MAEC Conficker Worm Characterization (a case study)

Ivan Kirillov
Penny Chase – Desiree Beck – Robert Martin

MITRE

Why Characterize Conficker?

- Why do this at all?
  - Intellectual exercise to generate MAEC discussion
  - Test how well MAEC's 3-tier approach deals with real malware
  - Identify problems/limitations/questions
    - Useful for fleshing out other aspects of MAEC

- Why Conficker?
  - Well-studied with many analyses available
  - Still highly active (~7000K bots as of 11/09)
  - Complex, blended threat relying on variety of attack vectors
  - Multiple variants exist

Outline

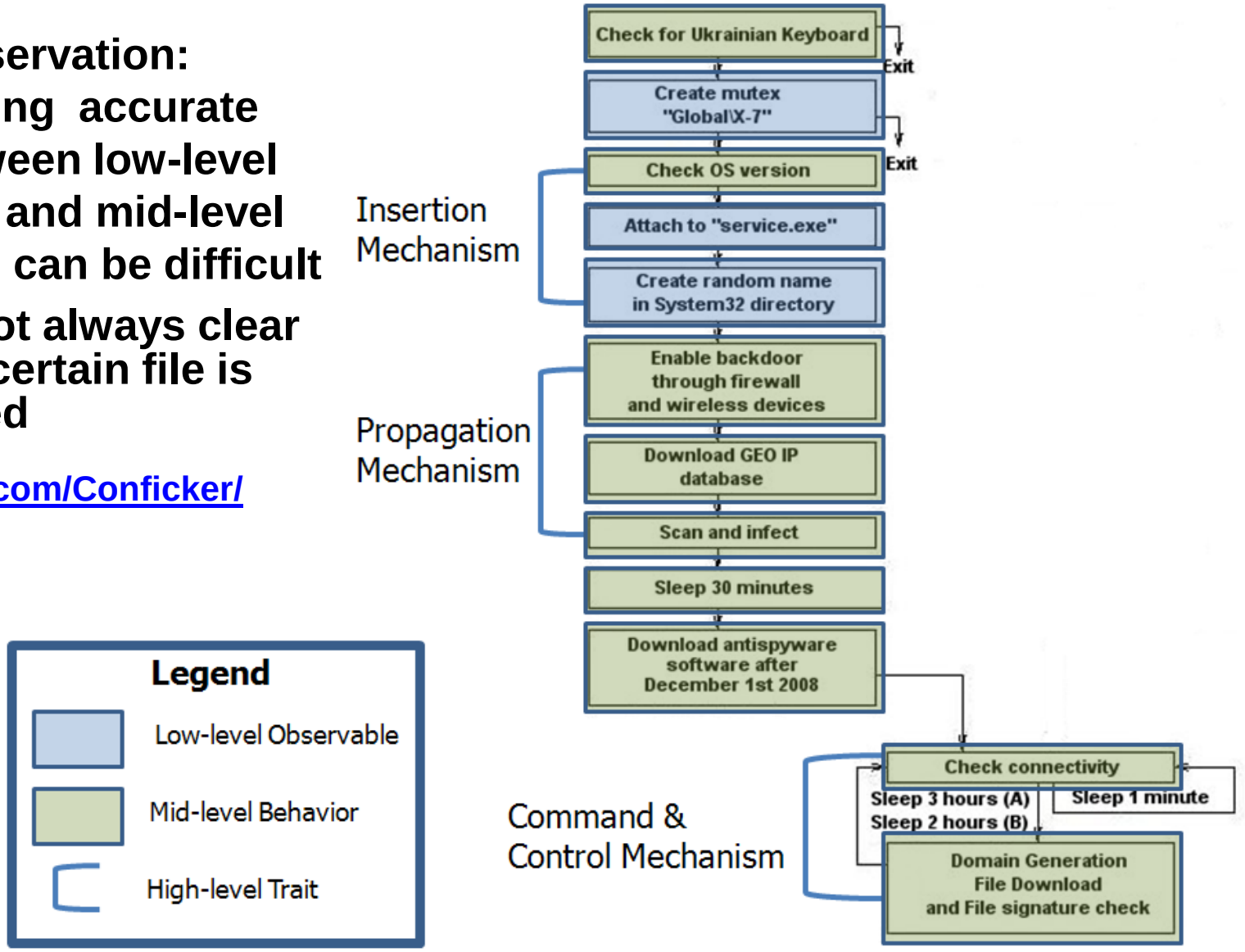
- **Simple MAEC example: SRI Conficker.A analysis**
  - **Top-Level control flow**

- **More elaborate MAEC examples: compilation of Conficker.B analyses**
  - **Self-Defense**
  - **File Sharing Propagation**
  - **Armoring/Capabilities**

Characterizing SRI's Conficker.A Analysis*

- **Initial Observation: Establishing accurate links between low-level attributes and mid-level behaviors can be difficult**
  - **E.g., Not always clear why a certain file is dropped**

*http://mtc.sri.com/Conficker/



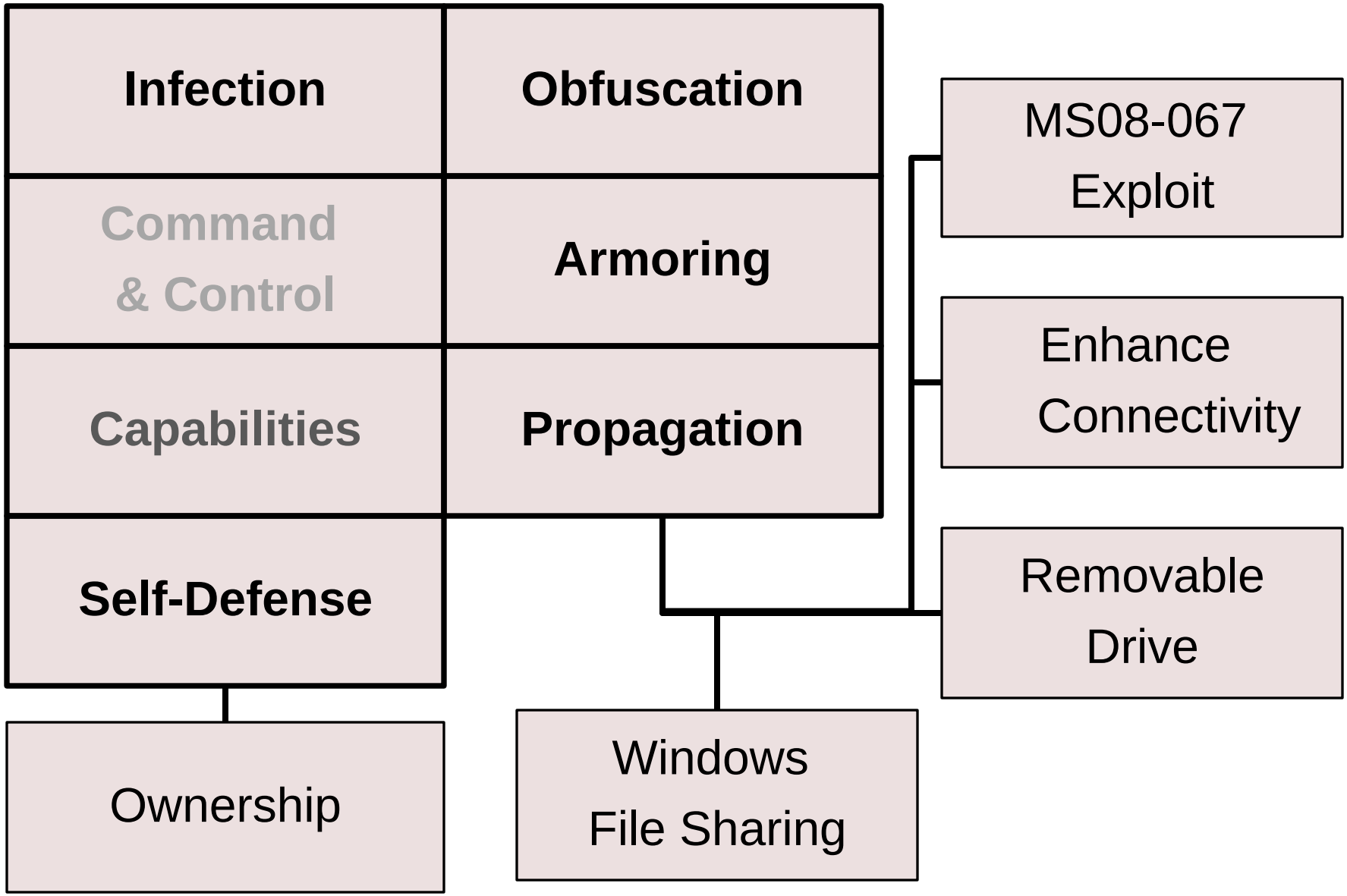
MITRE

Secondary attempt at characterization

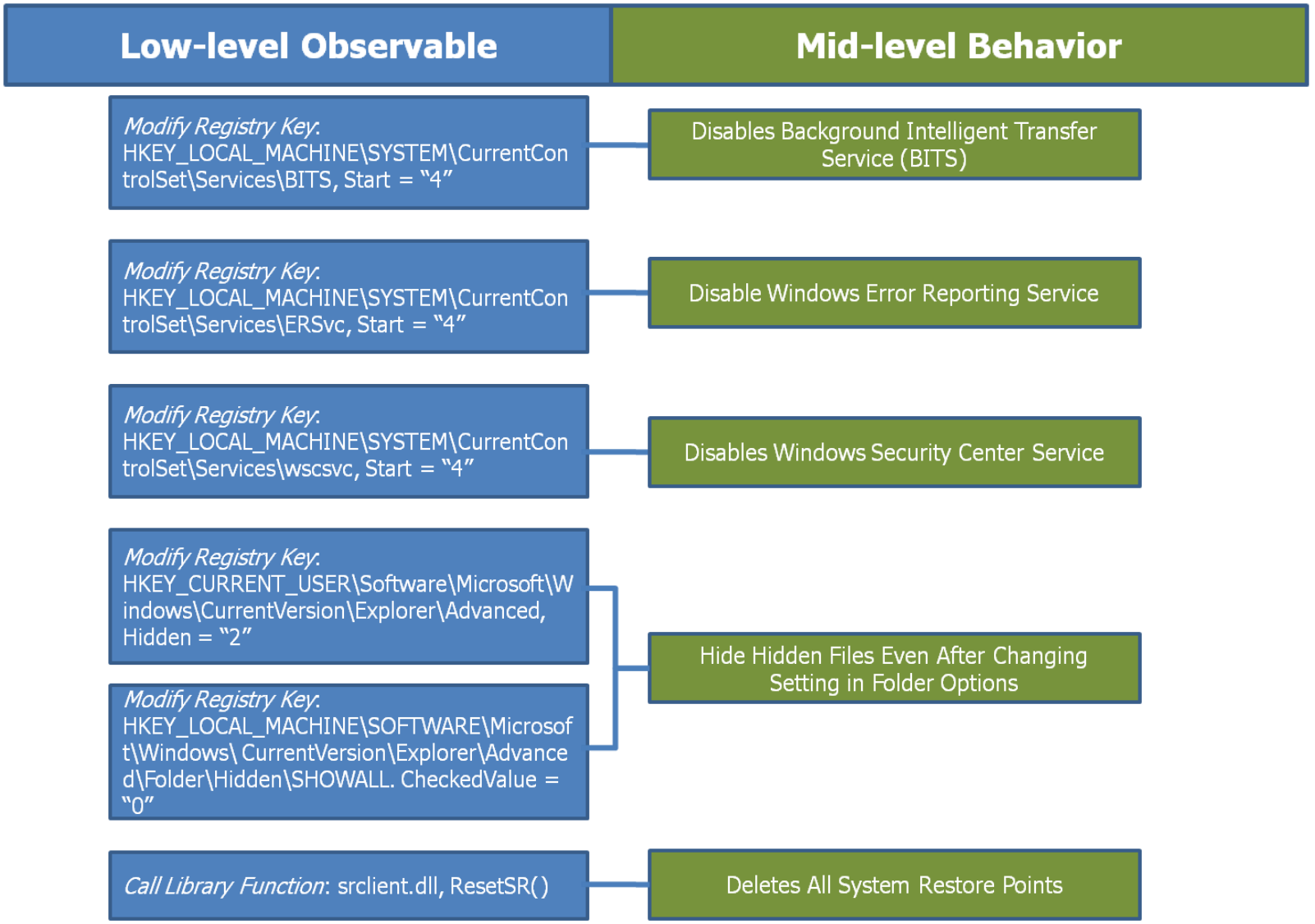
- We characterized Conficker.B
  - More interesting behavior than Conficker.A
  - Used multiple data sources

- Two-step process:
  - 1) Bin analysis data into MAEC's lower two tiers
    - Low-level attributes
    - Mid-level behaviors
  - 2) Group together attributes/behaviors into pre-defined high-level mechanisms

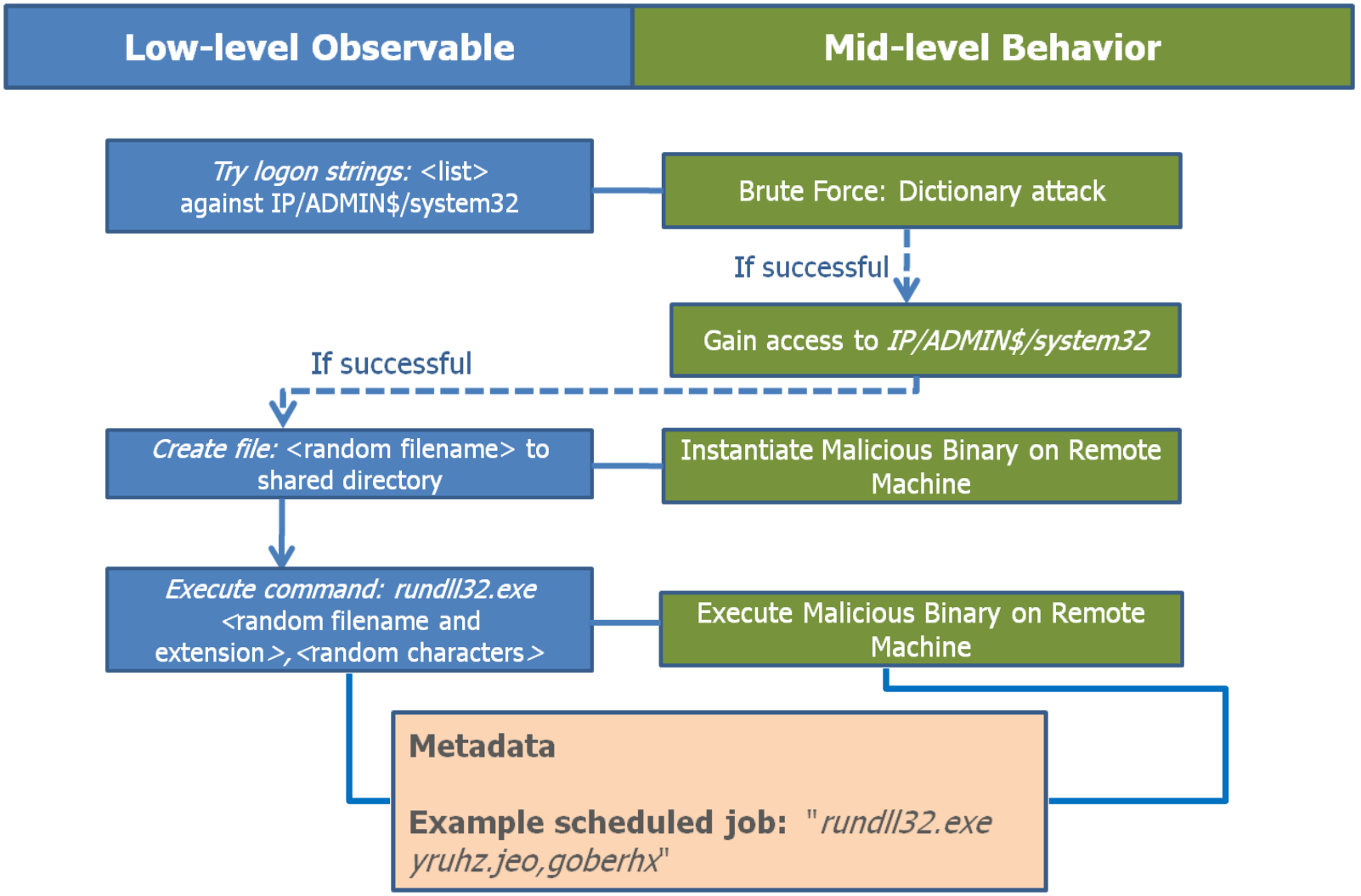
‘Full’ Conficker.B Characterization



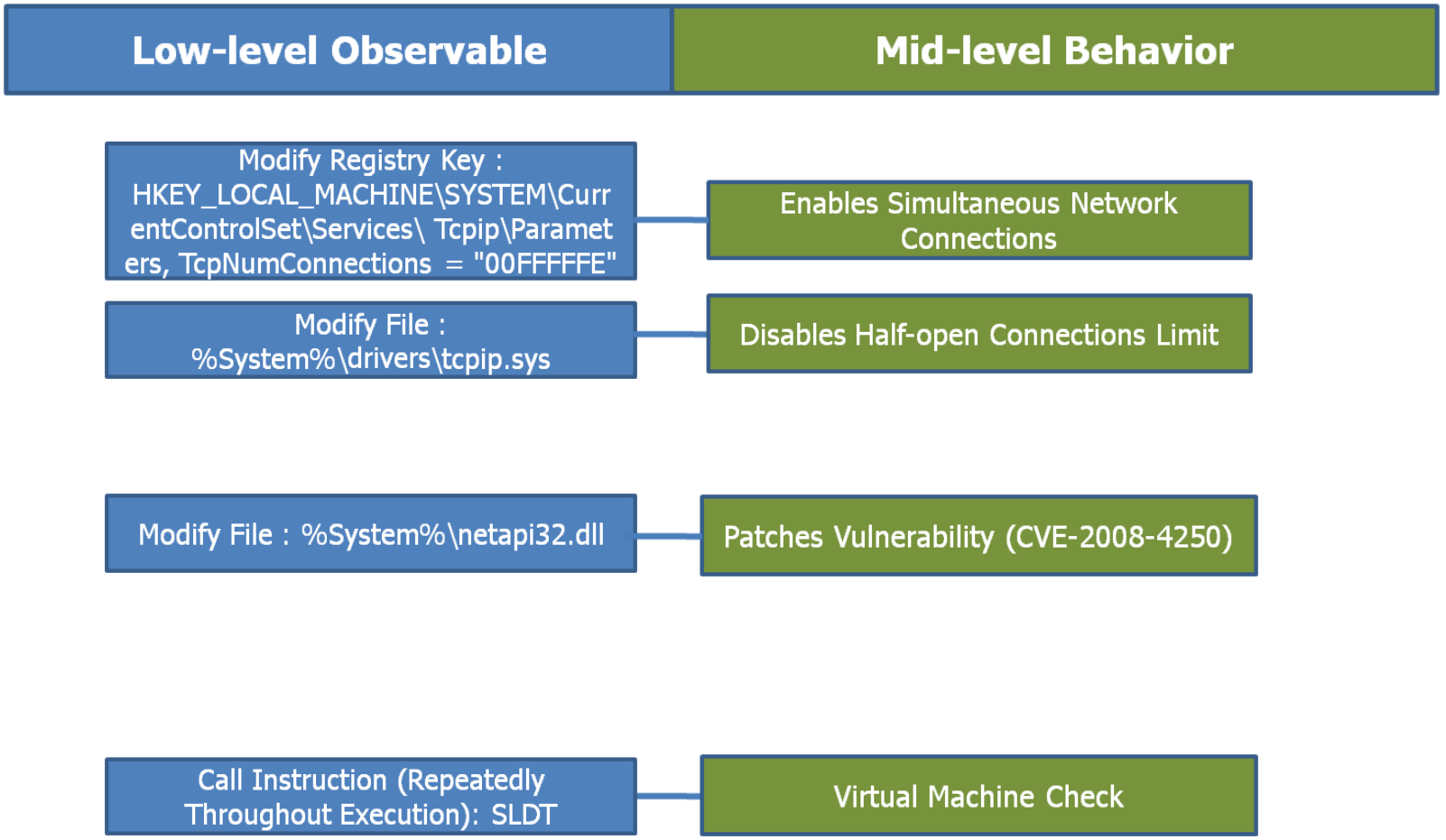
Conficker.B Self-defense



Conficker.B File Sharing Propagation



Conficker.B Propagation/Self-Defense/Armoring



MAEC Characterization Summary

- To gain a broader sense of how MAEC might be applied, we studied a variety of Conficker A and B analyses (see references)
  - AV/security vendor analyses
  - Open source research papers/analyses
  - Blogs

- General observations:
  - Overall, bottom-up approach was fairly intuitive
  - Mid-level behaviors often described in plaintext, without corresponding low-level attributes
  - Basic grammar for observables:
    - <Action> <object type> : <specific object>
    - E.g. Modify File: \%system32%\netapi32.dll

MAEC

Observations

Open Questions

Potential Features

Challenges

Attributes Observations (I)

- Low-level attributes are not purely artifacts, but can also refer to:
  - Change of state (e.g., creation/deletion of a registry key or file)
  - Transient behaviors (e.g., network traffic, creation of a process)
  - Features discovered during analysis (e.g., strings, encryption algorithm)
- Each MAEC tier needs to be finer grained, i.e., incorporate sub-levels, or else too restrictive
- Need to explicitly define appropriate level of detail for each attribute tier, or else risk overlap/ambiguity
  - Ex. “Attach to service.exe,” low-level or mid-level?

Attributes Observations (II)

- **Associating behaviors with a specific higher-level mechanism can be ambiguous**
  - **Can be difficult to partition between capabilities and other behaviors**
  - **E.g.**
    - **Conficker patches *tcpip.sys* to remove the half-open connections limit. Capabilities or propagation mechanism?**
    - **Conficker patches *netapi32.dll* to fix the MS08-067 NetPathCanonicalize exploit. Self-defense or capabilities?**

Attributes Open Questions (I)

- Does a payload consist strictly of actions & effects, or can it also encompass downloaded files?

- How to characterize machine code?
  - Provides valuable information on protection mechanisms, algorithms
  - Various formats – text description, actual code, bytes, etc.

Attributes Open Questions (II)

- Metadata – what exactly is it?
  - "One person's data is another's metadata"
  - Possible definition: any information not directly related to the actions and behaviors performed by malware
  - Some standard types: File Hashes, Dates, Threat Assessment, Comments, etc.
  - May not be possible to enumerate all other types

- Characterization of unintentional malware 'side-effects'
  - E.g. locking of user account due to brute-force dictionary attack
  - Difficult to correlate with malware execution
  - Simply include as metadata?

Schema Observations

- **Analysis perspective often based on reverse engineering vs. intrusion detection**
  - **Perhaps MAEC's schema should incorporate views based on this?**
  - **Relates to the two main classes of MAEC use cases**
  - **Could MAEC bridge the gap between the two?**

Schema Open Questions

- **Many mechanisms & processes (ex. Conficker.B's file sharing propagation) have logical, chronological, and other relationships**
  - **How should these relationships be represented?**
  - **Potentially adds a significant layer of complexity**
  - **However, useful for analysis, remediation, & detection**

- **How should MAEC incorporate enumerations of static attributes? (i.e. strings, IP addresses, etc.)**
  - **Useful data, but cumbersome to deal with due to size**
  - **Store this in MAEC clusters?**

Potential Features

- **Characterization based on Viewpoint**
  - **Particularly relevant for network mechanisms**
  - **E.g. propagation**
    - **‘attacker’ vs. ‘victim’**
    - **Victim machine sees different actions/behaviors than attacker**
- **Infrastructure identification/linkage**
  - **Identify at what specific level a malware behavior or effect is observable**
  - **E.g., is a behavior only host-level, or it can be seen at various levels of the network infrastructure?**

Challenges (I)

- **Repeatability**
  - **Would a MAEC characterization (of the same malware instance) be identical from analyst to analyst?**
    - **100% repeatability is unlikely**
    - **Large number of potential attributes to choose from**
    - **Unforeseen "emergent" malware behavior**
  - **In theory, a well-defined vocabulary and schema should ensure repeatability in most cases**

- **Discernibility**
  - **How much information is necessary to positively identify a malware instance?**
  - **Should MAEC deal with this?**
    - **E.g., confidence index**
  - **Important for typing and classifying malware by family**

Challenges (II)

- Usage & Adoption
  - MAEC will likely evolve to be large and fairly complicated
  - How can we ensure that the community will be able to use MAEC correctly?
- Implementation
  - Approach
    - What would be the best process for building up MAEC?
    - 1st Step: Low-level Attribute Enumeration
    - 2nd Step: Schema?
  - Enumerating & gaining consensus on mid-level behaviors

Conficker References

1) http://mtc.sri.com/Conficker/

2) http://www.ca.com/securityadvisor/virusinfo/virus.aspx?id=76852

3) http://www.honeynet.org/papers/conficker

4) http://www.sophos.com/sophos/docs/eng/marketing_material/conficker-analysis.pdf

5) http://isc.sans.org/tag.html?tag=conficker

6) http://www.symantec.com/connect/blogs/downadup-codex-edition-20

7) http://en.wikipedia.org/wiki/Conficker