Proposal
using Nagios as Intel AMT input

Auke Zwaan
auke.zwaan@os3.nl

Robert Diepeveen
robert.diepeveen@os3.nl

February 2016

1 Introduction

If a malware outbreak hits an organization, it is vital to respond very quickly to prevent any further damage. Within the first couple of minutes, the malware could already have spread and infected numerous machines on the local network [1] [2] [3].

One way to stop malware from infecting more machines is to disconnect the machine from the network in some way. Locating an infected machine and physically going there to do so could, however, take too much critical time. If this could be done remotely using a management console, the need of physically being there could be taken away. To provide this kind of management, Intel launched their ‘Active Management Technology’ which provides so-called ‘Out-of-band system access’. In this definition, ‘out-of-band’ refers to having a separate channel through which devices can be controlled remotely. Using AMT, all supported machines could, in theory, be taken off the network by switching them off or by applying any changes required to mitigate the infection.

There has been some work into network flow based malware detection, showing that this approach indeed can be effective [4] [5] [6]. Tools like Stealth-Watch[^2], Snort[^3] or Internet Security Systems (ISS)[^4], if provided with the right sets of rules, could therefore assumed to be effective in detecting malware. In this paper, the effectiveness of using a combination such an infrastructure monitoring tool and Intel AMT policies to discover and mitigate malware outbreaks in a large-scale network is reviewed.

[^2]: https://www.lancope.com/products-services-lancope
[^3]: https://snort.org/
2 Research question

The goal of this study is to determine whether it is possible to use an infrastructure monitoring tool and Intel AMT policies to effectively and timely discover and mitigate malware outbreaks in a large-scale network. Along with answering this question, we aim to deliver a simple proof of concept of an AMT quarantining tool.

3 Related work

While there appears to be some research on using Intel AMT for fighting malware indirectly [7], any precise details on the effectiveness and ‘duration’ of updating AMT policies in real-time are not yet available.

For any information on the implementation of AMT, the official documentation provided by Intel will be used. Furthermore, Amsden and Varol [8] provided more information on how to use AMT for malware-related tasks. They gave more information about how the blacklisting of specific hosts works, thereby providing useful insight in the features and limitations.

Yen et al [9] proposed a framework for network flow malware detection. Their framework was able to successfully identify malware traffic in a university network of which only 0.0097% of the hosts were infected. Using data from this framework could for example provide us with source or destination IP Addresses to block traffic from/to.

4 Approach

Using a test setup in a lab environment, a small malware outbreak is simulated. Note that, in this case, the malware is simply defined as a series of hardcoded events that can be detected by analyzing network behavior. An example of this could be a machine connecting to a specific IP or port multiple times.

In the test setup, there will be an ‘infected’ machine A that tries to infect a ‘clean’ machine B. After detecting the behavior that makes machine A ‘infected’ in our definition, the information of the attempted infection is used to update the AMT policies for other machines in the network. This way, AMT does not only prevent the infection of machine B from happening, but infections of any other machine on the local network as well.

The results of the infection simulation will be measured in time and used to review how fast AMT policies can be updated. Using these measurements, a conclusion will be drawn to show whether this approach would be effective for real-time malware mitigation.

5 Requirements

For this project we need at least 2 devices with Intel AMT technology embedded. During the project pitch, the teachers mentioned the availability of two. If there are no such devices, we need to know that as soon as possible to see if we can arrange something. Furthermore, we need access to Windows licenses (covered by Microsoft Dreamspark so far, which provides us with 2 licenses), considering that the management console runs best on Windows.

6 Ethical implications

As the setup and techniques described in this paper are all run in a lab environment, we do not expect any ethical implications. Any potential security vulnerabilities found accidentally during this project will, of course, be discussed with the OS3 Ethical Committee (OS3EC) immediately. This is, however, not the aim of this study.

Because we are not using real malware, we do not foresee any problems with outbreaks on the OS3 network. Furthermore, we will separate our test-setup completely from the OS3 network and the Internet.

7 Planning

The project broadly consists of four parts:

- Setting up the test environment (including the AMT hardware and software)
- Developing a small prototype
- Doing and evaluating measurements
- Concluding the research and writing the report

References


[8] Nathaniel Amsden and Cihan Varol. “Malware Detection from a Virtual Machine”. In: ().