AV-TEST
Tests of Anti-Virus-Software independent • qualified • fast

Testing Exploit-Prevention Mechanisms in Anti-Malware Products

Andreas Marx & Maik Morgenstern
AV-Test GmbH, Magdeburg, Germany

http://www.av-test.org

Presented at the CARO Workshop 2009
in Budapest, Hungary (May 5, 2009)

Contents

- Introduction
- Basics of the testing
  - Introduction of the exploit
  - Tracking the effect of the exploit
  - Tracking the reaction of the anti-malware product
  - Reproducibility and Comparability
  - Challenges
- Conclusion

Introduction

- Simple scanning of single files may still be enough to protect users in some, but not in all cases…
  - Gateway scanners vs. end-user (client) AV products
  - CRC detection of known “bad” files (easy to do) vs. full reliable exploit detection (requires a lot of research)
- Many of today’s products have much more protection features to offer
- Testing needs to reflect these additional protection mechanisms:
  - Whole product evaluation instead of only testing (possibly misleading) on-demand scanning capabilities

Introduction

- Therefore “Dynamic Testing” strategies were introduced:
  - The Importance of Re-creating In-the-wild Infection Conditions for Testing Multi-Layered Security Products (Symantec, CARO 2007)
  - Testing of “Dynamic Detection” (AV-Test, AVAR 2007)
  - AMTSO Best Practices for Dynamic Testing (AMTSO 2008)
- Malware threats are introduced via typical infection vectors and actually run (executed) on the system
- Anti-malware products can stop the malware at different stages, utilizing their whole set of features, e.g.:
  - URL Blocking, web reputation services
  - Static detection by signatures or heuristics
  - Dynamic detection by observing the behavior of the malware

Introduction

- Exploits can be an important part during the infection process (and often are today)
- Anti-malware products consider this and provide special detection and prevention capabilities
- Testing has therefore to consider exploits as important part of an attack as well
- On-demand scan of exploit code is not a sufficient test! But what else?

Basics of the testing

- The “Dynamic Testing” strategy will be applied and extended resp. specified:
  - Scenarios as realistic as possible in an isolated and safe environment are being used in the test
  - Explicitly looking at exploits
  - All stages of an infection and all possible reactions of the anti-malware product will be observed (the utilization of exploits being one of the stages)
  - This enables the tester to review the whole product, instead of only single features, e.g. the on-demand scanner
  - The view can always be narrowed to only the exploit part if desired (but usually shouldn’t unless for very special tests!)

Basics of the testing

- Basic approach
  - The exploit code (or malware that utilizes exploit code) is introduced to the system
  - The effects of the exploit (and further associated components) on the system are tracked
  - The reactions of the anti-malware product are being observed (if any)
  - The final system state is observed and the detection and prevention success of the product is assessed

Basics of the testing

- Introduction of the exploit
  - The vulnerable applications in the vulnerable version have to be installed on the system
  - Real in-the-wild exploit code should be the focus, instead of proof-of-concept code
  - The exploit code should be accessed through the usual infection vector, which is different for different targeted applications
  - It has to be proofed that the exploit code actually works

Basics of the testing

- Tracking the effect of the exploit
  - The state of the system before introducing the exploit has to be known
    - File system, processes, registry, loaded modules in processes, network traffic …
  - The changes to the system after executing the exploit can be tracked
  - This allows to tell what happens on the system
  - If nothing happens, the anti-malware product can not be blamed for not detecting anything
  - Compare our “Dynamic Testing” paper

Basics of the testing

- Tracking the reaction of the anti-malware product
  - Does the product present any messages or does it write any information to report files?
  - Are actions of the exploit code blocked?
    - Whole exploitation process detected and blocked?
    - Only parts of the process blocked?
  - Are malicious modifications, when detected, reverted afterwards?
  - System states are again carefully watched to observe what happens on the system and what is being prevented by the anti-malware product
  - Again: compare our “Dynamic Testing” paper

Basics of the testing

- Detection and prevention success
  - Did the product present any messages or wrote any information to report files?
  - What has been detected and how has it been detected?
  - What has been prevented?
  - What was the exploit able to do on the system?
  - Have these changes been reverted and created components been removed?

Basics of the testing

- Reproducibility and Comparability
  - Same problems as with “Dynamic Testing”, e.g.:
    - Online sources of malware may be dead or deliver other files (when an exploit tries to download further components)
    - Overall different behavior during different test runs (certain anti-malware products may be detected by the malware)
    - Malware might behave differently when recognizing it is being observed or being run in an artificial environment
  - Solutions are the same as well, e.g.:
    - Record queries once and replay them during the test
    - Perform a statistical significant number of tests to rule out the effect of behavior changing malware
    - Use minimal invasive means of observation
    - Use real systems, no virtual machines
    - Use real (but limited) internet connection, no simulated ones

Basics of the testing

- Further challenges
  - Sample throughput is usually small
    - Tests are very complex (many manual steps involved)
    - Analysis of the test results is very complex
    - Choice of viable samples may be limited
  - Relevance of samples
    - Not every sample is relevant (e.g. 2 year old exploits for outdated vulnerable application versions)
    - Exploits for fixed vulnerabilities in general
    - Exploits for very seldom used software
  - Database of vulnerable applications required
  - Exploits for vulnerabilities in anti-malware products
    - Valid for a test set? How to test?

Conclusion

- Testing the exploit-detection and prevention mechanisms of anti-malware products can be performed with approaches from “Dynamic Testing”
- Similar and some more problems occur
- Similar solutions can be found
- Still some open problems

Conclusion

- It is necessary to be able to process more samples to have statistically significant numbers
  - Reduce the complexity of the tests
    - Risk: tests may become irrelevant and unrealistic
  - Automate more steps
    - Risk: automation can be prone to errors
- Surely: the times of simple tests that process hundreds of thousands of files have passed away
- But can the complexity of new testing approaches be handled?

Thank you very much for your attention!

Are there any questions?