OPENC2: PROTECTING OUR FUTURE AT MACHINE SPEED

Dr. Lisa Mathews,
Security Engineer
Agenda

- Background
- Design Philosophy/ Principles
- OpenC2 Syntax
- OASIS Transition
- Current Plan
- Call for Participation
Background: The Motivation and Vision

- **Challenge**
  - Coordinated Defense in Cyber Relevant Time

- **Vision: Future Cyber Defense Tactics**
  - Sharing of indicators
  - Coordination of response actions
  - Automated, multi-part actions at machine speed

- **Strategy**
  - Decouple Functional Blocks and Standardize Interfaces
  - Identify and fill gaps as they pertain to Cyber Threat Indicator Sharing and Response
  - Participate in a diverse and collaborative environment

Standardization is a Key Enabler for Automation
Integration in the Absence of Standards
Decoupled Security Stack: Making Security More Manageable (& interchangeable)

**Sensing**
- STIX
- CybOX
- MITRE Cyber Analytic Repository
- OpenCOA / OpenPlaybook
- OpenC2 Actuator Profiles

**Sense Making**
- STIX Patterning Language

**Decision Making**
- OpenC2

**Acting**
- OpenC2 Actuator Profiles

**Identify**
- SENSORS
  - Instrumentation, raw telemetry, netflow, A/V logs, firewall logs, external threat intel

**Protect**
- ANALYTICS
  - Use cases, correlation, anomaly detection, heuristics

**Detect**
- COURSES OF ACTION / ORCHESTRATION:
  - Workbenches, Workflow Engines, Security Orchestration and Automated Response

**Respond**
- ACTUATORS
  - Things that block, move, stop/start, delete, quarantine, etc.
OpenC2 at a glance

- Unambiguous Machine-to-Machine Communication
- Simplicity
  - Low overhead on sensor and actuator
- Focuses on ‘Acting’ portion of cyber defense
- OpenC2 assumes the following has been done:
  - Sensing; ‘What’ triggers the action
  - Analytics; ‘Why’
  - Decision; ‘Which’ action
- OpenC2 will leverage pre-existing protocols and efforts
OpenC2 Focusses on ‘Acting’

- **STIX**
  - Standard Threat INTEL object
  - Supports Analysis

- **TAXII**
  - Standard Transport protocol
  - Supports Secure Exchange

- **OpenC2**
  - Standard Command Language
  - Supports Automated Response

OpenC2 is part of a Suite of OASIS Standards
OpenC2 Design Principles

- Lightweight
  - Efficient machine-to-machine communications
- Abstract
  - Focuses on ‘What’ to do versus “Device Specific”
- Extensible
  - Extensions enable additional precision and flexibility
- Agnostic
  - Transport, authentication, integrity controls etc.
  - Enables flexibility with respect to implementation

Enable Unambiguous Machine-to-Machine Command and Control Messages
OpenC2 Assumptions

- Basic Assumptions
  - The analytics have been done
  - The decision to respond has been made
  - The Transmitting and Receiving entities are authorized to do so
  - Assured transport
OpenC2 Parameters

- The Lexicon Decouples the aspects of the commands
  - ACTION: What is to be done
  - TARGET: What you are doing it to
  - ACTUATOR: Who is performing the command
- Extensions permit additional precision to the commands
  - SPECIFIER: Identifies general to specific targets or actuators
  - OPTIONS: Provide additional details for the command, target, actuator
- Benefits of decoupling
  - Facilitates integration of new technologies
  - Supports high level effects based AND device specific use case
# Example Actions / Targets

<table>
<thead>
<tr>
<th>Actions</th>
<th>Targets</th>
</tr>
</thead>
<tbody>
<tr>
<td>scan</td>
<td>device</td>
</tr>
<tr>
<td>locate</td>
<td>directory</td>
</tr>
<tr>
<td>create/query/set/delete</td>
<td>domain_name</td>
</tr>
<tr>
<td>report</td>
<td>email_addr</td>
</tr>
<tr>
<td>notify</td>
<td>email_message</td>
</tr>
<tr>
<td>deny/contain/allow</td>
<td>file</td>
</tr>
<tr>
<td>start/stop/restart</td>
<td>ipv4_addr/ipv6_addr</td>
</tr>
<tr>
<td>pause/resume</td>
<td>mac_addr</td>
</tr>
<tr>
<td>detonate</td>
<td>ip_connection</td>
</tr>
<tr>
<td>redirect</td>
<td>process</td>
</tr>
<tr>
<td>update</td>
<td>url</td>
</tr>
<tr>
<td>save</td>
<td>user_account</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

...
Example OpenC2 Command

# Block all ftp data transfers from hosts and request ack. Note that the five-tuple is incomplete

```json
{“action”: “deny”,
 “target”: {
  “type”: “openc2:five-tuple”,
  “specifiers”: {
    “Layer4Protocol”: “TCP”,
    “src-port”: 21
  }
}

“actuator”: {
  “type”: “openc2:firewall”,
  “specifiers”: {endpoint},
  “options”: {openc2: drop}
},

“command-options”: {
  {“id”:“UUID=123e4567-e89b-12d3-a456-426655440000”}
  {response=TRUE}
}
```
OpenC2 Use Case

- **Abstract Use Case**
  - **Mitigate Evil Domain**
  - **Local Orchestrator**
    - Deny Evil Domain
    - Scan evil.pdf
    - Contain Infected WS
  - **Actuator executes command**

- **Implement across**
  - Agnostic Transport Mechanism
Prototype Implementation

- Orchestrators and Actuators converge on the OpenC2 message fabric
- OpenC2 ‘Proxy’ maps to hardware API
- Converging on Message Fabric facilitates implementation
Demonstrate Vendor Agnostic

- Allows corporate wide sharing of cyber defense tactics
- Minimizes impact when changing components
Demonstrates Architecture & Technology Agnostic

- Deny Command is executed REGARDLESS of product
- Simplifies integration of new technologies that achieve similar actions
- Unified tactical approach independent of equipment set
OpenC2 Documentation Approach

- Core Language Specification
  - Actions
  - Default target namespace
  - Semantics, syntax
  - Profile framework
  - Minimum to implement

- Actuator Profiles
  - Scope and applicability
  - Required and optional actions and nuances in the context of the actuator
  - Applicable targets
  - Specifiers and options for a class of actuators

- Implementation Guides
Transition to OASIS

- OASIS Kickoff meeting on June 7, 2017
- OASIS Technical Committee
  - Monthly Meetings
- OpenC2 Sub-committees
  - Language SC meets weekly
    - Specification of syntax, actions and targets
    - Actuator Profile SC meets biweekly
    - Gathering of ‘frenemies’
  - Implementation Considerations SC meets monthly
    - External dependencies such as IA, Transport etc.
OASIS Technical Committee Organization

OpenC2 TC
Joe Brule / Sounil Yu

Language
Jason Romano / Duncan Sparrell

Actuator Profiles
Jyoti Verma / Dave Kemp

Implementation Considerations
Bret Jordan / David Lemire
Language: Proposed Work Plan

- Sequence of Committee Specification Drafts (CSDs)
  - November TC CSD version 0.1.0 (approved on 11/14/2017)
    - Document layout
    - Actions
  - January TC CSD version 0.2.0
    - Targets
    - Mandatory-to-Implement (MTI) Encoding (JSON) structure agreed to
  - February TC CSD version 0.3.0
    - Responses, Alerts and Modifiers
    - Address unresolved details from 0.1 and 0.2
  - March TC Committee Specification Version 1.0.0
    - Actuator Information
Profile Development Approaches

- **Bottom-up**
  - Start with full list of 30-40 actuator functions, then factor out common functionality into a higher-level profile (2nd level Endpoint / Network / Manager, or top level Generic)

- **Top-down**
  - Start with 2nd level profiles
  - Split off more granular profiles as necessary

- **Actuator SC will pursue both approaches**
  - List of specialized actuator functions under development
  - 2nd level profiles also being considered
  - “Meet in the Middle”
Actuator Profile Granularity

Roadmap / lists of potential profiles:

https://docs.google.com/document/d/1nlXzQOD0xT-SMp4vfIELvmV8AiWhmXwn8PoEjMEe3g

Successive refinement:
Implementation Considerations

- Subcommittee focuses on interoperability
  - OpenC2 Ecosystem
  - Message Transfer Mechanisms
  - Information Assurance Features

- Primarily guidance (non-normative) products

- Tracking and complementing Language and Actuator SCs
Request of the Cybersecurity Stakeholders

- Use Cases
  - Exercise the Language & Identify Gaps

- Actuator Profile Data Call
  - Which Actions from the Language Specification will be used?
  - Which Targets from the Language Specification do you act upon?
  - What Specifiers do you need?
  - What Options are available in your product?
We Welcome Your Support

- Please show your support on https://wiki.oasis-open.org/openc2/UsersSupportingOpenC2
- Ask your vendors to show their intentions on https://wiki.oasis-open.org/openc2/ProductsWithOpenC2
Questions? Comments? Complaints?

- OpenC2 Leadership
  - Joe Brule (Co-chair)
  - Sounil Yu (Co-chair)
  - Joyce Fai (Executive Secretary)
  - Duncan Sparrell (Language Subcommittee)
  - Jason Romano (Language Subcommittee)
  - Dave Kemp (Actuator Profile)
  - Jyoti Verma (Actuator Profile)
  - Dave Lemire (Implementation Considerations)
  - Bret Jordan (Implementation Considerations)

- Contact us at openc2-chair@lists.oasis-open.org
Backups
Observations

- Actuator profiles are the mechanism by which industry-specific knowledge is incorporated into the OpenC2 standard
  - Industry participation will enable success
  - Industry collaboration will define the distinction between the standard and product differentiators

- Actuators to be defined by capabilities
  - ‘Hardware’ based approach is redundant and does not support NFV
  - Multiple ‘profiles’ may be required
  - ‘Foundational’ profile?
OpenC2 External Dependencies

- OpenC2 is necessary but insufficient
- OpenC2 Assumes
  - Decision has been made
  - Action is warranted
  - The command can get there intact and securely.
  - Recipient is authenticated and authorized.
- OpenC2 Focuses on the ACTING portion of cyber defense

OpenC2 Implementations will FAIL without a robust means to convey commands!
威胁攻击跨 enclave C2 — 一个角色可能:

- 改变 C2 消息以降级或阻止防御性反应。
- 发送欺诈性命令打开 enclave B 进行攻击。
- 查看 C2 交通以获得对防御性反应的警告。
- 扰乱网络服务以防止 C2 消息的传递。

Diagram:
- ENCLAVE A
- ENCLAVE B
- Orchestrator
- Device
Threats against intra-enclave C2 – an actor may:

- alter C2 messages to degrade or halt defensive responses,
- send false commands to open up an enclave for attack,
- Spoof C2 replies to disrupt defense or confuse defenders,
- Flood devices to prevent delivery of C2 messages.
OpenC2 Derived Security Requirements

To combat or mitigate threats against inter- and intra-enclave cases, OpenC2 may need:

- **Confidentiality** – ability to control visibility of OpenC2 messages to only authorized recipients.
- **Integrity** – assurance that OpenC2 message sent is the message received
- **Authorization** – limit sending and receiving to authorized parties only
- **Authentication/Proof-of-Origin** – ability for all recipients to know the source of a message or identify of the sender
- **Availability** – assurance that messages can always be sent
- **Reliability** – assurance that messages are delivered to all intended recipients
Prototypes Posted on Github

- Yuuki
  - University of Maryland
  - Implements OpenC2 as multiple dispatch on type
  - Actuators are dynamically created and hot swappable

- OrchID
  - Zepko
  - OpenC2 proxy built in Django

- OCAS
  - S-fractal
  - OpenC2 API Proxy written in ERLANG

- Pub-sub on bsd
  - G2
  - Implementation of OpenC2 on open source firewall written in C
Additional Prototype Efforts

- OpenDXL Message Fabric
  - Joint INTEL/ G2

- Cisco ASA Prototype Implementation
  - Orchestrator issues DENY and ALLOW to Cisco ASA based on CTIA update

- Reactor Master/ Reactor Relay
  - Zepko
  - Use of OpenC2 in inter-domain commanding use case

- IACD Course of Action Implementation
  - JHU/APL on behalf of NSA
  - 15 OpenC2 Actions issued to Nine actuators
  - Implemented in Java
Actuator Roadmap

- **Goals**
  - Define initial set of cyber defense functions
  - Identify initial set of profiles to be developed

- **Approach**
  - Identify Product Categories & perform Market Survey
  - Identify the set of functions/features common across the category
  - Define conformance tests
  - Create Profile
OpenC2 as a Concept
At the Language Description Level
OpenC2 at the Actuator Profile Level
End Notes

Contribution: Status and Way Forward Brief by Joe Brule, Executive Director, OpenC2

Cisco® is a registered trademark of Cisco Systems, Inc.

Oasis® is a registered trademark of Oasis, Inc.

Yuuki © Joshua Brule

OrchID © OpenC2

OCAS © sFractal Consulting