25th Annual INCOSE International Symposium (IS2015)
Seattle, WA, July 13 – July 16, 2015

Adaptive Knowledge Encoding for Agile Cybersecurity Operations

Keith D. Willett, Rick Dove, and Mark Blackburn
Stevens Institute of Technology
Castle Point on Hudson, Hoboken, NJ 07030
kwillett@stevens.edu, rdove@stevens.edu, mblackbu@stevens.edu

Abstract. An agile cybersecurity operation is the dynamic adaptability of security services and mechanisms including people, process, technology, and environment to ensure organizational efficacy. Two key roles in agile security operations are cybersecurity operators and cybersecurity analysts. Both are overwhelmed with data and information, but underwhelmed in knowledge and understanding. Adaptive knowledge encoding introduces cybersecurity decision patterns (CDPs) and a cybersecurity decision pattern language (CDPL) as formal knowledge representation and a formal knowledge repository to capture, codify, and share knowledge that supports cybersecurity operators and analysts ability to perform timely agile cybersecurity operations. The Knowledge Engineer applies CDPs and the CDPL to provide a cybersecurity cognitive schema that dynamically adapts by assimilating new CDPs in the CDPL structure and acclimating the CDPL structure to new knowledge. CDPs and the CDPL together with applied fundamentals of agile systems engineering help facilitate the design and sustainment of agile cybersecurity operations.

Introduction

The system of interest for this paper is *cybersecurity operations*, which includes a dynamic interaction of people, process, technology, environment, results, consumption of results, and application of results to preserve organizational efficacy. Cybersecurity decision patterns (CDPs) and a cybersecurity decision pattern language (CDPL) capture explicit organizational knowledge, leverage subject matter expertise throughout the organization, and facilitate shared understanding and coordinated decision making; thus providing *adaptive knowledge encoding*. Where *design patterns* encode development knowledge, *decision patterns* encode operations knowledge. The uniqueness of this paper is to introduce the concepts of CDPs and the CDPL as adaptations of design patterns; and, to apply the fundamentals of agile systems engineering (Dove and LaBarge 2014) to show the application of CDPs and the CDPL in the design and sustainment of agile cybersecurity operations.

The INCOSE *A World in Motion – Systems Engineering Vision 2025* report describes cybersecurity, composable design, and decision support; plus, indirectly advocates knowledge management, knowledge representation, knowledge assembly, and knowledge delivery all as part of the future state of systems engineering ((INCOSE) 2014). The intent of CDPs and the CDPL is to provide composable cybersecurity decision support via adaptive knowledge management. The initial focus herein is on imminent use of CDPs and the CDPL for knowledge sharing among cybersecurity operators and analysts in support of incident response; i.e., people-to-people sharing. The future use for CDPs and the CDPL is to train artificial intelligence (AI) for machine enhanced cognition as an integral part of cybersecurity automation; the *Future:*

Cybersecurity Automation section provides a glimpse at this conceptual application. Part of this future glimpse includes the integration of cognitive systems engineering into systems engineering practices in order to produce a joint cognitive system (JCS) that provides for mutualistic symbiosis among humans and machines in cybersecurity operations. CDPs provide knowledge representation in this JCS and the CDPL provides a part of knowledge management.

Christopher Alexander described a pattern language as *the timeless way of building*. We should understand the *timeless way* as referring not to any specific pattern language but only to the underpinning conception of high-quality observation and design." (Hafiz and Johnson 2006) Perhaps we will eventually achieve a timeless way of incident response design, but we begin with achieving a more useful way to capture and share organizational knowledge on incident response. In security overall, there are recurrent practices that lend themselves to abstractions in order to capture and share knowledge of *best known applicable practices*. For example, securing the physical perimeter (Garcia 2007) may provide security patterns to raise awareness of campus security (e.g., fences, gates, cameras, lighting), building security (e.g., door locks, automated personnel entry systems), and room security (e.g., cipher locks). Likewise, there are cybersecurity recurrences of abstractions that lend themselves to codification (Blakley and Heath 2004). For example, identity and access management, secure communication, password strength, and securing data at rest. These abstractions can codify the concepts, but the specifics, especially for cybersecurity, have a limited useful life. As adversary capabilities advance, current safeguards become less effective and therefore must be replaced or upgraded. Likewise, new technologies may change paradigms thus causing a shift in the way we think about security (e.g., cloud security (Takahashi, Kadobayashi, and Fujiwara 2010)). The implications are to decouple technical specifics from the patterns and adapt the patterns to accommodate an ever-changing adversary and ever-changing technical environment. In other words, we need an agile cybersecurity approach.

The *CDPs and the CDPL in Operations* section elaborates on operational phases and a concept of operations (CONOPS). The interim sections present foundational details to justify the need for and influence the structures and content of CDPs and the CDPL.

Foundations

The foundations on which to build adaptive knowledge encoding for cybersecurity include the concepts of agility, agile-architecture, agile-systems, agile-security, agile-cybersecurity, risk dynamics, differentiating among data, information, knowledge, understanding, and wisdom (Ackoff 1989), and elaborating on the concepts around sharing knowledge and understanding. These all contribute to defining, elaborating, and applying a disciplined systems engineering approach to knowledge management as a contributing factor to agile cybersecurity operations.

Agility

Agility is "that characteristic which allows an organization to thrive in an environment of constant and unpredictable change." (Dove 1992) There is a difference between agile-systems engineering and agile systems-engineering (Haberfellner and De Weck 2005). The former addresses the process of engineering an agile system and the latter addresses an agile process of engineering a system. An agile system is flexible in its existing operating state as well as able to change operating states rapidly, and possesses such characteristics as being dynamic, adaptive, extensible, and scalable (Haberfellner and De Weck 2005). "Agile systems are defined in counterpoint to their operating environments. Words used to describe the general nature of the

target environment often include and combine dynamic, unpredictable, uncertain, risky, variable, and changing, with little attention to clear distinction among them." (Dove and LaBarge 2014)

Agile Cybersecurity Operations – Incident Response

Agile security is the well-coordinated ability to move quickly and easily in order to maintain an acceptable state of exposure to harm or danger. To be secure is not a goal, it is a state of being. The measure of being secure is not absolute, but is a factor of many continually changing influences including threats, assets, vulnerabilities, risk, and risk tolerance. Figure 1 shows a Systemigram (Boardman and Sauser 2008) of risk dynamics. The *workflow* represents an interaction of *assets* to achieve a desired effect (e.g., fulfill the organizational mission). Any potential disruption to workflow is a *threat*. Any weakness within workflow is *vulnerability*. The potential of a threat to exploit a vulnerability that brings about loss or harm is a *risk*. The *risk posture* is an intentionally assumed position to address all risk in terms of accept, share, transfer, or mitigate. The *desired security posture* is the intentionally assumed position to employ security services and mechanisms that share, transfer, and mitigate the risks in the risk posture.

Risk dynamics include agile cybersecurity in part represented by the continuous monitoring of the asset space to obtain periodic snapshots of operational states that represent the *current security posture*. A comparison of the current security posture against the *desired security posture* provides input to a gap analysis that result in either 100% alignment or a list of gaps that drive corrective action limited by available resources. Part of continuous monitoring includes the detection of events that may indicate cybersecurity anomalies that require incident response.

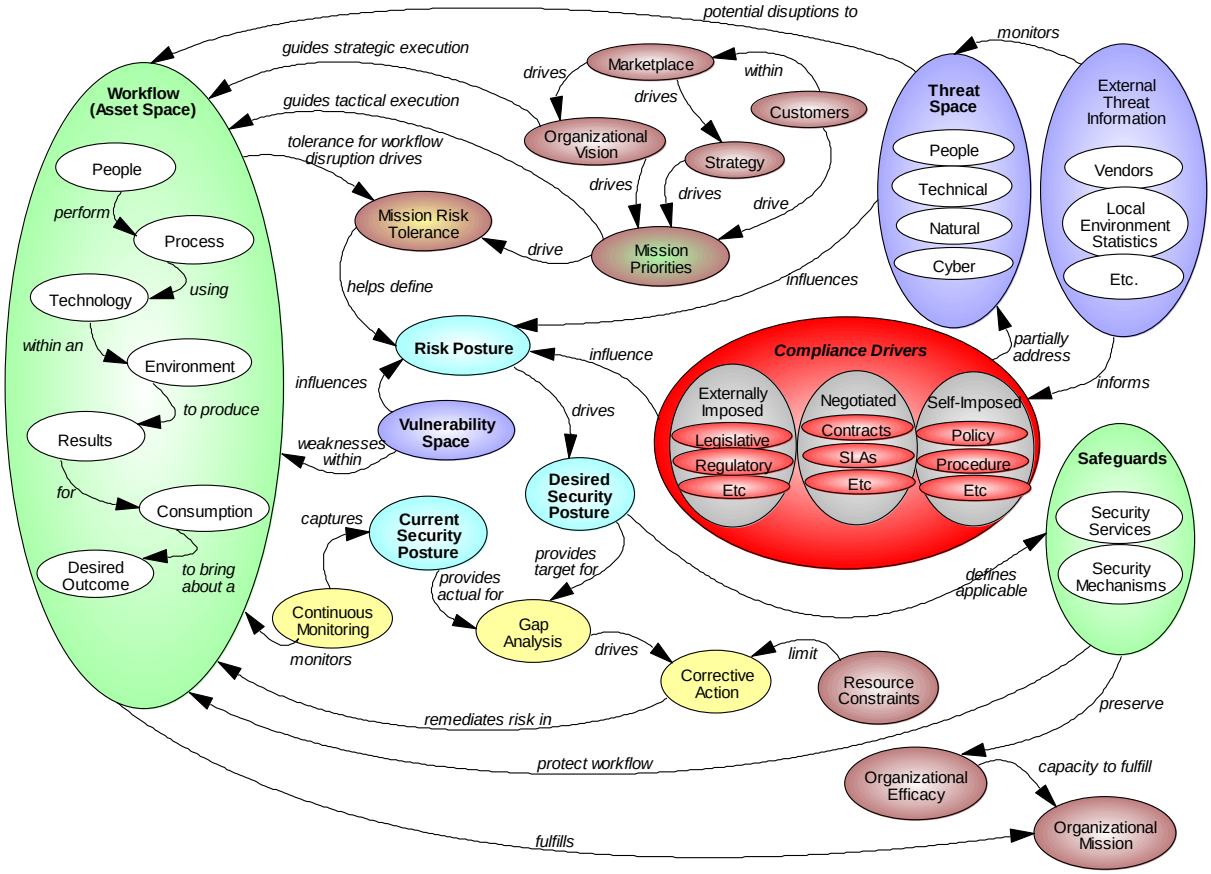

Figure 1. Risk Dynamics

Cybersecurity operator and analyst daily tasks are partially judged within non-static boundaries of acceptable *accuracy* and *timeliness* of incident response (IR). CDPs and the CDPL measures of effectiveness are in the increase of accuracy by virtue of encoding and sharing knowledge and in the reduction of time within various phases of the incident response by virtue of providing OODA-support (Figure 2).

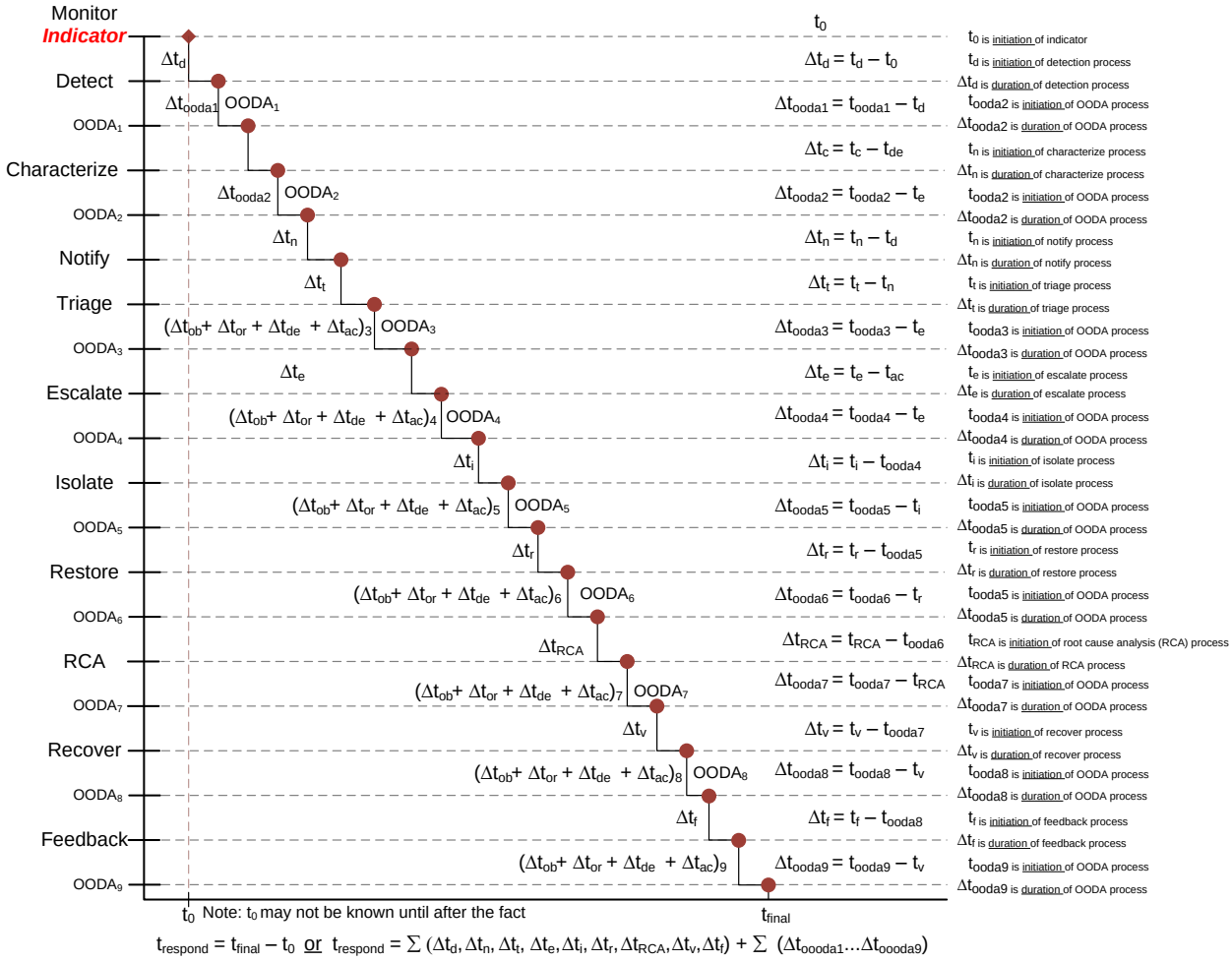

Figure 2. Incident Response Time Metrics

Encoded knowledge has dated relevance by the nature of ever-changing circumstances that surround the application of that knowledge. Adaptive knowledge encoding considers and captures the implications of these influences in a cycle of continual refinement to keep knowledge relevant. From a cybersecurity operator perspective, CDPs and the CDPL apply during continuous monitoring (observe), gap analysis (orient, understand), and corrective action selection (decide). From a cybersecurity analyst perspective, CDPs and the CDPL capture threat, vulnerability, security, and risk details that go into malware analysis and remediation.

Sharing

Risk dynamics entail many functional exchanges to share details among many entities. To provide a greater context and narrow the scope and focus of CDPs and the CDPL, let us examine who and what shares (i.e., the sharing entities), what is shared and why, and, the methods for sharing. In cyberspace, the sharing entities and their relationships are *people-to-people*, *people-*

to-machine, and *machine-to-machine*. The sharing entities may share details that include *data*, *information*, *knowledge*, *understanding*, and *wisdom* for purposes of *awareness exchange*, *content exchange*, or *command and control exchange*. Two basic methods of sharing are *synchronous* and *asynchronous*.

To elaborate on the distinctions among data, information, knowledge, understanding, and wisdom, data may take the forms of *native data*: data that resides on a cyberspace asset, *raw data*: data collected into some sensor subsystem, or *refined data*: data aggregated into a common tool or repository in a purposeful format. Information may take the forms of *raw information*: compiled narrative in a general context (e.g., an economic forecast for the United States for 2016), or *refined information*: compiled narrative in a specific context (e.g., the same economic forecast applied to company ABC for 2016 strategic and tactical planning). Knowledge is structured details that capture context, problem, and solution. Understanding captures relationships among knowledge and essence of wisdom is the anticipation of consequences (Cousins 1978).

Data is raw material that may or may not be facts; i.e., false data is still data, but not a fact. Data may be structured, but without context it lacks meaning. Information is unstructured narrative that provides some context and meaning for data. Sharing information occurs regularly in reports, journal papers, magazine articles, and other similar artifacts. Both data and information require time and energy to read, understand, and interpret to solve problems. Knowledge representation captures and conveys a *context*, a *problem*, and a *solution*; in this sense, knowledge is immediately actionable if found to be applicable to the situation at hand. *Understanding* encompasses the perception of meaning and the inference of meaning; *wisdom* focuses on judgment in some evaluation of data, information, knowledge, or understanding. Data has a low level of meaning and hence a relatively low value, where knowledge has a high level of meaning and hence a relatively high value. (Rowley 2007)

In the context of cybersecurity operations, the terms data and information are often used interchangeably. Most cybersecurity operators and analysts are overwhelmed with data from their own environments. Sharing more data just adds to the burden. Sharing information is somewhat more helpful, but still requires reading a narrative to discern relevance and application to the local environment. The formal encoding of knowledge that captures context, problem, and solution adds value by leveraging lessons learned by one among many. Formal knowledge representation, knowledge repository, and knowledge management is the essence of the learning organization. Sharing knowledge is of greater imminent-application value than sharing data or information.

Shared understanding comes from establishing and sustaining a shared cognitive schema that organizes knowledge and knowledge relationships. In this case, the CDPL represents that shared cognitive schema. Anticipating consequences requires predictive analytics that look ahead to the implications of a decision and subsequent action to examine potentials for desired/undesired outcomes and expected/unexpected consequences. The formalization of knowledge in CDPs and a CDPL may provide the foundation for future artificial intelligent system (e.g., expert system) training and mining to facilitate and share the *anticipation of consequences*.

The reasons for sharing details among people include the provision of situational awareness, individual and shared understanding, individual decision-making, coordinated distributed decision-making, individual actions, coordinated distributed actions, and command and control.

More than just decision support, these details also provide support for awareness, understanding, and action.

OODA-Support (Beyond Just Decision Support)

Three words may summarize the nature of reality: *choice*, *chance*, and *certainty*. Certainty results from our understanding of the sciences (e.g., chemistry, biology, cognitive, physics); certainty is predictable. Chance is randomness among unknown or knowable possibilities with varying degrees of confidence as determined from the application of analytics (e.g., probability, statistics, modeling, simulation). Choice is core to people and begs for support in becoming *aware* of events, to *understand* events and the influences of other inputs from chance and certainty, to identify and select among viable options, and to carry out actions as a result of the decision. In other words, there is need for support in terms of *observe*, *orient*, *decide*, and *act* (John Boyd's OODA loop) (Osinga 2005). For cybersecurity operators and cybersecurity analysts, *observation support* helps them become more aware; *orientation support* helps them better understand; *decision support* helps them identify and select among viable options; and *action support* provides action proxies to relieve them from the burden of rote activities (e.g., automated courses of action).

CDPs and the CDPL provide one medium for observation, orientation, and decision support. They help observation by codifying observables in the CDPs that guide what to look for in operations. They help with understanding by virtue of the CDPL representing a cybersecurity cognitive schema that includes knowledge relationships. The structure of the cognitive schema facilitates assimilating new knowledge into the existing schema and acclimating the schema to new knowledge. This ongoing dynamic adaptation to unpredictable change makes CDPs and the CDPL an agile security-engineering construct to design and sustain agile cybersecurity operations.

Unpredictable, Uncertain, Risk, and Variation Framework Extension (UURV++). The UURV++ framework provides terms of nuance that help in being precise in OODA-support. The terms clarify the OODA-support intent and actual ability. The nature of being *agile* accommodates unpredictable, persistent change; however, we run into reality that is unpredictable and uncertain as well as predictable and certain. We run into reality that repeats itself exactly as well as repeats itself in variations. We also run into reality that may be characterized by risk. We need to accommodate the nature of reality in terms of choice, chance, and certainty. In terms of agile systems, they have effective situational response options, within mission, under the following UURV framework (Dove and LaBarge 2014):

- **Unpredictable**: randomness among unknowable possibilities
- **Uncertain**: randomness among known possibilities with unknowable probabilities
- **Risk**: randomness among known possibilities with knowable probabilities
- **Variation**: randomness among knowable variables and knowable variance ranges

We may extend the UURV framework (UURV++[1]) to include the following:

- **Predictable**: lack of randomness among knowable possibilities
- **Certainty**: lack of randomness among known or knowable possibilities with knowable probabilities

[1] Allusion to the C program language unary operator that increments by one

- **Chance**: randomness among unknown or knowable possibilities with varying degrees of confidence
- **Choice**: the act of deciding and selecting among options

The CDPs and the CDPL use the UURV++ framework to characterize knowledge in terms as close to reality as possible to facilitate practical application by cybersecurity operators and analysts. With respect to anomalies, known-knowns are predictable and have a degree of certainty on how to address them. Known-unknowns have been encountered before but we have yet to determine specifics on what to do about them, therefore they have a degree of both certainty and uncertainty. Unknown-unknowns are more ones of chance than predictable or certain; e.g., zero-day attacks. Unknown-known is a posteriori conclusion of a mischaracterization where variation may have interfered with accurate identification. All possess a degree of risk, a degree of chance, and result in the need for choice by cybersecurity operators and analysts who must deal with anomalies.

Knowledge Management and Knowledge Engineering

Knowledge management is the process of creating value from an organization's intellectual capital including human capital, structural capital, environmental capital, and customer or relationship capital (Liebowitz 2001). Knowledge engineering is the process to capture, represent, encode, and test and evaluate expert knowledge (Liebowitz 2001). The definition of knowledge is "facts, information, and skills acquired by a person through experience or education; the theoretical or practical understanding of a subject."[2] Focusing on knowledge as practical understanding includes the ability to recognize the situation for knowledge application, recognize the need for knowledge application, and identify the knowledge actually to apply.

Figure 3 presents a view of knowledge scales (Hughes 2006) with the addition of knowledge types along a loose analog scale of tacit to explicit. Individuals, groups (teams), or the organization may retain knowledge. Organizational knowledge is more durable by the nature of it being more tangible and more explicit. Knowledge retained only by individuals is lost when the individual leaves the organization and thus less durable than organizational knowledge. Moreover, individual knowledge tends to be tacit in the form of heuristics and personal mental models. The scope of knowledge represents how much is known on a particular topic and tends to be broader for individual knowledge and narrower for organizational knowledge.

[2] http://www.oxforddictionaries.com/us/definition/american_english/knowledge, last accessed 8-Sep-2014

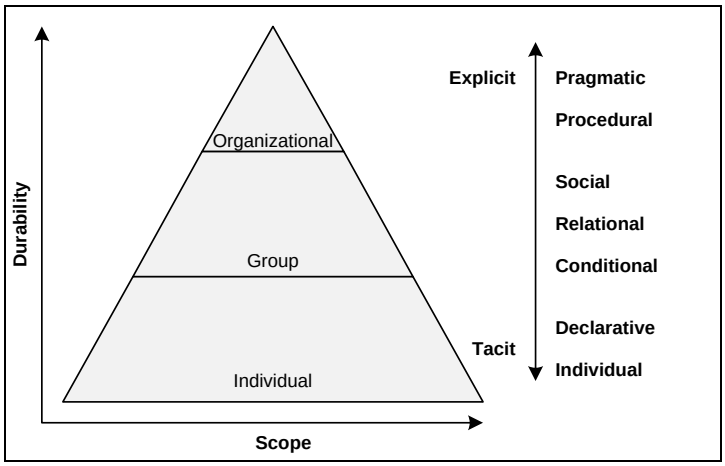

Figure 3. Scales of Knowledge and Knowledge Types

Table 1 (Kesh and Ratnasingam 2007) elaborates on the knowledge types to the right side of Figure 3. The primary goal for CDPs and the CDPL is to create explicit organizational knowledge. This includes capturing and codifying individual knowledge and converting it from tacit to explicit in the forms of relational, procedural, and pragmatic knowledge. CDPs and the CDPL also capture group knowledge and existing organizational knowledge in a standard format to facilitate knowledge sharing and reuse. The knowledge engineer role elicits details from all types of knowledge to create and maintain CDPs and the CDPL.

Table 1: Knowledge Types for Cybersecurity Operations

Knowledge Type	Short Description	Comments and Cybersecurity Operations Example
Pragmatic	Useful organizational knowledge	Explicit representation of best practices; e.g., codified organizational cybersecurity incident response details as derived from actual experience.
Procedural	Know-how	Explicit. Codified knowledge on security procedures; e.g., how to implement a firewall.
Social	Created by the group	Collective planning and actions; e.g., specific security mechanism acquisition, implementation, and deployment.
Relational	Know-with; interrelations	Knowledge of people, processes, and technologies interactions to achieve the desired effect; e.g., cybersecurity incident response workflow that engages multiple individuals, workgroups, and technologies. Segue from knowledge to understanding.
Conditional	Know-when	Knowledge of appropriate time to employ particular security services and mechanisms. Essence of balancing risk mitigation with resource constraints; e.g., when to add behavior-based or content-based security monitoring to signature-based.
Declarative	Know-about	Awareness of Risk Posture and associated security services and mechanisms that comprise the Security Posture (Figure 1); e.g. existence of anti-malware software or intrusion detection system.
Individual	Created by the person	More tacit. Cybersecurity operator and analyst individual practices to facilitate operations, identify malware, isolate malware, and remove it.

Cybersecurity Decision Patterns and Cybersecurity Decision Pattern Language

Christopher Alexander introduced the concept of design patterns for physical structure architecture in his landmark book entitled A Pattern Language (Alexander, Ishikawa, and Silverstein 1977). Design patterns encode design knowledge. CDPs encode cybersecurity operations knowledge in the form of problem, solution, and context. The CDPL provides an organizing repository for the CDPs and captures pattern interrelationships to facilitate a user's ability search, find, retrieve, determine applicability, and apply the appropriate CDP(s). The operational workflow facilitates a dynamic feedback loop that provides results and lessons learned (i.e., success, failure, and the degree/specifics of both) from application for CDP and CDPL adaptation.

Cybersecurity Decision Pattern Structure and Content

Table 2 provides a partial CDP template derived from the three sources noted plus critical thought and imagination on what constitutes the essential elements of a CDP. The details of the CDP template are skewed toward incident response with the intent of producing CDPs useful for decision support in real-time cybersecurity operations. The initial application of CDPs is for formal knowledge encoding, knowledge sharing, and adapting knowledge representation and the knowledge repository according to the ongoing acquisition of new knowledge and new understanding. The Annex contains an example CDP.

Table 2: Partial Cybersecurity Decision Pattern Template (Meszaros and Doble 1998, Lowe 2006, Appleton 1997)

	Description
Name	Evocative name that emerges from natural language to reference this problem/solution pairing.

Context	The situation, the circumstances in which the problem is solved. The context imposes constraints and helps identify the relative importance of the forces. The context may include tactical and strategic perspectives, for example:
	• **Strategic**
	  ○ **Organization**: <organization identifier> (e.g., accounting)
	  ○ **Strategic Function**: <function> (e.g., collections workflow management)
	  ○ **Capability**: <details> (i.e., description of desired results)
	  ○ **Activity**: <details> (i.e., formal collection of activities producing desired results; may use the generic workflow as a standard way to represent activity)
	  ○ **Task**: <details>
	• **Tactical** (**design note**: we do <u>not</u> want to turn the pattern repository into a ticketing system, the details below are too detailed for a pattern. They have their place and may be referenced via some ticket # in the Examples essential element, but they belong in a complementary, separate system)
	  ○ Physical location(s): <details>
	  ○ Network identification: <details>
	  ○ Tool: <details>

Problem	The specific problem that needs to be solved. Describe the problem, the root need as a coarse abstraction and the specific need in less coarse terms. The problem describes the *what* and should not include the solution (the *how*). At the least, the problem description should include: • The root of the problem is… <the root of the cause… may be an archetype> to help frame the problem and to identify existing approaches to existing archetype problems • The desired result is… < express desired result agnostic of solution that produces the result> Spotting the problem: • Observables (indirect, symptoms, indicators): <details>; • Observables (direct, problem source): <details>

	Description
Solution	Describe *how* to solve the problem; how to produce the desired result expressed in the problem. There may be multiple potential solutions; the best is relevant to the context and resolves the highest priority forces. The solution description may read like an instruction/imperative. Notional solution structure: • **Monitor**: for <details> • **Detect**: observable <details> • **Notify**: who <details> according to detect details • **Triage**: priority <details> according to detect details and tactical and strategic mission • **Escalate**: to level of expertise <details> per detect and triage details • **Isolate**: containment <details> according to tactical and strategic details • **Restore**: achieve interim operations <details> according to tactical and strategic mission • **Root Cause Analysis (RCA)**: <details; or, explicit reference to external report> • **Recover**: achieve normal operations <details> • **Feedback**: systemic feedback <details>; CDP feedback <details>

Decision Patterns v. Design Patterns. The Security Pattern Catalog[3] claims to contain all security [design] patterns written by all security experts since 1997 and currently contains 97 security [design] patterns. Design patterns capture design knowledge to facilitate development activities; decision patterns capture decision knowledge to facilitate operational activities. Example security design patterns include *Authentication Enforcer* and *Authorization Enforcer*; these describe what to do during design and development. Related CDPs may include *False Claim of Identity* and *False Claim of Privilege*; these describe what to look for and how to respond in operations.

Invariance. All actual patterns are not arbitrary design ideas, but rather emerge from observation (Alexander 1977); i.e., actual patterns are mined from real experiences. All patterns start with a notional idea, a concept for a particular pattern. This level of maturity is a *candidate pattern*. Upon observations from experience, a candidate pattern matures to become a *proto-pattern* (Appleton 1997). If at least three instances can be found in experience, a proto-pattern is on its way to being considered as an *actual pattern* (Appleton 1997); the *rule of three* represents the minimum level of recurrence for a pattern to be labeled *invariant*. The label of invariant is necessary for a pattern to be considered an actual pattern. Subsequent observations of a fourth or a thousandth occurrence do not guarantee invariance, rather they strongly suggest invariance and broad pattern applicability to the field (Lowe 2006).

Cybersecurity Decision Pattern Language Structure and Content

The CDPL facilitates the derivation of meaning; it facilitates the process of understanding by housing knowledge and capturing knowledge relationships. The role, fit, and function of the CDPL is to facilitate the ability to categorize and bin CDPs, search and find them, determine their relevance, and facilitate their application to the current situation. Part of the CDPL operational process is to be adaptive to new user needs and user feedback on the effectiveness and efficiency of its ability to fulfill role, fit, and function.

A pattern language codifies the interaction of human beings with their environment (Salingaros 2014). The CDPL is a framework to capture quality observation, orientation, and decisions within the domain of cybersecurity operations, with specific initial focus on incident response.

[3] http://www.munawarhafiz.com/securitypatterncatalog/, last accessed 8-Sep-2014

CDPs emerge from observation and each CDP has a well-defined place in the overall network of patterns, the collection of which constitutes the CDPL which itself is a vocabulary of decisions in both words and mental images relevant to practical action (Ulrich 2006). The CDPL does not provide rote answers, rather it contains raw material from which to both find answers and create answers according to the current need, circumstances, and environment. The CDPL structure emerges from the CDPs and their interrelationships; plus, from CDPL design principles, application in operations, and outcomes. There is no definitive, preconceived CDPL structure. However, rather than start with a blank page and hope for the best, the following section provides a notional CDPL structure with which to focus the initial compilation of CDPs.

Notional CDPL Structure. A key question to determine CDPL structure is how the prospective users of the CDPL encounter phenomena that prompt the use of the CDPL and how users most naturally capture details for encoding and subsequent sharing. In a general sense, cybersecurity operators and analysts are working in a situation that provides context (e.g., incident response); they encounter details in the form of a problem; and they find a solution that at least works if not one that is immediately optimal. Therefore, the incident response ontology of monitor, detect, notify, triage, escalate, isolate, restore, root cause analysis, recover, organizational feedback provides a notional structure within which to capture CDPs (Table 3).

Table 3: Notional CDPL Structure and CDP Alignment Guidance

IR Phase	Problem	**Helps Answer Operational Questions (Knowledge Types from Table 1)**
Monitor	Via situational awareness, I heard to expect something…	• When should I expect it? (conditional) • How do I keep it out? (procedural) • How do I find it? (procedural)
Detect	I see something…	• What is it? (declarative) • What does it do? (declarative) • How does it do it? (declarative) • Where does it come from? (declarative, relational)
Notify	I need to raise awareness…	• Who needs to know? (declarative) • What do they need to know? (declarative) • How do I notify them? (procedural)
Triage	I have something in hand…	• How do I determine the priorities? (procedural) • What incident is most critical to address first? (relational) • How do I determine the effects to the tactical mission? (relational, procedural) • How do I determine the effects to the strategic mission? (relational, procedural)
Escalate	I need to engage the appropriate expertise…	• Have we seen this before and know what to do about it (i.e., known knowns)? (declarative) • Have we seen this before and still not characterized it (i.e., known unknown)? (declarative) • Have we not seen this before (i.e., unknown unknown)? (declarative) • Posteriori, did we see it before but failed to characterize it correctly (unknown known)? (declarative)
Isolate	I need to stop it from proliferating… I need to stop its effects from spreading…	• How do I contain it? (procedural) • How do I contain its effects? (procedural)
Restore	I need to continue	• What is the tactical implication to mission? (declarative)

IR Phase	Problem	**Helps Answer Operational Questions (Knowledge Types from Table 1)**
	operations… I need to continue to fulfill [tactical | strategic] mission…	• What is the strategic implication to mission? (declarative) • How do I continue the mission? How do I fight through the attack? (procedural)
Root cause analysis	I need to find the root problem… I need to find the root cause… I need to define the root cause	• What is the root cause? (relational) • How do I get rid of it? (procedural) • How do I reduce the probability of recurrence? (procedural) • How do I stop it from happening again? (procedural)
Recover	I need to resume normal operations…	• How do I get rid of it? (procedural) • How do I modify the operating environment to accommodate knowledge of what I found and what to do about it? (procedural)
Organizational Feedback	I need to disseminate incident details and lessons learned to others…	• How can I capture and encode incident details, the problem, and the solution? (procedural) • How do I provide details to the organization for preventive or preemptive activity to minimize recurrence? (procedural)

Given that the CDPL intends to capture patterns for decision making in cybersecurity operations, alternative or complementary candidate influences for organizing patterns also include cybersecurity (e.g., threats, assets, vulnerabilities, risk, security services and mechanisms), decision making (e.g., OODA loop), and operations (e.g., generic workflow (Figure 1). Other contending or complementary structures to classify CDPs in the CDPL include security principles of confidentiality, integrity, availability, possession, utility, authenticity, privacy, non-repudiation, and authorized use (Willett 2008). Additionally, the general attack goal of *bring about a desired effect* via three objectives of *get-in*, *stay-in*, and *act*; and general defense goal of *minimize threat efficacy* via three objectives of *keep-out*, *throw-out*, and *restrain* are potential influences on CDPL structure. The STRIDE threat model (**S**poofing, **T**ampering, **R**epudiation, **I**nformation disclosure, **D**enial of service, **E**levation of privilege) (Hafiz and Johnson 2006) provides another potential organizing structure. At this point, all structures are open to consideration to help guide the emergence of the CDPL.

The knowledge types in Table 3 do not include individual, social, or pragmatic from Table 1; these are more knowledge sources and goals than knowledge types with respect to the CDPL. The job of the knowledge engineer is to elicit tacit and explicit knowledge from the individual and group and create explicit pragmatic knowledge that addresses the questions in Table 3 by providing specific answers.

Agile-Architecture for Adaptive Knowledge Encoding

Similar to agile-systems engineering and agile systems-engineering, there is agile-architecture design and agile architecture-design; the former represents a deliverable and the latter a process to develop that deliverable. An agile-architecture pattern provides guidance for designing and sustaining multiple instantiations using a common set of three elements and four activities (Dove and LaBarge 2014). The three elements are *modules* which are discrete standalone capabilities; *passive infrastructure* that provides plug-and-play connectivity for modules; and, *active infrastructure* that facilitates the performance of the activities to adapt the overall system to

ongoing change, both predictable and unpredictable. The four activities are *module evolution* to upgrade existing modules, add new modules, and remove old modules; *module readiness* to ensure module availability for assembly at need; *system assembly* to develop an instantiation; and *infrastructure evolution* to upgrade, add, and remove active and passive infrastructure according to changing rules and standards.

Figure 4 presents an agile-architecture pattern (Dove 2010, Dove and LaBarge 2014) for adaptive knowledge encoding to achieve (design) and sustain (operate and maintain) CDPs and the CDPL in day-to-day ongoing cybersecurity operations. The actual CDP classification scheme is yet to be determined; therefore, the Figure shows ten notional classes of modules (Table 3) that help organize the CDPs. The knowledge engineer is predominantly responsible for CDP evolution and CDP readiness. CDPs are modules stored in the CDPL as the active infrastructure. The users of CDPs, the incident response team initially, assemble the CDPs as befits the current incident response situation in order to obtain OODA-support. CDPs interconnect via the CDPL structure, plus via ad hoc interconnections. The passive infrastructure addresses plug-and-play interconnection rules and standards. Instantiation options depict three assembly variations of *single-CDP fixed response*, *multiple-CDP fixed response*, and *ad hoc adaptive response*. As implied by the name, the instantiations employ a single applicable CDP, multiple applicable CDPs, and dynamic identification and application of one or more CDPs respectively. The chief operations engineer is accountable to evolve the infrastructure that includes acclimating the CDPL structure to new knowledge and knowledge relationships, plus accommodating changes to rules and standards.

The generic passive infrastructure of sockets, signals, security, safety, and service correspond to CPDL specifics as shown in Figure 4. CDPL pattern interconnection includes knowledge management and pattern structure standards. CDPL communications protocols include messaging standards and interface standards to facilitate communications among incident response personnel and tools with the CDPL. CDPL security applies standards to ensure CDPL does not introduce vulnerability into cyberspace operations. To guard against inappropriate use, CDPL combinatorial requirements provide rules on which CDPs may and may not be combined. The CONOPS provides a user perspective that evolves with understanding to ensure effective and efficient understanding and application of CDPs and the CDPL. Pattern modules at the top of Figure 4 can be dragged and dropped into decision response assemblies enabled by passive infrastructure interface rules; the active infrastructure designates parties responsible for constant evolution of modules and infrastructure, and for assembly of decision responses.

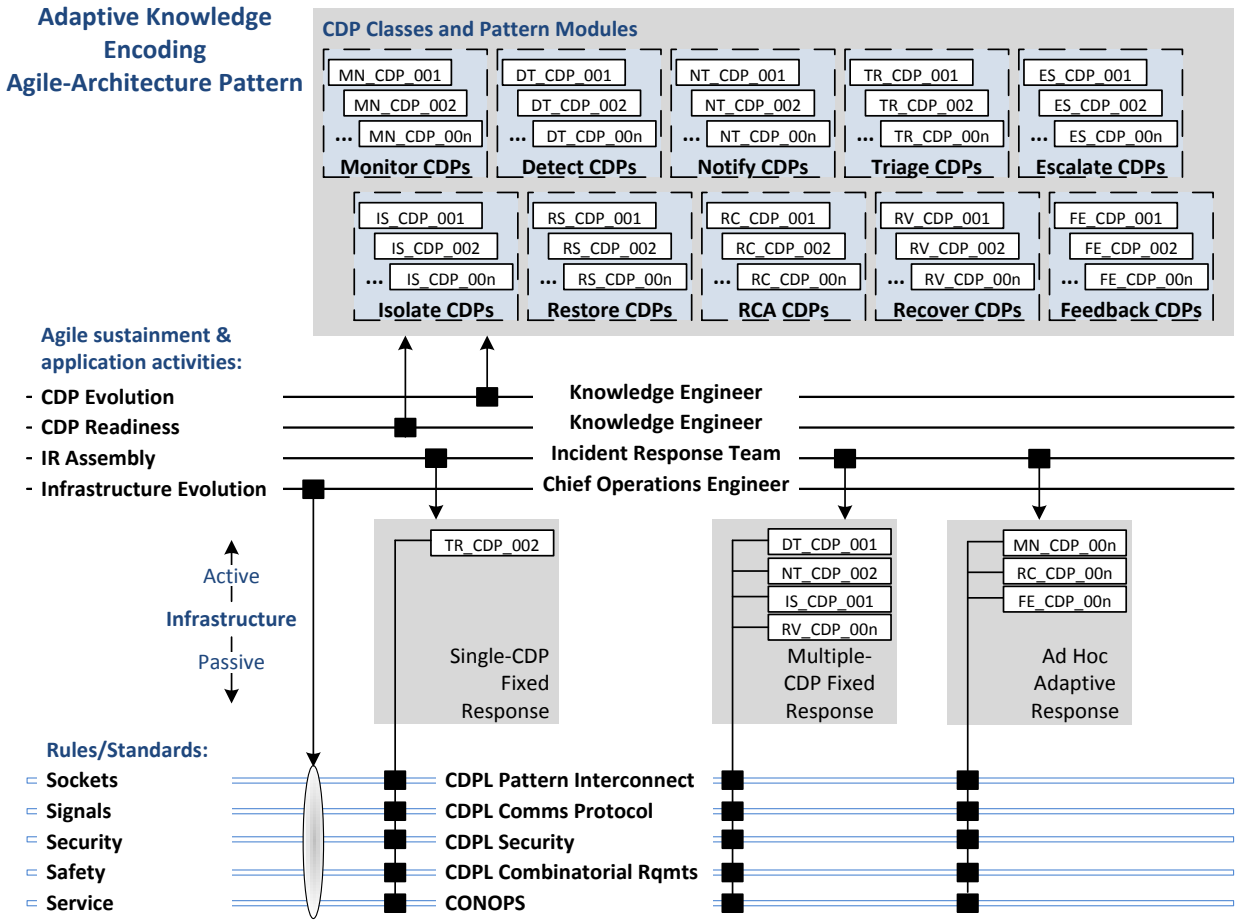

Figure 4: Adaptive Knowledge Encoding Agile-Architecture Pattern

Capabilities are *expressions* of desired results where *solutions* produce those results. The modules are in terms of *capabilities*. External to the agile-architecture pattern is the relationship of one or more solutions that produce the desired results. The point is not to assemble a solution based on available tools, but to assemble a solution based on the desired result (i.e., mission need) as expressed in the capabilities and then associate the appropriate tools according to many drivers and constraints; e.g., availability, budget, scale, existing contract vehicles, desire for a homogeneous or heterogeneous operating environment, etc.

CDPs and the CDPL in Operations

To convey the application of CDPs and the CDPL in operations, the sections below provide operational phases and a CONOPS that covers several short use cases. The Annex provides an example CDP that includes an expanded version of the template and notional details.

Operational Phases

There is a difference between establishing adaptive knowledge encoding and sustaining adaptive knowledge encoding for day-to-day ongoing use in cybersecurity operations. The workflow to establish and sustain CDPs and the CDPL includes operational phases of *create*, *stage*, *assemble*, *post-implementation evaluation*, and *evolve*. The key roles in CDP operations are the *knowledge engineer* to perform disciplined management of CDPs and the CDPL, the *cybersecurity operator* to facilitate secure workflow, and the *cybersecurity analyst* to provide deep analytical insight

into malware and other aspects of threats, assets, vulnerabilities, risk, and security services and mechanisms in support of cybersecurity operations. The sections below address the perspectives, tasks, interrelationships, and interactions with CDPs and the CDPL within the context of the cybersecurity workflow.

Create CDPs and the CDPL. *Mining* patterns is deriving them from real world experience by abstracting specific occurrences of successful operations (Lowe 2006). *Creating* is the application of inventive thought to produce from theoretical knowledge. By definition, a pattern is an abstraction of real world successes; therefore, all patterns must be mined and not created. However, the creative thought process still plays a role in pattern development. The application of intuition and experience contribute details for the pattern abstraction not directly based on observation, but based on supposition with a level of confidence in their eventual use and practicality. While creativity has a role, no pattern or details of a pattern will be labeled an *actual pattern* without satisfying the minimum level of invariance.

To create CDPs, the knowledge engineer facilitates the elicitation indirectly or directly elicits knowledge from the cybersecurity operators and analysts and then encodes and stores the details in CDPs. The knowledge engineer also uses vicarious sources like the national computer security incident response teams (CSIRTs) publications and regular updates, input from partnerships/alliances with other organizations; plus industry publications like the annual Verizon Data Breach Investigations Report. The knowledge engineer ensures normalized representation and adapts the CDPs according to new experience to ensure the most up to date organizational knowledge is available to cybersecurity operators and analysts. The knowledge engineer then stores the CDPs in the CDPL and creates or adapts associations with other CDPs to facilitate the emergence and maturity of understanding through knowledge relationships.

Stage. Staging refers to the knowledge engineer performing module readiness (Figure 4) which is preparing CDPs and the CDPL for access and assembly by cybersecurity operators and analysts. If there is a single CDPL and a single place where it resides, then staging is inherent in creation and evolution. Otherwise, staging includes duplicating the CDPL in other areas to maintain multiple copies for more efficient access, and/or synchronizing multiple independent CDPLs into a standardized collection of CDPs via replication. The outcome of staging is to have the latest relevant CDPL and CDPs available for assembly.

Assemble. In Figure 4, system assembly occurs when cybersecurity operators and analysts search the CDPL to find relevant CDPs, determine their applicability, and apply the CDPs to the current incident. Table 3 provides a list of relevant applications for CDPs and the CDPL in operations and analytics to support operations, specifically cybersecurity incident response. The three current resulting options of assembly are single-CDP fixed response, multiple-CDP fixed response, and ad hoc adaptive response. Maturity of the CDPs may refine or add assembly categories.

Post-Implementation Evaluation. When applying design patterns to physical structure design and development, a post-occupancy evaluation (POE) surveys occupants of a building in order to objectively assess the building's livability (Lowe 2006). A similar assessment may occur with CDPs and the CDPL by inserting a quality feedback process as part of the incident response workflow. The post-implementation evaluation (PIE) assesses whether the CDPL is searchable, readable, useable, useful, and accurate; and, assesses whether CDPs are findable, applicable, accurate, and complete enough for practical application. Also, the PIE may assess the cognitive

workload required to use the CDPs, which is a measure of the level of effort for using the CDPs; e.g., such an assessment may adapt the subjective workload assessment technique or Cooper-Harper scale (Warr, Cole, and Reid 1986). The combination of performance level and level of effort provides a foundation for a cost/benefit analysis to show the value (or not) of CDPs and the CDPL and helps to manage their application in operations.

Evolve. To evolve the modules and infrastructure, the knowledge engineer adapts CDPs and the CDPL to new knowledge based on elicitation of details from cybersecurity operators and analysts. Adaptation of CDPs includes adding new CDPs, adding new details to existing CDPs, and modifying/deleting existing details in CDPs to accommodate new knowledge and refined knowledge according to new experiences. Adaptation of the CDPL *assimilates* new knowledge into the CDPL structure that includes adding, refining, modifying, or deleting CDP relationships that facilitate understanding; and, *acclimating* new knowledge by modifying the CDPL structure.

Concept of Operations

The CONOPS walks through several use cases of CDPs and the CDPL providing OODA-support to cybersecurity operators and analysts in the context of the incident response phases introduced in Table 3. Assume the cyberspace environment has an array of security services and mechanisms deployed in a defense-in-depth and defense-in-breadth design and the quality of the design, configuration, and operation is adequate to perform the tasks described. Further, assume there is an existing set of CDPs in a CDPL and that the knowledge engineers, operators, and analysts are trained and sufficiently familiar with CDPs and the CDPL including lexicon and application method to perform the tasks.

Foundational to all incident response is monitoring. Cyberspace monitoring devices (e.g., intrusion detection, anti-malware scanners, and firewalls) are configured to look for known or anticipated anomalies. The CDP *observables* provide influence to monitoring configuration; i.e., CDPs codify *things to look for*. Observables may be indirect symptoms or indicators, or may directly identify the problem source. One benefit to CDPs is to formally capture and evolve details to get smarter over time in refining monitoring device configuration.

Use case one is an example of encountering a known-known where an intrusion detection system monitoring the cyberspace environment detects an anomaly and notifies a cybersecurity operator with details of system identification, date, time, nature of detection (e.g., signature, behavior, content), and anomaly type (e.g., configuration variation, unexpected user presence, or unexpected data transfer). In this case, the observed details are a user id active at 2:00AM Sunday morning that is usually active Monday through Friday between 7:00AM and 5:00PM. Upon notification, the operator normalizes details in terms of the CDP structure and content, specifically in terms of the observables. The operator enters the details on a CDPL interface screen; this becomes increasingly automated with maturity of the CDPL and its interfaces to the cybersecurity assets. The CDPL then returns any results from previously encountered CDPs of similar observables and prompts for further known details to help narrow the search. This interaction facilitates the cognitive process of the operator by quickly narrowing focus to missing relevant details. The operator no longer faces an unbounded set of options, but a narrow set of options directly related to the current observables. The recommended CDP with the highest level of confidence contains problem investigation and solution details from previous experience. The CDP solution steps suggest calling the Operations Manager, who then verifies that user was performing authorized work at that time. The CDP assisted in avoiding the time and resources

put into chasing a false positive; and produced quick resolution, thus minimizing the cognitive burden on the cybersecurity operator. This is an example of a single-CDP fixed response.

Use case two is an example of a partial known that starts with malware detection on a user desktop computer via malware signature scan. Upon notification, the cybersecurity operator normalizes details to help identify and assemble the appropriate CDPs. Given the signature, the CDPL returns a CDP that contains isolation and removal steps in the CDP solution details. There is no automated remediation in this case; however, previous research results by a cybersecurity analyst were captured in the CDP and provide the manual steps for the operator to treat the symptoms. A complementary CDP returns details on a recent phishing campaign that uses the identified malware. The solution steps include a review of the e-mail logs and anti-malware logs that yield details on fifty phishing e-mails found, forty-eight blocked, and two passed through; subsequent investigation finds one was deleted and the other opened on the machine reported in the anomaly details. The CDPs help treat the symptoms and provide guidance to find the root cause. The operations manager schedules a meeting with the anti-malware vendor to discuss why two e-mails were allowed through and uses the CDP details to discuss adding automated malware removal in a future release. This is an example of a multiple-CDP fixed response.

Use case three addresses a known-unknown that starts with a potential anomaly detection of restricted data traversing an outbound session to the Internet. Upon notification, the operator normalizes the details to facilitate CDPL search and finds results. Operations policy characterizes the nature of the anomaly as high priority and preliminary details identify Cybersecurity Analyst SME Group A as the escalation point. Upon preliminary investigation, the analyst assigned to work the anomaly concludes she has seen this before, but has yet to characterize the symptoms adequately, define the problem, or find the root cause. Having recently read a SysAdmin, Audit, Networking, and Security (SANS) report on Skype and data exfiltration, she identifies characteristics of this anomaly that look similar. Details from the SANS report provided clues to indicators on Microsoft Windows computers, the same as the initiating server of the anomalous communications session in question. Subsequent isolation of the server, root cause analysis, and recovery steps found and removed rogue software that initiated a Skype-like session with covert channel for data exfiltration.

The nature of cybersecurity operators and analysts is to focus more on the next problem than to meticulously capture details of their genius for a problem just solved. Therefore, the knowledge engineer works with the analyst to capture raw details of the incident and then the knowledge engineer creates a candidate-CDP for this context, problem, and solution. The knowledge engineer may also reference the SANS report and CSIRT details to discover additional real-world occurrences that contribute to the richness of the candidate-CDP and provide enough recurrence to satisfy the minimum level of invariance such that the candidate-CDP becomes an actual-CDP. Given this particular organization is a global enterprise with data operations across four continents, the lessons captured in this new CDP are staged to five other regional CDPLs for reuse by other cybersecurity operators and analysts.

Use case four addresses a zero-day attack representing an unknown-unknown. By the nature of the required minimum level of invariance, CDPs and the CDPL do not contain details about newly encountered threats. However, by virtue of the details not appearing in the CDPL, this enables first line operators to identify quickly an unknown-unknown. Similar to use case 3, subsequent investigation may, or may not yield details that spark the creation of a candidate-CDP. In the domain of cybersecurity, capturing details of a single occurrence will likely be

appropriate to share critical knowledge sooner than later. For example, the first encounter of Stuxnet warranted the creation of CDP candidate-pattern to capture and share knowledge of the environment in which the malware operated; the problem in terms of what it does; and the solution in terms of how to find it, contain it, and remove it. Such details would reside under a label of candidate-pattern until meeting the minimum level of invariance requirement.

If at a later time details were uncovered by an analyst that show an anomaly was indeed known but the CDPs lacked adequate detail, then this case may be characterized as an unknown-known. While such a characterization has little to no value in real-time, this posteriori characterization is useful to identify and track needed improvements in CDPs and the CDPL.

In all use cases, a PIE elicits user feedback in the form of a simple merit-rating scheme involving mouse clicks on a number of stars or on radial buttons representing a simple quality scale. In the event of a low rating, the PIE may prompt for additional details; and, prompt the knowledge engineer to elicit details and take corrective action in a manner that minimizes the involvement of the cybersecurity operator or analyst.

Future: Cybersecurity Automation

The *cybersecurity automation capability* is the ability to keep the cyberspace environment free from harm or danger with minimal manual intervention. Fundamentally, cybersecurity automation has two layers: 1) management, and 2) mechanistic manipulation. The mechanistic manipulation is the tactical execution of command and control (C2) in the cyberspace environment; i.e., the means to direct cybersecurity services and mechanisms and related cyberspace components relevant to sustaining a secure operating environment. The management is the logical reasoning and directing of C2; i.e., the governance and adjudication to ensure effective, efficient, and secure cyberspace operations.

What does it take for automated logical reasoning behind governance and adjudication decisions necessary for cybersecurity management? At the least, it requires *tactical* awareness, data, and knowledge; *mission* awareness, data, and knowledge; and *strategic* awareness, data, and knowledge. Can a machine possess and process knowledge? No, not in the sense of free thought; however, it may in the sense of formally encoded knowledge that represents a context, a problem, and a solution (i.e., CDPs and the CDPL). This formally encoded knowledge is part of the foundation for machine enhanced cognition (e.g., training an artificial intelligence (AI) system) to search, analyze, and produce conclusions and evidence from a breadth and depth of details in far faster time than a human. In time, the AI system will produce an answer (i.e., a human independent conclusion that leads to a subsequent automated course of action). At times, the AI system may produce options and recommendation for OODA-support. In all cases, there is persistent need for humans in the loop to continually train the AI system in the cybersecurity context and validate the decisions as accurate, useful, and usable. Therefore, AI supplements human thought and increasingly acts as a decision-proxy, but never entirely supplants the human in the loop. CDPs and the CDPL are the bare beginnings in formally encoding cybersecurity knowledge to establish a foundation for training AI in the cybersecurity operations domain.

Conclusion

This paper introduces the unique concepts of *cybersecurity decision patterns* and a *cybersecurity decision pattern language* as a disciplined approach to cybersecurity knowledge management. Details herein apply agile systems engineering fundamentals to the CDP and CDPL concepts to

show how they facilitate formal knowledge encoding and the adaptation of that knowledge over time to reflect new knowledge, refined knowledge, and refined understanding. CDPs and the CDPL provide a formal discipline for a knowledge engineer to elicit, codify, and capture cybersecurity operator and analyst experiences for knowledge sharing. Knowledge sharing transcends traditional data and information sharing by providing specific context, problem, solution, and supporting details that leverage the results of one successful effort across the entire organization. The organizational impacts include generating and retaining organizational knowledge, accelerated learning, increase in productivity across operators and analysts, higher quality responses, and improvements in response time and accuracy with intent to minimize threat efficacy.

Subsequent work to apply and expand on these concepts will elaborate on the CONOPS for CDP and CDPL application including their use in quantified analytic models (e.g., Bayesian Belief Networks) that add greater objectivity to their application and build a foundation for ever-increasing automation of CDP search, identification, determining applicability, and application. Figure 1 provides a foundation for identifying, isolating, and analyzing many systems dynamics modeling (SDM) archetypes. SDM archetypes are recurring patterns that help derive insight for the problem at hand from other problems and their solutions that fit the same archetype.

Additionally, subsequent research will include influence to CDPs from structures conducive to train artificial intelligence (AI) systems with the intent of exploring the potential of CDPs to begin building an AI training repository as a step along the path toward the goal of cybersecurity automation. A notional research roadmap to further CDPs, the CDPL, and their application to cybersecurity operations is as follows:

- Phase 1 (current; PhD dissertation in progress): person-to-person knowledge sharing
    - CDP and CDPL structure and content
    - CDP binning within and retrieval from the CDPL
- Phase 2 (current; PhD dissertation in progress): formal approach to conjoining cognitive aspects of cybersecurity operations and SE
    - Elaborate on the integration of cognitive systems engineering (CSE) into formal SE practice (e.g., INCOSE V model)
    - Apply SE and CSE to produce a joint cognitive system (JCS) that includes machine enhanced cognition in cybersecurity operations
        - Include CDPs and the CDPL in the JCS
- Phase 3 (future): modeling and analytics
    - Isolate quantifiable attributes to develop decision support systems (e.g., Bayesian Belief Networks) using CDP content and CDPL repository
- Phase 4 (future): train AI systems
    - Refine the CDP and the CDPL structure and content to be conducive to both train an AI system and to be mined by an AI system in the domain of cybersecurity operations; e.g., IBM Watson
- Phase 5 (future):
    - Employ the AI system for real-time decision support to cybersecurity operators and analysts; i.e., applied deductive and abductive logical reasoning
- Phase 6 (future):
    - Employ modeling and simulation to anticipate consequences of a proposed course of action in cybersecurity operations prior to taking that action; i.e., applied

inductive, predictive logical reasoning to look ahead for undesired outcomes and unexpected consequences

Annex – CDP Example

Table 4 provides a notional example of a cybersecurity decision pattern. The level of detail to include in a decision pattern is still under consideration. The CDP should have enough detail to be useful but not so detailed as to compete with a trouble ticketing system, where the details are very specific and could be down to bar code numbers of the assets involved.

The pattern name *Found Rootkit*, provides an idea of the role, fit, and function of the pattern. During monitoring, some observable indicated the presence of a Rootkit. The Found Rootkit decision pattern describes the problem, how to identify the problem, the context, forces, etc. This example provides a glimpse at the expanded CDP structure, which contains more detail than shown in the template (Table 2).

Table 4: Notional Example of a Cybersecurity Decision Pattern (Found Rootkit)

	Description
Name	Found Rootkit

Problem	The root of the problem is adversary presence in the cybersecurity environment and their executing steps to stay in by proliferating malware in the cyberspace environment. The desired result is to detect, find, and remove the malware and discover the method of proliferation and stop proliferation.
	• The root of the problem is unauthorized access and privilege escalation; the tactical and strategic implications are relevant to the attacker's specific use of the rootkit
	  o May include spyware, traffic monitoring, keystroke recording, backdoor creation
	• The desired result is a computer system that contains only the software that is part of the standard desktop environment
	
	Spotting the problem:
	• Observables (indirect, symptoms):
	  o Change from baseline taken by host-based intrusion detection software
	  o Strange behavior of system commands
	  o System files previously listed in file list no longer appear
	• Observables (direct, source)
	  o Using a binary editor, load <file>.sys and look for the occurrence of the string <string>. If present, this is a strong indicator that this is a rootkit.

Context	A rootkit is a collection of malware tools that enable administrator access to a computer. It allows the attacker to hide the intrusion and use the local computer as a launching point to other computers on the network. Rootkits reside on computer hosts.
Forces	• The search and fix is on internal computer assets; no issue with authorization • The tools to search and fix are readily available within the organization • The directions to find the malware permit a moderate level of expertise to find and fix; may use Help Desk resources verses specialized malware analyst • A systemic fix remains unavailable at this time; must be found and fixed on an individual basis
Invariance	High. This pattern addresses a repeatable way to find and fix the <name> rootkit.
Maturity	☐ Candidate; ☐ Proto-Pattern; ☒ Pattern; ☐ Anti-Pattern
Solution	Finding • Tool(s): system process analyzer (e.g. Sysinternals' RootkitRevealer) • Use the system process analyzer to look for <details> Fixing • Removing the rootkit piecemeal is a tricky business; inherent proliferation scheme is robust and

	Description
	the return of the rootkit is likely if not addressed systemically • Recommend backing up all relevant data • Reimage the system with standard desktop environment • Ensure all relevant software resides on the system; install any approved non-standard software • Restore data
Resulting Context	The resulting context is a system that is in an initial installation state with regard to software and contains the latest data available. The rootkit is now off the system.
Rationale	• The reimaging is the best solution available for this type of rootkit. • The forces are resolved because the system is restored to full operation without the presence of the rootkit and any other malware that may not have been detected.
Related Patterns	**Upstream** (predecessors): • Scan for Anomalies; Found Anomaly • Scan for Malware; Found Malware • Identify Malware → Found Rootkit **Downstream** (successors): • <decision pattern> • <decision pattern>
Known Uses	This decision pattern was applied to (latest first): • <location> on <date> and proved successful in finding and fixing the rootkit; refer to incident # IR-2015.01.09-fd • <location> on <date> and proved successful in finding and fixing the rootkit; refer to incident # IR-2014.09.22-kw and IR-2014-09-24-nd
Background	NA
Diagram	NA
Source	NA; developed in-house

References

(INCOSE), SE Vision 2025 Project Team. 2014. A World in Motion – Systems Engineering Vision 2025. San Diego, CA: INCOSE.

Ackoff, R.L. 1989. "From data to wisdom." *Journal of Applied Systems Analysis* 16:3-9.

Alexander, Christopher, S Ishikawa, and M Silverstein. 1977. "Pattern languages." *Center for Environmental Structure* 2.

Appleton, Brad. 1997. "Patterns and software: Essential concepts and terminology." *Object Magazine Online* 3 (5):20-25.

Blakley, Bob, and Craig Heath. 2004. Technical Guide: Security Design Patterns. The Open Group.

Boardman, John, and Brian Sauser. 2008. *Systems thinking: Coping with 21st century problems*: CRC Press.

Cousins, Norman. 1978. *The Saturday Review*, 15-Apr.

Dove, Rick. 1992. "The 21st Century Manufacturing Enterprise Strategy, or What is All This Talk About Agility." *Paradigm Shift International*.

———. 2010. "Pattern qualifications and examples of next-generation agile system-security strategies." IEEE International Carnahan Conference on Security Technology (ICCST), San Jose, CA, USA, 5-8 Oct. 2010.

Dove, Rick, and Ralph LaBarge. 2014. "Fundamentals of Agile Systems Engineering - Part 1." *Proceedings International Symposium 2014, INCOSE*.

Garcia, Mary Lynn. 2007. *Design and Evaluation of Physical Protection Systems*. Burlington, MA: Butterworth-Heinemann.

Haberfellner, Reinhard, and Olivier De Weck. 2005. "Agile systems engineering versus agile systems engineering." INCOSE 2005 Symposium.

Hafiz, Munawar, and Ralph E Johnson. 2006. "Security patterns and their classification schemes." *University of Illinois at Urbana-Champaign Department of Computer Science, Tech. Rep*.

Hughes, Michael. 2006. "A pattern language approach to usability knowledge management." *Journal of Usability Studies* 1 (2):76-90.

Kesh, Someswar, and Pauline Ratnasingam. 2007. "A knowledge architecture for IT security." *Commun. ACM* 50 (7):103-108. doi: 10.1145/1272516.1272521.

Liebowitz, Jay. 2001. *Knowledge Management: Learning from Knowledge Engineering*: CRC Press; 1 edition (March 28, 2001).

Lowe, James D. 2006. "A Design Pattern Language for Space Stations and Long-Term Residence Human Spacecraft." *American Institute of Aeronautics and Astronautics* AIAA 2006-7317.

Meszaros, Gerard, and Jim Doble. 1998. "A pattern language for pattern writing." *Pattern languages of program design* 3:529-574.

Osinga, Frans. 2005. *Science, Strategy, & War - The Strategic Theory of John Boyd*. The Netherlands: Eburon Academic Publishers.

Rowley, Jennifer E. 2007. "The wisdom hierarchy: representations of the DIKW hierarchy." *Journal of Information Science*.

Salingaros, Nikos A. 2014. "A Theory of Architecture Part 1: Pattern Language vs. Form Language." *ArchDaily*, 23-Mar-2014.

Takahashi, Takeshi, Youki Kadobayashi, and Hiroyuki Fujiwara. 2010. "Ontological approach toward cybersecurity in cloud computing." Proceedings of the 3rd international conference on Security of information and networks.

Ulrich, Werner. 2006. "The Art of Observation: Understanding Pattern Languages." *Journal of Research Practice* 2 (1):Article R1.

Warr, Dartanian, Herbert A Cole, and Gary B Reid. 1986. A Comparative Evaluation of Two Subjective Workload Measures: The Subjective Workload Assessment Technique and the Modified Cooper Harper Scale. DTIC Document.

Willett, Keith D. 2008. *Information Assurance Architecture*. New York: Auerbach Publishing.

Biographies

Mr. Keith D. Willett has a BSc in Computer Science with Mathematics minor from Towson University (1984); an MSc in Business and Information Systems from University of Baltimore (1986); an MSc in Information Assurance from Norwich University (2005); and is a PhD candidate in Systems Engineering Security at Stevens Institute of Technology (est. 2016). Mr. Willett holds $(ISC)^2$ CISSP and ISSAP certifications and has over 30 years of experience in technology and security as an educator, programmer, database administrator, operations manager, systems engineer, enterprise architect, and enterprise security architect. Mr. Willett is the co-author of *How to Achieve 27001 Certification* and *Official $(ISC)^2$ Guide to the ISSMP CBK*; and sole-author of *Information Assurance Architecture* all published by Auerbach Publishing.

Mr. Rick Dove is an INCOSE Fellow, and an adjunct professor at Stevens Institute of Technology teaching graduate courses in agile systems and systems engineering. Rick chairs the

INCOSE working groups for Systems Security Engineering and for Agile Systems and Systems Engineering. He is CEO/CTO of Paradigm Shift International, specializing in agile systems and agile security R&D and education. As Principle Investigator (PI) he has led agile self-organizing system security R&D on US DHS and OSD funded projects. He was co-PI on the 1991 OSD funded Lehigh study that introduced the concepts of agile systems and enterprises, and led the subsequent DARPA-funded research during the nineties that established basic system fundamentals for agile systems of all kinds. He is author of *Response Ability* (Wiley 2001).

Dr. Mark R. Blackburn is an Associate Professor with Stevens Institute of Technology and primarily responsible for research focused on methods, modeling, simulation, visualization, and automated tools for reasoning about computer-based systems. He is the Principal Investigator (PI) on a Systems Engineering Research Center research task sponsored by NAVAIR investigating the most advanced and holistic approaches to model-centric engineering, and co-PI on a related task for Quantitative Risk. He has also been the PI on research tasks for the National Science Foundation, Federal Aviation Administration, and National Institute of Standards and Technology.