and on the parameters, which have to be properly tuned.

We define the following quantities at each sampling time t = 0,1,... Let $p_t \in R_+$ be the power consumed by a process at time t. Such a quantity is measured and can be considered as an input. Let $w_{t+1} \in R_+$ be the prediction of the power needed by the process itself at the next time instant t + 1, i.e., it is an output.

At least in principle, the consumption of a process at time t + 1 may depend on the energy requirements at the previous time instants, from 0 to t. However, to avoid dealing with vectors of increasing dimension as the time grows, we consider a "fading memory" assumption, which consists in assuming that the output of the system (i.e., the power consumption of a process at time t + 1) depends only on a finite number q of past inputs [49]. In this perspective, we define a regression vector (also called regressor) as the collection of the past input variables from time t − q + 1 up to time t, i.e., $p_t \triangleq \text{col}\,(p_{t-q+1}, p_{t-q+2}, \ldots p_t) \in R^q$, where q is a positive constant value.

Thus, the training set for the creation of the model has the form of a set of input/output pairs

$$\Sigma_N \triangleq \{\underline{p}_t, w_{t+1}\}_{t=1}^{N}$$

Where, N refers the total number of available measures. The goal is to find a model that is able to capture, at each time t, the functional relationship between past and future consumption, i.e., the mapping $p_t \rightarrow w_{t+1}$.

$$\tilde{w}_{t+1} = \gamma(\underline{p}_t, \underline{\alpha})$$

Where $\tilde{w}_{t+1}$ is the estimated output at time t+1 and γ is a function belonging to a family of one-hidden-layer feed forward neural networks or binary decision trees. Notice that, when the regressor is made up by long series of past observations, the dimensionality of the problem rapidly increases; hence the need of efficient approximators arises.

Once the best model within the class has been found by the offline training procedure, the second step of the RBD method requires the definition of a detection rule. Specifically, the detection is performed online by feeding at each time step t the trained model with the past q measurements of the consumption collected into the regressor p_{t-1}. Then, the estimated consumption $\tilde{w}_t$ is compared with the real measured value w_t in a time window moving over time. The same procedure is repeated at the next time steps, and a prediction error e_t is defined as follows:

$$e_t \triangleq \frac{1}{\tau} \sum_{k=t-\tau+1}^{t} |\tilde{w}_k - w_k|$$

Where, τ is a given time horizon. We assume that a covert communication among two colluding applications is present if the prediction error is greater than a certain positive threshold ξ. Otherwise, it is considered absent. The rationale is that the approximate model, if properly trained, is able to predict the future behavior of the energy consumption of the "clean" system with a good level of accuracy. Hence, severe deviations reveal the presence of two processes covertly exchanging data.

VII. CLASSIFICATION BASED DETECTION

The The CBD method consists in solving a classification problem starting from a set of measurements both in the presence and in the absence of colluding applications. It requires the definition of a set of "features" representing the power consumption of the device in a concise, effective manner.

More specifically, we focus on three different features characterizing the energetic behavior of a process at each time t, collected into the vector $\underline{x}_t \in R^3$:

- The average power consumption from time $t - \lambda + 1$ to time t, where λ is a positive constant defining a window of past measurements,
- The total variation of the power consumption from time $t - \lambda + 1$ to time t, and
- The instantaneous consumption at time t. Thus, we focus on the following vector of features at each time t:

$$\underline{f}_t \triangleq \mathrm{col}\left(\sum_{l=t-\lambda+1}^{t} p_l, \sum_{l=t-\lambda+1}^{t} |p_l|, \; p_t \right) \in \mathbb{R}^3$$

Each vector f_t refers to a single measurement and is associated to a certain class k among two possible ones, corresponding to the cases in which covert channels are used to exchanging data between colluding applications (k = 1) and no covert channels are established (k = 0). The training set for the creation of the model takes on the form of a set of N input/output pairs:

$$\Sigma_N \triangleq \left\{ \underline{f}_t, g_t \right\}_{t=1}^{N}$$

Where, the scalar output g_t is equal to k if the input vector f_t belongs to the class k. The goal is to find a model able to recognize the class containing a given input vector that is not among the N used for the training. As in the case of the RBD method, we rely upon models belonging to a certain family of one-hidden-layer feed forward neural networks and binary decision trees as:

$$\tilde{g}_j = \gamma(\underline{f}_j, \underline{\alpha})$$

Where, $\tilde{g}_j$ is the class assigned to the input vector f_j by the model. The output $\tilde{g}_j$ must be one of the two possible classes. Clearly, the goal of the classification is to ensure that the assignment of the model is correct, i.e., the difference between $\tilde{g}_j$ and g_j is as small as possible. To this end, as in the case of the RBD, a suitable training phase is performed offline to find the optimal values of the parameter vector $\underline{\alpha}$. Once the best model within the family has been found, at each time t the detection whether a covert channel is present is performed online by feeding the trained model with the vector f_t of the current features and analyzing the value of the output $\tilde{g}_t$.

VIII. COVERT CHANNEL IMPLEMENTATION

A covert channel is a mechanism that can be used to breach a security policy and by allowing information to leak to an unauthorized process. For experimental purpose eleven covert channels has been used.

a) **Type of Intents** : The source sends a broadcast message to the sink and encodes the data to be transmitted into the type of the intent rather than directly exchanging the data as the extra payload of the message.

b) **File lock** : One application can send signal either by locking or unlocking a file. The other application can get back the signal by checking whether the file is locked or not.

c) **Volume settings** : Any application can change the volume settings and every time this setting is changed the system sends a notification to interested applications. Changes in the volume setting are not automatically broadcasted, which means that two applications communicating through this channel have to set and check the volume.

d) **Unix socket discovery** : The secret data is sent by encoding the information within the socket.

e) **System logs :**Android internally creates multiple different log files related to different contents.

f) **File size :** The secret sender sets the size of a shared file and the secret receiver interprets it as a byte.

g) **Memory load :** The secret receiver acquires the initial memory load of the secret sender. Then, the secret sender inflates the allocated data to send a 1 or releases memory to send a 0.

h) **Screen :** The change of the screen states (on or off) triggers a notification mechanism, which Android uses to inform applications of the screen setting.
i) **Vibrator status :** Any application can change the vibration settings and every time this setting is changed the system sends a notification to interested applications.
j) **Battery :** Mobile devices typically have a Lithium-ion battery, with limited charge capacity. Parallel use of multiple resources discharges the battery at different rates depending on the component used. This property can be exploited for encoding of information to form a covert channel.
k) **Camera usage :** The camera usage must also be stealthy in order not to raise the victim's notion. For example some phones have a LED that is automatically turned on when the camera is in use.

IX. ALGORITHM FOR MALWARE DETECTION

***Input*:** features (set of energy values)
***Output*:** presence of malicious software
Features are given as input to training dataset
***Regression based detection*:**

1. Training set for creating model has input and output pair of form,

$$\sum_N \triangleq \{\underline{P_t}, w_{t+1}\}_{t=1}^{N}$$

//$\underline{P_t}$ is the collection of past input variables, $w_{t+1}$is the prediction of the power needed by the process itself at the next time instant t + 1, i.e., it is an output and N is the total number of available measures.

2. Model for generating a functional relationship among past and future consumption is of the form,

$$\overline{w}_{t+1} = \gamma(\underline{P_t}, \underline{\alpha})$$

// $\gamma(\underline{x}, \underline{\alpha}) = \sum_{k=1}^{v} c_k \sigma\left(\sum_{j=1}^{q} a_{kj} x_j + b_k\right)$, $\sigma$is the activation function, v is the number of basis function, $\underline{\alpha}$ is the vector of free parameters

3. Prediction error can be found using the equation:

$$e_t \triangleq \frac{1}{\tau} \sum_{k=t-\tau+1}^{t} |\bar{w}_k - w_k|$$

4. If the prediction error is greater than positive threshold covert communication is present else covert communication is absent.

Classification based detection:

1. Vector of three features characterizing energy behaviour at time t is taken
   i. Average power consumption from time $t-\lambda+1$to time t,where λ is aposiitve constant defining window of past measurements.
   ii. The total variation of power consumption from time $t-\lambda+1$ to t.
   iii. Instantaneous consumption at t

$$\underline{f_t} \triangleq col\left(\sum_{i=t-\lambda-1}^{t} p_t, \sum_{i=t-\lambda-1}^{t} |p_t|, p_t \in R^3\right.$$

   Each vector $\underline{f_t}$ refers to a single measurement and is associated to a certain class k among two possible ones, corresponding to the cases in which covert channels are used to exchanging data between colluding applications (k = 1) and no covert channels are established (k =0).

2. Training set for creating model has input and output pair of form,

$$\sum_N \triangleq \{\underline{f_t}, g_t\}_{t=1}^{N}$$

   //g_t is equal to k if the input vector $\underline{f_t}$belongs to class k

3. The model to recognize class containing a given input vector is of the form,

$$\overline{g}_j = \gamma(\underline{f_j}, \underline{\alpha})$$

// $\overline{g}_j$ is the class assigned to the input vector $\underline{f_j}$ by the model.

4. Created model is used for detection of malicious software

X. CONCLUSION

Modern malware uses advanced techniques to hide from static and dynamic analysis tools. As a consequence, it is important to investigate how to reveal the presence of malicious software. The proposed approach uses the RBD, CBD technique to detect the malicious software and correlation analysis is implemented for improving the accuracy of the system. The proposed system effectives are proved by implementing it over eleven covert channels.

REFERENCES

[1] "MADAM: Effective and Efficient Behavior-based Android Malware Detection and Prevention"- Andrea Saracino, Daniele Sgandurra, Gianluca Dini and Fabio Martinelli, IEEE Transactions on Dependable and Secure Computing, 2016

[2] Luca Caviglione, Mauro Gaggero, Jean-Franc¸ois Lalande, Wojciech Mazurczyk, and Marcin Urba´nski,"Seeing the Unseen: Revealing Mobile Malware Hidden communicaton via Energy consumption and Artificial intelligence" IEEE Transactions on Information Forensics and Security ,Vol:11 ,No: 4,April 2016

[3] W. Mazurczyk and L. Caviglione, "Information hiding as a challenge for malware detection," Security & Privacy, vol. 13, no. 2, pp. 89–93, 2015.

[4] McAfeeLabs, "McAfee labs threat report," August 2014.

[5] J. Hoffmann, S. Neumann, and T. Holz, "Mobile malware detection based on energy fingerprints - a dead end?" in Research in Attacks, Intrusions, and Defenses. Springer, 2013, pp. 348–368.

[6] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, and L. Yang, "Accurate online power estimation and automatic battery behavior based power model generation for smartphones," in Proc. Int. Conf. on Hardware/Software Codesign and System Synthesis, 2010, pp. 105–114.

[7] L. Caviglione and A. Merlo, "The energy impact of security mechanisms in modern mobile devices," Network Security, vol. 2012, no. 2, pp. 11– 14, 2012.

[8] C. Marforio, H. Ritzdorf, A. Francillon, and S. Capkun, "Analysis of the communication between colluding applications on modern smartphones," in Proc. Annual Computer Security Applications Conf., 2012, pp. 51–60.

[9] A. Merlo, M. Migliardi, and P. Fontanelli, "Measuring and estimating power consumption in Android to support energy-based intrusion detection," Journal of Computer Security, vol. 23, pp. 611–637, 2015. [13] S. Haykin, Neural Networks, A comprehensive foundation. Prentice Hall, 1999

[10] "ICCDetector: ICC-Based Malware Detection on Android"-Ke Xu, Yingjiu Li, and Robert H. Deng, IEEE Transactions on Information Forensics and Security, 2016

[11] "High accuracy android malware detection using ensemble learning"-Suleiman Y. Yerima, Sakir Sezer, Igor Muttik, IET Information security, 2015

[12] K. Allix, Q. Jerome, T. F. Bissyande, J. Klein, R. State, and Y. le Traon, "A forensic analysis of Android malware-How is malware written and how it could be detected?" in *Computer Software and Applications Conf.*, 2014, pp. 384–393

[13] B. Dixon and S. Mishra, "Power based malicious code detection techniques for smartphones," in *IEEE Int. Conf. on Trust, Security and Privacy in Computing and Communications*, 2013, pp. 142–149.

[14] J.-F. Lalande and S. Wendzel, "Hiding privacy leaks in Android applications using low-attention raising covert channels," in *Int. Conf. on Availability, Reliability and Security*, 2013, pp. 701–710

[15] A.Merlo, M. Migliardi, and P. Fontanelli, "On energy-based profiling of malware in Android," in *Int. Conf. on High Perf. Computing & Simulation*, 2014, pp. 535–542.

[16] W. Mazurczyk and L. Caviglione, "Information hiding as a challenge for malware detection," *Security & Privacy*, vol. 13, no. 2, pp. 89–93, 2015.

[17] W. Mazurczyk and L. Caviglione, "Steganography in modern smartphones and mitigation techniques," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 334–357, 2014.

[18] "Detection of malicious code by applying machine learning Classifiers on static features: A state-of-the-art survey" Asaf Shabtai, Robert Moskovitch, Yuval Elovici, Chanan, information security technical report.

[19] "Catch me if you can: evaluating android anti-malware against transformation attacks "-vaibhav rastogi, yan Chen, and xuxian jiang, IEEE transactions on information forensics and security, vol. 9, no. 1, january 2014

[20] "A Study of Android Malware Detection Technology Evolution"-Hsieh, Wan-Chen, Wu, Chuan-Chi, Kao, Yung-Wei

[21] "Towards Energy-Aware Intrusion Detection Systems on Mobile Devices" Monica Curti, Alessio Merloy, Mauro Migliardiz, Simone Schiappacasse, 2013.

[22] S. Lee, W. Jung, Y. Chon, and H. Cha, "EnTrack: a system facility for analyzing energy consumption of Android system services," in Proc. Int. Joint Conf. on Pervasive and Ubiquitous Computing, 2015, pp. 191–202.

[23] Wade Gasior, Li Yang, Exploring Covert Channel in Android Platform, ASE International Conference on Cyber Security

