

Chapter 2

Bio-inspired Evolutionary Sensory System for Cyber-Physical System Security

Mohamed Azab and Mohamed Eltoweissy

Abstract Cyber-Physical Systems (CPS) promise advances towards smarter infrastructure systems and services, significantly enhancing their reliability, performance and safety. Current CPS Monitoring, Analysis, Sharing, and Control (MASC) technologies offer disparate and largely inadequate services for the realization of effective and efficient CPS security. Most current technologies did not consider that cyber and physical convergence would need a new paradigm that treats cyber and physical components seamlessly and pervasively. Further, information sharing was severely curtailed by enforcing parameter defense to preserve the privacy of the system to be secured, the Target-of-Security system (ToS). These limitations negatively impact the quality, reliability, survivability, and promptness of security services. In this chapter, we discuss the current challenges to CPS security, survey relevant solutions, and present a novel system, CyPhyMASC, towards realizing pervasive MASC for enhanced CPS security. CyPhyMASC is a bio-inspired intrinsically-resilient, situation-aware system utilized by Security Service Providers (SSPs) to provision MASC services to ToSs comprising numerous heterogeneous CPS components. CyPhyMASC is unique in that it acts as a generic middle layer between the SSPs and the ToSs creating a uniform interface that isolates ToS scale and heterogeneity aspects from control and management aspects. Such isolation and uniform representation facilitate interoperable security services. CyPhyMASC intelligently mixes and matches heterogeneous tools and control logic from various sources towards dynamic security missions. CyPhyMASC is also elastic where situation-driven MASC solutions can be dispatched using dynamic sets of sensor and effector software capsules

M. Azab (✉)

The City of Scientific Research and Technological Applications, Alexandria, Egypt
e-mail: Mohamed.m.azab@gmail.com

M. Eltoweissy

Bradley Department of Electrical and Computer Engineering, Virginia Tech,
Blacksburg, Virginia, USA
e-mail: toweissy@vt.edu

circulating through the ToS rather than using pre-deployed MASC components. Such approach provides evolvable, pervasive and scalable MASC services.

2.1 Introduction

Major physical infrastructure systems such as the water distribution systems and the electric power grid are large-scale complex systems that are expected to be highly reliable and trustworthy. Modern versions of these infrastructure systems go far beyond simple measures to integrate intelligence and automated control into the system through tightly coordinated and integrated cyber components constructing large-scale Cyber-Physical Systems (CPS).

CPS safety and security are prerequisites to assure stability, reliability, and survivability of such mission-critical systems. Security services for CPS are highly dependent on the promptness and accuracy of the Monitoring and Analysis (M&A) mechanisms employed. Traditional M&A approaches do not treat sensing and effecting for cyber components and physical components seamlessly. The current M&A mechanisms were designed based on a set of assumptions that unintentionally neglect the real-time interaction and the tight coupling between these converging components. The assumption was that physical components were protected by isolation and parameter defense while real-time response was not a primary factor for cyber components. Further, they assumed that there is no need to employ privacy preservation techniques as the Target of Security (ToS) privacy is implicitly protected by cyber and physical parameter defense. Additionally, they assumed that resource heterogeneity and scale could still be resolved by a distributed set of heterogeneous, pre-deployed platform-dependent defense tools with fixed resource profiles.

Research works in [1, 2] as well as our own have disputed the validity and correctness of such assumptions as they lead to drastic problems and limitations negatively impacting the quality and promptness of the CPS security service provisioning. Current CPS Security Service Providers (CPS-SSPs) fail to provision trustworthy robust and reliable monitoring and evaluation of the ToS components due to the use of scattered, uncoordinated, uncooperative, unaware, isolated and heterogeneous monitoring tools, and reporting mechanisms. Such limitations increase the use of resources due to redundancy, increase the risk of conflicts, and failures due to limited awareness and coordination, lower the defense quality due to the poor, and boundary limited feedback, increase the latency in security provisioning and in detecting attacks giving the attacker the advantage to spread the attacks through multiple networks, the tool heterogeneity and uncooperative nature massively complicates automating its management, the static nature of such tools complicates attempts to autonomously adapting to changes in the surroundings.

The problem is not only at the monitoring and evaluation phase of the defense provision process, but also at the analysis and investigation phase where the collected feedback gets analyzed searching for attack signs and indications. When the network scale grows exponentially, it becomes almost impossible to analyze the feedback from

all the sensors, compile that feedback together, and extract valuable information from it efficiently, and promptly. Additionally, due to the isolation between the monitoring technology and the analysis technology, and the lack of computational power needed to expand the sources of feedback, the current analysis technology does not have enough data to be fully aware of what is globally happening in the network under investigation. Having most of the conventional analysis mechanisms designed to share the same host/host-network for the sake of protecting their privacy, lead to serious limitations. The limited investigation search space, being easy to be targeted by attackers, Can be used to cause a DoS attack, and cannot cooperate in analyzing feedback or share information with out-of-perimeter nodes are examples of such limitations.

In addition to the presented set of limitations in the field of monitoring and evaluation, and analysis of feedback, the control phase has another set of serious limitations too. Control phase represents the stage where the defense system takes actions regarding detected threats face a serious set of limitations.

Control related limitations are mainly the result of lack of cooperation and awareness that limit the defense tools capability to resolve or even contain persistent fast spreading attacks. For example, it is too hard for such uncoordinated, scattered tools to marshal and coordinate task force to hunt down the attacks spreading all over the network or a set of interconnected networks. The reason behind such complexity is the difficulty in autonomously and promptly controls and coordinates both the SSP, and the ToS tools and equipment to block attack access given the current centralized management technology. Further, without appropriate global control, and situational awareness it is too hard to block the source of dynamic fast-changing remote attacks. Such limitations can be utilized to cause DoS attack by keeping the SSP busy treating infected files and strike more and more files. This chapter presents an Evolutionary Sensory System (CyPhyMASC), designed to induce a new paradigm for security service provisioning that intrinsically and comprehensively addresses the aforementioned challenges facing conventional techniques. CyPhyMASC is a biologically-inspired, intrinsically-resilient, intelligent, situation-aware sense and response system to effect biological-immune-system-like security provisioning. We address ToS heterogeneity and scale by enabling dynamic security resource elasticity.

CyPhyMASC is designed to separate the main security provisioning concerns; the tool logic, management and control, delivery mechanism, and physical resources. CyPhyMASC Utilize our smart, biologically inspired, resilient, adaptable, self and situational aware, elastic, and autonomously managed building blocks (the Cell) to construct mobile, dynamic, and runtime-reprogrammable defense carriers to pervasively distribute accurate, trustworthy, and prompt security services. CyPhyMASC acts as a middle layer between the SSPs and the ToS creating a uniform security interface that hides ToSs scale and heterogeneity concerns from control and management.

This uniform representation enables interoperable and cooperative defense. Further, such isolation maintains security provisioning survivability in case of ToS failure and DoS attacks. Additionally, CyPhyMASC autonomously and dynamically profile ToS hosts and direct security services based on the host dynamic behavior and

attachments. CyPhyMASC works on a biologically-inspired, intrinsically-resilient, adaptable foundation based on the Cell Oriented Architecture (COA). The COA provides intrinsic dynamic, distributed, resilient resource management and allocation needed to support CyPhyMASC pervasive M&A.

CyPhyMASC manages a vast number of elastic and intelligent containers (Cells) to host/abstract cyber/physical sensing and effecting tools. The COA Cell (the containers) can be described as a micro virtual machine capable of hosting the code logic of a single task. CyPhyMASC mimics the human blood stream circulation effect by utilizing its adaptable infrastructure to circulate these context-driven, functionally customizable sensor and effector Cells into the ToS body to pervasively monitor, analyze and control the TOS components. CyPhyMASC sensors and effectors are used to execute defense missions provisioned by SSP. A defense mission is a mixture of sensing and effecting tasks involving information gathering, partial analysis, control, and manipulation of the ToS elements.

CyPhyMASC can alternate/mix different defense/control missions from different SSPs to provision security services to the same ToS in a process called vaccination. The vaccination process involves sharing security experience and tools between SSPs in terms of abstract missions, and sensing and effecting packages. Vaccines are autonomously checked for privacy violations and maliciousness before utilization or storage. It is exactly like in biological systems where antibodies can be extracted from one immune body to another to create a healthy up-to-date defense community.

CyPhyMASCs main contributions presented in this chapter can be outlined as follows: Enable pervasive autonomously managed monitoring and analysis; Uniform security service provisioning for heterogeneously-composed multi-enclave CPS systems; Enable trustworthy, interoperable multi organization cooperative, dynamic, autonomous security; and Facilitate early failure/attack detection and resolution.

2.2 Attack Scenario

To further motivate our research and to illustrate the effectiveness of CyPhyMASC in achieving its mission we utilize the following working scenario depicting a hypothetical CPS attack named the BlackWidow attack [3]. The players are two competing companies ABC which is the victim, and XYZ that recruited a resourceful attacker to attack ABC.

The BlackWidow (BW) is designed to split into a set of code parts and spread in different directions and locations to decrease the probability of detection. The distribution of parts and the interconnection between the parts in different hosts weave a large web. This web is bi-directionally traversed to send any harvested data from the attacked target and to update the malware with new tools and missions. The BW is designed to be as generic as possible; it is not oriented to any specific application. By constructing the BW web the attacker can start to task the BW towards its designated mission based on the attackers target. The tasks might be remotely assigned through the Internet or preprogrammed in Internet-inaccessible locations (Fig. 2.1).



Fig. 2.1 Commercial security example

The attack is designed to be stealthy by hiding from the security system sensors searching for attack signatures. The attack will target an intermediate host machine that will contain the BW and command and control channel communications. In order to do so, the BW is designed to not harm the host or change any of its settings that might raise the Anti-Malware (AM) alerts. The malware will use minimal resources and will work in a very slow fashion not to alert the network security systems by its existence. The BW uses stolen digital certificates to authenticate its existence in the host machine in the form of drivers. The only way to detect this malware is through deep analysis of the logs of all the communicating nodes, which is computationally very costly to the current systems that share the same host machines. This sharing guarantees host privacy preservation, but limits the capabilities and the awareness of the security tools. The attacker utilizes these limitations to his advantage as illustrated later.

The malware is intended to be targeted, but due to the intentionally random deployment method, the code works in two modes as follows: (1) Benign mode where the malware infects other machines that do not belong to the target space. The machines might be used later in case of target change, or as a base for future attacks; and (2) Malicious mode, where the BW works only on the target host systems. The attacker feedback can determine the mode. The default will be benign unless the attacker changes that or predetermined targets have been programmed.

Attacker assumptions

1. The security and management system shares the same network or host with the target of attack/security system. [Note: security system might be exposed to attack by compromising the ToS. Additionally, Stolen passwords can simply be used to modify rules of IDS, routers, switches, firewalls, proxies, etc.]
2. The attack target security system, or major parts of it, uses COTS and signature based security products
3. The system is computationally incapable of being fully situation aware of all its components in a massive-scale network, in real-time.

3. Security systems are not resilient against attacks, and have weak recovery mechanisms. [Note: most of them assume that they will not be the target of an attack as long as they were able to secure their ToS.]
4. Cyber security is oblivious of and is not coordinated with physical security to protect the target cyber-physical system. Human intervention is need to facilitate such coordination. [Note: the attack can make them conflict with each other to bypass both of them]

Attack procedure (on Air-gapped Target). The attacker uses phishing attack that targets users emails and social network personal pages. The attacker uses social networks as a source of information to generate more convincing phishing emails. These emails will be directed from one of the closely related contacts to the victim. The attacker selects a group of employees working in different branches of ABC. These branches are distributed in various geographical locations, and the victims that will be the malware couriers have no direct relation with each other. The BW is programmed to search the user network for connected computers then it starts using one of the zero days exploits to clone itself into these computers. The attack victims will receive parts of the malware. Each of these parts will contain a fraction of the designated mission and a simple communication module. The communications module will be used to open a direct channel with the attacker and to search and establish communication with other parts. Directions to other parts locations might be sent by the attacker to minimize the search time. The attacker uses malware fractions to construct logical executable entities in the form of mobile software agents targeting different objectives. The first objective will be to search and infiltrate the network for data stores. The malware will sniff network traffic searching for predetermined signatures for such locations. The second objective will be to attack such data stores using the zero day exploits and the stolen certificates to locate targeted industrial secrets, and any available access keys to the protected area behind the air gap. The malware will frequently update the attacker of its findings based on a predetermined update methodology. After successful reception of this data, the attacker will use it to generate legitimate keys to access the air gap. The attacker will use the malware to locate the workstations controlling the surveillance cameras. The malware will record periodic video feeds to be sent to the attacker. These videos with the help of the attacker generated access keys will guide a recruited insider into infecting the air gap with the BW. The malware controlling the cameras will make sure that this process wont be recorded to protect the recruited insider. The air gap malware is programmed to increase the operational hours of certain machines that use specific raw materials manufactured by XYZ to increase XYZ profits. The malware can easily identify such machines by searching a predetermined fixed identifier that must be added to all the programming files targeting such machines. Further, the attacker will use the stolen secrets and designs to equip the malware with the needed logic to randomly manipulate the operational motors frequency in the production machines to induce random defects in the output products to lower its quality. XYZ shall benefit from ABCs loss due to its low quality products. Additionally XYZ will maliciously gain both financially and more control over ABCs production lines by,

for example, carefully adjusting the amount of consumed and supplied raw materials. We will revisit this scenario to illustrate how CyPhyMASC secures the ToS against such attack. Additionally we illustrate how CyPhyMASC invalidates the BW design invariants and attacker assumptions.

2.3 CPS Attack Detection and Resolution Techniques

2.3.1 Malware Detection

A malware is malicious software designed to infiltrate or damage a Cyber system or Cyber Physical System (CPS) without the owners informed consent [4]. There are many malware types with different shapes and entry points. Most of these software objects share similar purposes while they are expected to behave differently at time of infection. Viruses, worms, botnets, wabbits, Trojan-horses, exploits backdoors, spyware scumware, stealware, parasiteware, adware, rootkits, blended threats, evolving threats, keyloggers, hoaxes are examples of the different malware types. Figure lists the different types of attacks and the usability ration of each one of them [33].

Each malware group has its own way of being undetected. Modern malware detection tools utilize multiple detection mechanisms to be able to detect multiple malware categories as presented in Fig. 2.2. Malware especially viruses are either memory resident or non-memory resident. Non memory resident are simple attacks that can easily be detected an entry point with a cleaver detection tool.

The memory resident attacks are more complex and efficient that stays in memory and hides their presence from detection tools. These attacks are either fast infectious aiming to infect as much files as possible locally within the infected host or remotely through the host network, and network shares. The second category of memory resident attacks is the slow infectors. Slow infectors are the most dangerous type of malware as it uses stealth and encryption techniques to stay undetected as long as it can. They are powerful attacks that can be a combination of multiple processes working together towards certain objective. Malware detectors use signature based detection techniques to detect known attacks. Signature based detection became very efficient way of detecting known threats [5]. Finding a specific signature in one of the executable codes can accurately identify any enclosed threats within such code. Attack signatures are frequently updated and stored on the local anti-malware database. Unfortunately this technique is inefficient if the attack has a malformed signature either by the programmer or by a mutation engine.

Heuristic techniques are one the most efficient ways to detect such mutated attacks. Heuristic and metaheuristic techniques are used to spot unknown or known attacks with polymorphic behavior. By definition, heuristic technique is an informal technique to solve problems efficiently and in a way close to the optimal path [5]. Heuristic techniques are commonly used to rapidly reach a solution that is somehow close to the best possible solution. The metaheuristic technique is a heuristic method for



Fig. 2.2 Classification of malware related attacks

solving many of the computational problems by combining user-given black-box procedures in a hopefully efficient way [5].

Most of the modern malware detection techniques that use metaheuristics to detect attacks utilize a set of isolated tools utilizing different techniques hoping in detecting one of the attacks that there is no specific way to detect it. Most of these tools utilize one of the following mechanisms, Pattern matching, automatic learning, environment emulation, neural networks, data mining, bytes networks, and hidden markov models. There are other metaheuristics techniques but most of them are built based on one or more of the aforementioned mechanisms.

The main concept of heuristic based detection techniques is to detect attacks without knowing too much about its internal structure. Heuristic techniques mainly focus on examining the behavior and the characteristics of the executing software to anticipate whether it is acting maliciously or not. The most successful heuristic based detection technique named as The Heuristic Scanning Technique utilizes a mixture of multiple metaheuristic techniques such as pattern matching, automatic learning, and environment emulation.

Heuristic scanning in the common sense uses pattern matching to examine the assembly language instruction execution sequence, and qualifies them by their potential dangerousness. Heuristic scanning usually follows a set of built-in rules with pre-assigned weight on each rule. In case of violation of any of any of the rules the weight of the violated rule is added to the total violated rule by the same program or process. The program is flagged as malicious only if the total sum of added weights exceeds certain threshold. Figure 2.3 illustrated the idea of a single layer classifier with pre-determined threshold. The feedbacks from the different scanners are fed into global summarizing point that follows a certain metaheuristic mechanism as illustrated in Fig. 2.4. The overall result will decide whether to flag the scanned object or not. As the detection techniques gets more cleaver, the modern attacks or malware also emerge to more complicated attacks utilizing more sophisticated stealth techniques. Such techniques give them the advantage of being invisible to traditional scanners. Moreover the use of real-time encryption, and anti-heuristic sequences made them looks totally harmless to traditional malware scanners.

Heuristic scanners that use single metaheuristic mechanism that focuses only on monitoring the execution flow of the instructions of a certain program are deceivable

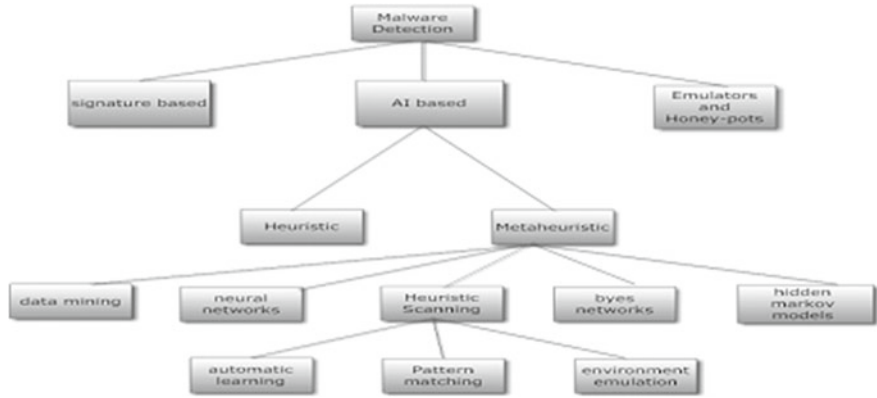


Fig. 2.3 Classification of malware related attacks

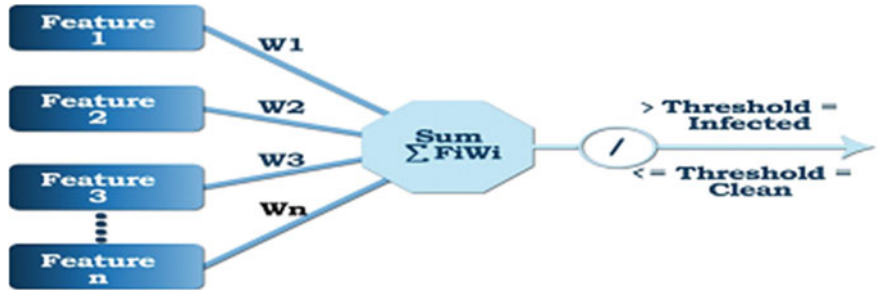


Fig. 2.4 Single layer classifier

by code obfuscation. Code obfuscation occurs by embedding some meaningless instructions within a malicious code. The same technique deceives detectors utilizing heuristic and signature scanning combined together. One of the successful mechanisms to resolve the aforementioned problem is the use of artificial runtime environment emulation. However, it is not a light weight detection mechanism, but it has high success rates in detecting unknown attacks. Environment emulation utilizes the idea of virtual machines; the malware detection tool provides a virtual machine with independent and isolated operating system and allows malware to perform its routines freely within the virtual environment. The execution behavior of the suspicious application is being continuously examined while the malware is not aware. Most of the stealth and anti-heuristic techniques are irrelevant in this case, as the detection tools scan the behavior from outside the box with a clear vision of what is really happening inside.

The main problem facing such technique is the massive resource consumption and the expected delay needed to construct the virtualization environment, and infiltrate the harmful instructions from being executed on the real machine. Another problem that arises with using heuristic methods for detecting malwares is the possibility of

false positives. A false positive event occurs when a benign program gets flagged as malicious by the heuristic scanner. The problem occurs frequently specially with noncommercial programs having suspicious routines through their encryption functionalities.

The use of automatic learning is a good resolution of such problem, where the detector learns from its mistakes. The main issue with this technique is it requires an advanced user. In order to resolve such problem autonomically, detection scanners have to increase their scanning depth, and combine feedback from multiple heuristic mechanisms. Also external consultation is one of the most efficient techniques, where an external resourceful node gets consulted for guidance related to suspicious programs with weights that parley cross the threshold line. The only issue with that solution is the possibility of privacy violation due to sending specifics about the suspicious events.

Recently more complicated attacks were introduced that depends on infecting and controlling multiple hosts creating an automated taskforce targeting multiple objectives. Such attacks usually have dynamic objectives, and construction components. Additionally, they are frequently and autonomically get updated using a dynamic up/down link between the attacker and the malware itself. Detecting such attacks is a very complicated task given the uncooperative nature of the conventional modern detection tools, and the fact that they share the same host, or host network with their ToS.

Sharing the same network or host with the ToS makes them an easy target for attackers to deceive, or destroy [6, 7]. Additionally, the successfulness of the malware detector depends mostly on the fast real-time, and deep analysis of the scanners feedback. Such process, especially when it involved creating a runtime emulated execution environment is a computationally costly process for a tool that shares the ToS resources.

2.3.2 Standalone and Distributed Monitoring and Evaluation Solutions

Defense services for CPS are highly dependent on the promptness and accuracy of the Monitoring and Analysis (M&A) mechanisms employed. Traditional M&A approaches do not treat sensing and effecting for cyber components and physical components seamlessly. The current M&A mechanisms were designed based on a set of assumptions that unintentionally neglect the real-time interaction and the tight coupling between these converging components. The assumption was that physical components were protected by isolation and parameter defense while real-time response was not a primary factor for cyber components. Further, they assumed that there is no need to employ privacy preservation techniques as the Target of Defense (ToS) privacy is implicitly protected by cyber and physical parameter defense. Additionally, they assumed that resource heterogeneity and scale could still be resolved by

a distributed set of heterogeneous, pre-deployed platform-dependent defense tools with fixed resource profiles.

Research works in [1, 2] as well as our own have disputed the validity and correctness of such assumptions as they lead to drastic problems and limitations negatively impacting the quality and promptness of the CPS security service provisioning. Current CPS Defense Service Providers (CPS-SSPs) fail to provision trustworthy robust and reliable monitoring and evaluation of the ToS components due to the use of scattered, uncoordinated, uncooperative, unaware, isolated and heterogeneous monitoring tools, and reporting mechanisms. Such limitations increase the use of resources due to redundancy, increase the risk of conflicts, and failures due to limited awareness and coordination, lower the defense quality due to the poor, and boundary limited feedback, increase the latency in security provisioning and in detecting attacks giving the attacker the advantage to spread the attacks through multiple networks, the tool heterogeneity and uncooperative nature massively complicates automating its management, the static nature of such tools complicates attempts to autonomously adapting to changes in the surroundings.

Research presented in [8–11] attempted to resolve some of the problems resulting from such assumptions using more flexible sensing and control elements. They devised a mobile multi-agent based attack detection system. The presented solutions were situation unaware and offered limited defense-tools pervasiveness and coordination. Generally speaking, provisioning security services while sharing the same host with the ToS exposes the ToS to DoS attacks, and limit the systems scalability and interoperability.

Works in [11, 12] utilized a multidisciplinary approach to intelligently resolve some of the presented limitations. They combined multiple artificial intelligence techniques to build a complex smart attack detection system. Unfortunately, these techniques were bounded by the available technology constraints; they were designed to provision dedicated security service while sharing the ToS host or host network. They were unable to overcome the curse of complex systems dimensionality. With the increase of system complexity and numerousness of input features, the processing time involved with clustering system events might badly affect system, and attack detection timeliness. Time constraints may sometimes force the system to prune less important features (dimensionality reduction) to maintain system timelines. However, the pruning approach is not always possible as it might compromise the detection accuracy.

All the above mentioned approaches were mainly concerned with security service provisioning for cyber components. The work presented in [13, 14] is a hardware based static detection system capable of supporting the requirements of both cyber and physical components. Using hardware based detection and analysis techniques guarantee prompt, and resource efficient response for quickly spreading attacks. A major disadvantage of technology is its limited flexibility, adaptability, interoperability, and maintainability. These systems are designed to work for specific target and cannot seamlessly adapt to match different targets. Multiple attack detection solutions were presented utilizing mixtures of the abovementioned methodologies employing different M&A techniques [15, 16], Unfortunately, none of these systems

where capable of presenting a comprehensive, autonomous, interoperable, globally situational aware and scalable solution that can guarantee adequate security provisioning quality and promptness while maintain the ToS survivability, operability, and privacy. Up to our knowledge CyPhyMASC is the first solution that can provide such features comprehensively and pervasively with low overhead.

2.3.3 CPS Related Control Solutions

In addition to the limitations presented in the previous two sections, in regards to monitoring and evaluation, and analysis of feedback, the control phase; where the defense system takes actions regarding detected threats face a serious set of limitations [7]. The limitations are mainly due to the lack of cooperation and awareness that limit the defense tools capability to resolve or even contain persistent fast spreading attacks.

For example, it is too hard for such uncoordinated, scattered tools to marshal and coordinate task force to hunt down the attacks spreading all over the network or a set of interconnected networks as it is hard to control the SSP, and the ToS tools and equipment to block attack access to the shared network. Further, without appropriate global control, and situational awareness too hard to block the source of dynamic remote attacks. Such limitations can be utilized to cause DoS attack by keeping the SSP busy treating infected files and strike more and more files.

Research work has been focusing on presenting a resolution for some of the control problems in CPS environments. Researchers in [17] presented what is called Autonomous Multi-agent Cooperative Problem Solving (TEAM-CPS), and successfully applied it on one of the critical CPS, the public telephone networks. They used multi intelligent agents that were designed to work together to provide distributed control for such system. Unfortunately, the system was not scalable enough to suit large scale systems. The limitations against this approach and other agent passed approaches like the work presented in [18, 19] is the high resource consumption nature of the agents, and the fact that they are designed to share the host resources. These limitations limit the approach capability to scale.

From another perspective, the use of intelligent agents lacks the support of the physical part of the network. The used agents are not aware of the interactions between the cyber and the physical parts of the system. Such unawareness increases the chance of conflicts, errors, and failures.

A more advanced version of this line of research was resented by the work of [7, 18] as they used multiple AI techniques to control a pool of mobile agents performing control tasks. The use of AI guided the management platform towards smarter decisions. Unfortunately, they shared the same problem of their insisters, the lack of situational awareness, and the inconsideration of isolating the control platform from the host under control. Such limitations limited the scalability of such systems, and their ability to suit CPS applications.

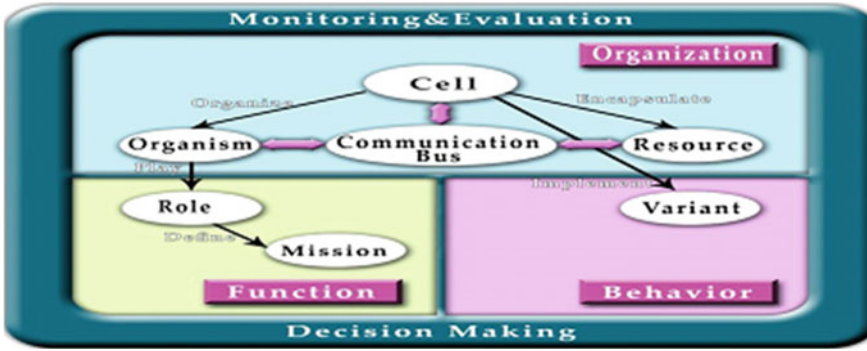


Fig. 2.5 Components of the COA

2.4 Evolutionary Sensory System (CyPhyMASC)

2.4.1 Overview of the Cell-Oriented Architecture

The COA employs a mission-oriented application design and inline code distribution to enable adaptability, dynamic re-tasking, and re-programmability. The Cell, is the basic building block in COA, it is an abstraction of a mission-oriented autonomously active resource. Generic Cells (Stem Cells) are generated by the host middle-ware termed COA-Cell-DNA (CC-DNA for short), then they participate in varying tasks through a process called specialization. Cells are intelligent, independent, autonomous, single-application capsules that acquire, on the fly, application specific functionality in the form of an executable code variant through the specialization process. Cells are also dynamically composable into larger structures Organisms representing complex multi-tasking applications. An Organism is a dynamic structure of single or multiple Cells working together to accomplish a certain mission. Figure 2.5 illustrates the different aspects and components of the COA. Applications built over this COA are constructed as a group of cooperating roles representing a set of objectives. An Organism represents a role player that performs specific mission tasks. Organisms are bound to functional roles at runtime. Note that different Organism variants can be bound to the same functional role at the same time leading to heterogeneous redundancy to support enhanced resilience.

2.4.1.1 The Cell

Conceptually, the Cell is the basic atomic active resource in a distributed computing platform. Cells act as a simple, single application module capsules (virtualization environment, or sandbox) isolating the executable logic from the underlying physical resources. Cells acquire, on the fly, application specific functionality in the form of an

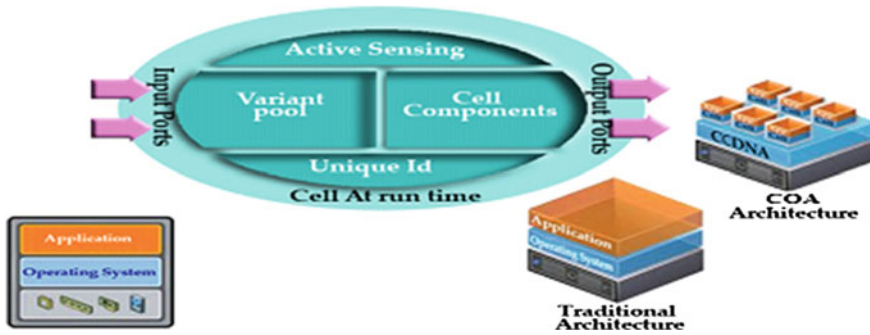


Fig. 2.6 COA cell at runtime

executable code variant. COA Cells are equipped with multiple tools supporting their intrinsic features regarding resilience, dynamic performance optimization, Cell self and situation awareness, and online programmability and adaptability. Figure 2.6 illustrates an abstract view of a COA Cell at runtime. A single workstation can host one or more Cells, providing a flexible way to share the physical resources among multiple applications. The COA Cell encapsulates the running application via elastic dynamic emulation of a suitable execution environment. Such encapsulation isolates the running application from the host platform decoupling the executable logic from the underlying physical resources. The COA Cell is designed to work on a redundant copy of the applications critical data, while all the application critical data are always saved in a remote safe warehouse. The intelligent COA management platform described in [20] works on maintaining all the aspects related to these operations seamlessly.

Decoupling data, logic, and physical resources enable the COA Cells to seamlessly move at runtime between different hosts regardless of their configuration heterogeneity. A successful utilization of such mobilization for security objectives is presented in [21]. CyPhyMASC also utilizes these features by employing the COA Cells as security-service carriers. CyPhyMASC circulate these carriers through the ToS network for host data collection and security service delivery as illustrated in the next sections.

2.4.2 The Foundation

CyPhyMASC is an evolutionary sensory system designed to enable real-time pervasive monitoring and analysis towards autonomous context aware security service provisioning. CyPhyMASC security provisioning platform is composed of three main layers, the management layer, sensor and effector abstraction layer, and sensor and effector tools layer as presented in Fig. 2.8. The three layers are founded over a COS based foundation. The management layer rules are played by a set of organ-

isms composed of Cells. The abstraction layer uses COA Cells to encapsulate attack investigation and resolution tools defined as binary code variants (APIs) constructing a set of platform independent sensing and effecting capsules. CyPhyMASC constructs a biological immune system like, defense environment by circulating generic streams of such capsules into the (Target of Defense) ToS body to induce a blood stream like effect. The following section illustrates and provides more technical details about the defense-capsules creation process, and the security provisioning methodology of CyPhyMASC.

2.4.2.1 CyPhyMASC Organisms and Capsules Composition

CyPhyMASC leverages the COA ability to abstract, encapsulate, and virtualize heterogeneous physical and logical resources into unified programmable objects Cells. Cells are sandboxes internally construct a suitable working environment for heterogeneous tools. Externally, it is capable of changing its characteristics to work with many targeted architectures. Regardless whether the sensing target was a computer in a network, or a physical sensor with COA-ready digital interface COA Cells will hide these differences from the enclosed sensing/effecting API.

CyPhyMASC uses the middleware (CC-DNA) installed on the ToS host machines to instantiate, deploy, and host sensor and effector Cells. CyPhyMASC sensors and effectors are a set of precompiled APIs with specific sensing or effecting tasks. Sensors and effectors come with a detailed specification file describing the targeted platform, estimated computational Wight, needed libraries to support it, possible conflicts, ... etc.

CyPhyMASC defense mission (organism role) is defined using a custom-made programming language used to generate scripts defining the structure, workflow, and the set of tasks for the sensing and effecting organisms. Additionally, it also defines the type of sensors and/or effectors needed to execute that mission.

Organism creation starts when the host CC-DNA receives the logic script. CC-DNA interprets this logic to construct the organism sensor or effector Cells. In COA, resources can easily be acquired when needed. CC-DNA might ask the local or remote logic reservoir for any sensor or effector APIs that are required to execute the designated sensing or effecting mission in case they were not already available on the targeted host.

2.4.3 CyPhyMASC Security Provisioning Methodology

CyPhyMASC manages and controls sensing and effecting Cells based on specific mission objectives provided by the SSP. CyPhyMASC works as a middle layer between the SSP and the ToS as illustrated in Fig. 2.8. CyPhyMASC leverages the uniform abstract representation of sensing and effecting Cells to circulate sensors and effectors intelligently throughout the ToSs. Additionally, CyPhyMASC leverages



Fig. 2.7 CyPhyMASC abstract view

such features to enable security service interoperability; where different SSPs can share in security provisioning for the same ToS in a privacy-preserving manner as will be explained shortly. CyPhyMASC security provisioning includes two main modes, a SSP-guided mode where CyPhyMASC blindly execute predetermined defense missions provided by the SSP; and an evolutionary-mode that involves evolutionary sensing, and effecting. The SSP-guided modes use CyPhyMASC as a delivery platform that executes certain commands blindly without being involved in the details of the process. CyPhyMASC collects runtime commands and deliver real-time feedback to the SSP. This mode is highly un-scalable and not recommended for large systems. As this is a limited version of our Evolutionary sensory system, we will move forward and illustrate more about the more generalized mode the evolutionary-mode (Fig. 2.7).

2.4.3.1 Overview and Initial Configuration

Evolutionary sensing aims to detect malicious abnormal behaviors without prior knowledge of that behavior. In both modes, CyPhyMASC maintains minimum level of security by maintaining the normal work mode of the currently-being-used tools on the ToS. CyPhyMASC treats such tools as part of its sensing and effecting arsenal. The Evolutionary sensing process involves analyzing and correlating different information feeds from multiple sources to magnify up-normal behavior deviation identifying possible attack indications. Evolutionary effecting involves utilizing the pervasive control feature of CyPhyMASC to autonomously deploy safe-resolution tools “that doesn’t conflict with the running applications,” or to contain such attacks

within certain perimeter while waiting for administrators to provide clear resolution procedure to execute. The deployment or containment mechanism works based on an intelligent and dynamic profiling mechanism.

The profiling mechanism works on the fact that attacks can be directed attacks working towards certain objective, or undirected attacks that seek maximizing the victim losses. Even for undirected attacks they can be considered as directed attacks at certain levels. For example, at the operating system levels, windows based attacks cannot infect Unix operating hosts, Java based attacks cannot infect C based software packages, etc. The working environment of a certain host can significantly limit the type of attacks infecting this host even for undirected attacks. Based on that, we can easily classify attacks into groups based on certain classification protocol. Such classification can be attributed with a set of parameters determining the likelihood of having an attack under this class/group in one of the hosts within certain boundaries.

CyPhyMASC has a set of pre-deployed manually/automatically generated profiles used to direct the sensor circulation and the basic deployment package for each host. CyPhyMASC adapt such profiles all the time to maximize the efficiency and accuracy of security provisioning. The dynamic adaptation of profiles adjusts the definitions defining the needed type and density of defense missions within each profile. Each profile is configured to match the host, host network, attached organization or enclave settings.

2.4.3.2 Joining CyPhyMASC Network

Joining an CyPhyMASC-equipped SSP network procedure starts by installing the CC-DNA on the host machines, registering the hosts' physical IPs, hosts configurations, and their security and privacy policies to the SSP host database, and classifying the host based on one of CyPhyMASC Profiles. Usually the host follows the general profile of the enclave/organization that it belongs to. However, CyPhyMASC can assign a more fine-grained profile to a certain set of hosts within the same network if they are supposed to behave differently at runtime. The configuration of such fine grained profiles can change over time if the behavior of the host or the surrounding changes.

The host configuration profile illustrates all the details regarding the host platform, computational capabilities, the organization /enclave id(s) for that host if any, and any special consideration regarding the applications running on it. The security and privacy policy defines the needed security level, the scope of cooperation, and the type of allowable sharing materials.

Upon registration of a new cyber/physical host, CyPhyMASC is notified to start the initial evaluation of the host to determine the profile that the host will follow, identifying the basic sensor deployment-package composition-profile. CyPhyMASC interprets the host record in the SSP database to identify the appropriate types of sensors and effectors APIs that match the host configuration-profile. CyPhyMASC will deploy the evaluation-sensors package to initiate the initial checkup to verify that the host the minimum requirements needed to join the SSP network. In case of

any problems, CyPhyMASC will autonomously deploy the appropriate effectors to resolve it.

CyPhyMASC frequently change the basic deployment-package by circulating new sensors to replace old ones. The process is guided by the global sensing feedback not only at the host but also through the network, and the security provisioning profile that the host is following. CyPhyMASC use a grading system to continuously evaluate sensors on each host based on their success to detect up-normal behaviors. In normal situations, at each evaluation-round, one of the new most successful sensors within each profile replaces the least successful sensor in the basic deployment-package of the host. The details about sensor circulation are illustrated later.

2.4.4 Evolutionary Sensing and Effecting Framework

After joining CyPhyMASC Network, the host defense related aspects will be handled by one of CyPhyMASC management units. The management unit works as a part of CyPhyMASC sensing and effecting framework presented in Fig. 2.8. CyPhyMASC sensing and effecting framework is classified into three main layers management layer, sensing and effecting abstraction layer, and the defense delivery tools as illustrated in Fig. 2.8. The sensing and effecting tools layer is a set of logical sensing or effecting APIs stored in the local reservoirs. These tools are autonomously abstracted at runtime into uniform sensing and effecting Cells participating in the construction of organisms playing certain defense missions. CyPhyMASC organisms are anonymously constructed, managed, and controlled at runtime by CyPhyMASC management layer. This layer is responsible for collecting, correlating, and analyzing sensor feedbacks. Additionally, this layer is responsible for taking decisions based on the sensing feedback, previous historical events, and SSP guidelines. Such decisions might involve composing more capable effecting defense missions for resolution or new sensing missions for deeper investigation.

CyPhyMASC management layer is a tree-like hierarchical construction, where hosts are connected to leaf-Brains “decision making organisms” to be monitored and controlled as presented in Fig. 2.5. Based on CyPhyMASC administrator settings, each leaf-Brain manages a specific number of hosts. leaf-brains frequently reports to their parents “Higher-level brains” for more comprehensive guidelines.

2.4.4.1 Feedback Management and Representation

CyPhyMASC Sensors feedback is a score-sheet like report that compares the behavior deviation regarding the sensing target to a predetermined threshold. Sensors are classified into different sets representing their targeted sensing objectives “ex, memory, communications, privacy, .. etc.” Thresholds are dynamically adjusted based on the nature of each host, and the number of false negatives/positives reported by the Sensor. Figure 2.9 illustrates the feedback analysis process. Score sheets from

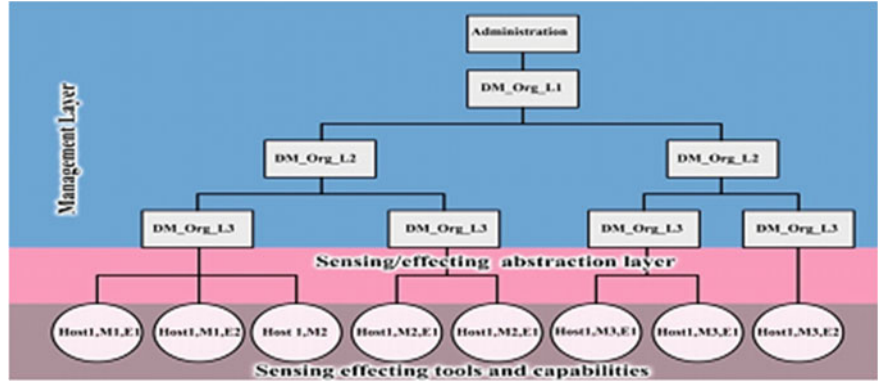


Fig. 2.8 CyPhyMASC abstract view

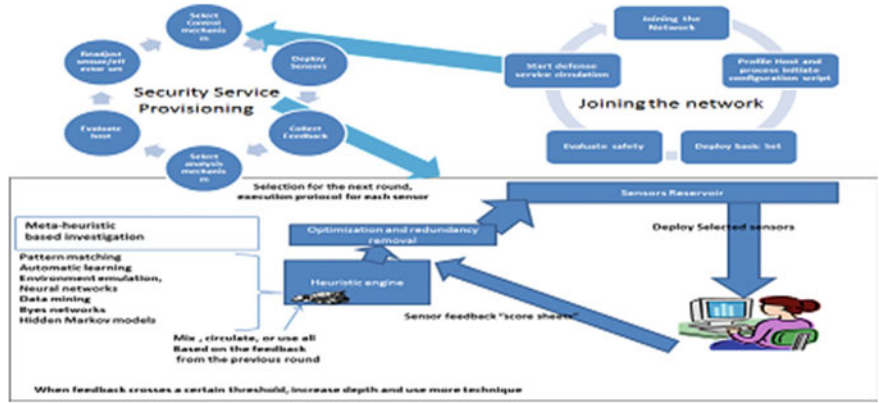


Fig. 2.9 The evolutionary security provisioning process

different sensors for the same host are sent to the leaf-brain to compose comprehensive score-sheets to be checked against the defense rules database. Each defense rule has a score-sheet attached to it. Rule-sheets have values for different objects “ex, memory, communication, .. etc” reflecting the behavior patterns “attack signature” of each object in case of infection. Behavior pattern description can be discreet or continuous. Rule description also includes the host sampling procedure. Sampling procedure describes the needed number of samples per object and the duration of each sample, and the sensors needed to takes such samples.

The leaf-brain checks the partial similarity between the sensor feedback “score-sheet” and the existing rules-sheets to allocate the most useful rules for the next deployment round. Based on that selection, the leaf-brain will compose a new organism, holding the list of sensors mentioned in the rule description, with a set of preprogrammed tasks based on the rule sampling procedure. Based on the feedback, threats might be detected, and the resolution mechanism described in the rule will be

followed. The leaf-brain will compose a new effector organism with list of effectors mentioned in the rule description and the execution workflow described in the rule.

Rules are under full time update by the parent-brains, and the SSP. Leaf-brain experience is frequently reported to the parent-brains for further guidelines. Parent-brains can construct a more comprehensive view of the whole network by correlating the leaf-brain feedbacks. Such views can magnify certain behavior deviation across the network, which will guide the composition of new defense roles to be executed by the leaf-brains for further investigations.

2.4.4.2 CyPhyMASC Information Sharing (Vaccination)

At the higher levels of the hierarchy “the parent-brains”, the collected incident reports with the rules, sensors, and effectors used are archived for sharing. The Clearing-House Organism (CHO) role will be easy in this case, as it will check the ToS privacy policy against sharing of such materials. As described before, the shared materials carry no indications about the specific incident source. It is similar to defense related tips that encourage SSP to apply a specific rule because it might reveal certain threats. The reason that motivated SSP to share such information might be the rule successfulness to detect a threat at one of her ToSs or it is a new rule that was developed with promising results. However, the ToS privacy policy will be checked in case it prophes even that level of information sharing.

The CHO role becomes more significant when the SSP asks cooperating SSPs to provide a solution for a problem she has. In this case, the reported suspicious score-sheet and the sensors used to extract it will be shared with other SSPs. CHO will check that material to make sure that the information enclosed dose not contradict with the ToS privacy policy. CHO rejected authorizations are reported to the administrator to manually override or discard. CHO will also check the SSP feedback regarding such requests, new rules, sensors, and guidelines might be provided as a resolution for the problem. Authorized solutions will be deployed, and rejected ones will be reported to the administrator for further guidelines. The details and the framework of defense mission sharing will be described later.

2.4.5 CyPhyMASC Brain Architecture

Figure 2.10 illustrates CyPhyMASC brain architecture and composition of organisms and the interactions between these organisms to achieve CyPhyMASC goals. I will briefly describe each component and its dedicated task. The Analysis and investigation organism is responsible of analyzing the continuous feedback from the ToS deployed sensors. The analysis and investigation organism sends reports to the Decision making organism to take decisions regarding composing new defense missions, authorizing the use; selecting the type of the effectors if needed. The Role composer organism will use the decision making organism guidance to compose new

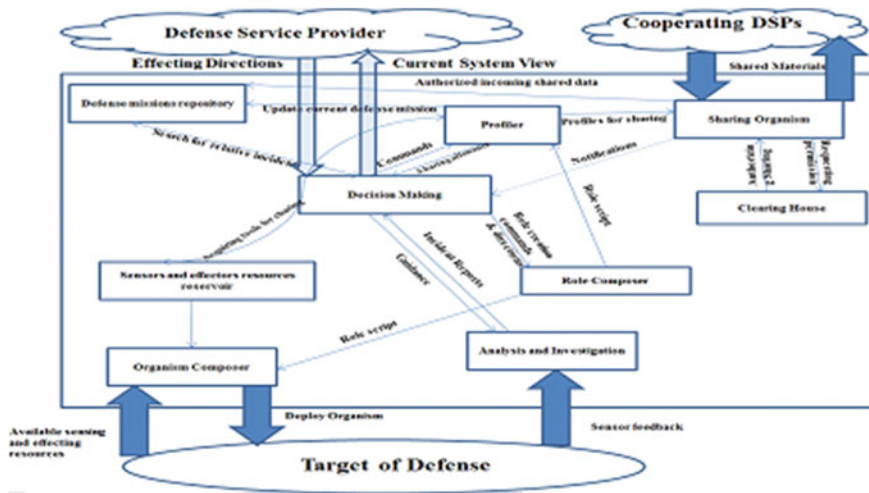


Fig. 2.10 CyPhyMASC architecture

defense roles. The role composition is described using our custom made mission definition language. The role script will be sent to the organism composer. The organism composer is a part of the CC-DNA installed on the host. The organism composer organism is responsible for the resource virtualization process. It will abstract the host resources to compose the requested organism Cells. These resources might be physical resources memory, processor, or logical resources in the form of conventional defense tools, or any locally stored sensors or effectors. If any of the resources needed to compose the cell did not already exist on the host, the cell composer will acquire these resources from the Sensor and effector Reservoir. Upon generating and testing new defense missions the decision making unit might decide to instruct the Profiler to generate a profile for the mission with the used tools to be shared with other cooperating SSPs. The sharing process is handled through the sharing organism that will manage sending and receiving shared materials between SSPs. Sharing or employing shared materials has to be authorized by the Clearing House organism. The Defense missions repository organism will hold history of defense mission usage either locally within the same SSP or globally through feedbacks from other cooperating SSPs with an evaluation for such missions for future reference.

2.4.5.1 Information Sharing and Exchange Protocol Within CyPhyMASC

One of the main contributions of CyPhyMASC is the trustworthy information sharing and exchange. CyPhyMASC share attack events and detection/resolution materials between different management organisms locally within the same organization, and globally between cooperation SSPs. Figure 2.11 illustrates CyPhyMASC defense mission sharing protocol.

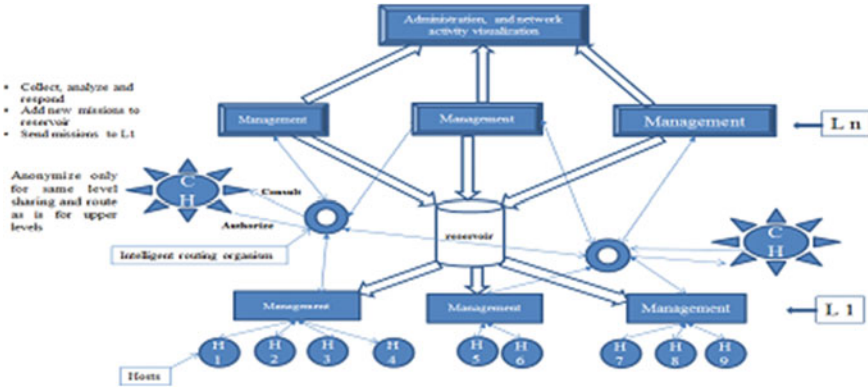


Fig. 2.11 CyPhyMASC defense mission sharing protocol

As mentioned before, the security provisioning process is managed by a hierarchy of management organisms. At the leaf nodes we have the management layer that manages defense mission circulation and execution on the ToS hosts. Such layer collects the sensor feedback, and the detected list of events that was sent to it through the score sheet technique described before. The collected materials are forwarded to one of the distributed routing organisms.

The routing organisms collect the incoming reports and send them identifying the report-source to the higher layer management. Additionally, all reports that was classified as important events by the report source is anonymized and checked for privacy policy violation by the clearing house organism to be forwarded directly to the others local management organisms. The clearing house organism, ask the local broadcasting organism to handle this task. The broadcasting organism will remove any duplicated reports regarding same event, or same defense mission and instruct the leaf management units to raise the score of the missions related to the reported events based on the severity of the event. Raising the score of a certain mission will increase the chance of applying it in the next mission circulation round. At the upper level of the management hierarchy, the collected reports from the lower level management units are correlated and analyzed to be presented to the network administrator through visualization software. Using such software will make it easier for the administrator to visualize the attack activity, the security provisioning process, and to provision manual override or further guideline if needed.

The higher management units will analyze the collected reports providing further guideline to the lower layers. It is also responsible for generating new missions to be stored in the missions repository. The upper layer management organisms layer provides guidelines to leaf management organisms. The upper management layer organisms can add a new package of sensing or effecting missions to be executed on a specific or a set of ToS hosts at specific time event; or by prioritizing or de-prioritizing in case of many false positives one of the missions already being used by this management unit. SSP level sharing is the responsibility of the higher management layers

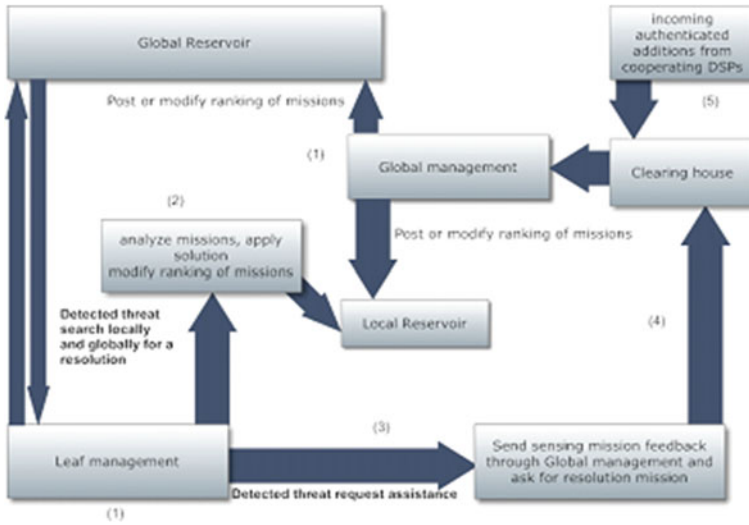


Fig. 2.12 The defense mission lifecycle

only. This level of sharing occurs upon the reception of a highly suspicious incoming event. The management organisms mark such event and its relative defense mission for sharing. A dedicated organism named as the sharing organism, anonymizes the shared material, and authorizes it for sharing via the clearing house organism. If the shared materials were not in violation of any of the ToS privacy policies, the shared materials are sent to all cooperating SSPs either as an alert or asking for a resolution guidelines. SSP attack alerts should include details about the attack and detection/resolution methodology in terms of defense missions and sensing/effecting tools. If the SSP was asking for an advice related to certain malicious activity, the abstract information related to how such activity is identified and formalized as a sensing only mission, all the tools needed to execute the mission is added to it to be broadcasted to the cooperating SSPs.

Upon reception of such missions, the receptor makes sure that there are no privacy violations. The mission is then tested in a controlled environment before applying it to the SSP attached ToSs. If the request was for a resolution guideline, and the receptor SSP has the resolution methodology, the resolution is composed in a defense mission format attaching all the needed sensing and effecting elements to execute it and sent back to the source. The source test the resolution tools in a controlled environment after making sure that there is no privacy violations then based on the administration opinion it might be deployed to resolve the reported attacks. Figure 2.12 represent the defense mission life Cycle.

2.4.5.2 Intelligent Attack Detection and Resolution

The main objective of any attack detection mechanism is to accurately and properly direct the correct defense tools towards matching attacks. Researchers proved that accurately identifying attacks is an NP-Complete problem [22], while others proved that it might be considered as an NP-hard problem as well [23]. The conclusion is that the problem cannot be solved in realistic time as the problem space expands exponentially over time. The use of Heuristics is always considered to be a good solution for such problems [24]. The most successful malware detection mechanisms uses heuristic scanning and signature based mechanisms to detect attacks. Heuristics scanning in its basic form is an implementation of three metaheuristics mechanism, the pattern matching, automatic learning, and environment emulation. Due to the high computational cost of running such heuristics based techniques, modern anti-malware techniques that are usually shares the same host or the host network of their ToS use a limited set of the available metaheuristics techniques. The reason behind that is to save the computational resources and to speed the process of classifying the executable tasks without interrupting their execution. CyPhyMASC is designed to work in total isolation of the ToS, and to isolate the main design concerns of malware detection and resolution, sensing, effecting, and control logic. Working in isolation from the ToS enabled CyPhyMASC to use more heuristics techniques and increase the depth of learning and investigation of such techniques without any negative impact on the ToS performance, or resources. Most of the workload on the analysis and investigation is waved to the SSP platform, and the ToS participates only in hosted sensing and effecting elements for a limited time frame. As presented before separating the security provisioning design concerns enabled CyPhyMASC to optimize the process of sensing and effecting saving more of the ToS resources. CyPhyMASC intelligent sensor reuse mechanism uses the same sensor to feed multiple heuristic techniques, to save a considerable amount of the computational power of the ToS. The next sections illustrate the sensor circulation, selection and reuse mechanisms of CyPhyMASC.

2.4.5.3 Profile-Guided Sensor Circulation

CyPhyMASC circulates its sensors and effectors to execute defense missions. The sensor circulation protocol depends on the tool requesting sensor deployment. The security provisioning process involves cooperation between multiple detection and resolution tools. Each tool submits a sensor deployment request to the sensor repository to deploy the requested sensors. The requests pass through optimization unit to remove sensors that were recently deployed before. Figure 2.13 illustrates the sensor selection and deployment procedure.

Classification based on profiles

As mentioned before, CyPhyMASC attach hosts to certain profiles based on the host engagement with its organization and enclave, host configuration, behavior,

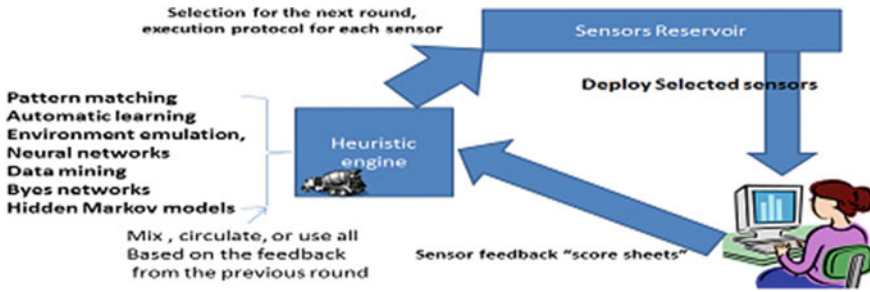


Fig. 2.13 Sensor selection and deployment

usage pattern, etc. some of these profiles are static and preloaded with CyPhy-MASC management units, and we call them the coarse grained profiles. Such profiles focusses on static classification aspects, like organization id, enclave id, platform configuration, network protocols, etc. this profiles determine the general security provisioning pattern for the host. The second type of profiles is a fine grained dynamic profile that determines the usage behavior of the host and mostly reflects the user behavior using the host. The type of applications being frequently used, the hours of operation are good examples for the aspects controlling such level of profiling. This profile type is dynamically adjusted based on changes on the usage behavior. CyPhyMASC uses host resident sensors to monitor such changes. The main objective behind using such profiling system is to optimize the utilization of security provisioning tools by directing only tools with high success ratio. Attacks are somehow targeted, either from the application-objective perspective, or from the technical perspective. CyPhyMASC circulates defense missions while favoring the activation of defense tools targeting attacks that match the host profile. CyPhyMASC do not limit tool activation to only those who match the host profile to cover any unexpected out of profile attacks. Using profiles to guide tool activation minimize the search space, enhance the detection accuracy and promptness and optimize resource usage on the host and on the SSP. The detection mechanism relays on multiple control techniques. Signature based technique is used by the resident unit to maintain minimum level of security at all times. The evolutionary sensing mechanism use heuristic technique to guide and control sensor circulation and to analyze sensor feedback to detect unknown attacks, or to identify maliciously acting components.

Identifying unknown threats

The utilization of heuristic / metaheuristics is necessary to enable attack prediction, and detection of unknown attacks. CyPhyMASC utilizes its ability to isolate the main security provisioning concerns sensing, effecting, and control logic to extract abstract, privacy friendly information regarding running processes. The feedback is safely sent to remote analysis units to apply whatever logic is needed to detect attacks. The selected logic metaheuristics technique determines the type of sensors to be used, execution pattern for such sensors, sample collection protocol. Additionally, that logic is the one responsible for processing the feedback coming for such sensor

```

(mechanism-id) Select-mechanism( last used, weight, severity level)
(Sensor-list, activation-protocol) Activate (mechanism-id, suspicious-item, item-type)
(deployment-package) Call optimization-unit ((Sensor-list, activation-protocol)
(feedback) Call deploy-sensors (deployment-package)
(res) Process-feedback (feedback, mechanism-id)
If res>threshold
fire-alert (item-type, mechanism-id, deployment-package, feedback)
else
if res < Dynamic(10)% threshold
(mechanism-id) Select-next-mechanism( last used, weight, severity level)
Repeat process
Else
Discard event

```

Fig. 2.14 Example of the hubristic mechanism selection procedure

to determine whether the host is safe or not. The process starts when the selection mechanism selects the metaheuristics technique to be used for the next detection round.

Metaheuristics selection mechanism

CyPhyMASC can use single metaheuristics technique or multiple metaheuristics technique at the same time to investigate certain issue. The selection of how many techniques to be used and the type of techniques used depend on the severity of the situation under investigation. The default is the use of only one technique, while if the last utilized technique reported high level of attack certainty that is close, but do not cross the required safety threshold, the system use other techniques to enhance the quality of calculation. In CyPhyMASC, metaheuristics techniques are ordered and have weights assigned to each one. The metaheuristics technique selection mechanism always prefers techniques with highest weight value. When an additional mechanism is needed the next highest value technique is selected. The weights for each technique is not fixed, it is dynamic and gets assigned based on the success or failure of each technique to detect attacks, and the number of false positives or negatives of each technique. This evaluation occurs independently at each management unit within CyPhyMASC framework.

Figure 2.14 presents a simple representation of the metaheuristics technique selection process.

2.4.5.4 Sensor Reuse

CyPhyMASC optimize ToS resource usage by minimizing the number of deployed sensors and effectors on the host. CyPhyMASC utilize it ability to separate the main concerns and the availability of abstract sensing and effecting elements to remove any duplication of sensor deployment requests. The separation between the tool and

the control behind it enabled CyPhyMASC to reuse previous sensor feedback within a certain time frame to feed multiple control components.

For example, let us assume that the selection protocol at time (t) selected pattern matching metaheuristics technique to investigate events occurring on host (h1). The investigations focused on suspicious memory behavior of a certain process. The sensors are selected based on that objective, and the sampling protocol is automatically generated using the predetermined syntax. The list and the protocol selected are sent to the optimization unit. The optimization unit checks if there was any previous valid match for that request. If any, it removes it and report the recorded feedback directly to the analysis unit. If not, the list is sent to the reservoir to program and deploy the sensors. The feedback is sent back to be analyzed by the pattern matching algorithm to determine whether there was an attack or not.

The optimization unit applies certain aging policy to determine the validity duration for a certain sensor feedback to be reused. The expiration date is dynamic and adjusted automatically based on the ToS host workload, nature, types of application, level of changes, amount of data flowing from and into it, number of application reinstalling, deployment, .. etc. If the requested sensor list contain any sensor that was used before with a valid expiration date and compatible deployment protocol measurement sequence, time, sampling rate, etc. then it will not be deployed again, and the old feedback will be reused. Doing so, is expected to save a considerable amount of host resources.

2.5 Example of CyPhyMASC Security Mission

The task of each CyPhyMASC component is further illustrated through a discussion of one of CyPhyMASCs automatically composed security missions that search for memory behavior deviation within a predetermined timeframe, and the dispersion of such deviation through the ToS host networks.

Goals

1. Detect massive deviation in memory usage
2. Locate the area under suspicion
3. Collect information about processes with suspicious memory usage
4. Identify critical and non-critical processes
5. Resolve the problem.

Tools

1. Memory usage monitoring sensors
2. Processes information collection crawler sensors
3. Process killing effectors.

2.5.1 Detection and Resolution Scenario

This synthetic scenario illustrates the event of using X123 to secure organization ABC. ABC is a large organization composed of multiple enclaves. The following incident happened in enclave E1. On the regular inspection round on E1 hosts, with an active heuristic mechanism X, the feedback collected by the deployed sensing organisms and analyzed by the analysis organism indicated an unidentified strange behavior in host A. The decision-making organism calculated the weights from the score sheets and followed the heuristic rules and the comparison between the calculated weights exceeds the threshold. The decision-making organism sent its guidelines to the role-composer based on the analysis reports with the list of high similarity rules to the reported feedback score-sheet. The role-composer composes new security mission that mixes the sensing part of all the similar rules. X123 was the newly generated mission that was built to investigate possible memory-related behavior deviations within E1. X123 have three main roles played by three organisms, sensing, analysis, and effecting. The sensing organism uses the memory scanning Cells to take multiple snapshots of the memory usage within host A for a certain period. The analysis organism applies some predetermined statistics to evaluate the detected behavior deviation that will be evaluated and compared against certain threshold to determine the next step. Based on the result, further investigations might be needed.

These investigations will be handled by the sensing organism that will deploy processes-information-crawler sensor Cells. Crawlers will collect information about processes with high memory usage. The collected data will be sent to the analysis organism that will generate a comprehensive report to the decision-making organism. The decision-making organism will decide whether to discard the incident, or to activate effector organisms on X123 to resolve the situation.

The decision-making organism might decide to share the mission profile with other SSPs asking for external feedback, or deploy X123 locally for further investigations. Based on the sharing command search-scope and the clearinghouse permissions X123 will be re-deployed. The deployment scope might be limited to only the hosts within enclave E1, all over ABC enclaves, or globally between SSPs searching for similar behavior deviation pattern. If the decision was to activate X123 effecting organisms for quick resolution, X123 will be instructed to kill some of the suspicious non-critical processes and re-evaluate the situation. If the redeployment came-out with multiple incoming alerts for the same memory behavior deviation, the decision-making organism will raise the severity level of the situation indicating global wide spreading attack. Based on that, commands will be issued to the role composer to customize a new containment mission based on the attack reported parameter. Meanwhile CyPhyMASC will be applying resolution effectors of X123. The containment effectors will be deployed over host A, other hosts in communication with host A, and the intermediate communication elements routers, switches, etc. to construct a quarantine area around the malicious host. After successful containment and resolution the whole process will be profiled and stored in the security mission repository for future reference. These profiles will hold details about the containment, and res-

olution methodology, and all the sensors and effectors API used to compose the used missions. Sharing authorizations and scope of these profiles will depend on the local clearing-house decision.

2.5.2 CyPhyMASC Addressing the BW Attacker Assumptions

In this section we discuss the BW attack design invariants and attacker assumptions that were mentioned in earlier illustrating how CyPhyMASC adequately invalidates them. CyPhyMASC is designed to work in total isolation from the ToS, invalidating assumption (1). CyPhyMASC acts as a buffer between the SSP and ToS. Neither CyPhyMASC nor the SSP share the network or the hosts of the ToS. The security services are delivered to the ToS in an isolated network that connects the ToS to CyPhyMASC. The security delivery vehicles are secured using our Moving-target defense (ChameleonSoft) described in [22], which invalidates assumption (4).

CyPhyMASC is an active defense system founded over a smart foundation managed by our automated management system (CyberX) described in [21] and secured by ChameleonSoft [22]. One of the main tasks of ChameleonSoft as is monitor and secure the COA based foundation against threats and attacks. Having ChameleonSoft and CyberX handling such details waves this workload away from CyPhyMASC giving it more space to focus on provisioning security services to the ToS. Additionally, CyPhyMASC is designed to support large scale systems and computationally expensive tasks. CyPhyMASC design supports distributing the tasks over a hierarchy of independent management entities composed of fine grained components managed by CyberX. The fine granularity of such components and the isolation between its logic, data, and physical resources enabled CyberX to fractionize large tasks over multiple hosts constructing a cloud like platform with virtually infinite resources. With that unique feature, CyPhyMASC invalidates assumption (3).

CyPhyMASC uses signature based detection tools as a part of its arsenal, while the major part of that arsenal relay on an evolutionary sensory system. CyPhyMASC utilizes multiple intelligent mechanisms to detect unknown attacks based on monitoring suspicious activities and up normal behavior. Further, CyPhyMASC is not a commercial product available for conventional users. Even though, the foundation of CyPhyMASC is highly dynamic and autonomous inducing high level of dissimilarity between identical copies of the same system, and that invalidates assumption (2).

One of the main objectives of CyPhyMASC is to promote the security system situational awareness of the different ToS components and to isolate the platform composition heterogeneity enabling seamless security provisioning. CyPhyMASC pervasive monitoring and analysis, and the intrinsic trustworthy sharing and cooperative security enhance the situational awareness all ToS components. CyPhyMASC collect events from different entities of the ToS, correlate the collected information to generate a global image of the entire system to be analyzed by high management units. Doing so, enables CyPhyMASC to detect slow moving attacks, and attacks using remote bots to lunch attacks on remote hosts. The aforementioned aspects success-

fully invalidate assumptions (3 and 5). By invalidating all the attacker assumption, we surmise that it is virtually impossible for the BW attack to succeed in attacking an CyPhyMASC protected system.

2.6 Conclusion

In this chapter, we presented CyPhyMASC whose unique construction and functionalities provide pervasive, seamless and scalable MASC services to enable interoperable and dynamic security employing situation-appropriate tools; early failure/attack detection and resolution; and trustworthy scalable cooperative security amongst heterogeneously composed and interacting targets like CPS. CyPhyMASC is built upon our novel Cell-oriented architecture and acts as a trustworthy elastic intelligent middle layer uniformly interfacing SSPs and ToSs. There are several interesting challenges that remain to be addressed. These include formalizing the sensing and effecting autonomous management and mission generation framework; autonomous inspection and profiling of security vaccines; correlating sensors feedback into comprehensive real-time global views; and optimally adjusting sensors circulation based on the dynamic changes of these views.

References

1. Knight, J.C., Leveson, N.G.: An experimental evaluation of the assumption of independence in multiversion programming. *IEEE Trans. Softw. Eng.* **12**(1), 96–109 (1986)
2. Jorstad, N., Landgrave, T.S.: Cryptographic algorithm metrics. In: 20th National Information Systems Security Conference, Baltimore (1997)
3. Parra, C.: Towards Dynamic Software Product Lines: Unifying Design and Runtime Adaptation. Universit Lille (2011)
4. Stberg, P.-O., Elmroth, E.: GJMF - A Composable Service-Oriented Grid Job Management Framework (2010). <http://www.cs.umu.se/ds>
5. Chen, X., Andersen, J., Mao, Z.M., Bailey, M., Nazario, J.: Towards an understanding of anti-virtualization and anti-debugging behavior in modern malware. In: International Conference on Dependable Systems and Networks (2008)
6. Sze, S., Tiong, W.: A Comparison between Heuristic and MetaHeuristic Methods for Solving the Multiple Traveling Salesman Problem. World Academy of Science, Engineering and Technology (2007)
7. Podgrski, W.: Artificial Intelligence Methods in Virus Detection & Recognition—Introduction to Heuristic Scanning (2012). <http://podgorski.wordpress.com>
8. Haack, J., Fink, G., Fulp, E., Maiden, W.: Cooperative infrastructure defense. In: Workshop on Visualization for Computer Security (VizSec) (2008)
9. Maiden, W.M.: DualTrust, A Trust Management Model for Swarm-Based Autonomic Computing Systems. Washington State University (2010)
10. Maiden, W.M., Dionysiou, I., Frincke, D.A., Fink, G.A., Bakken, D.E.: DualTrust: A Distributed Trust Model for Swarm-Based Autonomic Computing Systems. Data Privacy Management and Autonomous Spontaneous Security (2010)

11. Lee, Y.: A Pre-kernel agent platform for security assurance. In: IEEE Symposium on Intelligent Agent (IA) (2011)
12. Nguyen-Tuong, A., Wang, A., Hiser, J., Knight, J., Davidson, J.: On the effectiveness of the metamorphic shield. In: The Fourth European Conference on Software Architecture ECSA 10, pp. 170–174 (2010)
13. Abraham, A., Jain, R., Thomas, J., Han, S.Y.: D-SCIDS: distributed soft computing intrusion detection system. *J. Netw. Comput. Appl.* **30**(1), 81–98 (2007)
14. Wu, S., Banzhaf, W.: The use of computational intelligence in intrusion detection systems: A review. *Appl. Soft Comput.* **10**(1), 1–35 (2010)
15. Mukherjee, S.: FPGA based Network Security Architecture for High Speed Networks, MTech (2001)
16. Otey, M., Parthasarathy, S., Ghoting, A., Li, G., Narravula, S., Panda, D.: Towards nic based intrusion detection. In: The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 723–728 (2003)
17. Santos, I., Penya, Y., Devesa, J., Bringas, P.: N-Grams-based file signatures for malware detection. In: The 11 International Conference on Enterprise Information Systems (ICEIS) (2009)
18. Lemos, R.: White House Network Attack Highlights Need for Stronger Defenses (2012). <http://www.eweek.com/security/white-house-network-attack-highlights-need-for-stronger-defenses/>
19. Prayurachaturporn, S., Benedicenti, L.: Increasing the reliability of control systems with agent technology. *ACM SIGAPP Applied Computing* (2001)
20. Box, D.: *Essential COM*. Addison Wesley, Reading Mass (1998)
21. Knight, J.C., Davidson, J.W., Evans, D., Nguyen-Tuong, A., Wang, C.: Genesis: A Framework for Achieving Software Component Diversity. Technical Report AFRL-IF-RS-TR-2007-9, University of Virginia (2007)
22. Te-Shun, C., Sharon, F., Wei, Z., Jeffrey, F., Asad, D.: Intrusion aware system-on-a-chip design with uncertainty classification. In: The 2008 International Conference on Embedded Software and Systems-ICESS (2008)
23. Azab, M., Hassan, R., Eltoweissy, M.: ChameleonSoft: a moving target defense system. In: 7th International Conference on Collaborative Computing: Networking, Applications and Work-sharing, (CollaborateCom 11) (2011)
24. Spinellis, D.: Reliable identification of bounded-length viruses is NP-complete. *IEEE Trans. Inf. Theory* **49**(1), 280–284 (2003)

Bio-inspiring Cyber Security and Cloud Services: Trends
and Innovations

Hassanien, A.E.; Kim, T.-H.; Kacprzyk, J.; Awad, A.I. (Eds.)
2014, XXII, 585 p. 235 illus., 17 illus. in color., Hardcover
ISBN: 978-3-662-43615-8