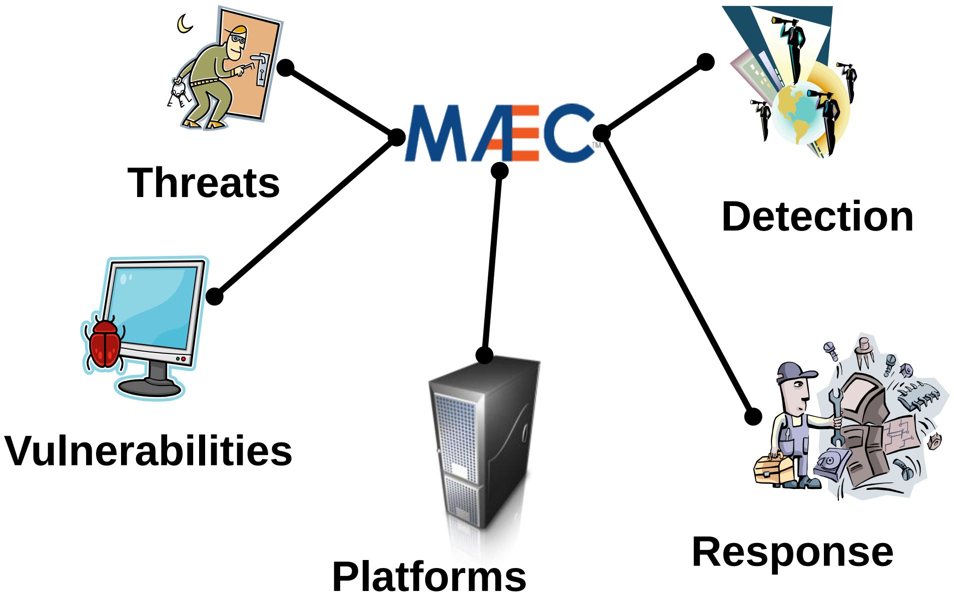
MAEC 2.x Explored

Penny Chase

3 October 2012

MITRE

Malware Attribute Enumeration and Characterization (MAEC)



Threats

Vulnerabilities

Platforms

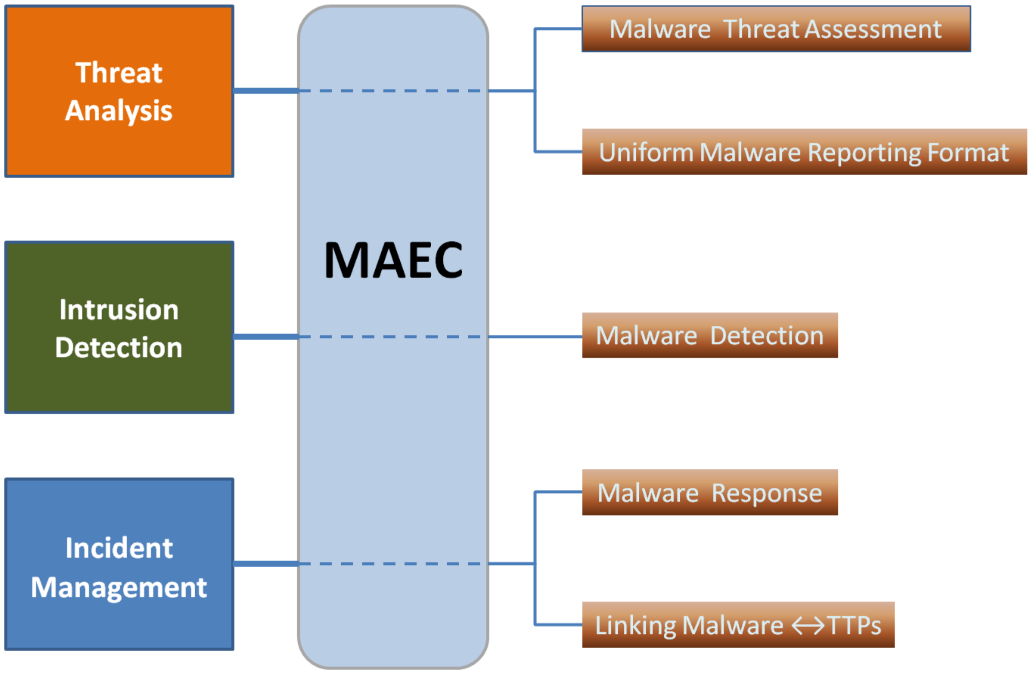
Detection

Response

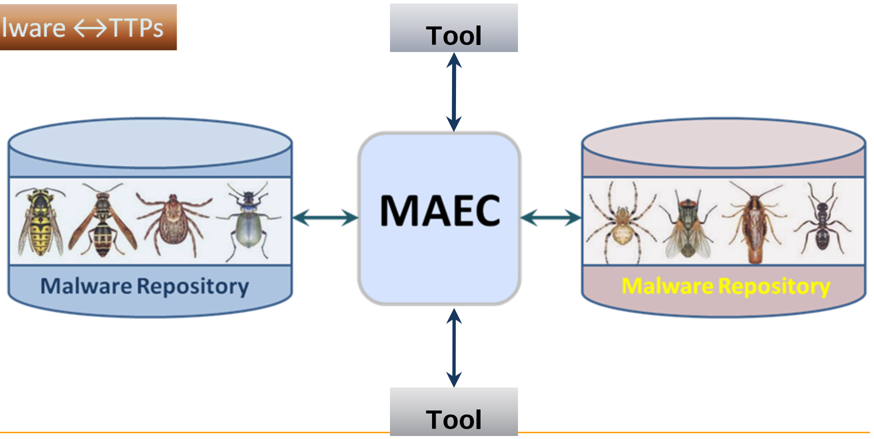
- Language for sharing structured information about malware
  - Grammar (Schema)
  - Vocabulary (Enumerations)
  - Collection Format (Bundle)
- Focus on attributes and behaviors
- Enable correlation, integration, and automation

MAEC Use Cases

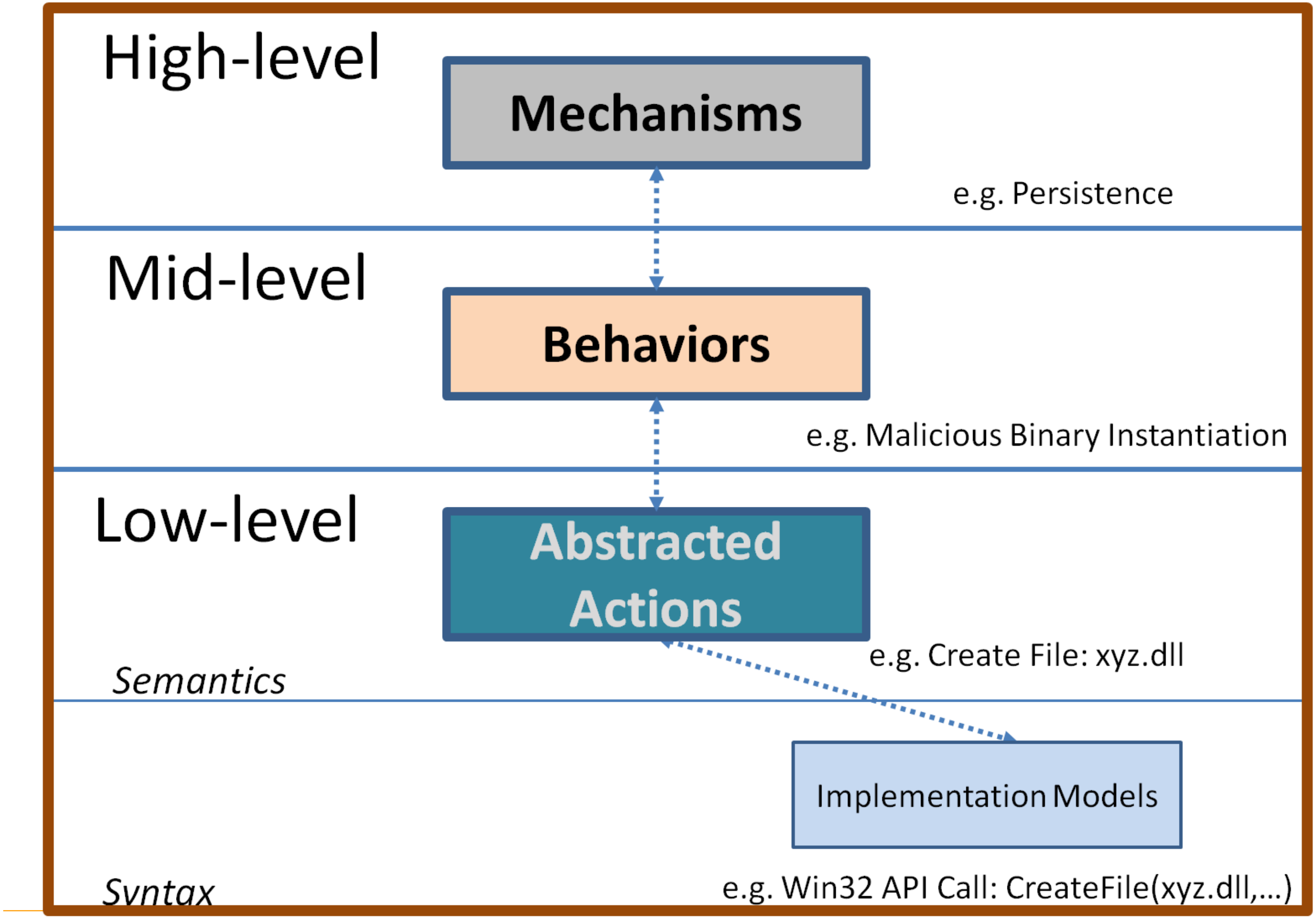
- Operational



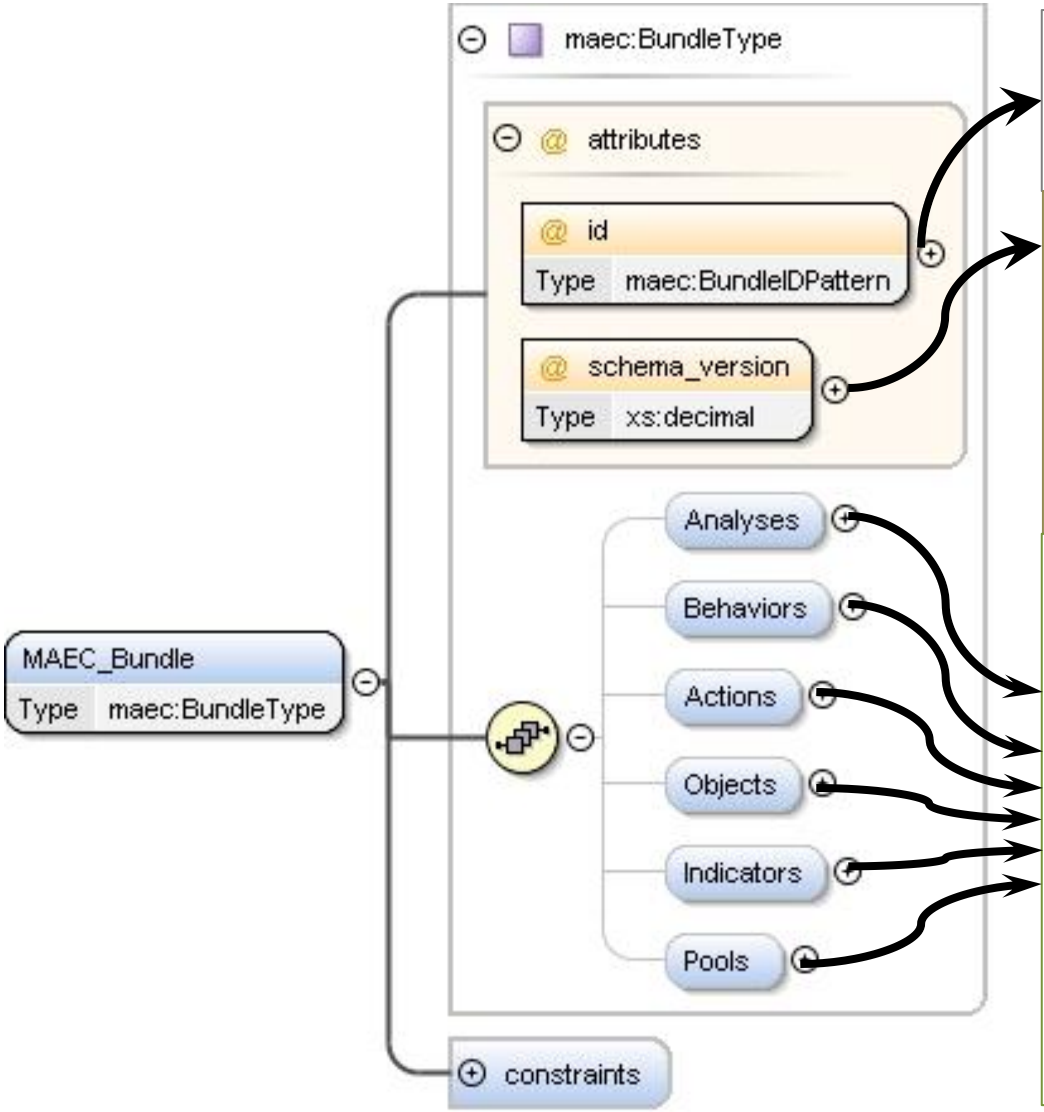
- Analysis
  - Help Guide Analysis Process
  - Standardized Tool Output
  - Malware Repositories



MAEC Structure Overview



MAEC's Bundle



MAEC Bundle ID

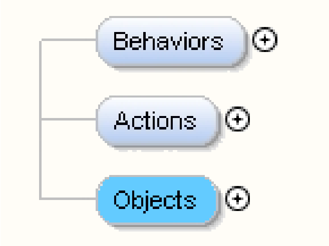
- Globally unique identifier

Schema Version

- Version of schema used to create bundle
- Used for validation

MAEC Components

- Attributes and metadata of a particular malware instance, family, class, etc.
- All optional
- Identified through various forms of malware analysis



MAEC & Malware Analysis Process I

Stage One: StaticTriage

11010101001

Malware Binary (PE)

1.

Imports

Exports

Resources

Strings

etc

2.

```
<maec:Object id="maec:tst:obj:1">
…
 </maec:Object>
```

3.

MAEC Bundle

Behaviors

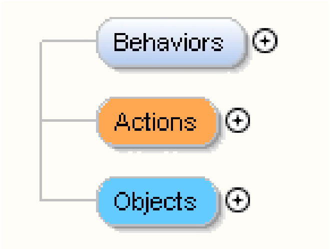
Actions

Objects

Obj 1

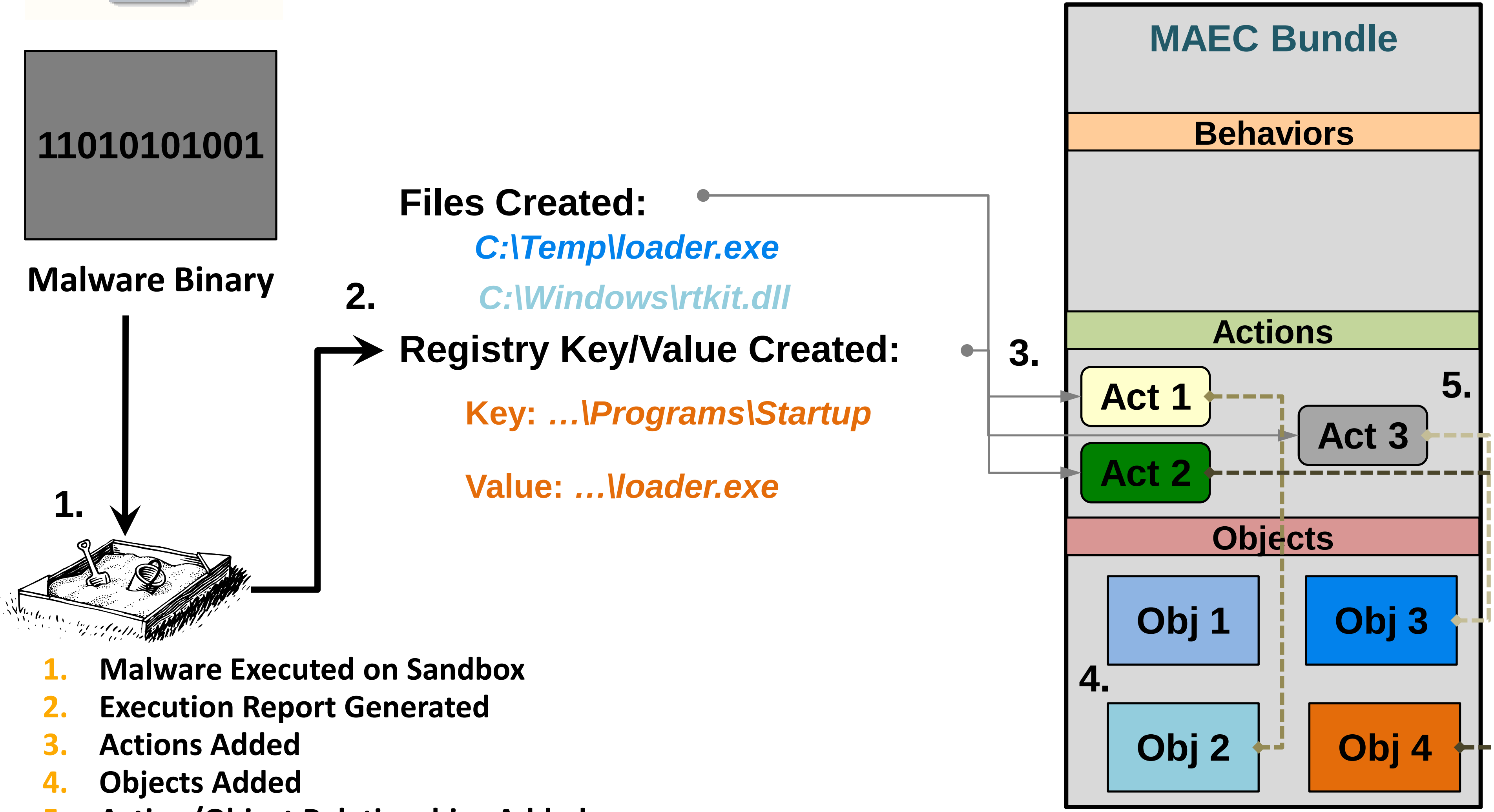
1. Features Extracted
2. MAEC Object Created
3. Object Added to Bundle

MITRE



MAEC & Malware Analysis Process II

Stage Two: Dynamic Analysis Triage



11010101001

Malware Binary

1.

2.

Files Created:

C:\Temp\loader.exe

C:\Windows\rtkit.dll

Registry Key/Value Created:

Key: *…\Programs\Startup*

Value: *…\loader.exe*

3.

MAEC Bundle

Behaviors

Actions

Act 1

Act 3

Act 2

5.

Objects

Obj 1

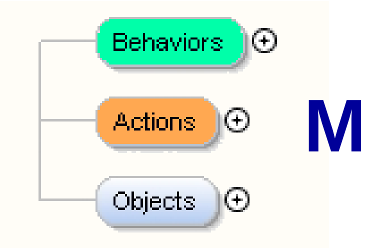
Obj 3

4.

Obj 2

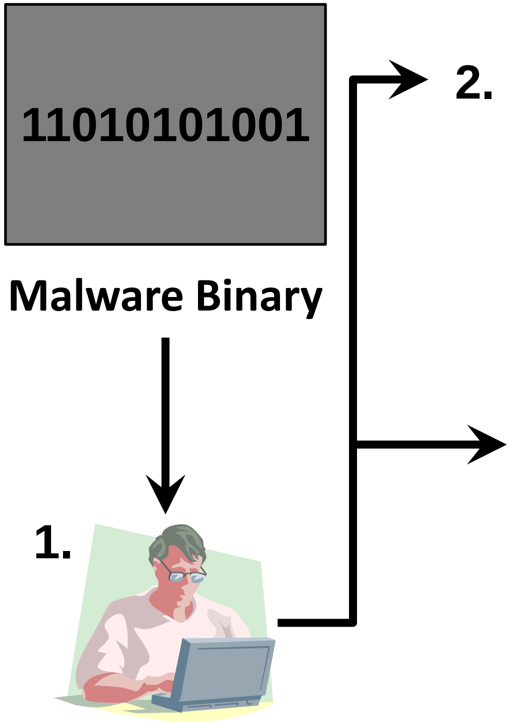
Obj 4

1. Malware Executed on Sandbox
2. Execution Report Generated
3. Actions Added
4. Objects Added
5. Action/Object Relationships Added



MAEC & Malware Analysis Process III

Stage Three: In-depth Manual Analysis



11010101001

Malware Binary

1.

2. Actions:

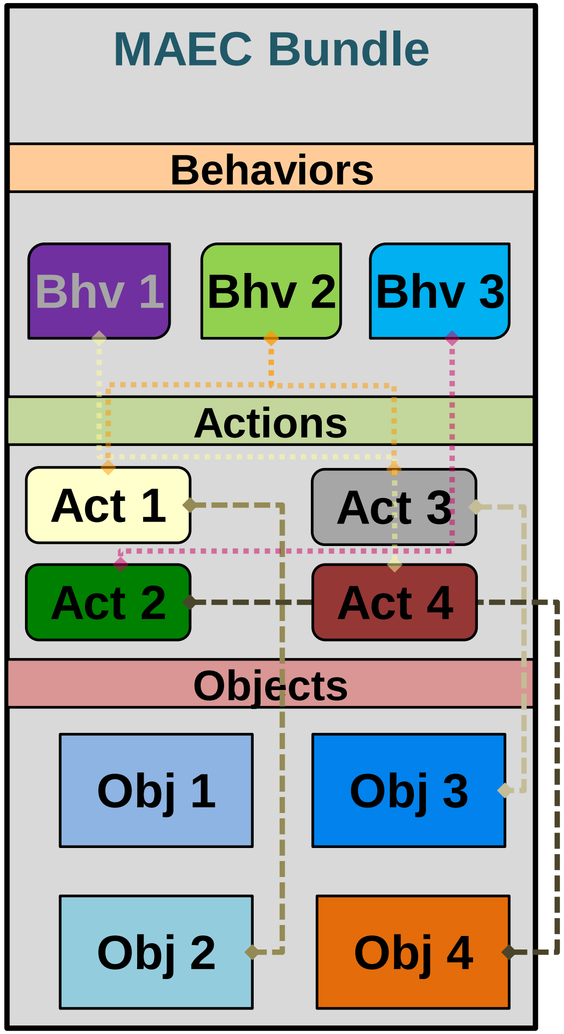
Start Winsock

3. Behaviors:

Winsock Startup

Malicious Binary Instantiation

Registry Persistence



MAEC Bundle

Behaviors

Bhv 1 Bhv 2 Bhv 3

Actions

Act 1 Act 3

Act 2 Act 4

Objects

Obj 1 Obj 3

Obj 2 Obj 4

1. Malware Analyzed Manually
2. New Actions Extracted and Added
3. Behaviors Extracted and Added

MAEC v 2.x

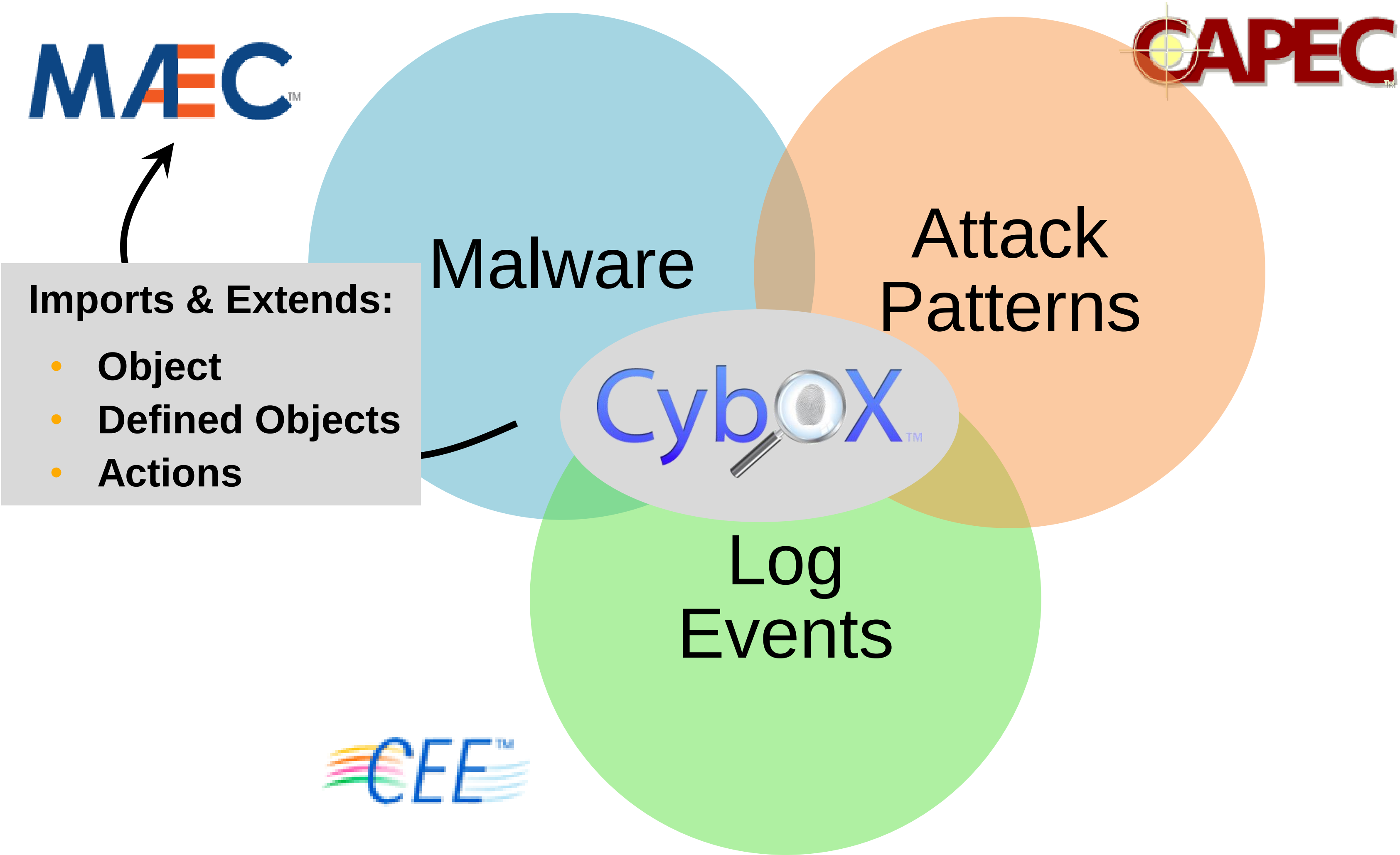
- XSD Schema Evolution
  - v1.0 – June 2010
    - Initial release
    - Focused on dynamic analysis output
  - v1.1 – January 2011
    - Added static analysis capability (PE attributes)
    - Schema changes, proper versioning implemented
  - v2.0 – January 2012
    - MAEC object model replaced with CybOX v 0.7
    - ActionType simplified
    - EffectType refined
    - Lots of ‘under the hood’ tweaks and minor additions
  - V 2.1 – April 2012
    - Support for CybOX v 1.0 (Draft)

MAEC™ v2.0 Additions

- Indicator Management Capability
  - Permits standard method of defining anti-malware indicators.
  - Linkages to other MAEC entities where appropriate. E.g. objects for specifying indicator used in detection.
- Relationship Support
  - Allows defining simple relationships between MAEC entities in an easy to use fashion. Examples: ParentOf, ChildOf, PrecededBy, etc.
- Many new enumerated types
  - Actions, Effects, Relationships, etc.

CybOX™

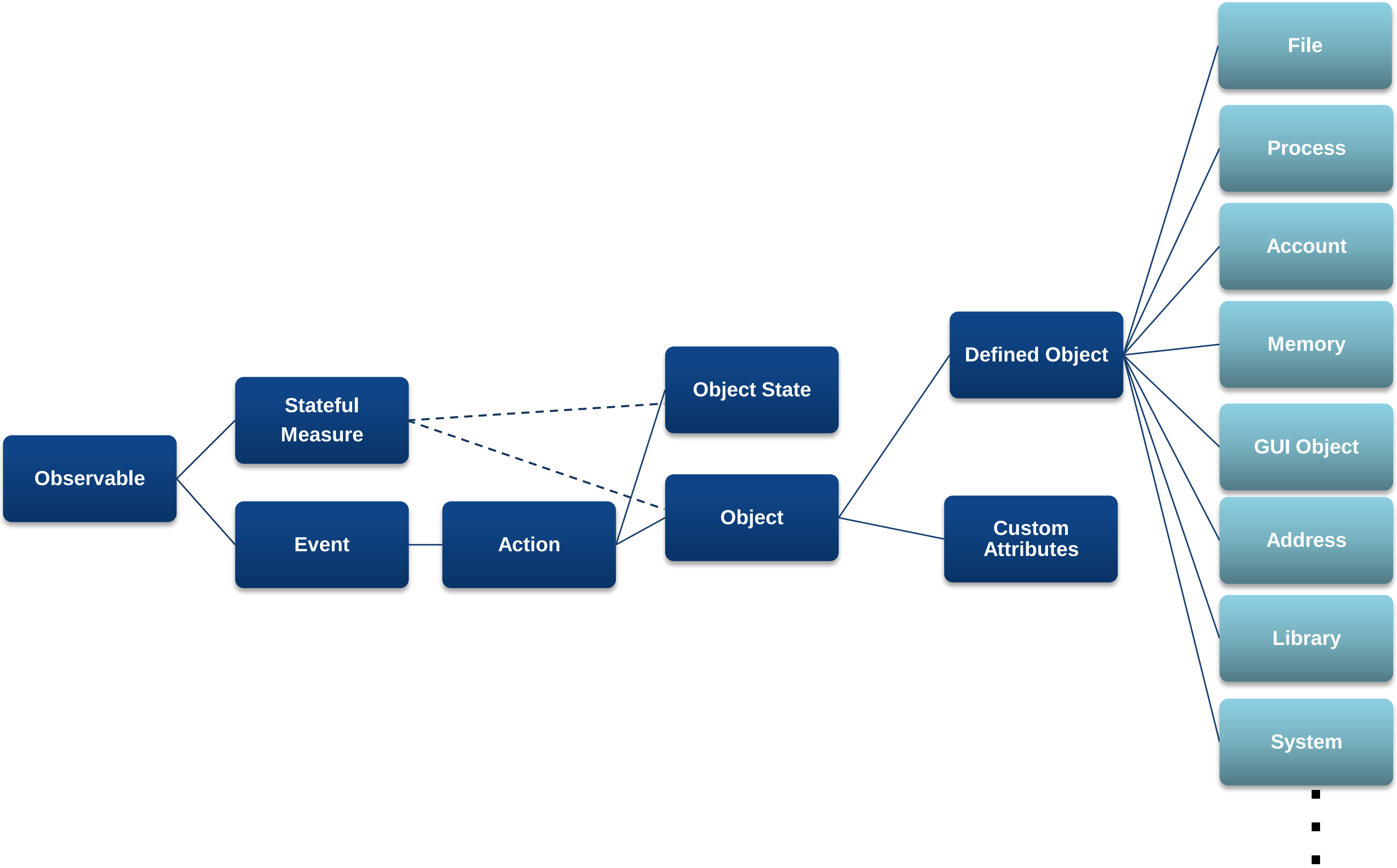
- **What is a cyber observable?**
  - **a *measurable event* or *stateful property* in the cyber domain**
    - **Some measurable events: a registry key is created, a file is deleted, an http GET is received, …**
    - **Some stateful properties: MD5 hash of a file, value of a registry key, existence of a mutex, …**
- **Cyber Observable eXpression (CybOX) is a standardized language for encoding and communicating information about cyber observables (http://cybox.mitre.org)**



Imports & Extends:

- Object
- Defined Objects
- Actions

Cyber Observable eXpression (CybOX) Schema Simple Overview



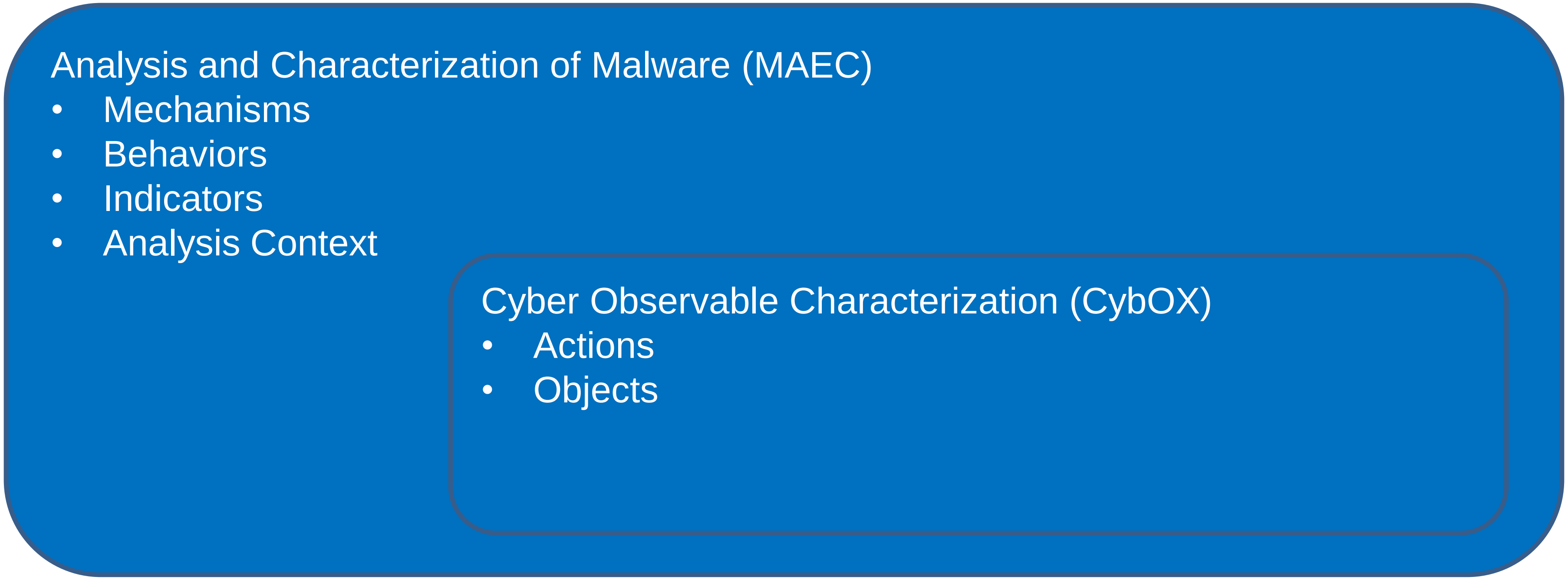
Various Defined Object Schemas

- Account
- Address
- API
- Code
- Device
- Disk
- Disk Partition
- DNS Cache
- DNS_Record
- Email Message
- File
- GUI
- GUI Dialog Box
- GUI Window
- Library
- Linux Package
- Memory
- Mutex
- Network Flow
- Network Packet
- Network Route Entry
- Network Route
- Network Subnet
- Pipe
- Port
- Process
- Product
- Semaphore
- Socket
- System
- Unix File
- Unix Network Route Entry
- Unix Pipe
- Unix Process
- Unix User Account
- Unix Volume
- URI
- User Account
- User Session
- Volume
- Win Computer Account
- Win Critical Section
- Win Driver
- Win Event
- Win Event Log
- Win Executable File
- Win File
- Win Kernel
- Win Kernel Hook
- Win Handle
- Win Mailslot
- Win Mutex
- Win Pipe
- Win Network Route Entry
- Win Network Share
- Win Pipe
- Win Prefetch
- Win Process
- Win Registry Key
- Win Semaphore
- Win Service
- Win System
- Win System Restore
- Win Task
- Win Thread
- Win User Account
- Win Volume
- Win Waitable Timer
- X509 Certificate

…

(more on the way)

MAEC and CybOX



Analysis and Characterization of Malware (MAEC)

- Mechanisms
- Behaviors
- Indicators
- Analysis Context

Cyber Observable Characterization (CybOX)

- Actions
- Objects

MAEC Tools and Utilities

- Python Bindings
  - For MAEC and CyBOX
  - Supports the development of MAEC tools and utilities
- MAEC Content Generation
  - Dynamic and static tool output translation
  - Native MAEC output
- Convert MAEC to other Formats
  - MAEC → HTML
  - MAEC → OVAL

MAEC Schema Bindings

- Permits:
  - Creation of new MAEC content
  - Manipulation of existing MAEC content

- Currently for Python 2.x
  - Full CybOX 1.0 draft support
  - Created with GenerateDS
    - http://cutter.rexx.com/~dkuhlman/generateDS.html

MAEC Tool Roadmap

Tool	Class	Language	Current Support	Avail.	License
MAEC/CybOX Python Bindings	Bindings	Python	MAEC v2.1/CybOX 1.0	Now	New BSD
MAEC → OVAL	Translator	Python	MAEC v2.1	Now	New BSD
Anubis → MAEC	Translator	Python	MAEC v2.1	Now	New BSD
GFI Sandbox → MAEC	Translator	Python	MAEC v2.1	Now	New BSD
MAEC → HTML	Translator	XSLT	MAEC v2.1	Now	New BSD
ThreatExpert → MAEC	Translator	Python	MAEC v2.1	Now	New BSD
MAEC Comparator*	Analysis	Python	MAEC v2.1	Now	New BSD
CuckooBox**	Native	Python	MAEC v1.1	Now	GNU GPL v3
Thug (Honeyclient)***	Native	Python	MAEC v1.1	Now	GNU GPL v2
PEFile.py → MAEC	Native	Python	n/a - in develop.	10/2012	New BSD
FireEye → MAEC	Translator	Python	n/a - in develop.	12/2012	New BSD
Norman Sandbox → MAEC	Translator	Python	n/a - in develop.	12/2012	New BSD
MAEC → Suricata	Translator	Python	n/a – in develop.	12/2012	New BSD

* Blake Hartstein (iDefense), MITRE updated to MAEC v2.1

** Cuckoo Team

*** Angelo Dell'Aera (Honeynet Project)

MAEC Development (1/2)

- Collaboration between industry and government
- Leverage existing resources, such as
  - IEEE Industry Connections Security Group's Malware Metadata Exchange Format schema v 1
  - Mandiant's openIOC
- Participate in standards efforts
  - IEEE ICSG Malware Metadata Exchange Format WG
    - Adding capability to MMDEF schema for capturing blackbox behavioral metadata about malware
    - Will likely import MAEC/CybOX, especially MAEC Objects and Actions
  - IETF Managed Incident Lightweight Exchange (MILE) WG
    - MAEC may be part of the MILE Structured Cybersecurity Information RFC (extensions to IODEF)

MAEC Development (2/2)

- Community contributions
  - Schema development
  - Support MITRE’s tool development
    - Provide schemas, documentation, examples to support translator development
  - Tool development
    - Blake Hartstein’s comparator script
    - Incorporate MAEC in open source projects (e.g., CuckooBox, Thug)
    - Discussions with vendors to provide native MAEC support (e.g., GFI, Norman, FireEye)

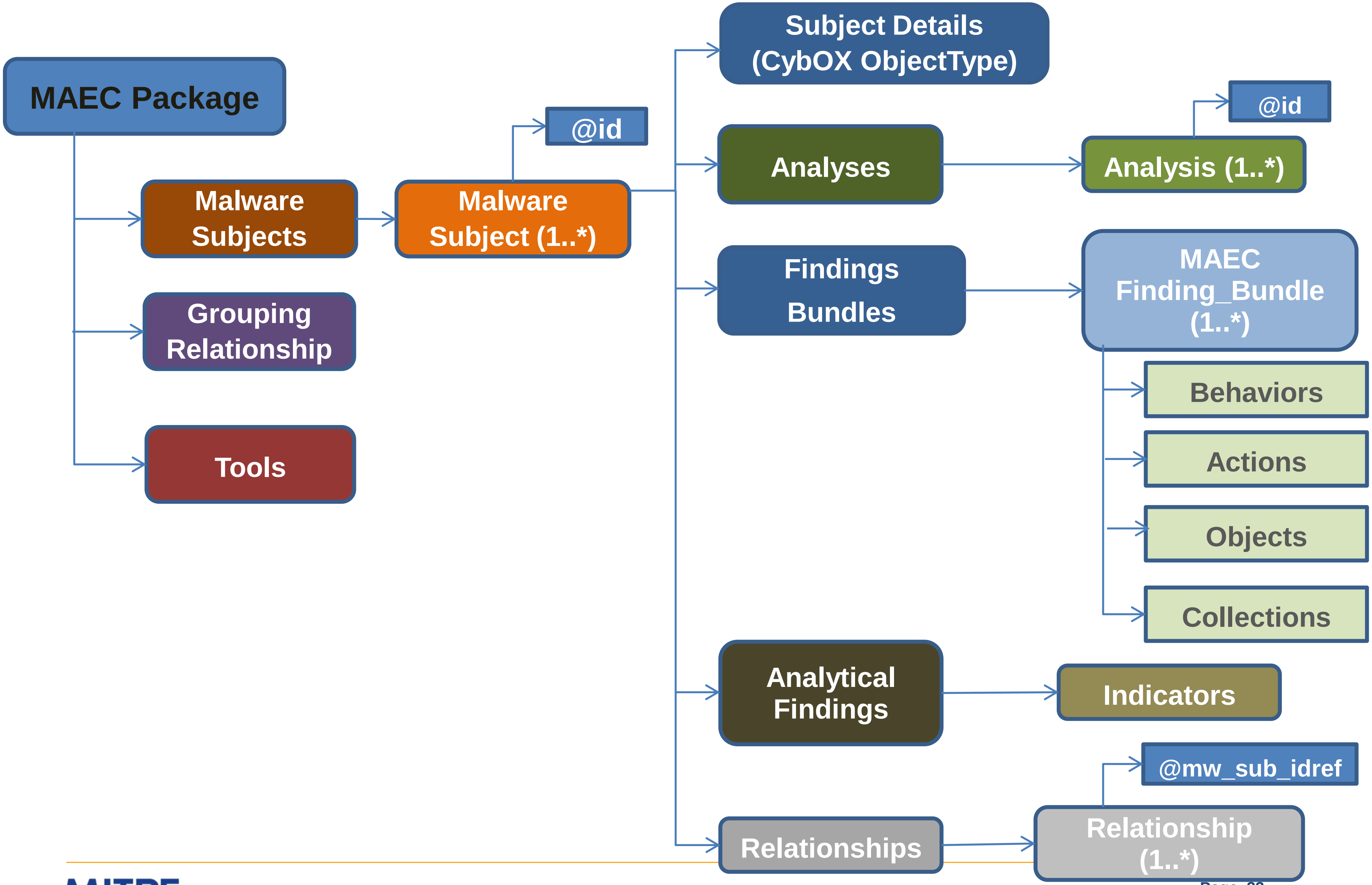
MAEC 3.x Plans

- MAEC 3.0
  - Refactor MAEC bundle to support bundle management and abstract bundle
  - End of October 2012
- MAEC 3.1
  - Support CybOX v 1.0 Final
  - End of December 2012
- MAEC 3.2
  - Initial implementation of mechanisms and required modifications to behaviors and relationships
  - End of March 2013

MAEC 3.0: Bundle Refactoring

- Goals:
  - Support bundle management:
    - Merging bundles created by multiple analyses
    - Collections of (MAEC v 1 and v 2) bundles
      - Algorithmically (e.g., clustering)
      - Related files (e.g., dropper and dropped files)
  - Abstract bundle
    - Enable MAEC to characterize malware without being tied to specific samples
- In MAEC 1.x and 2.x a bundle was created as the result of one or more analyses of a single malware sample

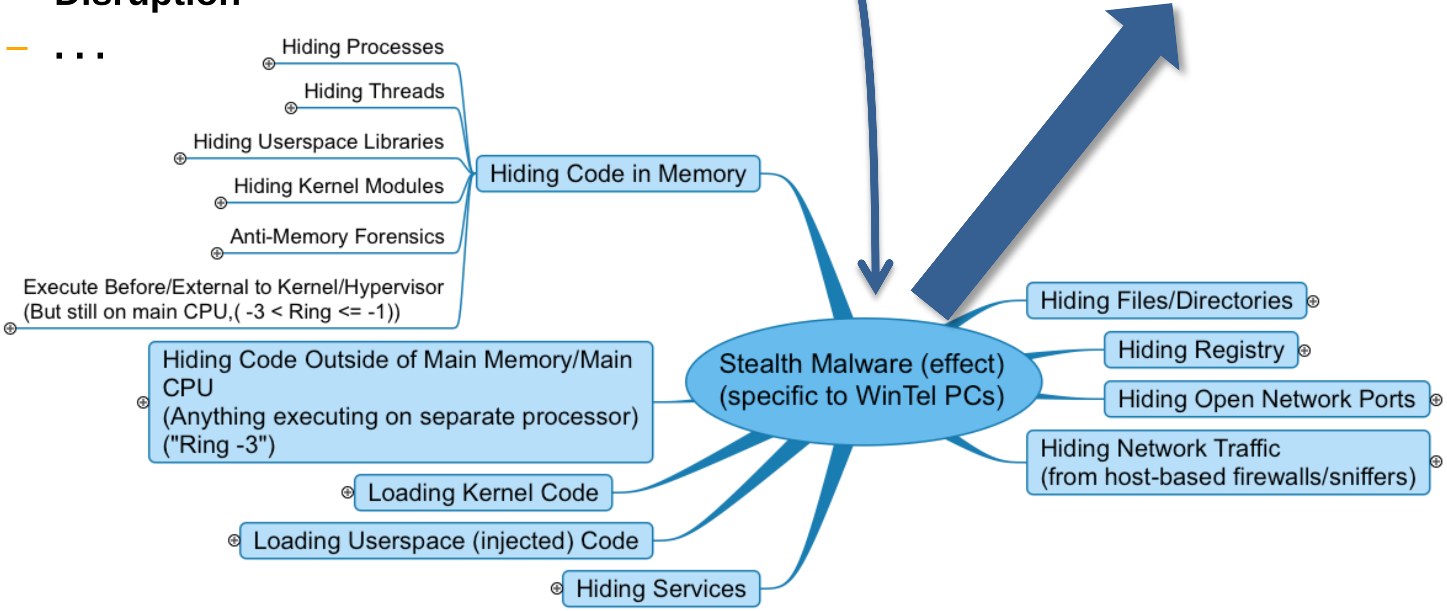
Proposed MAEC 3.0 Structure



MAEC 3.2: Mechanisms and Behaviors

- Enumerations
  - Exploit/Infect
  - Stealth
  - Self-Protection
  - Code Obfuscation
  - Persistence
  - Propagation
  - Command and Control
  - Information Stealing
  - Disruption
  - . . .

- Stealth Mechanism Schema
  - ID
  - Name
  - Parent
  - Children
  - Privilege Level
  - Objects
  - . . .



Stealth Malware (effect)
(specific to WinTel PCs)

Hiding Code in Memory

Hiding Processes

Hiding Threads

Hiding Userspace Libraries

Hiding Kernel Modules

Anti-Memory Forensics

Execute Before/External to Kernel/Hypervisor
(But still on main CPU,(-3 < Ring <= -1))

Hiding Code Outside of Main Memory/Main CPU
(Anything executing on separate processor)
("Ring -3")

Loading Kernel Code

Loading Userspace (injected) Code

Hiding Services

Hiding Files/Directories

Hiding Registry

Hiding Open Network Ports

Hiding Network Traffic
(from host-based firewalls/sniffers)

MITRE

Future MAEC Tools

- MAEC API
  - Allow users to generate valid/usable MAEC content without perfect knowledge of the schema
  - Current ‘MAEC Helper’ is a very simple, early take on this concept
- MAEC Bundle Management
- MAEC View Construction

For More Information

- **Web site: http://maec.mitre.org**
- **Mailing list: http://maec.mitre.org/community/discussionlist.html**
- **MAEC Development Group: http://handshake.mitre.org**
- **Github: https://github.com/MAECProject/Tools**