ARP Request Trend Fitting for Detecting Malicious Activity in LAN

Kai Matsufuji
The University of Tokyo
mattun@hongo.wide.ad.jp

Satoru Kobayashi
National Institute of Informatics
sat@nii.ac.jp

Hiroshi Esaki
The University of Tokyo
hiroshi@wide.ad.jp

Hideya Ochiai
The University of Tokyo
ochiai@elab.ic.i.u-tokyo.ac.jp

ABSTRACT

Security of local area networks (LANs) attract enormous attention these days. LANs are not safe even under a well-configured firewall, because malware can be easily delivered through network applications. Thus we need to take counter measures against malicious activities by focusing on each device itself. However, we cannot adopt many of existing methods to the devices which have limited resource capacity like IoT. Accordingly, we consider that the request pattern of address resolution protocol (ARP) may provide some indicators for finding such malicious activities without putting a burden on each device. In this paper, We propose a method to detect malicious network activity by ARP monitoring. The detection method is based on a fitting model of ARP request trend. We especially focus on the destination devices of ARP request in outlier detection with the model. We made an experiment with a data of monitored ARP requests in laboratory network. We also discuss parameter tuning and validity of the model based on three notable requirements.

CCS CONCEPTS

• **Security and privacy → Network security**; *Intrusion/anomaly detection and malware mitigation*; • **Networks → Local area networks**;

KEYWORDS

Address Resolution Protocol, Local Area Network, Security

ACM Reference Format:
Kai Matsufuji, Satoru Kobayashi, Hiroshi Esaki, and Hideya Ochiai. 2018. ARP Request Trend Fitting for Detecting Malicious Activity in LAN. In *Proceedings of IMCOM '19: (IMCOM '19).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Security of local area networks (LANs) is getting more and more serious these days. Malware makes some suspicious activities for expanding the malware itself, for searching sensitive data, or for overriding IoT application-layer protocols. Even if there is a firewall at the entrance of the network, malware can be easily delivered into LANs via phishing emails or malware infected smartphones through Wi-Fi. Thus we need to take counter measures against malicious activities in those cases by focusing on each device itself. However, it places a large load on the end-node devices to measure the network activity of them. It is difficult to make counter measures against malware, which works in the end-node devices, due to their resource capacity limitation. Some existing techniques [3] are proposed in recent days, but most of them are not reasonable in such as IoT networks. We propose based on address resolution protocol (ARP) to solve such problem. Because ARP request packet is broadcast for finding target MAC address in most cases, we can observe the packet in the other devices connected to the LAN. Therefore, we can observe the ARP request packets by adding a monitor machine to the LAN. This does not place an additional load to the end-node devices. Moreover, there is another advantage that we can introduce the ARP monitoring method without changing existing devices.

We can judge whether malware invades into a LAN or not by observing the trend of ARP request. When malware invades into a device, it tries the scanning attack to invade into other devices in the LAN or interrupts particular connections. When we observe ARP request of that time, the number of destination hosts in certain period increases or decreases exponentially unlike that in usual. Therefore, by observing the change of the number of destination hosts about ARP request, we can judge whether malware invades into LAN or not in the period.

In this paper, we propose a fitting model which predicts the next situation of the number of destination hosts about ARP requests by using the number so far and judge whether the connection in the LAN is normal or not based on it.

2 RELATED WORK

LAN security is paid more and more attention nowadays because many kinds of malware such as ransomware become a big threat in the modern society. However, we need to choose which method is the best when we adopt it to devices which have capacity limitation.

Kolbitsch et al. [3] propose a malware detection approach which learns a program of malware and judges whether the malware invades or not. Their approach is implemented at the end host and makes the host free up some resources for the approach. However,

IMCOM '19, January 4–6, 2019, Phuket, Thailand

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

we should not adopt such methods to the devices which have limited resource capacity like IoT.

In studies featuring LAN security, some of the studies focus on ARP just like our study.

Whyte et al. [5] record active systems within the network cell and devices they are trying to connect with by using ARP request and use them as an indicator for anomaly detection by counting ARP request which don't correspond to them. The study is related to ours in terms of anomaly detection using ARP request. However, it is difficult for their approach to deal with configuration changes because their method don't rewrite the record. Thus we need a method considering time variation and rewriting records.

Pandey [4], Jinhua et al. [2] and Hou et al. [1] study about LAN security focusing on ARP. However, these studies attach importance to taking measures against ARP spoofing, and it is difficult for us to adopt these studies to malware detection.

3 ARP REQUESTS CLASSIFICATION METHOD

As mentioned in Section 1, we take measures against malicious activities by observing ARP request packets. We propose a fitting model to ARP request trend patterns. The fitting model predicts the trend of ARP request in the next day by using that in one or more days before.

3.1 Requirements of Fitting Models

We list the requirements for fitting models in our study as follows:

Learn Automatically
The model shouldn't require knowledge such as the list of malwares manually, otherwise we would update the list manually forever.

Follow Time Variation
The model should expect the situation of the next day by learning the past trend. Because the system configuration in a LAN changes frequently, the model should be always updated.

Correspond to Amplitude
The model should memorize drastic change experienced recently. It shouldn't be too sensitive to the trends. Otherwise, we can't adopt the model to the shaking trends.

3.2 The Extraction of the Degree of Destination

Our approach first makes a connection graph for a certain time duration of a network, i.e., device-to-device connections based on ARP request. We focus on the number of connected hosts in ARP (hereinafter, this is called the degree of destination). Fitting model estimates the future trends learning from the past trends (Figure 1).

We consider ARP request packets which a device (S) sends. The frame of ARP requests is broadcast, and we define them which is observed by another device of the same segment in a day(D_s) as:

$$D_s = (IP_1, IP_2, ...IP_k) \quad (1)$$

Thus the degree of destination about S in a day ($X_s(n)$) can be showed as follows:

$$X_s(n) = k \quad (2)$$

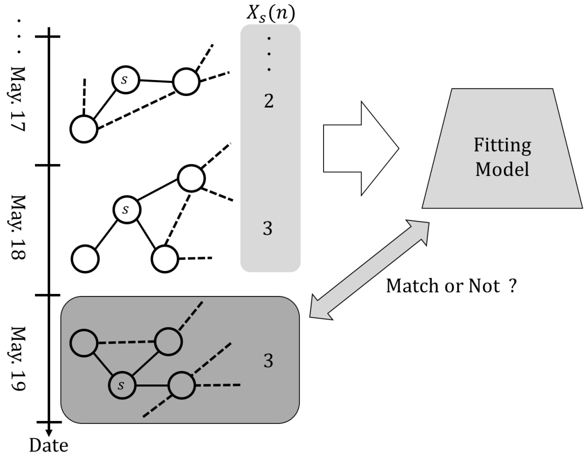


Figure 1: Estimation the future trends of the degree of destination. The connection graphs are generated from the observations of broadcasted ARP requests.

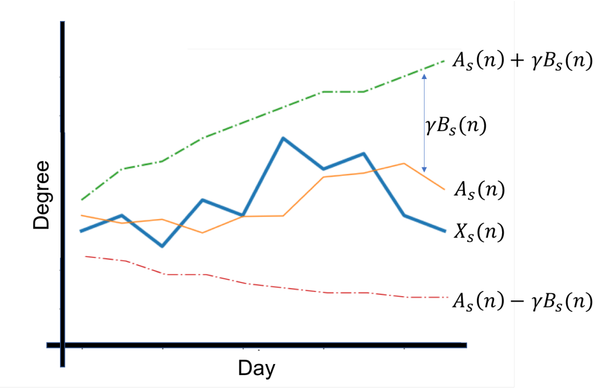


Figure 2: An Example of Fitting Model

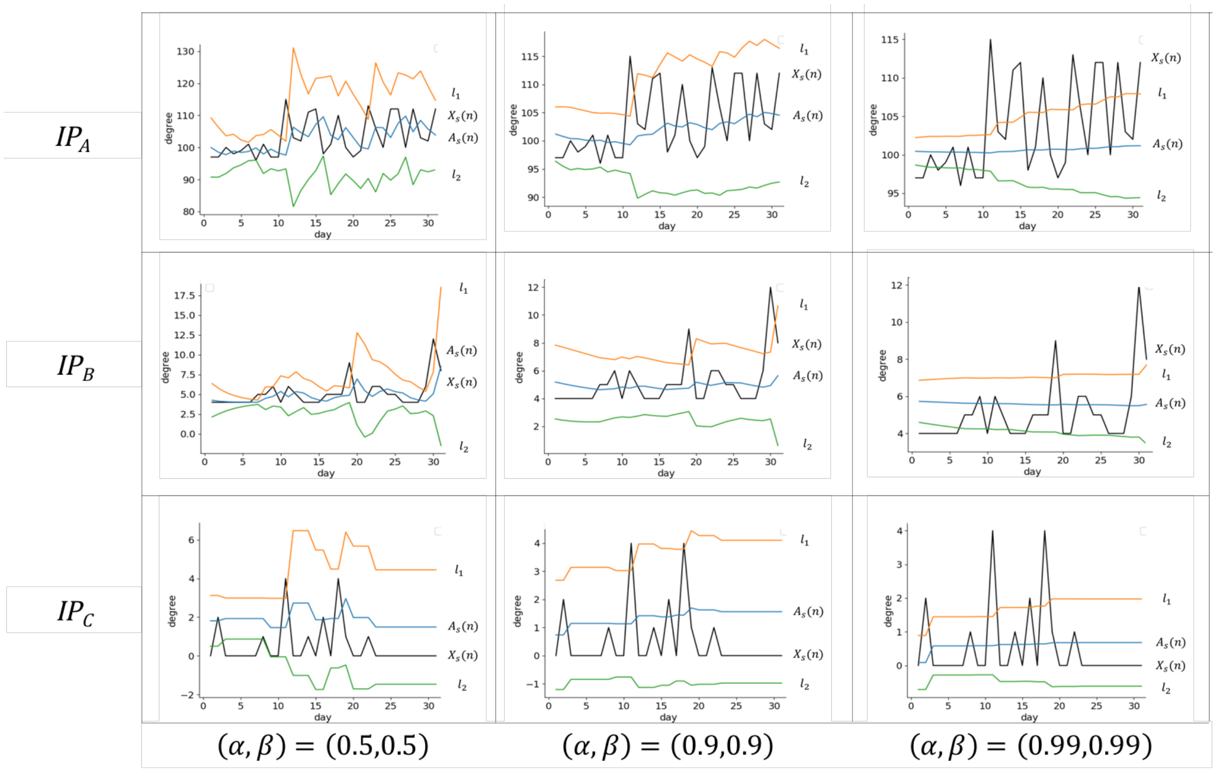
Here, n represents the date.

3.3 Definitional Formulae of Fitting Model

The fitting model, we propose in this paper, gives an upper limit and a lower limit. It expects that in the next time duration the degree of destination will be within those limits. We admit that, to actually show the fitness, we must introduce some fitness evaluation models, such that the gap between the upper and lower limits should be appropriate. However, this kind of evaluation model is dependent on the network operation, so we left this discussion open, in this paper.

In Section 3.1, we defined $A_s(n)$ and $B_s(n)$ considering three requirements. Further, we define an upper limit and a lower limit of fitting model as $A_s(n) \pm \gamma B_s(n)$. By defining the two limits in this way, fitting model performs more efficiently meeting three requirements.

Fig 2 shows the example of fitting Model. $A_s(n) - \gamma B_s(n)$ and $A_s(n) + \gamma B_s(n)$ is the lower limit and the upper limit of fitting model



$(\alpha, \beta) = (0.5,0.5)$ $(\alpha, \beta) = (0.9,0.9)$ $(\alpha, \beta) = (0.99,0.99)$

Figure 3: The Transitions of $X_s(n), A_s(n), l_1$ and l_2 Changing α and β to Various Values about three IP addresses in May,2018.

and when $X_s(n)$ doesn’t lie between the boundaries, $X_s(n)$ is judged to be abnormal.

Now, when considering about definitional formulae of fitting model, we shouldn’t define all cases in only one formula. This is because devices which always connect to somewhere every day are different from what occasionally connect to somewhere. In the former, the value of 0 in $X_s(n)$ is should be judged as abnormal because as long as devices connected to somewhere, they keep on sending ARP request every certain period. In the latter, on the other hand, 0 in $X_s(n)$ is should be normal because they don’t always need to send ARP request. Thus, we define formulae of fitting model distinguishing between the two cases.

3.3.1 Fitting Model about Permanently-Connecting Device. Many of the devices which always connect to somewhere (Permanently-Connecting devices) connect some application servers(e.g., DNS, syslog, DHCP, Nagios and DB) during normal times, and it is assumed that the degree of destination approaches a constant value. Now we define the moving average about $X_s(n)$ ($A_s(n)$) as:

$$A_s(n+1) = \alpha A_s(n) + (1-\alpha)X_s(n), A_s(0) = 0 \tag{3}$$

and the variance ($Y_s(n)$) as:

$$Y_s(n) = \{X_s(n) - A_s(n)\}^2 \tag{4}$$

and the moving average about $Y_s(n)$ ($B_s(n)$) as:

$$B_s(n+1)^2 = \beta B_s(n)^2 + (1-\beta)Y_s(n)^2, B_x(0) = 0 \tag{5}$$

Further, we define the determination condition to decide whether devices fit the model or not (C) as:

$$C : A_s(n) - \gamma B_s(n) < X_s(n) < A_s(n) + \gamma B_s(n) \tag{6}$$

and we perform anomaly detection about S based on the inequality.

3.3.2 Fitting Model about Occasionally-Connecting Device. It is assumed that the degree of destination about devices which occasionally connect to somewhere basically exhibits the same behavior as mentioned in Section 3.3. However, we need to consider separately about when devices don’t connect to anywhere, and the degree of destination evaluates to 0. In the above case, we redefine $A_s(n)$ and $B_s(n)$ as mentioned before as:

$$A_s(n+1) = \begin{cases} \alpha A_s(n) + (1-\alpha)X_s(n) & (X_s(n) > 0) \\ A_s(n) & (X_s(n) = 0) \end{cases} \tag{7}$$

$$B_s(n+1)^2 = \begin{cases} \beta B_s(n)^2 + (1-\beta)B_s(n)^2 & (X_s(n) > 0) \\ B_s(n)^2 & (X_s(n) = 0) \end{cases} \tag{8}$$

and C as:

$$C : X_s(n) = 0 \lor A_s(n) - \gamma B_s(n) < X_s(n) < A_s(n) + \gamma B_s(n) \tag{9}$$

4 EVALUATION

4.1 Experiment Settings

To validate our approach, we collected ARP request packets from the network in lab from April 17 to June 16,2018. We investigated the degree of destination per day by source IP address. Hereinafter, we call the upper and lower limit of fitting model as mentioned in Section 3 as l_1 and l_2. In this experiment, we define γ as 2 ($l_1, l_2 : A_s(n) \pm 2B_s(n)$). Also, if over 1/4 of measurement period about a device is 0, we call the device as Occasionally-Connecting device. Under these conditions, we describe l_1 and l_2 changing α and β and validate which parameter(α and β) fits the behavior of $X_s(n)$ the best.

We use first seven-days time-series (except days without observed communications) as a learning-period, which makes $A_s(n)$ stable enough for outlier detection.

4.2 Parameter Tuning

In this section, we discuss parameter tuning in the formulae of the fitting model. In other words, we consider how the fitting model works if the parameters are not tuned properly.

Figure 3 shows the transitions of $X_s(n)$, $A_s(n)$, l_1 and l_2 changing α and β to various values about three IP addresses in May,2018. Now, IP_A and IP_B are Permanently-Connecting devices,and IP_C is Occasionally-Connecting device (not including the learning period).

In the case that both α and β is 0.99, $A_s(n)$ is almost constant in all three IP addresses in spite of the large change of $X_s(n)$. This is because $A_s(n)$ is fixed to data memorized in transient state because of too large α. Further, we can see that although $X_s(n)$ experienced large oscillations, the oscillations are not reflected in l_1 and l_2. This is because the amplitude memorized in transient state is reflected too strongly because of too large β.

On the other hand, we also evaluate the case which both α and β are 0.5. In $A_s(n)$, we can see that the value changes drastically compared with graphs as mentioned so far in all three IP addresses. This is because the value of last $X_s(n)$ is reflected too strongly because of too small α, and $A_s(n)$ doesn't memorize previous data. Further, we can see that unlike in 0.99, l_1 and l_2 change very intensely after $X_s(n)$ experienced drastic changes. This shows that the two value are too sensitive to last change because of too small β, and they can't predict correctly.

4.3 Evaluation of Requirements

As mentioned in Section 3, $A_s(n)$ and $B_s(n)$ in the formulae about fitting model meet the three requirements. Now, we list an example which had a good behavior as fitting model as a result of experiment (i.e., when both α and β is 0.9) and confirm that the fitting model in that case surely meets the three requirements(Figure 3).

Learn Automatically
This condition is met obviously because the fitting model doesn't require any information like the lists of malwares.

Follow Time Variation
It can be said that this condition is met because l_1 and l_2 change in response to changes of $X_s(n)$.

Correspond to Amplitude
To judge whether this requirement is met or not, we need to confirm drastic changes in Figure 3. In all three IP addresses, when we observe drastic change of $X_s(n)$ for the first time, the day's data is judged to be abnormal because fitting model doesn't expect this change. On the other hand, when we observe them for the second time or later, fitting model memorizes drastic change experienced before for several days. Therefore, we can see that in IP_A, large shakes occurring immediately after first shake is not detected as abnormal value, and in IP_B and IP_C, oscillation occurring some time after first oscillation is judged to be abnormal.

From the above, we derive the result that fitting model we proposed functions meeting the three requirements when we choose the parameter properly.

5 DISCUSSION

In this paper, we have experimented targeting a network in lab. As a future work, we need to discuss other networks for confirming the reliability in our study. Also, we focused on the degree of destination in ARP request. In the future, we consider the different features such as the locality or frequency. Thanks to these future works, the achievement in this study will be more successful.

6 CONCLUSION

In this paper,we have proposed a fitting model of ARP request trend patterns. Firstly, we proposed three requirements which composed the fitting model. Secondly, we showed the formulae in accordance with the three requirements and defined an upper limit and a lower limit. Finally, we investigated and evaluated the transition of the fitting model changing parameters in the formula to various values. As a result, we concluded that the fitting model functioned meeting the three requirements when we choose the parameter properly. As a future work, we need to discuss other networks or consider the different features of the degree of destination.

REFERENCES

[1] Xiangning Hou, Zhiping Jiang, and Xinli Tian. 2010. The detection and prevention for ARP Spoofing based on Snort. In *Computer Application and System Modeling (ICCASM), 2010 International Conference on*, Vol. 5. IEEE, V5–137.

[2] Gao Jinhua and Xia Kejian. 2013. ARP spoofing detection algorithm using ICMP protocol. In *Computer Communication and Informatics (ICCCI), 2013 International Conference on.* IEEE, 1–6.

[3] Clemens Kolbitsch, Paolo Milani Comparetti, Christopher Kruegel, Engin Kirda, Xiao-yong Zhou, and XiaoFeng Wang. 2009. Effective and Efficient Malware Detection at the End Host. In *USENIX security symposium*, Vol. 4. 351–366.

[4] Poonam Pandey. 2013. Prevention of ARP spoofing: A probe packet based technique. In *Advance Computing Conference (IACC), 2013 IEEE 3rd International.* IEEE, 147–153.

[5] David Whyte, EVANGELOS Kranakis, and P Van Oorschot. 2005. ARP-based detection of scanning worms within an enterprise network. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC).*