OpenC2

302 Sentinel Drive Suite 300, Columbia MD 21046

Information assurance implementation considerations

Michael Larmie and Patrick Izzi @ G2 Inc.

2016

Contents

[Introduction 2](#_Toc454798875)

[Preface: 2](#_Toc454798876)

[Focus 2](#_Toc454798877)

[Threat Landscape 3](#_Toc454798878)

[Operating Environment 4](#_Toc454798879)

[Security Implications of Language Structure 5](#_Toc454798880)

[Security Topics 5](#_Toc454798881)

[Authentication 5](#_Toc454798882)

[Authorization 5](#_Toc454798883)

[Accounting 6](#_Toc454798884)

[Auditing 6](#_Toc454798885)

[Non-Repudiation 7](#_Toc454798886)

[Integrity 7](#_Toc454798887)

[Confidentiality 8](#_Toc454798888)

[Encryption 8](#_Toc454798889)

[Cryptographic Hash Functions 9](#_Toc454798890)

[Verb Whitelisting 10](#_Toc454798891)

[Architecture & Implementation Considerations 11](#_Toc454798892)

[Peer-to-Peer 11](#_Toc454798893)

[Full Mesh PVC 11](#_Toc454798894)

[Publish – Subscribe (Pub-Sub or Multi-tenancy) 13](#_Toc454798895)

[Strict Type Enforcement 14](#_Toc454798896)

[Integration with Configuration Management 14](#_Toc454798897)

[Out of Band Management (OOBM) 14](#_Toc454798898)

[Domain Federation 15](#_Toc454798899)

[Certification and Accreditation (C&A) – Regulatory Compliance 15](#_Toc454798900)

# Introduction

OpenC2 is a concept used in conjunction with the emerging strategy of Integrated Adaptive Cyber Defense (IACD). In context with IACD, an objective is to define a set of universally understood Integrated Defensive Cyber Operations (IDCO) Command and Control (C2) commands and a command structure that can be contextualized for specific environments. The universally understood command set, OpenC2, will cover the range of common actions expressed at a high level. These high level commands are then interpreted in terms of the context, the concerns and assets of the entity executing the commands. These interpretations need to be commonly understood locally, but not globally. They are only meaningful in the context in which they are used, so cross-context adjudication is not required. Each enterprise or enclave provides its own implementation via the specific sensors and actuators available, and so can specify the common command set to reflect the implementations of which it is capable.

This paper prepares developers and implementers of the OpenC2 language with a primer on some of the most prevalent security topics and architecture constructs which should be considered during the deployment of OpenC2.

## Preface:

Achieving adequate information security for organizations, mission/business processes, and information systems is a multifaceted undertaking that requires:

* Clearly articulated security requirements and security specifications
* Well-designed and well-built information technology products based on state-of-the-practice hardware, firmware, and software development processes
* Sound systems/security engineering principles and practices to effectively integrate information technology products into organizational information systems
* Sound security practices that are well documented and seamlessly integrated into the training requirements and daily routines of organizational personnel with security responsibilities
* Continuous monitoring of organizations and information systems to determine the ongoing effectiveness of deployed security controls, changes in information systems and environments of operation, and compliance with legislation, directives, policies, and standards and
* Information security planning and system development life cycle management.

From an engineering viewpoint, information security is just one of many required operational capabilities for information systems that support organizational mission/business processes— capabilities that must be funded by organizations throughout the system development life cycle in order to achieve mission/business success.

# Focus

* Emphasis placed on separation of problem space and solution space in context to OpenC2 implementations
* Focus on implementation of OpenC2 in real world scenarios
* Introduce concept that a poorly designed/implemented OpenC2 may enrich an environment for potential adversaries
* Objective of OpenC2 should be to provide rapid response while not introducing new avenues of exploitation.

The development of a new information technology language can present many temptations for designers and developers to leap directly into solutions for the myriad of problem sets laid out before them. Incorporation of the OpenC2 language into the broader Active Cyber Defense ecosystem increases the complexities confronted by solution developers. The totality of this challenge calls attention to the need for a disciplined Systems Engineering (SE) approach incorporated with the Information Systems Security Engineering (ISSE) process to produce a holistic and secure solution. One of the cornerstone ideals from the ISSE process is to separate the problem space from the solution space. The problem space defines what the system will do, while the solution space defines how the system will solve the problems. This paper introduces fundamental security problem spaces for consideration; the specific solutions are left to developers for implementation.

The key to OpenC2 security design is to strike the right balance between operational performance and the security of the environment. In order to assist designers with achieving this objective, the focus of this paper is the introduction of security topics which warrant consideration during the development process. The broad range of threats and operating environments, which are introduced in the following sections, hints to the scope of the challenge when incorporating security into the OpenC2 language.

## Threat Landscape

* Management planes of systems/networks are high-value targets given the often weak access controls necessarily in place.
* CND systems are also key targets as part of subverting access controls and detection.
* Many attacks focus on protocol/language exploitation based on lax enforcement and lack of input validation.

The connected world has evolved from isolated, enterprise networks to a rapidly changing ecosystem emerging with the moniker Internet of Things (IOT.) As both the ubiquity and complexity of connected systems emerge, the corresponding attack surface presented by these systems will require comprehensive security designs to address the new threats.

The traditional management plane sends and receives traffic related to the configuration and operation of networks and devices. Securing the management plane includes, but is not limited to, authentication, authorization, accounting, encrypted connections, accurate time, and the collection of network traffic statistics and logging records. Management plane traffic if often permitted firewall exceptions and crosses VLAN boundaries for both necessity and ease of implementation. The enabling nature of management plane traffic dictates the protection of this data stream. Bad actors are fully aware of the opportunity presented by management plane traffic and the shortcuts many system administrators and developers leverage to simplify the complexities of hardening the management plane. OpenC2 is by nature a highly sensitive and specifically operational class of management traffic which requires a commensurate level of protection.

One of the most common application security weaknesses is the failure to properly validate input data. Input validation can thwart misconfigured actions before processing by the end device. A note of caution: poorly constructed input validation routines can be computationally expensive, meaning system resources could be needlessly consumed due to machine-speed operations. Auditing the input of commands and the outcome of actions can enhance the detection of malformed input data.

Traditional network defense models struggle to keep pace with the advancement of attack vectors. Network exploitation events are almost certain, which leads to the necessity for both robust security measures and enriched detection capabilities. The security analytics needed for early detection can only be provided with a rich set of auditing features. Auditing records should include who/what accessed the device, what occurred, and when it happened. Comprehensive audit record collection and analysis will often provide the coverage of discovery when network defense measures are misconfigured or inadequate.

## Operating Environment

* Mixed environments
* Mixed device smarts
* Inheritance of legacy devices and architectures
* Varied regulatory and compliance landscape leads to varied design choices
* The OpenC2 language should be thin and flexible enough to fit within a wide range of environments.

As networks merge towards the Internet of Things, the complexion of these systems will become increasingly heterogeneous. This mixed environment of devices will also come with varied degrees of capabilities. The OpenC2 language needs the flexibility to fit within this wide performance envelope. Coupled with the adoption of emerging technologies is the potential inheritance of legacy devices and architectures. Many older platforms do not support features now considered common or even out dated and their limited resources can present a significant challenge when addressing modern security structures. Computing resources (e.g., CPU, RAM/flash memory, interface speeds) are often sparse and the performance of the system may degrade beyond the point of satisfactory levels once tasked to process additional security layers.

Another aspect which feeds into legacy architecture designs is the varied and ever-evolving regulatory landscape. This factor should be considered when assumptions are made regarding the expected operating environment. The diversity and depth of various regulations can require very specialized skills to design architectures which realize compliance. Looming deadlines driven by governance mandates often lead to unorthodox solutions and the security of the OpenC2 framework should address the broad spectrum of compliance challenges.

## Security Implications of Language Structure

* Only essential and core security elements should be added to syntax
  + desired effect not achievable using message stack
  + applicable to all targets and actuators
  + applicable to all contexts
* Context-specific security elements as specifiers
  + ignored by devices that don't have the required capability
  + not mandatory in all domains
* Security analysis must account for possibility of specifier being ignored
* Enduring, enclave-wide context-specific security requirements can be articulated as policy, rather than sent with each command
* Hooks for future development

# Security Topics

## Authentication

Authentication is the process of verifying the identity or other attributes claimed by or assumed of an entity (user, process, or device), or to verify the source and integrity of data. In the overall operations of OpenC2 in context of machine-to-machine, the systems need to securely authenticate to verify it is the authorized system involved in any interaction and not a rogue entity. With the increasing number of Internet-enabled devices, reliable machine authentication is crucial to allow secure communication in automated network environments. In the Internet of things (IoT) scenario, almost any imaginable entity or object may be made addressable and able to exchange data over the network. It is important to realize that each access point is a potential intrusion point. Each network device that has an OpenC2 component needs strong machine authentication. In any architecture deployment, consider implementing authentication and verification calls within an Orchestrator and Actuators. The call out between components should at least verify that a unique identifier would be used to verify that each component verifies that they are identify who it is. There are many challenges to find the right authentication model that can support a machine-to-machine communication method at extremely fast rates of operating.

## Authorization

Following authentication, a user must gain authorization for doing certain tasks. After logging on to a component, such as an orchestrator, the user account may try to issue actions. The authorization process should be in place to determine whether the user has the authority to issue such actions. Authorization is the process of enforcing policies: determining what types of qualities of activities, resources, or services/actions a user is permitted. Usually authorization occurs within the context of authentication. Once a user has been authenticated, they may be authorized for different types of actions depending on the policy assigned. In the context of OpenC2, this policy enforcement is necessary because even though a certain device can execute any action it is tasked to perform, some actions should not be executed at only certain times, or not at all as they could lead to network or device compromise.

There are actions within the OpenC2 language that can be grouped by their general activity. Each group of actions may need to have some level of authorization to allow such actions to be performed. One set of actions that control permissions and accesses are a desired area of interest of an attacker that can use these actions for his advantages. The OpenC2 commands (e.g., DENY, CONTAIN, ALLOW, etc.) could lead to network compromise if used in the wrong context. Based on the security issues with authentication, the same concerns are with command authorization. OpenC2 syntax does not have inherited user schemas or even user to command mapped permissions that allow or not allow the ACTION to be performed.

Consider the following:

STOP (

[target (type=process, [id=123345])],

[actuator (type=endpoint ,[id=9876t6])],

[immediate]

)

The STOP construct would stop a process on an endpoint with an immediate action when the message is received. If this was a mission critical device that received the command without authorization, the network could be rendered inoperable.

## Accounting

Accounting will be necessary for a multitude of activity to improve communication paths and actions within an OpenC2 environment. Accounting will provide the ability to measure resources a user or system component (e.g., Orchestrator) consumes during access. This could include the amount of system time or amount of messages has sent or received during a communication session. Accounting is carried out by logging of session statistics and usage information and is used for authorization control, trend analysis, resource utilization, performance, and capacity planning. Overall all of these are important data captures to improve the configuration and deployment of OpenC2 components and a verification that intended operations are working as intended.

## Auditing

Audit trails are necessary in machine-to-machine communications. Audit trails maintain a record of system activity by system or application processes and by user activity. In conjunction with appropriate tools and procedures, audit trails can provide a means to help accomplish several security-related objectives, including individual accountability, reconstruction of events, intrusion detection, and problem identification. In audit terms, such activities are often called events, and auditing OpenC2 functions could be called event logging. Typical events include:

* Authentication exchange between each component (orchestrator, and actuator)
* Message generated, message sent, message received, message action taken
* Any type of success or failure of any OpenC2 exchange.
* (and any result of an OpenC2 C&C – e.g. an application changes configuration)

Actions and the following results that are the direct result of OpenC2 should be recorded and analyzed for security areas such as forensics, secure implementation, security architecture of impact changes within the environment, and completion of such tasks.  This type of auditing provides the essential ingredients for early detection of actions which violate security policy. Currently there are no enforcements within the OpenC2 framework that would directly accomplish such actions.

## Non-Repudiation

Nonrepudiation is the assurance that the sender of information is provided with proof of delivery and the recipient is provided with proof of the sender’s identity, so neither can later deny having processed the information.

A repudiation attack happens when an application or system does not adopt controls to properly track and log users' actions, thus permitting malicious manipulation or forging the identification of new actions. This attack can be used to change the authoring information of actions executed by a malicious user in order to log wrong data to log files. Its usage can be extended to general data manipulation in the name of others, in a similar manner as spoofing mail messages. If this attack takes place, the data stored on log files can be considered invalid or misleading.

Managerial control of enforcing confidentiality throughout an OpenC2 deployment architecture along with the implementation control considerations of adding encryption could be possible to enhance the operational effectiveness of OpenC2 usage and provide a level of technical assurance of message authenticity.

Modifications to the OpenC2 language structure in support of complex tasks such as encryption should be kept to a minimum to avoid re-inventing already established standards and taxing devices which might not be capable of the required processing overhead. This is an area where the deployed environment can best leverage existing and compatible capabilities to perform such services.

## Integrity

Integrity is the property whereby an entity has not been modified in an unauthorized manner. Without message integrity there are several security risks that are prevalent and currently used by attackers, applicable to OpenC2, which include:

* Man in the Middle – ability to intercept the messages while in transit and possibly use the information gathered for reconnaissance, and message tampering
* Forgery of the message – changing the content of the message so a different action is received
* Replay attack - Even a valid message may present a danger if it is utilized in a "replay attack". i.e. it is sent multiple times to the server to make it repeat the requested operation. This may be achieved by capturing an entire message, even if it is sufficiently protected against tampering, since it is the message itself that is used for attack now. Usual means to protect against replayed messages is either using unique identifiers (nonce) on messages and keeping track of processed ones, or using a relatively short validity time window.

## Confidentiality

Confidentiality is the property that information is not disclosed to system entities (users, processes, devices) unless they have been authorized to access the information. Confidentiality protections may apply to the entire message being processed, or only to certain parts of it.

Applying encryption requires conducting an extensive setup work, since the communicating parties now have to be aware of which keys they can trust, deal with certificate and key validation, and know which keys should be used for communication.

Modifications to the OpenC2 language structure in support of complex tasks such as encryption should be kept to a minimum to avoid re-inventing already established standards and taxing devices which might not be capable of the required processing overhead. This is an area where the deployed environment can best leverage existing and compatible capabilities to perform such services

## Encryption

Encryption is the conversion of electronic data into another form, called [cipher text](http://searchcio-midmarket.techtarget.com/definition/ciphertext), which cannot be easily understood by anyone except authorized parties.

The primary purpose of encryption is to protect the confidentiality of digital data stored on computer systems and/or transmitted via the network. Modern encryption algorithms play a vital role in the security assurance of IT systems and communications as they can provide not only confidentiality, but also the following key elements of security:

* Authentication: the origin of the message can be verified
* Integrity: proof that the contents of a message have not been changed since it was sent
* Non-repudiation: the sender of the message cannot deny sending the message

Applicable to OpenC2, the syntax passes content data as a plaintext message and encryption is the responsibility of the native infrastructure. The content message is encrypted using an encryption algorithm and an encryption key. This process generates cipher text that can only be viewed in its original form if decrypted with the correct key. Decryption is simply the inverse of encryption, following the same steps but reversing the order in which the keys are applied. Today's encryption algorithms are divided into two categories: symmetric and asymmetric.

Symmetric-key ciphers use the same key, or secret, for encrypting and decrypting a message or file. The most widely used symmetric-key cipher is AES which was created to protect government classified information. Symmetric-key encryption is much faster than asymmetric encryption, but the sender must exchange the key used to encrypt the data with the recipient before he or she can decrypt it. This requirement to securely distribute and manage large numbers of keys means most cryptographic processes use a symmetric algorithm to efficiently encrypt data, but use an asymmetric algorithm to exchange the secret key.

Asymmetric cryptography, also known as public-key cryptography, uses two different but mathematically linked keys, one [public](http://searchsecurity.techtarget.com/definition/public-key) and one [private](http://searchsecurity.techtarget.com/definition/private-key). The public key can be shared with everyone, whereas the private key must be kept secret. RSA is the most widely used asymmetric algorithm, partly because both the public and the private keys can encrypt a message; the opposite key from the one used to encrypt a message is used to decrypt it. This attribute provides a method of assuring not only confidentiality, but also the integrity, authenticity and non-reputability of electronic communications and data at rest through the use of digital signatures.

Either category could apply to an implementation and use case with OpenC2; however, architecture considerations and maintenance of key pairs, speed, and distribution would have to be thoroughly considered.

Multicast service model would benefit OpenC2 deployments as it is focused with a specific group and a specific set of users. However it does not provide group management which includes receiver authorization or transmission authorization. None of the IA tenants are addressed in current IP multicast service and architecture, which are authentication, authorization, encryption and data integrity. Source authentication could be achieved using IPSec services but not receiver authentication. Also lack of any access control in multicast may lead to serious problems and a number of possible attacks. One example is just flooding attacks. Another is a high degree of packet loss and this will affect OpenC2 messaging. Congestion, session collision arise since there are cases that sessions can use the same multicast address, and data could be interleaved. If IPSec and other secure tunneling services are added, there will be issues with key sharing (group key) and authorization which all have to be planned in advance. The problem with the simple protocol is that it does not scale well as the size of multicast group increases and membership changes become more frequent. The communication cost of updating group keys is linear in the group size, which can be quite expensive (resources) for most applications and systems.

Point to Point architectures in conjunction with OpenC2 could provide a great balance of both message and session based network deployment. Unlike a group model, Point to Point will be component based and key distribution and protection will be extremely important. Authentication and Authorization would be highly recommended to be deployed at both endpoints – orchestrators and actuators. Very strong key management needs to be deployed. If an adversary is able to gain access to the decryption key, the solution will be rendered vulnerable, even useless. Any device that is considered out of scope must not have access to the keys used to protect OpenC2 content and its distribution.

For any cipher, the most basic method of attack is brute force; trying each key until the right one is found. The length of the key determines the number of possible keys, and hence the feasibility of this type of attack. Encryption strength is directly tied to key size, but as the key size increases so too do the resources required to perform the computation.

Alternative methods of breaking a cipher include side-channel attacks, which don't attack the actual cipher but its implementation. An error in system design or execution can allow such attacks to succeed.

## Cryptographic Hash Functions

A cryptographic hash function plays a somewhat different role than other cryptographic algorithms. Hash functions are widely used in many aspects of security, such as digital signatures and data integrity checks. They take an electronic file, message or block of data and generate a short digital fingerprint of the content called a message digest or hash value. The key properties of a secure cryptographic hash function are:

* Output length is small compared to input
* Computation is fast and efficient for any input
* Any change to input affects lots of output bits
* One-way-value – the input cannot be determined from the output
* Strong collision resistance – two different inputs can’t create the same output

All of these properties would benefit any OpenC2 implementation with limited resources and lack of support for providing full encryption of the messages. Hash functions could be used as an alternative to encryption because the environment and requirements limit the reality of using encryption.

The ciphers in hash functions are built for hashing: they use large keys and blocks, can efficiently change keys every block and have been designed and vetted for resistance to related-key attacks. General-purpose ciphers used for encryption tend to have different design goals. For example, the symmetric-key block cipher AES could also be used for generating hash values, but its key and block sizes make it nontrivial and inefficient.

## Verb Whitelisting

Even though devices have support for a variety of methods of configuration management and operational change configurations, the ability to use such methods at any given time and at any given moment would be considered hazardous to network and device operations. It would be an operational method to form a whitelist of device operational commands and have a technical consideration of allowing certain commands to be used by certain operators and at different times of execution (off-hours, maintenance windows). An implementation model could be developed with the commands and OpenC2 syntax to make best use of creating a model of operating the devices autonomously.

For example, many devices such as a firewall are capable to accept commands such as DELETE, MODIFY, ACCEPT, DENY, START, STOP, RESTART; however, even if such a device is capable of accepting the commands and act upon an action, not all actions should take place especially during production operating hours. Actuators and Target devices will blindly act upon actions. In essence OpenC2 command hierarchy is too much of an enabling system without certain capability checks in place.

With respect to controlling the environment and keeping commands in sync with allowed permissions and commands, another consideration for architecture implementation is to map a controlled list of OpenC2 commands that are authorized and not authorized to various actuators. There should also be an automated capability for an over-ride of a command even if it was issued manually or out of band. A concern is that even an arbitrary message that may have been sent in-advertently due to some other cause may bring devices to a halt or lead to a whole system outage. Orchestrators and actuators that are responsible for sending configuration changes should have a validated list of authorized commands and have logic within the interpretation of messages to know if certain commands are not authorized to be executed. A logical mapping of device capabilities and production concerns should be considered when creating the authorized mapping of commands at any of the authorized endpoints – either orchestrator or actuator. A tightly coupled understanding of the devices to be managed to the flexibility of automated C&C to the point of integrated, managed and enforced will lead to fewer problems during production and likely lead to a more efficient authorization model when implementing OpenC2.

# Architecture & Implementation Considerations

## Peer-to-Peer

With-in a peer-to-peer architecture and without basic verifications of commands there are a few issues that could arise. Misinterpretation of commands may be the most common threat. The threat appears if the language of OpenC2, say a communications protocol in the simplest case, is not sufficiently verified. If the communication language is used which is more complicated than the messages of a fixed format, the semantic problems of understanding appear. The language implementation in a peer-to-peer architecture may need to be kept as simple as possible.

Misunderstanding of a situation could lead to wrong decision planning and execution stages. It is a threat that may be more prevalent based on the environment that OpenC2 is implemented into is sometimes unknown to the designer (the author of the OpenC2 implementation). The environment in any cyber space network is complex with many different operating systems, software platforms, protocols etc. The best recommendation would be to restrict the environment as much as possible by permitting the OpenC2 components to operate only on known platforms. Situational awareness always needs to be improved and verified, or at least carefully tested with respect to safety of the OpenC2 components behaviors.

Peer-to-peer architectures can present many atypical issues not found on other domain-based systems. Secure connections have different key management interface challenges and basic issue of trust must be addressed differently.

### Full Mesh PVC

Advantages

* All actuators would be connected to each other – no single point of failure
* High degree of reliability due to the multiple paths for data that are provided by a large number of redundant links in between all endpoints
* Fully connected network does not need to use packet switching or broadcasting
* Multiplexing or trucking in a best-effort network – data could burst up to the full line rate of the trunk
* Encryption throughout the network – provides all tenants of security topics above.

Disadvantages

* High percentages of redundancy overhead in many of the network connections
* Setup and maintenance of this topology is very difficult. Administration of the network becomes more complex and difficult as it expands
* Latency will be a factor to consider passing messages in real time.

Without providing any later of encryption through a mesh network, the possibility of attack and propagation of the actor within the network is immense. Any endpoints that get compromised would allow the attacker to take full control over the mesh network and start passing messages that could cause outages or device malfunctions. The attacker would have the ability to re-direct actions and possible outcomes based on how they choose to divert both traffic and operations of the connected devices.

Full Mesh PVC is an architecture that has a few IA concerns in regards to securing the infrastructure (core backbone, routers, network gear). As the PVC expands, Full Mesh suffers from the n-squared problem. A fully meshed topology requires [n(n-1)]/2 PVC’s where n is the number of routers being connected to the network. A fully meshed network of 5 routers requires [5(5-1)]/2 which would be 10 PVC’s. Each component will have to go through full security lifecycle review – and shortcuts may happen in the process that could leave the PVC’s vulnerable to attack at a component or even protocol level.

Insider threat becomes another topic to consider around Full Mesh PVC. The network will require continuous supervision because of the redundancy present in the network. Skilled network administrators may also take advantage of the network for their own purposes and the larger the network grows, the harder it will be to detect rogue actions from an insider.

Latency may also be a huge factor to consider, especially for defense in real time. Latency becomes an issue in smaller lower power, wide-area networks because it doesn’t have the processing capability to handle the messages. This is something to highly consider when planning the protocol being used and the latency the applications will require. Think in comparison to a Wi-Fi Mesh – messages are translated much quicker than a ZigBee mesh.

Concerns regarding adding more connection points would require further infrastructure maintenance, and security overview on the underlining deployment network. Growth must follow the three layer routing model in order for meeting high availability, acceptable performance, and low-cost requirements. The mesh could also suffer from over saturation – devices receiving OpenC2 messages that are non-relevant yet the device will have to use system resources to make decisions on what it should process. As the network expands other issues arise such as encryption key management with the expansion of the network, reliability and able to handle scale growth without sacrificing speed and efficiency passing messages.

During the planning stage of development there are a few variables to consider to alleviate communication issues as the network may mature or grow;

Maximum rate requirements – understand the bandwidth and number of PVC’s required to meet service needs

Committed Information Rate – CIR is the bandwidth provided to each PVC on physical links. Failure to properly calculate appropriate CIR level results in poor performance, and failure to meet service levels.

FECN/BECN Congestion Protocol – enable this to protect the network resources from over utilization

Virtual sub-interface and multiprotocol management – consider traffic management with high rate of speed and volume of OpenC2 messaging.

Partial Mesh / Star Deployment (Intra Domain)

* Orchestrators and Actuators may pass just messages instead of a fully configured network with established concurrent connections
* Orchestrators would be the management framework for all messaging, meaning white listing and permission mapping will be important to deploy
* Management infrastructure in place can be leveraged as the operation centers (management consoles) where Orchestrators may be deployed to pass messages to all endpoints either through a messaging framework or session based connections.
* Authentication and Authorization at each component will need to be bi-directional to safeguard message transport and verification for enhanced auditing

Intra Domain network routing security rests on confidentiality, integrity and availability. Three aspects of threat possibilities regarding this type of architecture deployment are – acquiring routing information, denial of service, and even routing path manipulation. Management platforms that can act as brokers to send information to orchestrators in this model should have local security configurations in place to protect the transmission of C&C instructions to the appropriate OpenC2 components. The basics is how the OpenC2 environment routing of messages will need the utmost protection – which would require apply the security controls such as authentication, authorization, and encryption.

## Publish – Subscribe (Pub-Sub or Multi-tenancy)

Actuators and Orchestrators in cyber operations and cyber defense can be used most efficiently in multi-tenant formations. Botnets could be an example, if bots are developed as actuators. However, the command and control in botnets has remained quite simplistic. One can expect that multi-tenant systems will become the main form of deployments in cyber operations. In this case, the botnets will negotiate between themselves and will cooperatively create a complex behavior for achieving the general goals stated by a commander known as the orchestrator. As a consequence, the strict control of behavior of every single bot will be weaker. Also, it will be more difficult to foresee all possible cases for decision making. Practically, it will be impossible to verify the outcome of multi-tenant behavior for all situations. It is possible that backdoors and forced destruction will have to be built into the system. Multicast control messages may be needed for emergency cases of the actuator control. Another option could be self-destruction of actuators if loss of contact occurs, i.e. if for some time no command and control messages are received. A specific threat of multi-tenant systems is the formation of unwanted coalitions by actuators. This can happen if the actuators achieve too much autonomy in decision making. Communication between the actuators will be only partially observable to human controllers in this case. This will require very careful selection of constraints on the behavior of actuators. However, there will never be an absolute guarantee of avoiding a misunderstanding of a situation by a team of actuators. Also, the danger remains that a collection of actuators may behave unintentionally in a harmful way.

There are a number of advantages to the pub-sub model:

* Accommodates dynamic network topologies
* Rapid implementation
* Potential for very large scale with peer-to-peer
* Raw speed

The pub-sub paradigm also has several dis-advantages:

* Scalability limited with central orchestrator
* Use of IP multicast traffic requires specialized configurations
* Message delivery not natively guaranteed
* Focus on throughput at expense of availability and integrity.

Strict Type Enforcement

Strict type enforcement and the related input validation is essential to avoid one of the most commonly exploited vulnerabilities. Within an OpenC2 implementation, it must be possible to receive any arbitrary string of bytes and determine if it is a valid OpenC2 command. As long as all the elements in the message are validated against a controlled set of vocabulary, using optional modifiers would not be a problem. The absence of input validation may present opportunities for unintended code to execute on critical network defense systems.

## Integration with Configuration Management

Configuration Management (CM) is the application of sound programing practices to establish and maintain consistency of a product’s or system’s attributes with its requirements and evolving technical baseline over its life. Considering configuration management during OpenC2 development and deployment will allow for a better organized OpenC2 deployment and help with the overall system ecosystem. Having configuration management will allow a developer/engineer to search the configuration management database for specific CI’s that someone wants to designate as the different components to be deployed, such as orchestrators, or actuators, and targets for a release. Configuration management owners would be notified when configuration integrations are changed, for example, if server is updated with a new release of a business application, owners of both the server CI, which would include OpenC2, and the business application CI would receive notices.

The CM process is not administrative-only; when coupled with robust auditing, CM becomes a key component used for the enrichment of operations monitoring data. As the threat landscape evolves, a well-documented CM process will ensure the audit trail does not become obsolete. While CM was once typically only found in Federal Enterprise Architectures, the proliferation of both regulations and security best practices has encouraged the wide-spread adoption of CM into systems of all scale.

## Out of Band Management (OOBM)

The continued exponential growth and increased complexity of IT infrastructure — M2M, Cloud Computing, Internet of Things — drives the need for smarter real-time decision-making, deeper data analysis, and more robust data storage at the network edge. The demand for always-on connectivity and resilience in the face of network failure events also continues to rise. Rapid identification and remediation of connectivity issues between dispersed Internet-connected devices and remote infrastructure is critical. Out-of-Band management involves the use of a dedicated channel for managing network devices. This allows the network operator to establish trust boundaries in accessing the management function to apply it to network resources. It also can be used to ensure management connectivity (including the ability to determine the status of any network component) independent of the status of other in-band network components. OOBM is a common best practice with renewed focus based on the threat landscape listed earlier in this document. C2 systems are prime objectives for bad actors and OOBM offers another layer in the defense-in-depth model.

Company security policies, generally, will restrict or prohibit a dedicated connection to the OOBM ports through access control lists or other access methods. In practice though, the requirements of system availability often lead to unsecure access points. Those responsible for keeping devices and networks running may leave back-door access in place on these OOMB ports so that disastrous failures can rapidly fixed. To address these types of issues, a security plan should be implemented and enforced, focusing in these areas, which will enhance the entire security architecture of the enterprise:

* Definitions of vulnerabilities and risks of out of band access for OpenC2
* Review security architecture for mitigating those risks
* Proper balance between security and the need for timely out-of-band-access during critical events
* Systems of processes, equipment and technologies that provide, wherever required for OpenC2, integrity, confidentiality, and/or non-repudiation for out of band access.

## Domain Federation

A federation trust model is often found in partner organizations with explicit trust and a shared security and/or accreditation boundary. Federated identity management facilitates the trust of user and/or object identities and related attributes which are stored across multiple systems through the use of common standards. Single sign-on is perhaps the most common form of identity federation. The various models used to facilitate identity management should be accounted for under the broader umbrella of AAA and developers should be cognizant of the emerging trends and standards used to implement identity access and management.

## Certification and Accreditation (C&A) – Regulatory Compliance

Regulated industries such as public companies, health care providers, financial institutions and other organizations that handle PII and adhere to PCI will have to follow strict policy enforcements in validating security requirements for systems, applications, system software, and other technologies before they are deployed into a production environment.

This process is designed to ensure compliance with specifications, regulations, standards and objectives identified during each phase of a System Development Life Cycle (SDLC). With the previous topics discussed around OpenC2, when it comes to an implementation, the overall system architecture has to address fundamental security features (auditing, CIA, AAA, encryption) in order to meet the requirements listed in the various security requirements to pass compliance standards.

Evolving politics and security incidents born from poor implementations have produced a patchwork of diverse regulations and frameworks. Implementations of OpenC2 should integrate qualified ISSE processes to achieve both solid practical security and regulatory compliance. It is often said that compliance does not equal security, but strong security practices can transform the practice of risk management and exceed many mandates.