Using Infection Markers as a Vaccine against Malware Attacks

Andre Wichmann andre.wichmann@fkie.fraunhofer.de

Elmar Gerhards-Padilla elmar.gerhards-padilla@fkie.fraunhofer.de

cydef@fkie.fraunhofer.de

Fraunhofer FKIE

Outline

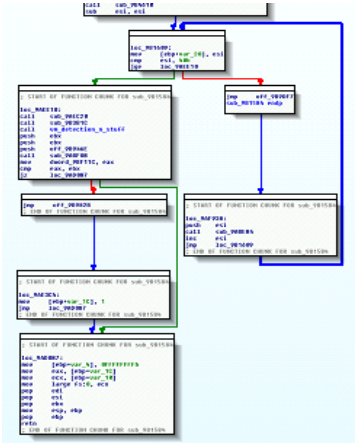
- Motivation and General Idea
- Infection Marker Taxonomy
- Automated Extraction Framework
- Evaluation
- Conclusion

Malware

- Malware is a big problem for (networked) systems
  - Not only on desktop PCs
  - Also for mobile and embedded devices

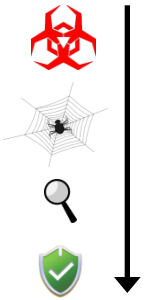
- Malware has a high value for their developers
  - Financial: Online banking, data theft, ...
  - Political: Espionage and sabotage

- Modern malware gets more and more complex
  - Sophisticated evasion techniques (*Lexotan32*...)
  - Advanced anti reverse engineering tricks
  - Complex code (*Stuxnet*...)

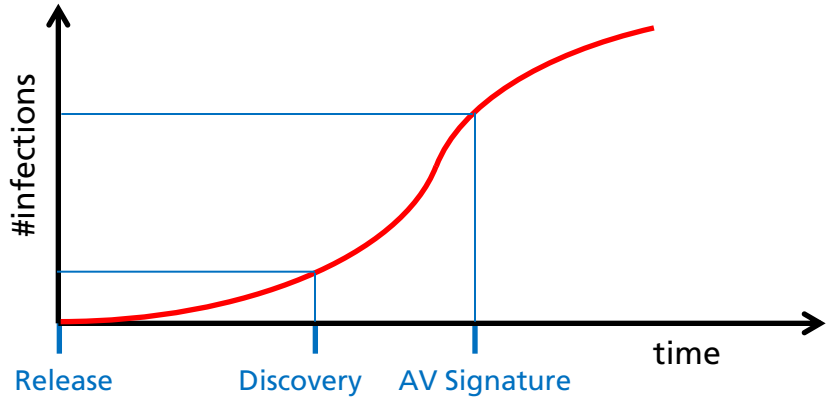


Malware Analysis

- Typical timeline:
  - New malware gets released
  - Malware is discovered by AV researchers
  - Malware gets analyzed
  - Detection and mitigation techniques are released

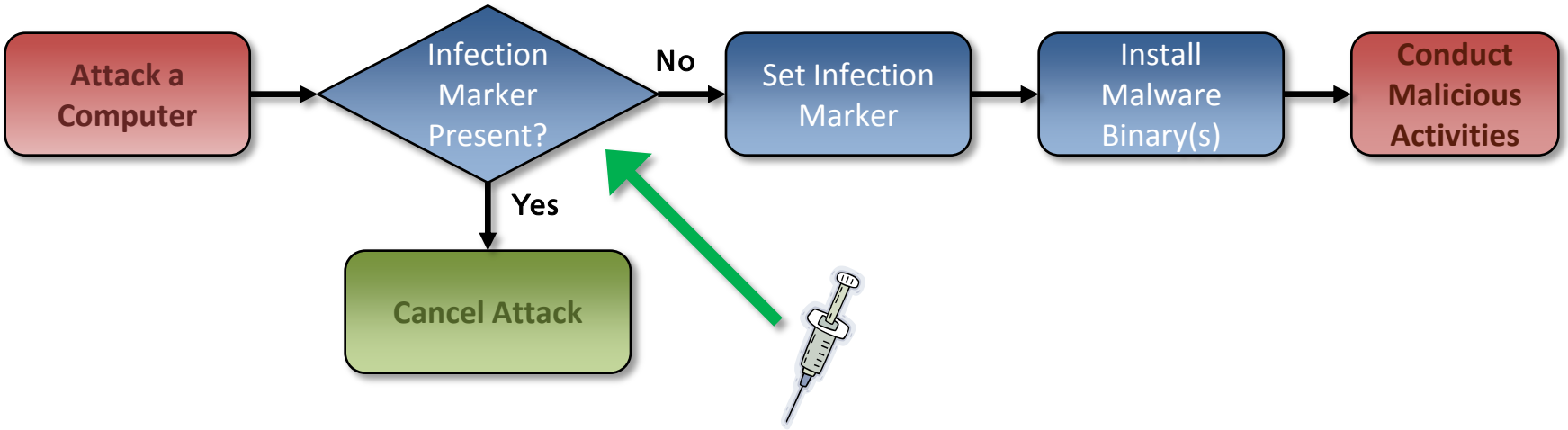


- Short analysis time is critical!
  - The longer malware can spread unhindered, the more damage potential it has



Using Infection Markers as Vaccine

- Typical stages of a malware attack (simplified):



- Idea: **Set infection marker on clean systems** to immunize them
- **Automate** the process of extracting infection markers
- During deeper analysis,
  - Propagation of malware is **mitigated**
  - Critical systems are **protected**

Infection Marker Taxonomy

Infection Marker Characteristics

- Developers of malware don't want to infect the same system twice
  - No additional advantage (system resources)
  - Could affect system stability
- Use *Infection markers* to detect installation of same malware family

- Infection markers must be *persistent/accessible* and *deterministic*
- Infection markers should be *unique* and *hidden*

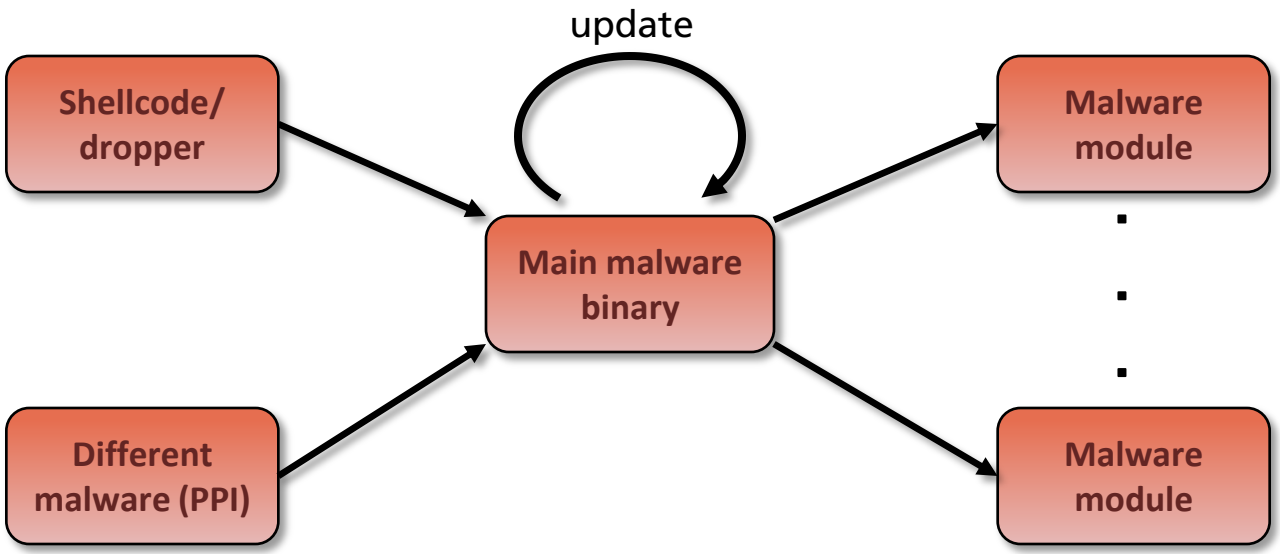
- Examples
  - Mutexes ("uterm12", "Microsoft Debugger", "kj65akjnlk264lk11")
  - Registry keys ("NTVDM Trace" = "19790509" - Stuxnet)
  - Presence of a file
  - ...

Infection Marker Taxonomy

- Marker location and lifetime
  - Permanent (registry key, BIOS, ...)
  - Volatile (mutex, named pipe, ...)
  - Volatile markers have to be set each system reboot
- Marker type
  - Static: Fixed for all malware instances (Stuxnet)
  - Dynamic: Different for each infected system (Conficker)
  - Dynamic markers harder to extract (algorithm!)
- Coupling with malicious functionality
  - Independent of malware functionality (mutex not used otherwise, ...)
  - Part of/dependent on malware functionality (autostart key, API hook, ...)
  - Take into account when using marker as a vaccine!

Infection Marker Taxonomy (cont.)

- Time/Location of marker check
  - Check for marker can be in any malware binary



- Could make extraction of marker harder
  - Not always easy to get hold of dropper

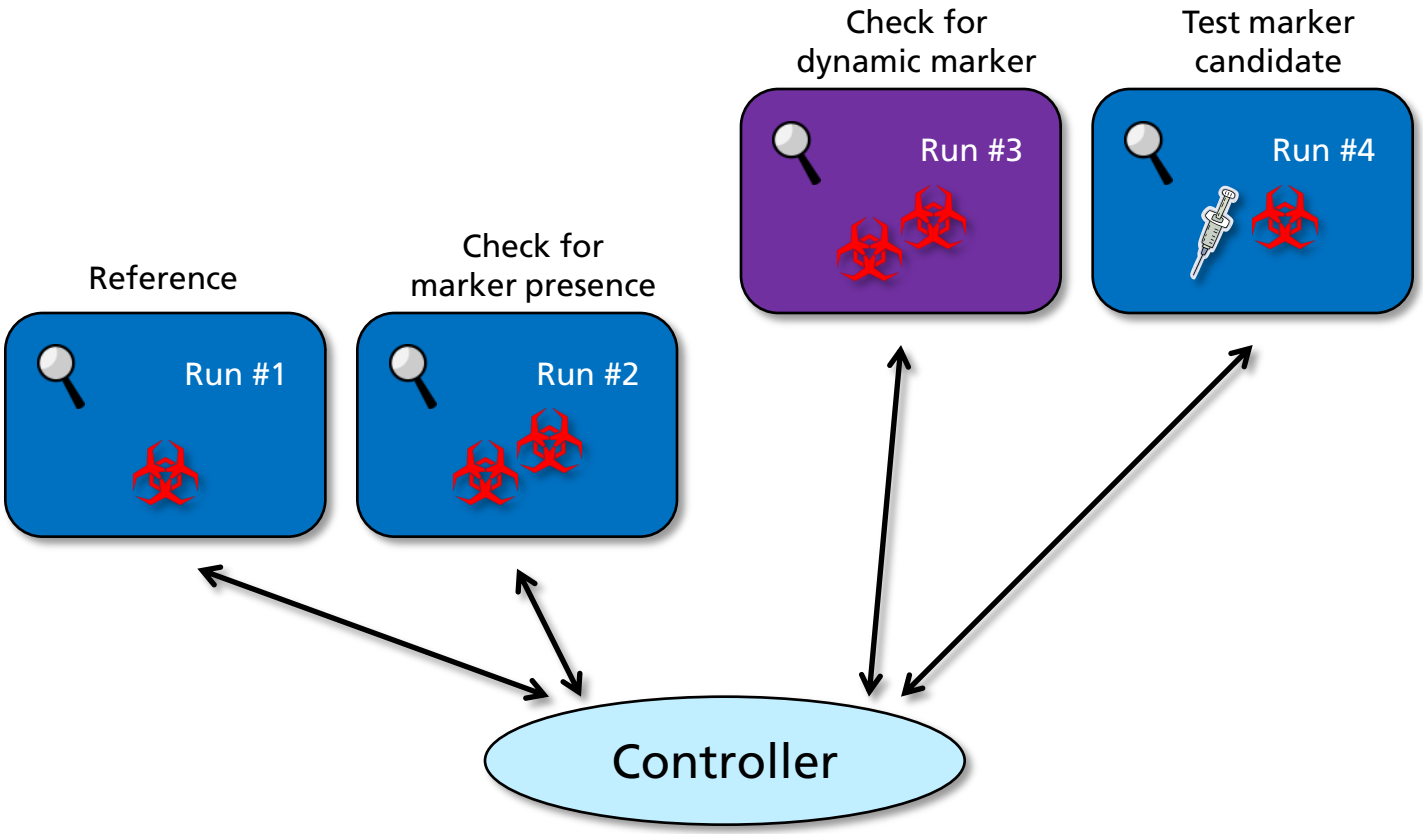
Automated Extraction Framework

General Idea

- In general, reverse engineering is very time consuming
  - Typical RE questions are very open in nature (“What is the C&C protocol?”, “What is the damage potential?”, ...)
  - Many intermediate steps can be automated...
  - ...but for special details and the big picture, a human expert is needed
- Extracting infection markers *can* be automated
- Assumptions:
  - Markers are set/checked for via confined set of OS APIs
  - Markers are checked early in the malware binary
  - If a marker is present, malware terminates quickly

Framework architecture

- *Controller* controls four virtual analysis environments
- *Process Observer* monitors relevant API calls of the malware



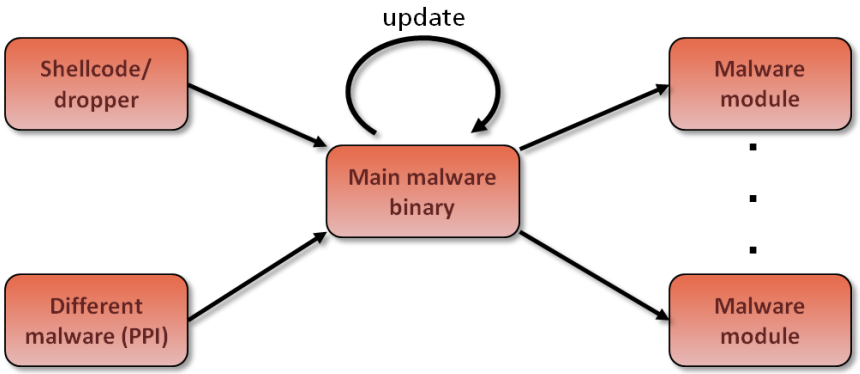
Evaluation

Evaluation Setup

- Questions to be answered
  - How many malware samples use infection markers?
  - What types of markers are used?
  - How many malware families are susceptible to vaccination?
- Corpus of 1496 malware samples
  - Randomly selected
  - From between 09/2009 and 09/2011
  - Sources: Honeypots, user submissions, spam traps
- Case studies
  - Sality
  - Conficker

Results: Corpus

- 889 out of 1496 samples (**59.4%**) use some kind of infection marker
- No statement can be made about the other 40.6%
  - Detected analysis environment?
  - “unwanted software” like keygens?
  - Only one component of a malware?



Results: Corpus (cont.)

- Rest of the results about the samples that *do* use an infection marker

- For **95.2%** of the samples, the marker could be determined
  - Run #4 shows that they are susceptible to vaccination!
  - Only for 4.8%, no conclusion could be drawn about marker type

- **98.4%** use mutexes
  - 1.0% use registry keys
  - 0.3% use named pipes
  - 0.1% use files

- 99.4% use static markers
  - Only 0.6% use dynamic markers (named pipe “AVIRA_<number>”, mutexes)

Results: Case Studies

- Qualitative analysis: Look at two most widespread malware families (Symantec Intelligence Report 02/2012)
  - Conficker, Sality

- **Sality:** Highly polymorphic file infector
- Creates global static mutex as infection marker
- Framework successfully identifies and extracts the marker
- Vaccination program is automatically created

- **Conficker:** Highly sophisticated worm
- Creates global dynamic mutex based on host name as infection marker
- Framework successfully identifies marker and its type
- Provides information for human expert to easily extract algorithm

Conclusion and Outlook

- Framework can automatically provide a vaccination program for a majority of malware
  - Mitigates propagation of new malware and protects critical systems
- Limitation: Dynamic analysis
  - Extend PoC framework to VM introspection
- Limitation: Unusual markers
  - Monitor on instruction level instead of API level
  - Use dataflow analysis to extract infection marker code
  - Works for dynamic markers, too
- Malware will likely use infection markers in the future, too
- Inherent properties make for a good counter-measure

Questions?