VOLUME 2 NUMBER 1

JOURNAL OF CYBER SECURITY & INFORMATION SYSTEMS

A Knowledge Management (KM) Primer

By Mark Addleson, PhD

It can be difficult to navigate your way around the field of knowledge management (KM). Whether you are just starting out and thinking about putting ideas about KM into practice, or you work in a organization that has had a KM initiative in place for years, at times it may be hard to see what KM is about, what people are doing, and why and how they are doing it. For, although the field has been evolving for at least twenty years, there is a very broad spectrum of ideas about what KM is (the theory and principles), how to do it (the practices) and what *not* to do (Fahey and Prusak, 1998; Snowden, 2007).

One sign of trouble in this field is that there are many definitions of KM.[1] Another is that, while lots of organizations claim to be 'doing KM,' their strategies often have little in common. A third is that KM can be very technical, so KM initiatives become complicated, often unnecessarily so.

In this primer, I want to answer three questions.

- Why KM?
- What is KM about?
- How do organizations undertake KM initiatives?

If you are involved in KM, I hope my perspectives will help orient you and I would be pleased to receive your questions, comments, or suggestions.

In the beginning there was management without knowledge

Management practices, as we know them today, began in factories, machine shops, and foundries towards the end of the Nineteenth Century, when mass-production methods became more prevalent (see Crainer, 2000; Witzel, 2012). The generally recognized starting point for contemporary management practices is 'scientific management,' indelibly linked to the name of Fredrick Taylor (1911), the inventor of time and motion studies and founder of the management consulting industry. By 'eliminating waste,' his object was to improve the productivity of manual workers and cut costs to make industrial organizations more efficient and more profitable.

Today, whether they work in government agencies or accounting firms and whether they are involved in aerospace engineering, health care legislation, or web design, most people are *knowledge workers* (Addleson, 2011). As Table 1, below, reveals, knowledge-work and factory-work are completely different. Because the different kinds of work have nothing in common, you can't manage knowledge workers – or their work – as if they were assembly line workers. In most organizations, however, you find principles and practices that evolved in factories, similar to those advocated by Taylor. It is hardly surprising, perhaps, that these conventional management practices are obstacles to doing knowledge-work.

Factory-work	Knowledge-work
Physical	Mental
Solitary (think 'production line')	Social (think 'network')
Routine and repetitive	Complex and dynamic
Talk is a distraction	Talk ('sharing knowledge') is necessary
Tools (like blueprints, machines, and breakeven charts) are essential.	Tools are needed, but all work starts with and is guided by 'talk' – people in conversation.

TABLE 1: Comparing factory-work with knowledge-work

One answer to the questions, where did KM come from and why are organizations doing it, is that KM provides tools and techniques to bring management into the Twenty-First Century. A well thought-out, fully implemented KM initiative can help to eliminate out-of-date industrial management practices. A good KM initiative will enable

people to organize and run today's organizations – government departments and agencies, for-profit businesses, as well as non-profits – as knowledge organizations need to be run, with employees – who often work in teams – collaborating and sharing knowledge (Bryan and Joyce, 2005; Linder, 2005; Sandow and Allen, 2005).

Knowledge management evolved from earlier 'change management' efforts that included 'Total Quality Management' (TQM) (Martínez-Lorente, *et al*, 1998) and 'Process Reengineering' (Macdonald, 1998) and 'Organizational Learning' (Yeo, 2005). These emerged about midway through of the Twentieth Century (Prusak, 2001; Lambe, 2011). The goal in each case was to *improve the way organizations worked* – to make them more effective and/or efficient. In retrospect, we can see that each of these efforts shared at least two important features, which contradicted old-style factory-management practices.

1. They advocated decentralization: greater reliance on 'local' knowledge and experience, instead of trying to run everything from the top with rigid rules, plans, and structures. The thinking here is that the people doing the work, with *practical knowledge based on their experience* of *how things work*, know most about how work processes can be improved and are the first to see problems when they arise. Workers 'on the ground' are usually in the best position to respond to changing circumstances, but they need to have *the authority to use their knowledge*, make decisions, and take action when necessary.

2. People need to 'share their knowledge'. If you want to devolve decision-making down to the local level, it is no good isolating individuals and groups in organizational silos (e.g. separating them by department) or behind top-down structures, which make it difficult for subordinates to communicate with superiors. You need to devise systems, structures, and cultures that make it easy to share knowledge, or 'move it around'.[2]

Knowledge management represents the further evolution of these ideas. Where TQM and Reengineering were devised originally with the object of designing new work practices and processes to make *industrial* firms more efficient, knowledge management is a creature of the *information age*.

These are some basic premises of KM:

a. people need knowledge and information to do their work;

b. today they have access to lots of information (and some people work almost exclusively with information);

c. accessing information and sharing knowledge enables them to do a better job – solve problems, work smarter, and produce better results;

d. there is technology available to help people access and analyze information and share knowledge;

e. conventional (factory-style) management practices don't pay attention to knowledge or information: to what knowledge/information people need, how they get it, whether they share it, and so on; and

f. in most organizations there are many barriers to accessing information and sharing knowledge

There isn't much agreement about what KM is

It is difficult to find a definition of KM that two people agree on and many fields, from IT to lawyers and librarians, claim KM as their own. As each has different knowledge-related needs, they advocate different practices.

Here is my attempt at a brief explanation of what KM is and why we need it.

When people do anything they *use knowledge and they often access information* (e.g. reading a book because they are doing research, mining a database for information about customers' buying habits). In most situations it requires more than one person to get something done and *people share knowledge*. At one end of the knowledge-sharing spectrum, in a person-to-person phone conversation, you might find a mechanic at a car dealership ordering brake rotors from a parts supplier. At the other end, where hundreds or even thousands of people, with different roles, responsibilities, and expertise are involved in a large-scale defense contract, at any moment, working in teams, various groups may be planning or reviewing some aspect of the design or testing of hardware or software.[3] Focusing on the interconnections between *work* (getting things done),

information, and *knowledge*, KM revolves around fundamental questions like:

- what information and knowledge do people need,
- how do they use it,
- where do they get it,
- do they have it,
- what obstacles are there to getting, sharing, and using it, and
- what will help them get it, share it, and use it more effectively

Wherever there are knowledge workers, questions like these help them accomplish whatever they are doing. The last one is most closely related to action, but, from the standpoint of KM, *all are practical questions,* in the sense that informed answers contribute to a better, more functional workplace.

The reason for asking these questions is usually framed in management language; for example, 'improving efficiency', 'making the organization more competitive (or more profitable)', 'getting things done quickly and cheaply', but the goal is the same. Unless people can do effectively what they aim to do (control air traffic, make new policies, protect critical infrastructure, care for the sick, and so on) organizational objectives won't be met. So, it is helpful, when doing KM, to keep the fundamental questions and people's work firmly in mind and *stay focused on the connection between these questions and the work people are doing*.

> *KM is about improving the experience and quality of knowledge-work, recognizing the importance of information and knowledge for getting work done well.*

Information and knowledge: is there a difference?

One of the considerations that trips up people doing KM is a lack of clarity about information and knowledge and their differences. Although a great deal has been written on questions like, is there a difference and, if so, does it really matter, the responses, unfortunately, often generate more heat than light. While they are philosophical, these questions are also intensely practical, because how you answer them shapes not only the way you think about KM, but also how you practice it.[4] To individuals who treat information and knowledge as interchangeable, the main purpose of KM, typically, is to provide employees with access to the right *technical information*. When they see knowledge and information as different, however, recognizing that people collaborate to get things done, the object of KM first and foremost is to create an environment where colleagues can readily share their knowledge with one another. In the former case, KM usually falls under IT; while, in the latter, instead of being subsumed under IT, KM may be the responsibility of a group in human resources or organization development.

Steering clear of philosophical debate about what knowledge is and how people acquire it, I will explain briefly why knowledge and information are different, although they are closely related in a symbiotic way. Anything you regard as *inform*ation informs – so, is useful – because you can and do place the material in the context of what you *already know*. Information 'fits' your (pre-existing) understanding. If something is beyond your knowledge and comprehension it is non-sense; it cannot inform.[5]

Knowledge

Knowledge is what you, or other people, know. If you have children you have knowledge about them: their ages, their likes and dislikes, their personalities, and so on. If you are a materials fabricator, you probably know what it takes to bend and cut and how to join metals and composites. Some of this you've probably acquired from books, the web, or from talking to colleagues.

It is common, nowadays, to distinguish between two types of knowledge: *explicit*, in the form of principles, theories, and facts about the world, lots of which fall under the heading 'technical knowledge'; and *tacit*, acquired largely from experience. These are sometimes referred to, respectively, as 'know *what*' (explicit) and '*know how*' (tacit). I know about the tensile strength of metals and the number of instructions a microprocessor is capable of handling every second and I know that the Empire State Building is 450 meters high, even though I've never experienced (seen) these directly. 'Know-how' implies an ability to get things done and to deal with problems or issues. I know how my children respond to different situations, I know how to jump-start a car, and I know how to stay upright on a bicycle without having to think about it.

It is also widely acknowledged that most of what we know is tacit and, among KM practitioners, there a fairly widely held belief that it is desirable, as well as practical, to turn tacit knowledge into explicit knowledge. Once 'captured,' they argue, it can be transferred to others (who will be able to access it as information) (Nonaka and Konno, 1998; Nonaka and

Toyama, 2003). Organizations, concerned about protecting their 'intellectual capital,' for example, are prompted by consultants to prevent useful knowledge 'walking out of the door' when employees resign or retire. They may then go to considerable lengths and incur significant costs to capture the knowledge of retirees, gained from years of experience on the job, then make it available to others

There are several reasons why, rather than rushing to embrace them, these kinds of initiatives, that include transferring knowledge in the form of lessons learned from people in one project team to another, should be treated with some skepticism and approached with caution. One reason is the growing recognition that tacit and explicit knowledge are different *types* of knowledge. It isn't practical to turn one into the other. Each is important and useful in its own way and they are complementary, not substitutes (Cook and Brown, 1999). There are also question marks over the knowledge that organizations manage to capture. Is it useful to others – either contemporaries or future generations – and, if so, in what form and under what circumstances?

The problem is that *knowledge always has a context* and you can't take it from its original context – the varied circumstances and life-experiences of the knowers – and put it into files or databases without it losing at least some of its *meaning*. One way to understand this problem is to consider how difficult it is to explain to someone who has never experienced a different culture how natives of the culture express their feelings. This is the kind of tacit knowledge you acquire through experience. You can explain to a stranger 'facts of the situation,' for example what people say and do when they greet one another, but this doesn't allow him or her to 'get' the culture. To know it, they have to experience for it for themselves, by participating in it.[6]

To clarify my position on the difficulty of capturing and transferring knowledge, it is time to return to the distinction between information and knowledge. There are many situations where people need and – as long as someone provides it – can *acquire information* that helps them either to do something they otherwise could not do, or to become more proficient at doing it. If they already have a common context of technical and other know how, doctors, engineers, lawyers, software developers, plumbers, or musicians can learn a lot from the information in instructions or other documents created by colleagues. But, with different backgrounds or fundamentally different experiences – when they have different ways of knowing and have to find common ground in order to proceed, when they have to *discover* what is going on, what others mean or intend, or what to do, when, and with whom – people's ability and willingness to collaborate and make sense of the situation (coming to understand it) together is paramount. Now, *sharing knowledge* takes priority over 'transferring information'. The information they can access is, at this point, less important than their ability to 'find a way forward' *together*.

Information

In contrast to knowledge, which people possess – they 'have knowledge,' think of information as 'out there' on websites, in databases, on menus, and in instruction manuals and blueprints. What was once someone's knowledge in the form of ideas, perspectives, or points of view, information is now in a kind of limbo waiting to be found.

It's not what is out there that is information. Whatever is out there becomes information only *when someone, seeing it as useful*, 'adopts' it and uses it. Whether they stumble upon it serendipitously or are consciously looking for ideas, a reference, or 'additional information' to help them with something they are working on, at the point at which they 'connect' with it, finding it interesting or believing it is useful, it becomes part of their knowledge (i.e. what they know) for a time. It is a common mistake to treat knowledge and information as if they are completely separate things. Knowledge – what we know and – information – which we acquire – are complementary. We find and use information because we have knowledge of how and where to look for it, plus an understanding of what we are looking for and some sense of what is likely to be useful and why.

Without a *context* of existing knowledge (i.e. what you already know), information is useless. In fact, without that context it is wrong to call it information, because it does not inform. Telling you the Empire State Building is 450 meters high is literally meaningless to you unless you know numbers, understand what a meter is, know what a building is and, more specifically, are interested in the height of buildings and the Empire State Building in particular. This is to say that 'stuff' is not information *unless people can make some meaning of it and,* when they do, *it is knowledge (i.e. it is what you, or they, know).* You are surely familiar with stories about inventors who, initially, were unable sell what later turned out to be very practical ideas (the invention of Xerography - photocopying technology – is one example), because potential investors who they tried to convince couldn't 'see' the significance of their ideas. They had no context for appreciating the information they were given.

They didn't have the knowledge to assimilate it.

Much of the knowledge that we use to first find information and then use it is tacit. If I am in a restaurant and want to know what there is to eat, I know to look at the menu, or to ask the person who comes to serve me, particularly if I can't understand (make meaning of) the menu because it is in a foreign language or it describes dishes from a country and culture I don't know. I know, too, that a search engine is my door to lots of potentially useful information, but, until I learn (and know) how to use it, all this information is 'hidden', as if it doesn't exist. When I buy a new piece of technology, I look for instructions on how to use it, but if the technology is far from what I already know, because I don't have a context of existing knowledge, I might not be able to understand the instructions. They won't provide me with useful information until I call a friend for help or ask an expert to help me.

Knowledge is social

These examples point to an important consideration about knowledge. Much of what we know isn't in our heads. It is social – held and shared in groups or communities (McDermott, 2002). Because knowledge (or knowing) is social, because we share experiences and the meaning of ideas, experiences, values, and beliefs, we're able to communicate, share knowledge, and collaborate.

As I'm sure you have discovered, however, shared experiences and shared meaning only go so far. You have been working on a project, with the same people, for some months and, just when you think you 'know how another person thinks' or believe 'you're all on the same page', someone's actions suggest that you really don't know what motivates them or, perhaps, that they haven't understood what you said or what you expected from them.

One of the complexities of organizational life is that we work with and are expected to share knowledge with people who have very different interests and experiences, even when they are from the same organization. Nowadays, the people we work with are often from different, even competing organizations (Addleson, 2011). When there are two or more prime contractors and many more subcontractors on a very large project – as you find, for example, with any Major Defense Acquisition Project (MDAP) – innumerable organizational, occupational, and interpersonal boundaries exist in the multiple networks of professionals who must interact and share knowledge in order to do the work. These differences contribute to breakdowns, when work gets done badly and the whole project may run into difficulties, which is one important reason why we have to pay attention to knowledge and really work at ensuring we are sharing it effectively.

Two approaches to knowledge management

It is important to understand the relationship between knowledge and information, because this has a bearing on how organizations approach KM and it also explains why many KM initiatives fail to live up to expectations.

Organizations undertake KM initiatives in order to improve efficiency, because they see KM as a way of becoming more competitive, of reducing costs, and so on. KM will have these benefits *if* it enables people to be more creative, to work smarter, to be more productive and, generally, to do better work. Earlier, I said that in order to make sense of KM – to appreciate what it is about and also to understand what works and doesn't work – it is necessary to keep an eye on the relationship between work, knowledge, and information. Now, we can begin to see why.[7]

The nature of knowledge-work

Here are a few examples of knowledge-work

- Organizing and coordinating teams designing the hardware for the navigation system of a surveillance drone.
- Deciding what kind of information to extract from a huge database of customers' purchases collected by a supermarket chain, then writing algorithms to extract the information.
- Tracking down the people responsible for committing a bank robbery.
- Assisting customers who are subscribers to your cloud-based hosting service to set up their sites.
- Designing a guidance system for an air-to-air missile.
- Developing a training program for employees in your HR department.
- Testing the security of a large government agency's information systems.

Now, here are a few of the characteristics of this kind of work:

Many people are involved in getting things done: employees of the organizations, their customers and clients, contractors, suppliers, and so on.

Typically, much the work is done by an assortment of project groups or teams. These may be comprised of people with different skills and technical qualifications. From time to time teams need to interact with other teams, both from the same or different organizations, who may be spread out across the globe.

The problems people deal with in order to get things done are often ill defined. They don't have clear-cut objectives, a predetermined time-line, and in some cases they don't even know who they are going to work with, as people are assigned and reassigned while the project or task is in progress.

Even before they begin to 'solve problems' their work involves 'setting' the problem: deciding what they are doing or what issues they are dealing with, then deciding what they're going to do about the situation and who should be involved, setting schedules, and getting commitments (Schön. 1983).

This work is what we call *'organizing'*. Knowledge workers spend a lot of time organizing.

They do this by interacting and talking to one another on the phone, in person, and by email. In fact, much of their work consists of conversations. Before a defense project is funded there are rounds of discussions and negotiations, among a multitude of stakeholders, including potential contractors, politicians, and senior officers, offering proposals, doing evaluations, and providing counterproposals. At different times these groups draw on individuals with a variety of skills, from negotiators to cost estimators to proposal writers. And, with the object of deciding what comes next as well as assessing what's been done, the pattern of *sharing knowledge* – talking, asking questions, offering advice, listening to what clients and colleagues have to say, getting commitments, providing updates on what they have accomplished, and so on, continues throughout the project until the contract is eventually put to bed, perhaps a decade or more later.

From these few points we conclude that:

- Work is very social. It involves people continuously interacting with one another.
- Knowledge-work is also cooperative in the sense that people *need to collaborate* in order to define and solve problems together.
- Conversations are central and, when you observe them at work, you realize how much time knowledge workers spend on the phone, on email, or talking to others in conference rooms and corridors. They can get little done unless they talk to each other, sharing their knowledge; and unless they are *willing* to collaborate they won't share knowledge. By talking to one another they find out what the issues are, what has been done so far, what needs to be done, what kinds of problems people are experiencing, and so on.

Now, these behaviors – working together collaboratively and talking to one another, sharing knowledge, are not the norm in most organizations. Under 'old' rules of management, which evolved in the factory system:

- Competition, rather than collaboration, is expected. People compete with one another in order to climb the ladder to the top or to earn bonuses and bigger paychecks.
- Action is valued more than talk. In fact, employees are generally discouraged from talking.
- Employees are expected to work alone, rather than cooperate.

The design of office space illustrates the last two points. You find employees sitting behind cubicle walls, isolated from each other.

KM version 1

One approach to knowledge management says the real purpose of knowledge management is to correct the deficiencies of conventional management practices. Knowledge workers can't function effectively in a factory-management culture. People need to talk to one another, they need to cooperate (collaborate) more than they need to compete, and as it takes a team (even teams of teams) to do the work, we should be rewarding team-effort rather than individuals.

From this standpoint, the fact that they are doing knowledge-work, not assembly-line-work, changes everything (Allee, 2000). KM, viewed as any and all actions that encourage and enable people to collaborate and, in the process, co-create and share knowledge, should be as ubiquitous, necessary and natural for organizations as breathing is for humans.

Adopting KM version 1 means recognizing that knowledge management is potentially deeply subversive. Its *purpose is to change the way we manage work* – making any and all changes necessary to ensure that knowledge workers are able to do and produce good work. In the interests of creating a culture where teams really do work as teams, where people

are able to leverage their combined knowledge to solve wicked problems (Conklin, 2006; Marshak, 2009; Rittel and Webber, 1978), produce good software, or deliver excellent services, we need to examine every practice to see if it stands in the way of generating and sharing knowledge. Nothing should be sacred.

KM version 1 begins with questions like:

- Our work depends on collaborating and sharing knowledge, what does it take to do it well?
- How well are we doing and what are the obstacles?
- How do we deal with them?

Only when people understand the relevance of these questions and have good answers should they address more 'technical' ones related to 'intellectual capital' and 'talent management' such as:

- What kinds of knowledge/experience do we need?
- Who has this knowledge?
- How do we ensure that the people who have it are connecting with those who need it and vice versa?
- What kinds of tools will help people collaborate and, how do we encourage people to use them in ways that foster collaboration? (see Wenger, White, and Smith, 2009)

KM version 2

A fundamentally different approach to KM, which is very popular, KM version 2 focuses on tools and data (or 'content') more than, and in many cases instead of, people and practices. Most organizations with KM initiatives actually do KM version 2, even if they talk as though they are doing version 1. There are probably two reasons for this. First, KM version 2 is not subversive. It fits well with conventional management practices. The other reason is that people who are responsible for KM often have not thought deeply enough about knowledge and work and haven't asked the deeper questions, about why they are doing KM, what they hope to accomplish, and what it takes to get there.

It is fairly easy to tell whether organizations are doing KM version 1 or 2. Version 2 is characterized by a highly technical KM language and by budgets that are heavily oriented to IT, to technologies like portals, databases, and search engines, and to activities like 'knowledge engineering,' knowledge capture and retrieval, information retrieval, enterprise architecting, data mining, and categorizing information (creating taxonomies or developing ontologies). KM version 2 is an approach that tends to see and treat knowledge and information either as completely separate (and to focus on information, mistaking it for knowledge) or to blur their differences. So, when people doing KM version 2 talk about 'collaboration', they often mean moving information or data around, rather than people interacting and sharing knowledge as they make meaning together (Addleson, 2013).[8] KM version 2 should probably be called 'information or data management' rather than KM.

Why it is necessary to keep these two approaches to KM separate

At the end of the day, doing knowledge-work well – *producing good results* – depends on people collaborating and sharing knowledge. This is the bottom line of knowledge-work and the object of KM version 1. People need to share knowledge and, no matter how sophisticated your technology, no matter how good your search engines, or how detailed your taxonomies (i.e. no matter how hard you pursue KM version 2), if they won't share knowledge or don't do so effectively you have a problem: your teams and project groups become dysfunctional and projects run into trouble and fall short or fail.

Most organizations struggle with the problem of sharing knowledge, but few are tuned into the reason for the struggle; they manage knowledge workers using outdated, high-control factory-management practices and KM version 2 is compatible with these practices. For example, knowledge workers need to network – and networks are loose and flexible – but organizations rely on rigid, top-down reporting structures. Instead of paying attention to these issues, to the culture the enables people to organize their work in fluid networks, with flexible plans (and deadlines if necessary) and agile practices – clearly this is a tough nut to crack because it requires everyone to think and act differently – organizations focus attention on KM version 2, opting for 'technological fixes.' Here, they get assistance from vendors who claim, misleadingly, to sell KM in a can (i.e. a computer/server). When they install this software, purchase this search engine, create a portal, build the right workflow processes, and so on, 'information rich,' employees will work smarter, quicker, be more productive, and organizations will be more innovative, more competitive, and more profitable.

Community of practice or community of interest?

Alongside portals, document repositories, directories, and search engines communities of practice (CoP) are an integral part of many organizations' KM initiatives. There is good reason to be pleased that word about CoP spread quickly (the term was coined by Jean Lave and Etienne Wenger in 1991) but there is a downside too. CoP is an over-used buzzword. What organizations call 'CoP' are often communities of interest (CoI). The difference is significant, as I will explain.

Knowledge-work is collaborative, creative, and synergistic. Knowledge workers get things done by interacting and sharing knowledge, when they draw on their experience to answer colleagues' questions or give advice, when they swap stories or try to fathom out – together – what is going on, what they need to do and how to do it. Communities of practice, as the name suggests, have to do with the way people do their work (i.e. with their practices) but, to appreciate what makes a CoP and why they are important for KM, the words 'community' and 'practice' must carry equal weight.

The word 'community' suggests a group of people who are quite intimate with one another; connected not only by intellectual interests (e.g. rocket scientists who design missiles) or formal work requirements (e.g. individuals designated as 'members of the "Blue Team"') but also by interpersonal relationships like friendship and/or loyalty and/or collegiality. Perhaps they show genuine affection for one another, with each one caring about what the others do or don't do. When people 'live in community,' they tend to see each other quite often, know quite a lot about what the others are doing, and be generous in helping and supporting one another when they need it. This describes the relationships among people in a CoP.

Turning to practices, Etienne Wenger identifies three elements of their work practices that give members of a CoP a sense of belonging to and participating in a shared enterprise (Wenger, 1998. 72-85. See also Wenger, 2004; Wenger *et al*, 2002).

- Their '*mutual engagement*,' or the fact that they are actively involved in doing the work together.
- The fact that they see their work as a *'joint enterprise'* and, as they interact, continuously discuss and clarify what they are doing, what constitutes 'good work' and whether what they've done is up to standard, and so on.
- '*A shared repertoire*' of resources in the form of shared routines, artifacts or tools, a common vocabulary, and, perhaps, similar ways of thinking, even dressing.

Julian Orr (1996) and others have written detailed accounts of CoP, explaining how they work and what makes them different from regular teams and the kinds of interactions people typically have in organizations. Some of the characteristics of CoP, which you might expect in a community, are:

- Limited hierarchy. Members treat one another as peers. Authority is based on age, experience, and expertise rather than rank.
- Limited competition. Relationships are collegial and cordial and competition is friendly (e.g. demonstrating problem-solving skills rather than rivalry for promotion).
- It is seldom 'strictly business'. When members chat they will talk about their families, share their concerns about bosses or colleagues, and so on.

From the standpoint of KM, CoP have virtues that are particularly important. One is that participants readily share knowledge. CoP are good – some might say ideal – knowledge sharing contexts. The other is that they are to a large extent self-organizing. Rather than compliance, relying on instructions and rules from above, it is participants' accountability to each other, as well as their mutual commitment to their 'joint enterprise,' that ensures the job gets done and gets done well. Self-organizing is a particular virtue in environments where things are constantly changing and experience is paramount. Rigid rules and formal structures impede rather than assist people in getting the work done.

As you might expect, organizations that understand and practice KM version 1, emphasizing flexible work processes, with groups sharing knowledge and organizing themselves, are generally better at supporting CoP. Sharing knowledge is everyone's business. This means a culture of openness. While CoP can emerge in all kinds of environments, they are more likely to thrive when there is openness rather than top-down control.

You'll often find groups called 'communities of practice' in organizations that have adopted KM version 2, emphasizing tools and technology ahead of people and practices. Most of the time, however, these are, at best, *communities of interest* (CoI). Members of CoI are interested in the same 'body of information' – not necessarily work-related – and, often, have little else in common. They may be members of the same profession (e.g. lawyers; scholars) and/or have a similar domain of expertise (tort reform; medieval religion). Sometimes a CoI is comprised of individuals with the same hobby such as sci-fi, model trains, or gardening.

As these examples suggest, the participants need not be in the same organization and CoI are often virtual groups of people who only contact one another online, as members of a user- or interest-group. In common with members of a CoP, sharing their ideas, knowledge, or information (in the form, say, of articles, drawings, or URLs) helps CoI members get things done, whether at work or play. This is valuable but, when it comes to KM, it is only part of the story. It is 'cooperation' rather than 'collaboration,' which means 'working together'.

Members of a CoP generally work together closely and, as joint contributors to the work, *co-create* the results, whether this is a

PowerPoint presentation or a piece of software. To understand the difference and why collaboration is highly desirable, you need to appreciate that knowledge-work is deeply creative. When software developers start a project, for example, they seldom know where they will end up. Their 'product' comes to life and evolves while they work, in the work, as they interact and talk among themselves and with their clients. Without the back-and-forth, the meetings, conversations, and networking, little would be accomplished.

The bottom line is that CoI are necessary, but not sufficient, for people to do good knowledge work. When a KM initiative is mainly tools and technologies (KM version 2), the IT department that takes the initiative in setting up SharePoint sites for team members to share ideas or ask for advice is helping the cause of KM; but by how *much* depends on a variety of factors. If the organization is hierarchical, their online contacts should help people work around the barriers to knowledge sharing (e.g. between superiors and subordinates) created by hierarchy. Yet, you can do this without affecting the culture and ultimately it is the culture (whether people do or don't want to share knowledge) that matters.

If, for example, their business involves working with 'big data,' or if they have highly sensitive information that needs to be secure, organizations surely must have a heavy IT focus (i.e. KM version 2). Focusing on IT, however, is never the end of the story in terms of getting work done. In fact, in most cases it is just a small part of the story. Knowledge-work means people getting together, interacting, talking, sharing knowledge and creating new knowledge in order to solve problems, deciding what to and how to do it, then guiding and assisting one another in actually doing it. Making this happen takes KM version 1. Organizations that are mainly doing KM version 2 won't get the results they want. Poor knowledge management practices and limited collaboration will consistently hamper them. It is worth thinking about your organization's stance on KM? Are you doing KM version 1, version 2, or both? And, how well does KM serve you?

To answer the question, how should we organize knowledge-work, you need look no further than Agile methods, like Scrum (among many sources of information on Agile, see the articles on Ken Schwaber's website http://www.controlchaos.com and Mike Griffiths' blog, 'Leading Answers' at http://leadinganswers.typepad.com/leading_answers/). Although they are associated with software development and project management, agile methods should serve as an example for all knowledge-work. As the name indicates, these methods have evolved with flexibility at their core. Agile recognizes tacitly that, in spite of their best efforts to do so, people may not be able to see and plan very far ahead. Instead, they figure out what to do (and, possibly, where they went wrong) while actually doing the work – discussing, planning, designing, and building – with one another, with other teams, and their clients. So, Agile practices rely on stakeholders interacting (i.e. cooperating and collaborating) frequently, sometimes daily (even if only for a few minutes) to share knowledge; and they rest on the premise that, as those doing the work know better than anyone what is going on, it is best for them to organize themselves (see Addleson, 2011).

The Team Software Process℠ embodies many of the best practices for supporting knowledge-work including: coaching, team building, collaborative planning, and regular meetings to assess and report on the status of the work and to revise plans. Watts Humphrey (2000), the developer of TSP, says it was clear, early on, that the success of TSP depended on management providing broad support for the process. This is true of KM in general and, as good KM practices frequently run counter to 'old' management practices, letting go of the old ways may well be the main obstacle to implementing a successful KM initiative.

About the Author

Mark Addleson is an Associate Professor in George Mason University's School of Public Policy, where he teaches in the Organization Development and Knowledge Management Master's program (ODKM). Mark taught for more than 20 years in his native South Africa at the University of the Witwatersrand's Graduate School of Business Administration and was a director of Econometrix, a consulting firm. An Associate of The OCL Group, LLC, he has consulted with a range of organizations – for-profit, non-profit and government – both in South Africa and the USA. His areas of research include organizing knowledge work, knowledge management, and organizational change and he is widely published. His book, *Beyond Management*, about how to organize knowledge-work, was published by Palgrave Macmillan in 2011. Mark also has a blog 'Management is Dead', at http://www.managementisdead.com.

References

Addleson, Mark. 2013. "Will the Real Story of Collaboration Please Stand up so We Can See It Properly?" *Knowledge Management Research and Practice* 11 (1) (February): 32–40.

Addleson, Mark. 2011. *Beyond Management: Taking Charge at Work*. Houndmills, Basingstoke, Hampshire, UK: Palgrave Macmillan.

Allee, Verna. 2000. "Knowledge Networks and Communities of Practice." *OD Practitioner* 32, Fall/Winter (4). Available at: http://www.vernaallee.com/images/VAA-KnowledgeNetworksAndCommunitiesOfPractice.pdf

Bryan, Lowell L., and Claudia Joyce. 2005. "The 21st Century Organization." *The McKinsey Quarterly* (3): 24–33.

Conklin, E. Jeffrey. 2006. *Dialogue Mapping : Building Shared Understanding of Wicked Problems*. Chichester, England ; Hoboken, NJ: Wiley.

Cook, Scott D.N., and John Seely Brown. 1999. "Bridging Epistemologies: The Generative Dance Between Organizational Knowledge and Organizational Knowing." *Organization Science* 10, Jul/Aug (4): 381–400.

Crainer, Stuart. 2000. *The Management Century: A Critical Review of 20th Century Thought and Practice*. San Francisco: Jossey-Bass Publishers.

Fahey, Liam, and Laurence Prusak. 1998. "The Eleven Deadliest Sins of Knowledge Management." *California Management Review* 40 (3): 265–76.

Humphrey, Watts S. 2000. "The Team Software Process (TSP)." Technical Report CMU/SEI-2000-TR-023. ESC-TR-2000-023. Available at http://www.sei.cmu.edu/reports/00tr023.pdf

Lambe, Patrick. 2011. "The Unacknowledged Parentage of Knowledge Management." *Journal of Knowledge Management* 15 (2): 175–97.

Lave, Jean, and Etienne Wenger. 1991. *Situated Learning: Legitimate Peripheral Participation.* New York: Cambridge University Press.

Linder, Jane. 2005. "How Do Things Really Work Around Here?" *Across the Board* 42 (6) (November): 24–29.

Macdonald, John. 1998. "The Quality Revolution - in Retrospect." *The TQM Magazine* 10 (5): 321–333.

Marshak, Robert J. 2009. "Reflections on Wicked Problems in Organizations." *Journal of Management Inquiry* 18 (1): 58–59.

Martínez-Lorente, Angel R., Frank Dewhurst, and Barrie G. Dale. 1998. "Total Quality Management: Origins and Evolution of the Term." *The TQM Magazine* 10, (5): 378–386.

McDermott, Richard. 2002. "Knowing Is a Human Act." *Upgrade* 3 (1): 8–10.

Nonaka, Ikujiro, and Noboru Konno. 1998. "The Concept of 'Ba': Building a Foundation for Knowledge Creation." *California Management Review* Vol 40, 1, Spring (Special Issue on 'Knowledge and the Firm): 40–54.

Nonaka, Ikujiro and Ryoko Toyama. 2003. "The knowledge-creating theory revisited: knowledge creation as a synthesizing process." *Knowledge Management Research & Practice* 1 (1): 2–10.

Orlikowski, Wanda J. 1993. "Learning from Notes: Organizational Issues in Groupware Implementation." *The Information Society* 9, 3: 237–250.

Orr, Julian E. 1996. *Talking About Machines: An Ethnography of a Modern Job*. Ithaca, NY: Cornell University Press.

Prusak, L. 2001. "Where Did Knowledge Management Come From?" *IBM Systems Journal* 40 (4): 1002–6.

Rittel, Horst M.J., and Melvin M. Webber. 1973. "Dilemmas in a General Theory of Planning." *Policy Sciences* 4: 155–169.

Sandow, Dennis, and Ann Murray Allen. 2005. "The Nature of Social Collaboration: How Work Really Gets Done." *Reflections, Society for Organizational Learning* 6, "Innovations in Practice" (4-5): 1–14.

Schön, D. (1983). *The Reflective Practitioner: How Professionals Think in Action*. New York: Basic Books.

Shingo, Shigeo. 1989. *A Study of the Toyota Production System from an Industrial Engineering Viewpoint*. Portland, Or.: Productivity Press.

Snowden, Dave. 2007. "1998 and all that, a return to sin." http://www.readability.com/articles/dxj6sxx6?print=1

Taylor, Fredrick Winslow. 1911. *The Principles of Scientific Management*. New York: W.W. Norton and Company Inc.

Yeo, Roland K. 2005. "Revisiting the roots of learning organization: A synthesis of the learning organization literature." *The Learning Organization* 12 (4): 368–382.

Wenger, Etienne. 2004. "Knowledge Management as a Doughnut: Shaping Your Knowledge Strategy Through Communities of Practice." *Ivey Business Journal* 68 (3): 1–8.

Wenger, Etienne. 1998. *Communities of Practice: Learning, Meaning, and Identity*. New York: Cambridge University Press.

Wenger, Etienne, Richard McDermott, and William M. Snyder. 2002. *Cultivating Communities of Practice*. Boston: Harvard Business School Press.

Wenger, Etienne, Nancy White, and John D Smith. 2009. *Digital Habitats: Stewarding Technology for Communities*. Portland, OR: CPsquare.

Witzel, Morgen. 2012. *A History of Management Thought*. 1st ed. New York, N.Y.: Routledge.

Endnotes

[+] I wish to thank Dennis Goldenson and Taz Dougherty for their advice and guidance, provided in conversations about this paper and in comments on earlier drafts.

[1] There is a collection of 42 definitions of knowledge management at http://www-958.ibm.com/software/data/cognos/manyeyes/datasets/43-definitions-of-km/versions/1

[2] Both sets of ideas are embedded in what has become known as the "Toyota Way" (See Shingo, 1989), which turns Taylorist management on its head.

[3] For examples of the need to share knowledge and the challenges of doing so in highly complex, networked organizational settings, we need look no further than the Major Defense Acquisition Programs (MDAPs), where thousands of people from many organizations, with widely different skills, interests, and affiliations, working in various teams, are contributing, in innumerable ways, to the development and production of a particular weapons system.

[4] As anyone knows who has delved into the distinction between knowledge and information, that it is a highly contentious area. Unfortunately, either because the matter is unsettled, or because people don't pay enough attention to the issues, knowledge and information are often treated as if they are interchangeable. People talk about 'knowledge' when they really mean 'information' and vice versa. Without claiming that my views are definitive or necessarily correct, I hope these ideas help both to reinforce the point that it is important to distinguish between knowledge and information and to stimulate you to think about the differences.

[5] People who, because of their training, or experience, or both, know (understand) differently, surely glean different information in the same circumstances; for example, a master mechanic and layman looking at an engine leaking something.

[6] The fact that much of what people know and need to know to 'understand the problem,' 'get the job done,' or 'find a way out of the mess' comes only from experience explains why it is so important to turn to those who have hands-on experience when drawing up plans, developing capability requirements for new systems, and so on. This fact also highlights a fundamental flaw in high-control management and administrative systems. In high-control organizations, the formal authority to act increases as you go up the chain of command and the greatest expertise is presumed to reside at the top of the organization. This combination often results in a particular type of hubris that leads to problems and breakdowns. Even though they have little or no practical knowledge on which to base plans or requirements, those at the top nevertheless plan and formulate requirements without advice from the people who have experience and they issue directives to subordinates who possibly understand the realities of the situation better than they do.

[7] For a fuller discussion of many of the points that follow, see Addleson (2011).

[8] By the 1980s, two kinds of software tools had appeared that supported collaboration. With one, like Ventana's GroupSystems, designed primarily to facilitate group decision-making, participants (typically aided by a facilitator) sat in the same room responding to common questions. The software aggregated their responses and seeing the results on a screen was a prelude to further conversations, debate, and deliberation. The other, like Lotus Notes, built as a client-server system, allowed virtual knowledge sharing, by participants who were possibly separated by both time and distance. Although this latter category of software, originally known as 'groupware,' has proliferated with the advent of internet-based social networking tools, in many organizations the tools still have not fulfilled their potential to support collaboration. 20 years ago, Wanda Orlikowski (1993), who had studied the roll-out of Lotus Notes in a large management consultancy, pointed out that the way tools are used reflects people's cognitive and technical frames, or perspectives. One reason why tools like SharePoint typically are used for storing data and accessing information, rather than as 'spaces' for sharing knowledge, is that the management mindset, which favors competition, doesn't 'get' the human-social dimensions of collaboration (as opposed to the technical possibilities for enabling it).

Search…Backwards

By Eric Treadwell

Department of Defense (DoD) procurement references a large number of military-unique specifications, standards, and handbooks (standardization documents). Project proposals and contracts with the DoD reference these standardization documents, often down to the sub-paragraph level. The process of finding only those paragraphs which apply to a given project, and listing them, is called "tailoring" and consumes considerable engineering time. Unfortunately, current instructions for military standardization documents (MIL-STD-961/962/963) offer neither guidance nor direction on this "tailoring" process. A process is needed that will extract only the applicable references for any project description. Using the proposed "Dynamic Tailoring" method, a Subject Matter Expert (SME) can easily write this needed "tailoring" process. Dynamic Tailoring allows rapid, accurate, and precise retrieval of information contained in standardization documents. Dynamic Tailoring also removes language barriers introduced by selection of key words, use of jargon, or unfamiliar terminology. Further, SME knowledge is preserved and made available in the form of the Dynamic Tailoring algorithm.

Background

Leafing through hundreds of pages of a requirements document in order to find the paragraph applicable to the task at hand is the lot of technical professionals. Searching in a Portable Document Format (PDF) (or other format), or worse yet, paging through a scanned document, does not unlock the promises of the information age. At best, if the document is used regularly, the user may have a collection of tags and bookmarks, which last until the document is updated. It is estimated that 30 percent to 50 percent of an engineer's time is spent searching for and validating information[1].

Dynamic Tailoring is a new method of sorting information in standardization documents and preparing it to be retrieved on demand (a patent is in process). The current approach is to trawl through an information set to find relevant information. The problem is that the person using the standardization document rarely thinks the same way as the SME. Using just one example, the expert and the casual user have very different terminology. For example: are we discussing bulldozers or track-loaders or wheel dozers? Even given a defined collection, a search engine relies on a limited set of keywords (or other tagging methods) to sift through data and return matching results. Looking beyond this, Dynamic Tailoring starts with the defined collection of directly and tangentially related data and sorts it in a manner relevant to answer all possible user queries. The question comes last: Search…backwards.

The path to the creation of this method was about as direct as a keyword search. The author was tasked to rewrite MIL-HDBK-1791, "Designing for Internal Aerial Delivery in Fixed Wing Aircraft," and update it to MIL-STD-1791A. MIL-STD-1791 communicates design requirements for aircraft physical and operational limits to designers and purchasers of new or modified equipment intended for transport via the United States Air Force's cargo aircraft. The standard must

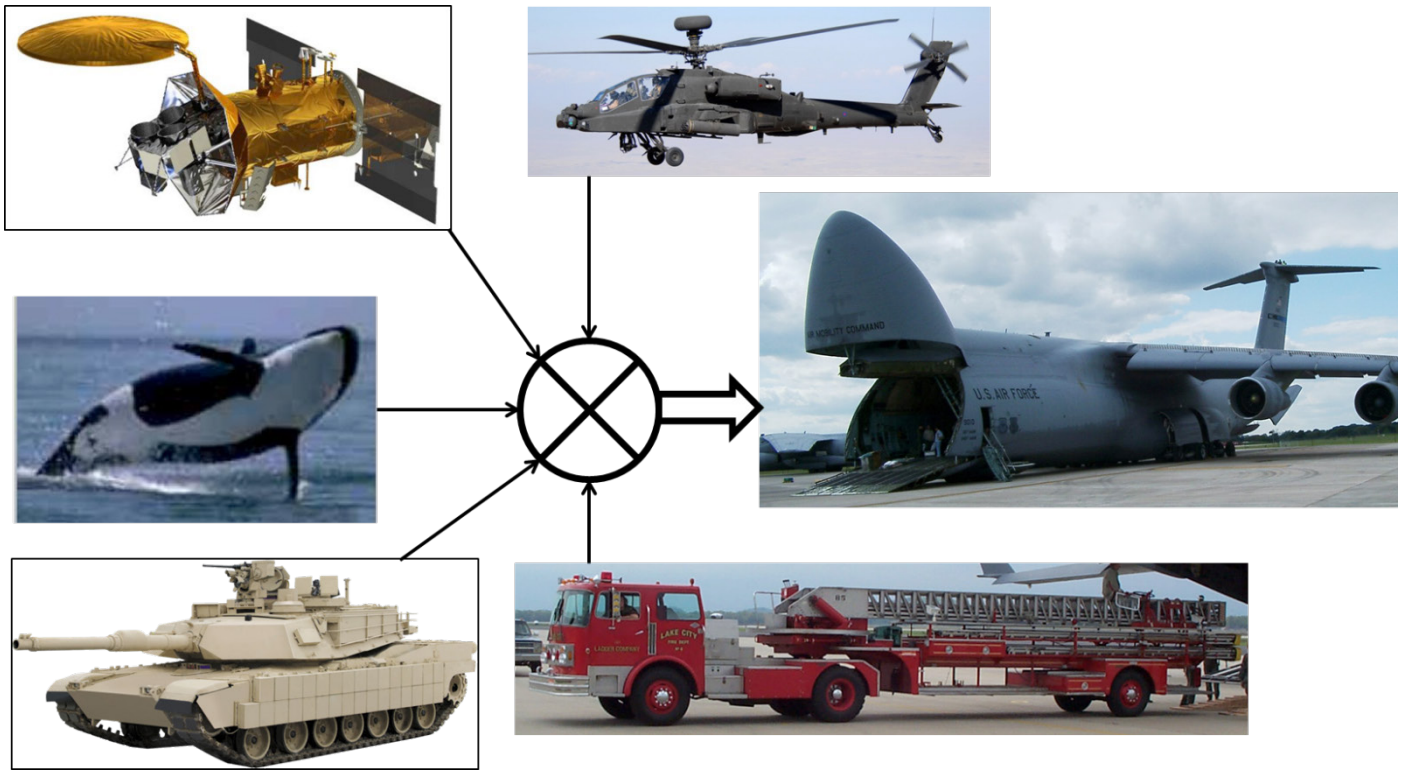


Figure 1: Schematic of MIL-STD-1791A Requirements

cover every conceivable type of cargo from any US government user: trucks, tanks, planes, helicopters, boats, satellites, and humanitarian cargo such as Keiko, the killer whale. In addition, the document contains guidance and lessons learned.

The process of corralling the wide-ranging information and fitting it to the nearly unlimited items which can be transported provided a challenge. Beyond just updating the document, the author developed Dynamic Tailoring to more effectively sort through the myriad requirements. The updated document was reorganized with a new structure for its content based on the constraints of Dynamic Tailoring. Finally, a new computer-based interface was implemented to facilitate Dynamic Tailoring.

What is Dynamic Tailoring?

Dynamic Tailoring is a method of preparing the content of a standardization document for the user of that document. The use of Dynamic Tailoring is an interactive process whereby the user describes to the standardization document (spec) the project at hand and the spec replies with only the applicable paragraphs. Notice that this definition focuses on the user's interaction with the spec.

Dynamic Tailoring does not change the content. It does not necessarily change the organization of the document. Dynamic Tailoring simply makes the information more accessible to the user. Users do not care about every single line; they only want to know what applies to their particular case. To help the user, the spec author/SME (or current Responsible Engineering Officer (REO)) must think through the way users interact with their document. The REO then performs the search…backwards, to prepare the document for Dynamic Tailoring. If the REO is not also the SME then a good deal of coordination is required.

The first task for the REO/SME preparing for Dynamic Tailoring is to perform a requirements analysis. The first step in this task is to identify the "core" requirements (those that always apply due to physics or laws). The second step is to identify "conditional" requirements. Next is to identify the conditions wherein "conditional" requirements apply. Lastly, separate and identify "procedural" requirements (as opposed to physical requirements). Why separate procedural limits? Frequently, procedural limits are based on risk acceptance or best practice, things like speed limits. With a good enough reason you can do things differently or perhaps change or

waive the restriction. Physical limits, like stall speed or structural strength, are immutable. Understanding the difference will help when the user comes for requirements relief.

Another word on requirements: The user does not care what types of requirements are written. The user only cares about compliance. Think hard about the conditions chosen. The conditions which will form the basis of the sorting routine are key to the user's successful interaction with the spec. In the example of MIL-STD-1791, the owning organization categorized at least 95 different types of cargo before abandoning the system. There is also a slowly changing roster of aircraft, with varying requirements, that one might try to organize the standard around. Rather than focus on either of these shifting categories, the author of 1791 took an interface approach to arrive at twenty-nine cargo characteristics and eight procedures. There may be an unlimited number of possibilities for cargo, but there are comparatively few ways that cargo can interact with an aircraft (see Figure 2). Keep in mind that while the examples in this article are physical items of cargo, the item or items governed by a spec may be physical items or a process and Dynamic Tailoring will still be applicable.

Requirements Analysis

1. Core Requirements
2. Conditional Requirements
3. Applicable Conditions
4. Procedural Requirements
5. Applicable Procedures

Once the requirements have been analyzed, task two correlates them. Conditions are correlated with conditional requirements, and procedures are correlated with procedural requirements. Three lists are then generated: the list of conditions and procedures, a correlation list for conditions/procedures and requirements, and the list of core requirements. Finally, those three lists are combined to produce the Dynamic Tailoring list. Thus Dynamic Tailoring is complete and the document is prepared for the user.

To use Dynamic Tailoring: first, the user identifies which conditions their project matches; second, the user identifies which covered procedures are being used; lastly, the user consults the Dynamic Tailoring list to identify the core requirements and applicable conditional and procedural requirements. Specification compliance verification may then proceed with a known, agreed-upon subset of applicable requirements. The lists may be considered agreed-upon (contractually, regulatory, etc.) because in the document the SME publishes the lists as a required part of the specification.

Requirement Correlation

1. List Core Requirements
2. List Conditional and Procedural Requirements
3. List Conditions and Procedures
4. Correlate List #2 and List #3
5. Combine the core list and the results of Step 4.

This process is simple to automate using current computer technology. In a manual ("user's guide") implementation, the user is presented with the three lists (they may be formatted as tables, etc.). In an electronic implementation the user interface is based off of the list of conditions and procedures and the computer program returns the core requirements along with any results that correlate to the selected conditions or procedures. In Figure 2, the "Loading Method" and "Special Consideration" columns both represent the conditions identified for MIL-STD-1791A.

The process may also be extended to include multiple, related documents. To stay with the transportation theme: MIL-STD-1366, "Transportability Criteria," references MIL-STD-1791, along with MIL-STD-209, MIL-STD-129, MIL-STD-810, and MIL-STD-461, all of which may be required for an item being transported by various Department of Defense assets. In order to link all the documents together for Dynamic Tailoring, all that is required is a wider effort.

Payoff

"Internet [keyword] searches require some judgment. If you don't use enough keywords to narrow your topic, you'll end up spending a lot of time scanning sites and trying to find the ones that are most relevant. On the other hand, a tightly focused search might overlook a relevant citation. There are no easy answers, but through trial and error you'll probably find the balance that works for your particular topic."[2]

Why should "educated guess" remain the rule of the day? Use of Dynamic Tailoring and its underlying requirements analysis prepares the data for the user such that the results are neither too broad to be helpful nor so narrow as to exclude critical information.

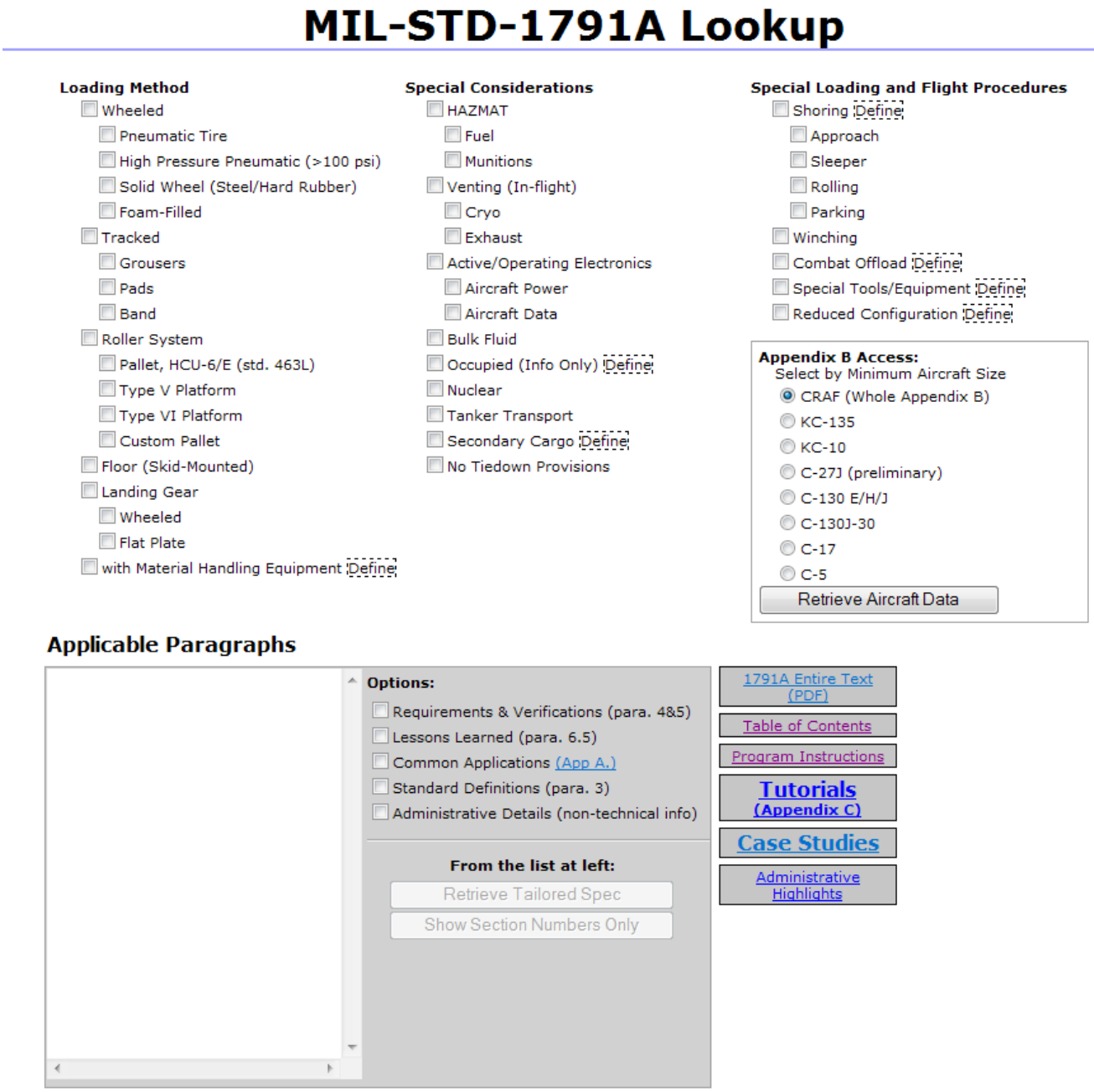


Figure 2: Screenshot of the Prototype

Based on research[1], it is estimated that tens of thousands of man-hours could be saved in the DoD through rapidly, accurately, and precisely retrieving information contained in standards documents. The first benefit of producing Dynamic Tailoring for a document is that the initial cost is offset by reduced requests for assistance, interpretation, and clarification. The following activities are also affected: identification of requirement applicability; easily comparing various vendors' offers against specification paragraphs; clear communication of design parameters; facilitating training on, use of and exploration of specifications ("what if" scenarios). Responses to requests for proposal can be more complete. Specification compliance verification can be streamlined since applicable criteria are known. The document publishing body is also expected to experience benefits in training new employees and preserving SME knowledge. In short, any organization that uses or produces standards to which Dynamic Tailoring applies stand to benefit: all uniformed services, NATO, etc.

While it is beyond the scope of this paper to estimate the cost to any given project, the cost of starting from scratch is known. The total cost of the prototype effort: approximately 3 semester hours were expended to develop the Dynamic Tailoring method and apply it during the rewriting of MIL-HDBK-1791. Approximately 3 more semester hours were spent rewriting the handbook. About 960 man-hours were expended in: revisions and editing of the document, designing the web-based interface, writing the computer code, and testing the electronic version of Dynamic Tailoring for MIL-STD-1791A. The only break down available for the 960 hours was a full update of the computer version's document text (copy-and-paste) with some paragraph numbering changes and associated algorithm modification. This update took approximately 30 man-hours. This also represents a mean maintenance cost per document update.

The electronic version was written in HTML, CSS, and JavaScript. Other file types utilized included PDF, JPEG, and GIF. This set of programing languages and file types assured multiple operating system compatibility, the only requirement is a modern web browser. Once this was assembled, it was realized that the planned CD distribution method was not fully utilized. The electronic version included additional examples, color photographs in the body of the standard, as opposed to the requirement for line drawings in the printed edition. Taking further advantage of the information storage space, additional full-color photographic case studies were developed and included as a supplement to the existing guidance and lessons learned. Future projects may reuse some of the work products, they are available for distribution.

Current Status

The project is complete and the Dynamic Tailoring method has passed internal testing. The manual Dynamic Tailoring list and table for MIL-STD-1791A were published as non-binding guidance in the standard. The dramatic change represented by Dynamic Tailoring is difficult for the current system to accept. The computer-based version faces further hurdles as the Department of Defense does not presently publish standards except on paper or in PDF. The anticipated workload of converting other eligible standards is seen as the main roadblock to wider adoption of the method, regardless of manual or electronic implementation. An education and training effort will be required to move forward with this time-saving advance in knowledge management.

Conclusion

In short, Dynamic Tailoring guides the user with a method that can be thought of as searching backwards. It starts with the answers and the Subject Matter Expert who holds them. The SME then investigates all the ways questions may be put to his document (requirements analysis). After this the SME creates the list of relevant questions and identifies the parts of his document that answer them (Requirements Correlation). This correlation is then provided to the user as the Dynamic Tailoring method for the document. Thus, any possible user's query is focused and guided to the correct answer before their search has begun. The time savings of both internal and external users of standardization documents will surely benefit the Government, or any other standards-issuing body.

About the Author

Eric Treadwell an engineer in the C-5 program office. He has spent twelve years working in the Air Transport Test Loading Activity (ATTLA) and various cargo aircraft program offices on air transport and air drop of cargo. In addition to updating MIL-STD-1791, Eric authored ATTLA's cargo analysis program, T-Loader. T-Loader consolidates geometric and weight limits from six different aircrafts' TOs to facilitate evaluation and automate routine calculations. To expedite fulfilling the DoD's 24/7 world-wide demand for ATTLA's air transport certifications, Mr. Treadwell spearheaded development of ATTLA's "community of practice" website, making the certificates available for download to authorized users. He is also vice president of a writer's group in Dayton and has a handful of unpublished novels.

References

Dick Bourne, Search …and find (Mechanical Engineering, Vol. 135, No. 09) pg.45, citing research by CIMdata of Ann Arbor, MI.

AFH33-337, The Tongue and Quill, 1 Aug 2004, page 29, [clarification added].

BYOD Topic: How Complicated Can Calendars Be?

By Michael Weir

Using a representative example from something as simple as calendar synchronization, the unintended consequences of moving too quickly into the Bring Your Own Device realm are highlighted, with some commentary on strategies that can help.

In a January 2013 blog post from Microsoft TechNet regarding excessive iOS6.1 logging on a Microsoft Exchange Server[1], author Bobby Pendino and Microsoft MVP Andy David trade comments regarding causes and fixes for an Apple-versus-Microsoft approach to managing calendar requests and synchronization. Now, these two companies are no stranger to complicated operating systems and complex code, but their differing approaches to a seemingly insignificant Information Systems (IS) component such as notification of calendar synchronization changes made a king-sized enterprise problem for anyone that had a combination of Apple consumer products and Microsoft server products (which is a pretty large number of organizations!). The primary incident discussed in this particular technical discussion described a single Apple device, automatically updated to iOS 6.1, and the resulting 50 Gigabytes of logging that inundated the Microsoft Exchange Server when the device rebooted and began checking calendar appointments automatically. In a web article from 8 February 2013[2], Author Ed Burnette notes that several corporate environments decided to shut off access by Apple devices, because there is no local control over the Apple automatic updating that their users had configured. The technical details flowed out over a few months, with a good reference for altering Microsoft Exchange mailbox handling to eliminate most of the problem, and leaving Apple users out in the cold until Apple released a fix (reference is here: https://devcentral.f5.com/community/group/aft/2165837/asg/50).

Clearly Apple and Microsoft worked quickly to remediate this particular flaw, but the real stakeholders that need to pay attention are enterprise IS managers and also corporate CIOs who are interested in bringing to fruition the business dreams of Bring Your Own Device (BYOD) that is all the rage these days. While it is a possible method for streamlining many of the difficulties of employees having various combinations of telephone, tablet, laptop, slate, and desktop computers that are personally or organizationally owned, this particular story highlights the real possibility that BYOD brings a whole new world of risk into the information systems strategic thinking bubble. Much of risk-based assessments of evolving IT and IS infrastructure assumes a certain predictability of likely futures. In a true BYOD enterprise, how does one postulate the risk over the universe of possible devices that might be brought to bear to access and interact with corporate data when the user has full flexibility?

The implications are significant. An organization that proposes to implement a BYOD strategy needs to provide policy-level guidance that can exist long enough for employees to effectively understand and use it properly, along with implementation procedures that are technically current enough to handle new devices and interactions as they evolve and occur.

If the policies of an organization change quickly over time to accommodate technology changes, employees will not have a firm enough basis to understand and comply with that policy. The result is an unwillingness or an actual inability to really comply with the guidance because it is too transient to succeed as a policy.

If the procedures an organization's IT/IS personnel implement are insufficient to meet the policy directives, then the risk increases significantly that some unusual/odd/uncharacteristic device may be introduced that has an unintended consequence that has great impact on the organization.

Typically, an organization includes policy statements such as "the incorporation of new software or hardware onto the network must be coordinated with the IS/IT department so that compatibility and impact analysis can be done." In the

situation described, there was no opportunity to vet the impact of a revised Apple iOS because it was automatically configured without user interaction and without coordination with the enterprise components of any organization. The decision of a single hardware provider can impact directly on an entire organization without any opportunity to develop remediation or integration steps effectively.

In my own experience managing corporate IS/IT, this type of wild-card aspect of risk management is what keeps dedicated IS staff up at night. At first blush, the combination of using "Big Data" and BYOD seems to clear some of the limiting hurdles of our current organization-centric approach to managing and using data to its greatest extent. In truth, there is no free lunch. During each attempt to squeeze more out of the IT department by either outsourcing or eliminating perceived bottlenecks (and in doing so, reducing staff internally), there is a concomitant increase in the actual resource support necessary from the IT staff or from the general employee population to keep up with the "improved" infrastructure and services. I've seen it in the transition during the 1980's and 1990's to eliminate administrative staff to support general office duties ("secretarial" IT support), and during the 2000's to consolidate and reduce help desk support. In both cases, the reduction of staff actually resulted in less productive work force statistics. In the first case of administrative IT support, every single worker in the organization(s) becomes less productive because they inherit the effort necessary to feed the automation tools directly and must take time out of every day to perform functions that used to be handled by a small but dedicated staff. In the case of help desk automation, every single worker in the organization(s) becomes essentially their own troubleshooter and interpreter of the automated (or outsourced) support function. Overall, the effectiveness of the organization is decreased, but the perception is that the business is running "leaner" because staff numbers are lower.

The "no free lunch" aspect of this particular risk set comes about through the strictly increased monitoring and immediate response capability that will be necessary to reduce this potentially large impact from significantly affecting the business or organization. It is a more difficult risk function to calculate, as there is not yet a set of "rules of thumb" or metrics that capture the indeterminate nature of the possible failures that could come with this type of BYOD failure/impact mode. Making an error on the safe side will increase the burden of IS/IT costs on the organization and may affect in some ways an ability to implement new ideas (limitation of funds/resources). Making an error on the risky side may put the company in a very bad situation when something adverse in fact does happen.

Logically, a more carefully considered policy for BYOD in the enterprise could provide a solid and long-term base for IS/IT strategy, but it will take a more determined approach on the part of the IS/IT management and strategic thinkers to help the business and organization strategists to understand that there really does need to be firm control of some aspects of adding the newest technology to the enterprise networks, and that the risk of adding new technology really does have an acceptance of a larger risk possibility.

As well, the functional-level procedures that are implemented by a company's IS/IT organization must come with very good automation to support understanding of the condition of networks and auditing/monitoring functions that provide usable and easy to understand statistics on the well-being of the network and resources used by the organization.

About the Author

Michael Weir, CSIAC Director, has been with Quanterion Solutions, Inc. for about four years with a primary focus in cyber security. Prior to joining QSI, Michael spent 32 years in and out of the Air Force and Air National Guard, maintaining ties with the Air Force Research Laboratory while supporting operational units in the US and overseas. He has degrees in Music Performance, Electrical and Computer Engineering, and Information Management along with several certifications in network and computer security.

References

[1] B. Pendino, "Is iOS6.1 causing Excessive Logging? Anyone Else See This?", web forum for Microsoft Technet, 31 January 2013, http://social.technet.microsoft.com/Forums/en-US/exchangesvradminlegacy/thread/d7f4b534-2eac-4291-9564-97a9875056ee, visited 10 February 2013

[2] E. Burnette, "iOS6.1 banned from coporate serviers due to Exchange snafu", ZDNet Forum, 8 February 2013, http://www.zdnet.com/ios-6-1-banned-from-corporate-servers-due-to-exchange-snafu-7000011064/, visited 10 February 2013

Metrinome – Continuous Monitoring and Security Validation of Distributed Systems

By Michael Atighetchi, Vatche Ishakian, Joseph Loyall, Partha Pal, Asher Sinclair, Robert Grant

Distributed enterprise systems consist of a collection of interlinked services and components that exchange information to collectively implement functionality in support of (sometimes mission critical) workflows. Systematic experimental testing and continuous runtime monitoring of these large scale distributed systems, including event interpretation and aggregation, are key to ensuring that the system's implementation functions as expected and that its security is not compromised.

To illustrate the need, consider an example Information Management System (IMS) that enables sharing of sensitive information between information publishing and consuming clients. Problems associated with configuration management can easily lead to situations in which the IMS allows unauthenticated clients to participate in information exchanges or allows unauthorized information to be disseminated to consumers. Furthermore, the loose coupling between subscribers and the IMS can lead to situations in which the IMS is unavailable and consumers believe that no new information is being published, causing significant misunderstandings across information sharing relationships. Finally, remnant vulnerabilities in the IMS can cause failures to happen at any time and cause significant damage to mission execution if not dealt with in a real-time manner. Unavailability of information sharing directly reduces situational awareness, loss of integrity can give adversaries control over mission execution, and loss of confidentiality can be detrimental to the reputation of actors and/or mission goals in general.

Monitoring and validation of IMS and client operations can aid in detection, diagnosis, and correction of situations like this. This is particularly important since 92% of reported vulnerabilities are located at the applications layer [1]. Despite the importance of experimental validation and continuous monitoring, and the increased support to adopt security assessment as part of the software development life cycle, current approaches suffer from a number of shortcomings that limit their application in continuous monitoring situations and their use in the validation of assurance claims.

First, current test practices favor unit tests over integrated tests for establishing correct functionality. Unit testing, e.g., performed via Junit [2], checks program functionality piece-by-piece but provides little to assess the overall information assurance claims of a system under test. Various tools exist for actively assessing the security of distributed systems, e.g., Nessus [3] and HP Fortify [4] to name a few, but their functionality is achieved by running specialized unit tests for security properties against either the code or the running system. In contrast, integrated end-to-end testing tools, such as YourKit [5] or Grinder [6], focus on performance and scalability. These tools enable operators to find bottlenecks or provision computing resources, but lack metrics associated with assessing security and correct functionality.

Second, integrated and end-to-end testing and experimentation is often postponed until software artifacts have matured significantly. This is because integrated testing and experimentation can be time consuming and effort intensive and the perception is that the cost of manually performing experiments early on frequently outweighs the benefits.

Finally, common testing and metrics frameworks add additional dependencies to existing systems, in the form of additional libraries that need to be loaded into the system under test and lines of code being added in support of instrumentation. This not only increases software complexity but more importantly can cause version dependency issues. It can also have unintended side effects on certification and accreditation as the software now has additional code that must be certified but that is not part of the core functionality, i.e., it is part of the continuous monitoring.

Distribution A. Approved for public release; distribution unlimited (Case Number 88ABW-2013-4215). This work was sponsored by the Air Force Research LaboratoryAFRL.

This article describes Metrinome, a metrics framework written in Java that is specifically designed to provide a platform for structured continuous security assessments throughout the software lifecycle. The novelty of Metrinome lies in its loose coupling with the system under test and its integration of end-to-end testing with continuous application-level remote monitoring. Specifically, Metrinome provides (1) runtime computation of a wide range of metrics from log messages generated by distributed components during system execution, (2) execution of assertions over the metrics to determine correct functionality while the system is operating, and (3) improved situational awareness via dashboard views and generation of experimentation reports. The outputs of Metrinome-based assessments can be used as input to Certification and Assessment (C&A) processes to precisely document the assertions that were previously checked to hold true in the system. Metrinome is available free of charge to government entities through AFRL.

II. Related Work

A. SNMP Dashboards

A number of management platforms exist that use the Simple Network Management Protocol (SNMP) for monitoring devices and nodes. Network Management Information System (NMIS) [7] operates at the networking level and enables monitoring, fault detection, and configuration management of large complex networks. Its main metrics deal with device reachability, availability, and performance. HP OpenView, IBM Tivoli, and Nagios provide similar functionality. Unlike these platforms, Metrinome specializes on monitoring at the application level and execution of fine-grained assertions.

B. Distributed Testing

Software Testing Automation Framework (STAF) [8] is an open source multi-platform, multi-language framework that enables a set of functionalities including logging, monitoring and process invocation for the main purpose of testing. STAF operates in a peer environment; a network of STAF-enabled machines is built by running STAF agents across a set of networked hosts. In contrast to STAF, the goal of Metrinome is more focused and hence no agents are required to be installed. Avoiding agents not only leads to reduced maintenance costs but also significantly reduces the attack surface across networked systems under test. Due to their complimentary nature, we have used Metrinome in conjunction with STAF for continuous testing and integration.

C. Application-level Metrics Frameworks

Several application-level metrics frameworks exist to monitor and measure the performance of applications. For example, Javasimon [9] exposes an API which can be placed into the code and allows inline computation of count metrics and measurement of durations. Metrics [10] is similar to Javasimon but allows data to be streamed to other reporting systems, e.g., Ganglia [11] and Graphite [12].

An important distinction between Metrinome and the above mentioned frameworks is Metrinome's use of log messages to provide the same monitoring functionality. This makes Metrinome loosely coupled with the system being monitored and makes it applicable to any application that generates log messages, e.g., using Log4j or Logback.

D. Reporting/Graphing Backends

Ganglia, Graphite, and Splunk [13] are examples of highly popular platforms that offer the ability to search, analyze, and visualize data in real-time. Typically these frameworks consist of a processing backend that collects and stores the data. They also use statistical methods that provide new insight and intelligence about the data. Metrinome provides functionalities that intersect with the above mentioned applications, such as dashboard views and experimentation reports. One difference is that Metrinome focuses less on scalability but rather on ensuring correct execution of a system under test through the validation of assertions.

E. SIEM Platforms

Security Information and Event Management platforms (SIEMs), e.g. ArcSight [14], adopt many of the technologies described above, such as SNMP dashboards and reporting backends, to provide users with the ability to query, and analyze security threats generated by both hardware and software applications. Unlike Metrinome, these platforms require the deployment of agents on networked hosts to collect and report events.

III. Design and Architecture

Metrinome is designed to achieve specific objectives in portability and ease of use.

- **Portability** – Metrinome can monitor a system independent of the implementation of the system.
- **Minimal coding overhead** – Rather than adding new instrumentation libraries to monitored processes (causing versioning conflicts and Java classpath pollution), Metrinome interfaces with existing logging and auditing frameworks, e.g., Logback [15].
- **Ease of use** – To be of immediate use to experimenters and administrators, it should be easy to specify metrics and assertions that must hold over the metrics in a systematic way. In addition, results of metric computation need to be readily accessible by humans or other programs through a well-defined Application Programming Interface (API) and Graphical User Interface (GUI).

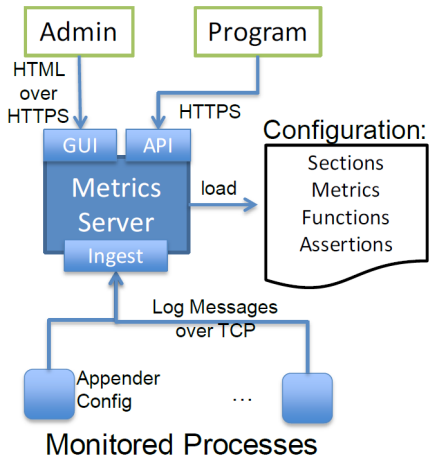


Figure 1: Metrinome High-Level Architecture

Figure 1 provides an overview of Metrinome high level architecture. Metrinome works with a set of monitored processes that have the ability to send log messages over TCP connections to the ingest API provided by the Metrics Server. Ingestion is performed via simple logging configuration changes on the monitored processes, e.g., by specifying the use of a SocketAppender in Logback to send certain log messages remotely to the Metrics Server over TCP connections in addition to or instead of sending those messages to the console or a local file.

Due to the fact that log messages issued by different processes may be similar, particularly if the processes are executing the same code base on different physical machines, the Metrics Server requires a descriptive unique process name associated with a specific logging instance as part of the log message. This requirement has already been built into most of the logging and auditing framework, enabling filtering of messages based on process names within Metrinome. The processing performed by Metrinome on received messages is defined using a XML-based Domain Specific Language (DSL), describing concepts such as sections, metrics, functions, and assertions. The Metrinome DSL allows administrators to specify processing logic in one file that can be dynamically loaded into the Metrics Server.

Finally, to ease access to information, Metrinome offers two interfaces: (1) a GUI, implemented in HTML and accessible through common web Browsers using HTTP(s), and (2) a RESTful [16] secure Web Services API for use by external programs.

IV. Modes of Use

The Metrinome framework supports a number of operational use cases and scenarios, including use during demonstrations, experiments, and continuous monitoring.

A. Runtime Visualization During Demonstrations

A major hurdle facing users during a demonstration is the ability to showcase a holistic view of the system operation while highlighting specific aspects that are being demonstrated, such as performance, load balancing, resistance to security attacks, etc. Metrinome's GUI equips the demonstrator not only with the ability to pinpoint the changes in the system as these events occur, but also to visualize these changes to the measurements graphically during runtime.

B. Experimentation

Metrinome seamlessly integrates with off-the-shelf continuous integration frameworks, such as Jenkins [17]. Users can easily specify assertions showcasing desired system behavior. Metrinome evaluates assertions at specific control points within an experiment or at the end of an experiment. Metrinome's HTTP interface also allows user controlled and on-demand evaluation of assertions at runtime. An HTTP response will indicate whether the assertion evaluation passed successfully or failed. In the case of failure, the HTTP response also includes information about the particular assertions that failed.

When an experiment is complete, Metrinome stores the state of all assertions along with metrics values, historical statistics, and definition of metrics. This process supports offline analysis and reproducibility of experiments, and can also generate inputs to C&A processes. Finally, Metrinome has the ability to export the metrics data into other programs using the Comma Separated Value (CSV) format which allows administrators to perform customized analysis over the data, using spreadsheet and visualization software of their choice.

C. Continuous Monitoring

Continuous monitoring is a desirable feature in enterprise environments because it decreases the time to react to occasional hardware and software failures and minimizes the time to mitigate security attacks such as Denial of Service attacks. While guidance for continuous monitoring is maturing [18], agencies have already started to struggle with compliance mainly due to implementation costs [19]. Metrinome reduces costs by virtue of integrating with existing logging and auditing frameworks. It also provides ready dashboard functionality that increases situational awareness at no additional implementation cost.

V. Interfaces

A. Metrinome Language

Metrinome processes receive messages based on user-specified processing logic, which is dynamically loaded into the Metrics Server. This processing logic echoes a user's perception of the desired system behavior and is declared in terms of metrics and assertions. Users are able to express such terms using a XML-based representation.

Figure 2 shows the XML schema for specifying the processing logic. The metrics element serves as an enclosing element to the entire document, while the section element serves not only to organize the metrics and assertions into different clusters, but also to limit the scope of assertions.

```
- <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  - <xs:element name="metrics">
    - <xs:complexType>
      - <xs:sequence>
        - <xs:element name="assert" minOccurs="0" maxOccurs="unbounded"
          - <xs:complexType>
            - <xs:sequence>
                <xs:element name="name" type="xs:string" />
                <xs:element name="metrifRef" type="xs:string" />
              + <xs:element name="function">
                <xs:element name="value" type="xs:string" />
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        - <xs:element name="section">
          - <xs:complexType>
            - <xs:sequence>
              + <xs:element name="assert" minOccurs="0" maxOccurs="unbou
              - <xs:element name="metric" minOccurs="0" maxOccurs="unbou
                - <xs:complexType>
                  - <xs:sequence>
                      <xs:element name="name" type="xs:string" />
                      <xs:element name="description" type="xs:string" />
                    + <xs:element name="function">
                    - <xs:complexType name="choice">
                      - <xs:choice minOccurs="1" maxOccurs="1">
                        - <xs:group name="unary">
                          - <xs:sequence>
                            - <xs:element name="event">
                              - <xs:complexType>
                                  <xs:element name="component" type="xs:s
                                  <xs:element name="regex" type="xs:string"
                                </xs:complexType>
                              </xs:element>
                            </xs:sequence>
                          </xs:group>
                        - <xs:group name="binary">
                          - <xs:sequence>
                            - <xs:element name="start_event">
                              + <xs:complexType>
                              </xs:element>
                            - <xs:element name="end_event">
                              + <xs:complexType>
                              </xs:element>
                            </xs:sequence>
                          </xs:group>
                        </xs:choice>
                      </xs:complexType>
                    </xs:sequence>
                  </xs:complexType>
                </xs:element>
              </xs:sequence>
            </xs:complexType>
```

Figure 2: Metrinome's DSL Schema

Thus an assertion specified for a particular section will not be triggered against metrics in another section

The core of the language consists of two major elements: Metric and assert. Metric is used to specify a measurement evaluation while assert – associated with a metric or a set of metrics – is used to specify the expected system behavior.

A metric element has a unique name and a description to provide information about the Metric. An assert element has a unique name and a metricRef element which uses regular expressions to allow the referencing of a metric or a set of metrics that the assertion will be evaluated on. Both elements encompass a function which expresses a statistical calculation to perform. Functions allow the user to configure the actual operation to be performed, which in the case of metrics occurs over the incoming message (e.g., counting the number of exceptions occurred) or in the case of assertions, through the specification of a logic expression (e.g., zero number of errors).

A function specified as part of an assertion is triggered when an experiment is complete or by an external entity request. The main purpose of assertion functions is to validate metric values, thus they tend to be logical in nature.

A function specified in a metric can be triggered by a single event which is equivalent to specifying unary functions, such as count, or two separate events (denoted start_event and end_event) which is equivalent to specifying binary functions, such as time difference.

An event consists of two parts: component and regex. A component outlines the actual set of processes whose log messages can trigger such an event. All processes not specified via the component element will not trigger the specific event. The regex specifies the message string to be processed. The processing engine allows the use of regular expressions in both component and regex, thus enabling easy specification of processes and messages.

Finally, a function element can have several attributes:

- **round:** rounds a numeric value of the measurement to the nearest specified number beyond the decimal point.
- **roundhistory:** similar to round but applies over the statistical calculations rather than individual values.

Two special attributes epochs and colors are used to indicate the staleness of a measurement as observed by the Metrics Server and can be customized per metric. A user can specify a staleness threshold and an associated severity color which will be highlighted on the HTML GUI Interface.

Metrinome provides a set of predefined functions for computing metrics, including the following:

- **count:** counts the number of occurrences of an expression,
- **ratio:** provides the ratio of two expressions,
- **diff:** calculates the time difference between two events,
- **absdiff:** calculates absolute time difference between two events, and
- **sum:** calculates the sum of two expressions.

Examples of assertion functions are equals, greater than, less than, and greater than or equal. The library of functions can be easily extended to support additional functions, which currently requires changes to the Metrics Server but not to monitored processes.

```
<metric>
    <name>reqAuthz_pass</name>
    <description>Number of passed client requests</description>
    <function round="0" roundHistory="1">count</function>
    <event>
    <component>CoTToPubSvc|CoTToSubSvc</component>
        <regex>.*Request failed authorization.*</regex>
    </event>
</metric>
```

Figure 3: Example Metric: Count

```
<assert>
    <name>No_OutOfMemory_Errors</name>
    <metricRef>error_outOfMemoryErrors</metricRef>
    <function>equals</function>
    <value>0</value>
</assert>
```

Figure 4: Example Assertion: No Out Of Memory Error

```
<assert>
    <name>non_error_gt_0</name>
    <metricRef>^((?!.*(error|Processing)).*)</metricRef>
    <function>greaterThan</function>
    <value>0</value>
</assert>
```

Figure 5: Example Assertion: Non processing and error metrics should be greater than zero.

Figure 3 highlights an example of a security assessment metric called 'reqAuthz_pass' which provides the number of requests that failed during authorization generated by processes containing 'CoTToPubSvc' or 'CoTToSubSvc' in their descriptive names, based on which they are sending log messages to the Metrics Server. This metric is useful especially for testing the authorization process of an application during high load or automated attacks.

Figure 4 shows a simple assertion example over a metric called 'error_outOfMemoryErrors'. As the name indicates, this is a useful assertion for testing that a system has no out-of-memory exceptions.

Another example shown in Figure 5 highlights an assertion that showcases the correct functionality of the system under evaluation. The assertion uses regular expressions to state that all metrics except the ones containing error or processing in their names should have values greater than zero values.

B. HTML User Interface

Figure 6 displays a screenshot of the GUI. The first column highlights the name of the metric as specified in the configuration file. The next column highlights the latest measurement of the metric. By default, Metrinome provides statistical information such as the average, median, and standard deviation across historical values. The last column highlights the changes in the value of the metric over time graphically. This feature is useful to quickly pinpoint measurement anomalies. Users can view metrics without the graphs by clicking on the "Metrics" link.

C. Metrinome API Interface

The service API consists of an Assertion and Metrics service accessible via HTTP.

The Assertion service offers the following functionality:

- HTTP GET http://localhost:8080/assertions
  - Triggers evaluation of assertions against the current status of the metrics, which either returns success in the form of a HTTP response code of 204, or a list of failed assertions, encoded as XML payload in the HTTP response.
- HTTP GET http://localhost:8080/assertions?SHOWDEFS
  - Displays a table of current assertion definitions.

The Metrics service offers the following API:

- HTTP GET http://localhost:8080/metrics?CSV
  - Returns the metrics values in a CSV format.
- HTTP GET http://localhost:8080/metrics?EVENTS
  - Returns the collected events that were used to generated the metrics.

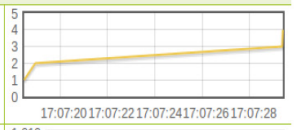
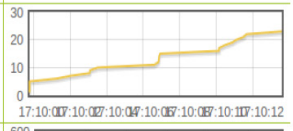
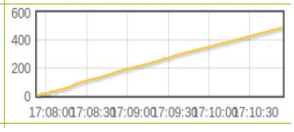
Metric	Value	mean	median	stddev	min	max	Graph
STEG Functionality:Internal Load							
subscription_count_t2e	5	2.2	2.0	1.3	1.0	4.0	
subscription_count_e2t	1	1.0	1.0	0.0	1.0	1.0	
publications_count_t2e	24	11.5	11.5	7.0	1.0	23.0	
publications_count_e2t	491	245.0	245.0	141.9	1.0	490.0	

Figure 6: Metrinome's Metrics with Graphs Interface

VI. Use of Metrionome During Red Teaming

We have successfully used Metrinome during internal security testing of software artifacts developed under the Secure Tactical to Enterprise Gateway (STEG) [20] R&D effort. To evaluate the security benefits of STEG, we build an internal threat model

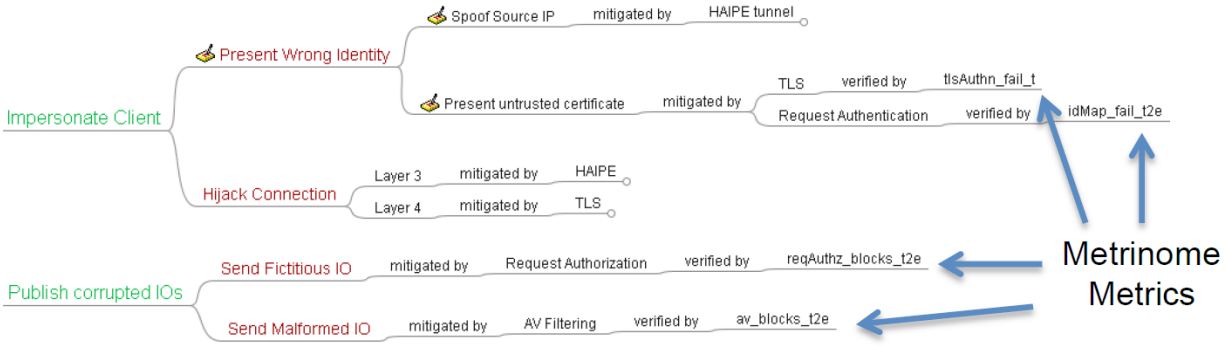


Figure 7. STEG Attack Tree for Loss of Integrity

that decomposes attacks into three main categories, namely, attacks that cause (1) loss of integrity, e.g., by corrupting service logic or changing data in transit, (2) loss of availability, e.g., by crashing critical components or exhausting shared resources, and (3) loss of confidentiality, e.g., by getting unauthorized access to sensitive information. The attacks are then further decomposed into sub-categories for each category (i.e., Integrity, Availability, and Confidentiality). The model can be visually represented as attack graphs, with annotations for defenses and logical arguments.

Figure 7 shows the resulting attack graph for integrity. The graph reads from left to right and first branches out into high-level attack strategies, e.g., Impersonate Client and Publish corrupted IOs. The next levels then provide functional refinements for the attacks. Attack refinement may lead to multiple alternatives (branches). The next level of the attack graph is annotated with mitigated by, indicating the defensive component that addresses the particular attack represented by the branch. Note that an attack strategy may have multiple mitigating defenses (indicated by the mitigated by annotation on a branch). For the cases where mitigation is verified by experimental observation or logical arguments, the attack graph is shown with an additional level, annotated with verified by describing how we determined that the STEG prototype actually addresses the threat.

We used Metrinome to establish and document correct security functionality by measuring a number of metrics listed in the attack tree, including TLS authentication failures, identity mapping failures, authorization failures, and anti-virus filtering failures.

VII. Conclusion and Future

Metrinome has proven to be an effective component in supporting runtime assessment and monitoring, demonstrations and scientific experimentation during execution of the STEG R&D effort. In particular, the integration of end-to-end testing into the continuous build cycle has helped identification and mitigation of run-time bugs.

Going forward, we expect Metrinome to grow as it is adopted by other efforts with extended requirement sets. In particular, we have plans to (1) make it easier to add custom functions without the need to recompile the Metrics Server through a plugin framework, (2) provide capabilities for more complex graph generation, e.g., by providing boxplots via integration with R [21], (3) provide the ability to define metrics over metrics and metrics capturing trends, and (4) implement an adapter layer for ingesting messages other than Logback.

About the Author(s)

Mr. Michael Atighetchi is a Senior Scientist in the distributed computing group at BBN and technical lead on several DARPA- and USAF-sponsored research projects. Mr. Atighetchi has a Master of Science degree in Computer Science from UMASS Amherst and a Master of Science in Informatics from the University of Stuttgart/Germany. Mr. Atighetchi is a Senior Member of the IEEE, member of ACM, and has authored over 60 publications in peer-reviewed conferences and journals on topics including adaptive security, Red Team assessments, identity management, and Cross Domain Solutions. Raytheon BBN Technologies, 10 Moulton St, Cambridge, MA 02138

Dr. Vatche Ishakian is a Scientist in the distributed computing group at BBN working on USAF-sponsored research projects. Dr. Ishakian's has a PhD degree in Computer Science from Boston University and is a member of the IEEE and ACM. His experience spans a broad set of disciplines across networking and distributed systems, including application-level scheduling and management, network economics, data placement, and network architecture. Dr. Ishakian has authored over 15 publications in peer-reviewed conferences and journals. Raytheon BBN Technologies, 10 Moulton St, Cambridge, MA 02138

Dr. Joseph Loyall is a principal scientist at Raytheon BBN Technologies. He has been the principal investigator for Defense Advanced Research Projects Agency and AFRL research and development projects in the areas of information

management, distributed middleware, adaptive applications, and quality of service. He is the author of over 100 published papers. He is a Distinguished Member of the ACM and a Senior Member of the IEEE and of the AIAA. Dr. Loyall has a doctorate in computer science from the University of Illinois. Raytheon BBN Technologies, 10 Moulton St, Cambridge, MA 02138

Dr. Partha Pal is a Principal Scientist at BBN Technologies. His research interest is in the areas of adaptive cyber-defense, resiliency and survivability. As the Principle Investigator in a number of past and ongoing projects sponsored by various agencies, he has been leading the development, demonstration and evaluation of innovative cyber-defense mechanisms, strategies and survivability architectures, and using them to build survivable distributed information systems. He is a senior member of the IEEE and a member of the ACM. He has over 80 publications in peer reviewed conferences and journals, and holds a PhD in Computer Science from Rutgers University. Raytheon BBN Technologies, 10 Moulton St, Cambridge, MA 02138

Mr. Asher Sinclair is a Senior Program Manager at AFRL's Information Directorate working in the Resilient Synchronized Systems Branch (RISB) at the Rome Research Site. His interests include research and development in enterprise systems management, service-oriented architectures, and Cyber security. He has contributed to more than 18 technical papers and conference proceeding publications. He holds a bachelor's degree in Computer Information Systems from the State University of New York and a master's degree in Information Management from Syracuse University. Air Force Research Laboratory, 525 Brooks Road, Rome, NY 13441, USA

Mr. Robert Grant works for the Air Force Research Laboratory Information Directorate in Rome New York. He has a B.A. in English from the University at Buffalo, a B.A. in Computer Science from Oswego State, and is currently working on his Masters in Computer Science at Syracuse University. Air Force Research Laboratory, 525 Brooks Road, Rome, NY 13441, USA

References

[1] Eoin Keary, Integration into the SDLC(Software Development Life Cycle), Retrieved Nov 06 2013, https://www.owasp.org/images/f/f6/Integration_into_the_SDLC.ppt

[2] JUnit Homepage, Retrieved Sep 06 2013, https://github.com/junit-team/junit/wiki/Getting-started

[3] Nessus Vulnerability Scanner, Retrieved Sep 06 2013, http://www.tenable.com/products/nessus

[4] HP Fortify My App, Retrieved Sep 06 2013, https://www.fortifymyapp.com/

[5] YourKit Profiler, Retrieved Sep 06 2013, http://www.yourkit.com/

[6] Grinder, Retrieved Sep 06 2013, http://grinder.sourceforge.net/

[7] Network Management Information System, Retrieved June 10 2013, http://www.sins.com.au/nmis/sample/

[8] Software Testing Automation Framework, Retrieved June 10 2013, http://staf.sourceforge.net/

[9] Java Simon - Simple Monitoring API , Retrieved June 10 2013, http://code.google.com/p/javasimon/

[10] Metrics, http://metrics.codahale.com, Retrieved June 10, 2013

[11] Ganglia Monitoring System, Retrieved June 10 2013, http://ganglia.sourceforge.net/

[12] Graphite - Scalable Realtime Graphing, Retrieved June 10 2013, http://graphite.wikidot.com/

[13] Splunk, http://www.splunk.com/ Retrieved June 10 2013.

[14] ArcSight, http://en.wikipedia.org/wiki/ArcSight

[15] Cody Burleson, "How to setup SLF4J and LOGBack in a web app – fast", Apr 10 2013, https://wiki.base22.com/display/btg/How+to+setup+SLF4J+and+LOGBack+in+a+web+app+-+fast

[16] Fielding, Roy Thomas, "Architectural styles and the design of network-based software architectures", Diss. University of California, 2000.

[17] Jenkins: An extendable open source continuous integration server, http://jenkins-ci.org/ Retrieved July 1 2013.

[18] Kelley Dempsey, Nirali hawla, Arnold Johnson, Ronald John-ston, Alicia Clay Jones, Angela Orebaugh, Matthew Scholl, and Kevin Stine, "Information Security Continuous Monitoring (ISCM) for Federal Information Systems and Organizations", Retrieved June 25 2013 http://csrc.nist.gov/publications/nistpubs/800-137/SP800-137-Final.pdf

[19] Jason Miller, "Agencies struggle with continuous monitoring mandate", Retrieved June 25 2013 http://www.federalnewsradio.com/513/2681377/Agencies-struggle-with-continuous-monitoring-mandate

[20] "R: Box Plot Statistics", R manual, Retrieved June 3 2013, http://stat.ethz.ch/R-manual/R-devel/library/grDevices/html/boxplot.stats.html

[21] "Secure and QoS-Managed Information Exchange between Enterprise and Constrained Environments", currently in submission to appear in Proceedings of ISORC 2014.

BEC: Applying Behavioral Economics to Harden Cyberspace

By Victoria Fineberg

This article describes several cybersecurity innovations. First, it proposes to integrate behavioral economics' findings of biases in judgment and decision-making into cyber strategies, policies, and guidance using a new framework called Behavioral Economics of Cybersecurity, or BEC. Second, it aligns BEC with NIST's Risk Management Framework by treating persistent human biases as a special type of vulnerabilities in the Risk Assessment phase and by controlling these biases in the Risk Response phase. Third, it defines the BEC structure using a Zachman-like two-dimensional framework of cyberactors (Users, Defenders and Attackers) from three cybersecurity perspectives (Confidentiality, Integrity and Availability). The paper also provides examples of how common cybersecurity exploits map into the BEC framework.

1 Introduction

Cyber practitioners regard a human as the weakest link of cybersecurity, yet cyber strategies and policies do not reflect this reality. An implied assumption of cyber guidance is that, if decent people are properly trained, they will do the right thing. So why don't they? People habitually take decision-making shortcuts with consequences ranging from undesirable to catastrophic. An exploration of faulty judgments and implementation of relevant countermeasures could enhance cybersecurity.

This paper proposes a framework for bridging the gap between theory and practice of the human role in cybersecurity through the identification and mitigation of persistent cognitive biases. The motivation for this work is the impact behavioral economics has made on standard economics by amending the rational-actor model with quantifiable irrationalities. Rational actors are assumed to know and do whatever is in their best interest. While economists have always been aware of various manifestations of irrational decisions, they previously disregarded them on the premise that individual irrationalities are random occurrences that cancel each other out without detriment to economic modeling. However, in the 1970s cognitive psychologists revolutionized the field by demonstrating that many biases are not random but rather typical and persistent, enduring even when individuals are aware of them. Their work gave rise to behavioral economics, which bridged the gap between economic theories and psychological realities.

This paper proposes a similar approach of bringing Behavioral Economics models into Cybersecurity to identify common cyberactor biases and develop mitigating models. Figure 1 illustrates the parallels between the two realms. Just as in the study of the marketplace, behavioral factors (B) modify economic models (E) demonstrating behavioral economics (BE) effects; similarly, in cyberspace, behavioral economics models (BE) of cognitive biases can enhance cybersecurity (C) in the proposed framework of Behavioral Economics of Cybersecurity, or BEC.

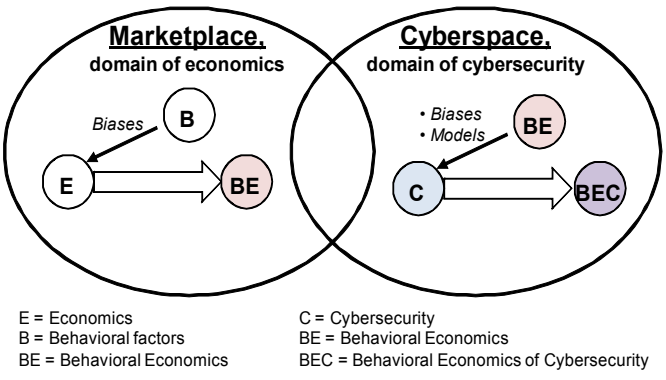


Figure 1. Analogy between BE and BEC studies

Section 2 describes how BEC fits into the existing cybersecurity work by extending the Risk Management Framework with persistent human biases. Section 3 defines principal cyberactors whose biases are addressed by BEC and breaks down actors' activities into cybersecurity perspectives, thus producing the overall structure of the BEC framework. Section 4 further refines categories of BEC actors and provides examples of how common cybersecurity exploits map into the BEC. Section 5 provides some examples of applying BE findings to BEC.

2 BEC Extension of the RMF

At the intersection of economics and cybersecurity lies risk, which creates a symbiotic relationship between the two realms. In the marketplace, risk is increasingly associated with cyberspace activities; and in cybersecurity, risk management is the principal economic model. In economics, there are as many definitions of risk as there are economists, but risk calculation always comes down to the product of the probability of an event and its impact. For example, Bodie provides an economics definition of risk as follows, "…investment risk is uncertainty that matters. There are two prongs to this definition—the uncertainty and what matters about it—and both are significant" (Bodie & Taqqu, 2011, p.50). In this definition, "the uncertainty" represents the probabilistic component, and "what matters" is the impact. In both economics and cybersecurity, human biases increase the probability component of risk. This section describes proposed modifications of the current Risk Management Framework (RMF) by incorporating into it human biases as a new class of vulnerabilities.

In the United States, the RMF developed by the National Institute of Standards and Technology (NIST) is an authoritative source on risk in cyberspace. NIST does not define risk but describes Risk Management as "a comprehensive process that requires organizations to: (i) frame risk (i.e., establish the context for risk-based decisions); (ii) assess risk; (iii) respond to risk once determined; and (iv) monitor risk on an ongoing basis" (NIST SP 800-39, p.6). The calculation of risk takes place in the Risk Assessment phase of the RMF that aims

> "to identify, estimate, and prioritize risk to organizational operations … resulting from the operation and use of information systems. The purpose of risk assessments is to inform decision makers and support risk responses by identifying: (i) relevant threats to organizations or threats directed through organizations against other organizations; (ii) vulnerabilities both internal and external to organizations; (iii) impact (i.e., harm) to organizations that may occur given the potential for threats exploiting vulnerabilities; and (iv) likelihood that harm will occur. The end result is a determination of risk (i.e., typically a function of the degree of harm and likelihood of harm occurring)" (NIST SP 800-30 Rev. 1, 2012, p.1).

The notion of cyber risk as a function of threats, vulnerabilities and impact is conceptualized by Nichols, Ryan & Ryan in a formula "Level of Risk = (Threat x Vulnerability) x Impact / Countermeasures" (2000, p.70), where Threats correspond to components of risk posed by hostile organizations or individuals, and Vulnerabilities are characteristics of friendly systems that constitute flaws exploitable by the threats. These relationships can be expressed as follows:

R = I * Pr / C, i.e., Risk (R) is Impact (I) times Probability of the incident occurrence (Pr), reduced by Countermeasures or Controls (C); and

Pr = T * V, i.e., Pr is the product of Vulnerabilities (V) and Threats (T) that could exploit them.

Thus, cyber risk calculation is essentially the same as that used in the economic risk calculation.

NIST SP800-30 rev. 1 enumerates vulnerabilities related to organizational, business, and technical issues (2012, Table F-1, p.F-1), however, the human vulnerability is missing from the current NIST description. If people are as predictably irrational as behavioral economists have repeatedly demonstrated, then cognitive biases represent a persistent source of vulnerabilities and should be incorporated in the RMF.

Significantly, the human side of cybersecurity is recognized in the Information Assurance Technical Framework (IATF) that defines the three components of Defense In Depth (DID) as people, operations and technology (2000, p.ES-1). In comparison, the current approach to risk management is focused only on operations and technology. BEC seeks to establish the human component in the cyber risk framework as shown in Figure 2.

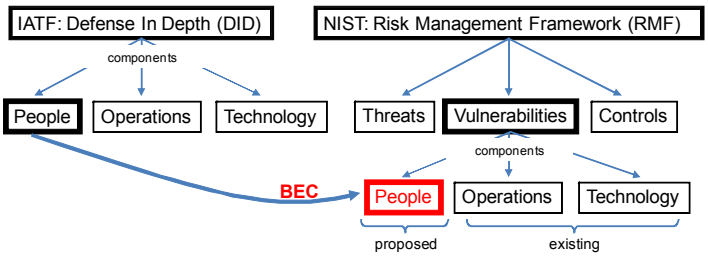


Figure 2. BEC alignment of DID and RMF

3 BEC Framework Structure

Human actors play various roles in cyberspace, and their biases affect different areas of risk. For the purpose of BEC, the three principal categories of cyberactors are Users, Defenders and Attackers. Users utilize cyberspace services and functions. Defenders create, operate and strengthen cyberspace. Attackers aim to penetrate cyber defenses and to exploit Users' systems and data. The irrationalities of Users and Defenders represent cyber-system vulnerabilities; the irrationalities of Attackers are potential opportunities to mitigate threats. Table 1 summarizes normal threats and vulnerabilities posed by cyberactors as well as their mitigation using BEC-based countermeasures.

Table 1. Risk components in BEC

Cyberactors	Risk Assessment		Risk Response
	Threats	Vulnerabilities	BEC Countermeasures / Controls
Users		*Irrational* behavior	*Mitigation* of irrational behavior
Defenders		*Irrational* behavior	*Mitigation* of irrational behavior
Attackers	*Normal* behavior		*Inducement* of irrational behavior

BEC affects two aspects of Risk Management, Risk Assessment and Risk Response. In Risk Assessment, BEC specifically adds focus on human vulnerabilities; and in Risk Response, BEC controls the human-related risk by using all three components of the DID, i.e., people, operations, and technologies. In fact, when the causes of human decision-making biases are understood, operational and technical countermeasures are frequently more effective than purely human countermeasures such as training people or raising their awareness. Table 2 provides examples of human, operational and technical countermeasures that control various vulnerabilities and threats, emphasizing those caused by people.

This paper refines the BEC approach further by presenting it similarly to Zachman's (1997) framework that organizes modeling artifacts along two dimensions, representative stakeholders and a set of perspectives. In the BEC framework, the stakeholders are Users, Defenders and Attackers, and the perspectives on their irrationality are the security services they undermine, i.e., Confidentiality, Integrity and Availability. Table 3 shows the structure of this framework.

Table 3. Structure of the BEC framework

	Confidentiality	Integrity	Availability
Users			
Defenders			
Attackers			

The BEC framework will help decision makers prioritizing security services that humans put at risk and selecting corresponding countermeasures. The focus could be different for government agencies whose highest priority is protecting Confidentiality of their information, utilities primarily concerned with maintaining Integrity of their Supervisory Control And Data Acquisition (SCADA) systems, and online businesses for whom Availability is the matter of survival. Section 4 provides an expanded approach to BEC and Figure 3 shows the overall view of the BEC framework[1].

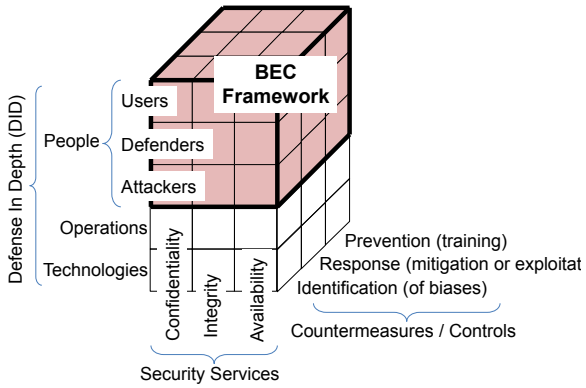


Figure 3. Complete BEC framework

1 This representation of BEC is an expansion of the *Rossi Cube created by the National Defense University, iCollege professor John R. Rossi.*

Table 2. Human vulnerabilities and corresponding countermeasures

Risk assessment (emphasis on people)		BEC Countermeasures/Controls (all DID areas)		
		People	Operations	Technologies
Vulnerabilities	People (Users, Defenders)	**Bias Mitigation**		
		Awareness, cognitive training	Policies, scripts, playbooks	Electronic reminders, gates, checks
	Operations	**(Existing mechanisms)**		
	Technologies			
Threats	People (Attackers)	**Bias Inducement**		
		Pretense of gullible employees	Pretense of important processes	Technical decoys

The BEC framework can be applied flexibly. Some organizations may focus on the threat entry points where human biases manifest themselves. For example, Users jeopardizing system Integrity by choosing weak passwords would be treated as the vulnerability of the User-Integrity (U-I) block of the framework. Others may consider the risk of human vulnerabilities primarily based on their impacts. For example, if Users' weak passwords ultimately put at risk the information Confidentiality, the focus would be on the mitigation of the User-Confidentiality (U-C) block.

4 Refinement of the BEC Framework

The general principles of BEC can be further refined by illustrating how known cyberactor biases map into this framework. Table 4 expands Table 3 with examples of Users' and Defenders' vulnerabilities and of Attackers' goals.

Table 4. Examples of BEC applicability

Cybersecurity Perspective / Cyberactor		Confidentiality	Integrity	Availability
Vulnerabilities (what Users and Defenders do)				
User	Individual	Self-disclosure in social media, response to phishing, poor Wi-Fi security, geolocation, loss of mobile devices	Poor anti-virus and firewall protection, poor Wi-Fi security, unprotected or weak passwords, use of pirated software, "risky" web surfing, excessive complexity of the digital life	Rare or no backups
	Organizational	Social engineering, improper discard of records, comingling of business and personal devices, malicious insider actions	Complexity in requirements, comingling of business and personal devices, insiders	Inadequate protection of data, blackmail of online bookmakers
	External	Medical and financial records	ISP, cloud providers	Botnet-based DDoS
Defender	Developer	Backdoors, development errors	Poor coding, errors in testing and integration	n/a
	Operator	Insiders, operational errors	Poor security architecture, overwhelming log data, complex processes, malicious insider actions	Attacks causing shut down, operational errors, poor COOP and contingency planning
Attack goals (what Attackers are after)				
Attacker	Foreign government	Diplomatic and intelligence secrets	Political pressure	Political pressure, blackmail by blocking cyber access
	Foreign military	Strategic military secrets	Disruption of tactical military operations	Disruption of tactical military operations
	Non-state combatant			
	Business	Business secrets, customer data	Sabotage of competitors	Sabotage of competitors
	Criminal	Exploitable business and individual secrets	Distortion of business operations and data	Distortion of business operations
	Hacker	Government, military, and business secrets	Government, military, and business penetration	Government, military, and business disruption
	Terrorist	Information	SCADA	SCADA

In the Table 4 examples, the primary types of the BEC actors, Users, Defenders and Attackers, are further classified into subcategories. The characteristics of these categories are as follows.

- **Users.** Vulnerabilities are associated with the use of cyber resources as they carry their business and leisure activities.
  - *Individual.* These users include individuals and small businesses in which personal and business technologies are not separated. Individuals choose and install their own hardware and software and provide limited defense of their own cyber environment. Their vulnerabilities are self-inflicted.
  - *Organizational.* These users differ from individual users in that their biases produce vulnerabilities for their organizations thus creating the principal-agent problem.
  - *External.* Vulnerabilities (externalities) that individual and organizational users create for the third parties, e.g., user devices joining botnets used to attack external targets.
- **Defenders.** Vulnerabilities are associated with the protection of cyber resources.
  - *Developer.* Vulnerabilities are introduced before a system becomes operational.
  - *Operator.* Vulnerabilities are created in the process of defending an operational system. These vulnerabilities also include unintended consequences of organizational policies and procedures.
- **Attackers.** Among various types of attackers, the following entities present distinct threats and may be controlled by different human-focused countermeasures: foreign government, foreign military, non-state combatant, business, criminal, hacker, and terrorist.
- **[Insiders.]** From the cybersecurity perspective, insiders act as attackers. However, their behavior is influenced and mitigated by their organizations, and thus they are considered as organizational users.

5 Examples of Applying BE to BEC

5.1 Defender biases in Risk Assessment and Risk Response

In the seminal Behavioral Economics paper Judgment under uncertainty: Heuristics and biases (1974), Amos Tversky and Daniel Kahneman described a heuristic adjustment from an anchor and the resulting biases of overestimating the probability of conjunctive events and underestimating the probability of disjunctive events. Influenced by the first of these biases, the subjects who were given a high probability (90%) of a certain event, were consistently betting that the probability of this event occurring seven times in succession was still high. The second bias is complementary; after the subjects were given a low probability (10%) of a certain event, they were consistently betting that the probability of this event occurring at least once in seven consecutive experiments was still low.

By extension, in cyber Risk Assessment, underestimating the probability of disjunctive events may lead to understating the risk when low-probability threats are introduced multiple times in a disjunctive fashion. Likewise, in cyber Risk Response, a series of conjunctive controls may be psychologically anchored on high effectiveness of individual controls without the recognition of a lower level of protection that they provide collectively. Underestimation of risks and overestimation of controls are likely to influence intuitive security decisions. However, these errors of judgment may also affect formal Risk Management processes in which decision makers have some preconceived notions. This usually happens due to the confirmation bias, a human trait of seeking and emphasizing the information that supports existing beliefs.

5.2 Defender and User biases undermining Integrity

In the book The (honest) truth about dishonesty, Dan Ariely (2012) describes the Behavioral Economics of honesty and dishonesty, some of which may have profound impact on cybersecurity. For example, the experimental findings that people cheat less when they are given timely reminders of ethical standards (p.41) could be used for preventing some Insider actions and warning Organizational Users against violating Acceptable Use Policies (AUP).

Financial conflicts of interest are exacerbated when financial instruments are complex, people are not dealing with real money, and all their colleagues are committing similar offenses (Ariely, 2012, p.84). Some of these situations can be prevented by standard measures such as separation of duties of financial advisers and financial managers (p.94). A similar potential for conflict of interest exists in cyberspace, and the International Information Systems Security Certification Consortium recommends administrative controls such as separation of duties of the individuals who are requesting and authorizing critical actions and expenditures; those performing backups and restoration; application developers of the development, testing and production environments; and personnel in other areas where abuse is likely ((ISC)2®, 2010, pp.12-13). In the context of the BEC framework, current (ISC)2's recommendations can be fine-tuned for specific cognitive biases.

Frequently people break rules under excessive cognitive load when they have so much on their mind that there is little room for resisting temptation (Ariely, 2012, p.99). Experiments show that "when our deliberative reasoning ability is occupied, the impulsive system gains more control over our behavior" (p.100).

In cyberspace, this phenomenon is particularly dangerous for Defenders whose cognition may be depleted by long shifts and continuing vigilance, thus leading to errors in judgment such as missing alarms or opening phishing messages. When these scenarios are well understood, it may turn out that the most appropriate cyber controls are not additional education and training but technical and operational reminders, checks, and constraints.

Altruistic cheating is a paradoxical phenomenon making "it easy for group-based processes to turn collaborations into cheating opportunities in which individuals cheat to a higher degree because they realize that their actions can benefit people they like and care about" (Ariely, 2012, p.222). In cybersecurity this bias may work against standard anti-collusion measures. Furthermore, people's desire to be nice and social is at heart of many successful social-engineering attacks. Ariely points out that "those in the spotlight: politicians, public servants, celebrities, and CEOs" should be particularly diligent in setting the right example, but in reality, they "are too often rewarded with lighter punishments for their crimes than the rest of the population" (p.215). Edward Amoroso, the Chief Security Officer of AT&T, observes a similar phenomenon in cybersecurity describing the symbolism of the managers' behavior and the perils of "executive exemption" when the most senior executives evade security controls and often commit security policy violations (Amoroso, 2011, pp.125-126). The issue is three-fold: the executives are the primary targets of attacks, they are frequently less knowledgeable about technology than their workforce, and their staff is reluctant to enforce the policies. Amoroso calls for "major national infrastructures solicitations" to support security staff in their efforts to control executives who outrank them. A formal, scientifically supported BEC framework will help reinforcing security at the highest organizational levels.

6 Significance of BEC

The marketplace and cyberspace alike are vulnerable to extreme occurrences that defy all past expectations. Nassim Taleb calls this type of an event a Black Swan and defines it by its three characteristics: "first, it is an outlier, as it lies outside the realm of regular expectations, because nothing in the past can convincingly point to its possibility. Second, it carries an extreme impact. Third, in spite of its outlier status, human nature makes us concoct explanations for its occurrence after the fact, making it explainable and predictable" (Taleb, 2010, p.xxii). In an earlier paper (Fineberg, 2012), I described the impact of Black Swans on the Continuity Of Operations Planning (COOP), but Black-Swan risks extend to all corners of cyberspace, and the susceptibility to concocting post factum explanations without accounting for the extreme nature of these events represents a persistent cognitive bias.

Philosophically, Taleb attributes Black Swans to Platonicity defined by him as the human propensity for categorizing data and substituting complex reality with its models. Platonicity misleads people to "mistake the map for the territory, to focus on pure and well-defined 'forms,' whether objects, like triangles, or social notions" (p.xxix). While "these intellectual maps of reality are not always wrong," the greatest danger is in the Platonic Fold, "the explosive boundary where the Platonic mindset enters in contact with messy reality, where the gap between what you know and what you think you know becomes dangerously wide. It is here that the Black Swan is produced" (p.xxx).

From this perspective, Behavioral Economics bridge the Platonic Fold between the pure, well-defined models of standard economics and the messy reality of the human psyche. Likewise, Behavioral Economics of Cybersecurity (BEC) bridge the Platonic Fold between the theoretical principles of cyberspace and the messy reality of the cyberhuman, its most susceptible link. Figure 4 illustrates the Platonic Fold concept and the roles of BE and BEC in bridging it.

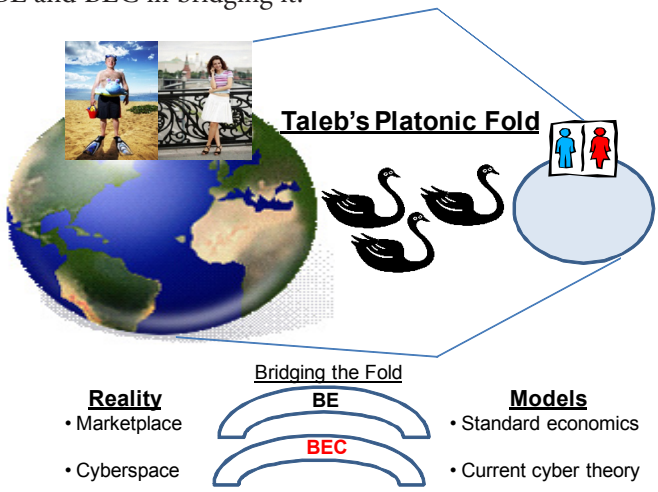


Figure 4. Platonic Fold of the marketplace and cyberspace

The significance of BEC is in reducing the potential for Black Swans of cyberspace.

7 Conclusion

Behavioral Economics experiments have firmly established that people exhibit common biases in their judgment and decision making and persistently violate assumptions of the rational actor model. Yet, cyber strategies, policies and risk management guidance are still geared towards rational cyberactors. This paper proposes to incorporate BE findings into the realm of cybersecurity by creating a new framework called BEC.

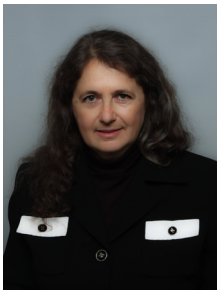
The paper demonstrates how the BEC framework can be integrated into the current NIST's Risk Management Framework and how

it can be structured as a matrix of cyber actors (Users, Defenders, Attackers) and security services (Confidentiality, Integrity, Availability). Examples of the BEC applicability are provided. The awareness and mitigation developed within BEC would smooth Taleb's Platonic Fold between the theory and practice of a human in cyberspace and mitigate the risk of future Black Swans.

8 Acknowledgment

The author greatly appreciates the comments provided by Yeva F. Byzek, David R. Harris, Alicia M. Martin, and John A. Wasko.

9 About the Author



Victoria Fineberg is a Principal Information Assurance Engineer at the Defense Information Systems Agency (DISA). She is a Certified Information Systems Security Professional (CISSP) and has completed Chief Information Officer (CIO) and Chief Information Security Officer (CISO) programs at the National Defense University's (NDU) iCollege. Victoria holds a Masters Degree in Mechanical Engineering from the University of Illinois at Urbana-Champaign, is a licensed Professional Engineer and a Senior Member of IEEE. Prior to DISA, Victoria worked for Bell Labs at Lucent Technologies. Her professional interests include cyber security, risk analysis, and the impact of cognitive biases on cyber operations.

10 References

Amoroso, E. (2010). Cyber attacks: *Protecting national infrastructure*. Burlington, MA: Butterworth-Heinemann.

Ariely, D. (2012). *The (honest) truth about dishonesty: How we lie to everyone—especially ourselves.* New York, NY: HarperCollins Publishers.

Bodie, Z. & Taqqu, R. (2011). *Risk less and prosper: Your guide to safer investing.* Indianapolis, IN: John Wiley & Sons.

Fineberg, V. (2012). COOP hardening against Black Swans. T*he Business Continuity and Resiliency Journal*. 3Q. http://www.businesscontinuityjournal.com/.

IATF Rel 3. (2000). *Information Assurance Technical Framework.* Retrieved from http://www.dtic.mil/cgi-bin/GetTRDoc?Location=U2&doc=GetTRDoc.pdf&AD=ADA393328.

(ISC)²®. (2010). *Official (ISC)2® guide to the CI SSP® CBK.* Second Edition. H. F. Tipton, Editor. Boca Raton, FL : Auerbach Publications.

Nichols, R.K., Ryan, D. J., & Ryan, J. C. H. (2000). *Defending your digital assets against hackers, crackers, spies & thieves.* New York, NY: McGraw-Hill.

NIST SP 800-30 Rev. 1. (2012). *Guide for conducting risk assessments.* Retrieved from http://csrc.nist.gov/publications/nistpubs/800-30-rev1/sp800_30_r1.pdf.

NIST SP 800-39. (2011). *Managing information security risk.* Retrieved from http://csrc.nist.gov/publications/nistpubs/800-39/SP800-39-final.pdf.

Taleb, N. N. (2010). T*he Black Swan: The Impact of the Highly Improbable.* New York, NY: Random House.

Tversky, A. & Kahneman, D. (1974). Judgment under uncertainty: Heuristics and biases. *Science, New Series,* 185(4157), pp. 1124-1131.

Zachman, J. A. (1997). *Concepts of the framework for enterprise architecture: Background, description and utility.* Zachman International. Retrieved from http://www.ies.aust.com/PDF-papers/zachman3.pdf.

Article Submission Policy

The CSIAC Journal is a quarterly journal focusing on scientific and technical research & development, methods and processes, policies and standards, security, reliability, quality, and lessons learned case histories. CSIAC accepts articles submitted by the professional community for consideration. CSIAC will review articles and assist candidate authors in creating the final draft if the article is selected for publication. However, we cannot guarantee publication within a fixed time frame.

Note that CSIAC does not pay for articles published.

AUTHOR BIOS AND CONTACT INFORMATION

When you submit your article to CSIAC, you also need to submit a brief bio, which is printed at the end of your article. Additionally, CSIAC requests that you provide contact information (email and/or phone and/or web address), which is also published with your article so that readers may follow up with you. You also need to send CSIAC your preferred mailing address for receipt of the Journal in printed format. All authors receive 5 complementary copies of the Journal issue in which their article appears and are automatically registered to receive future issues of Journal

COPYRIGHT:

Submittal of an original and previously unpublished article constitutes a transfer of ownership for First Publication Rights for a period of ninety days following publication. After this ninety day period full copyright ownership returns to the author. CSIAC always grants permission to reprint or distribute the article once published, as long as attribution is provided for CSIAC as the publisher and the Journal issue in which the article appeared is cited. The primary reason for CSIAC holding the copyright is to insure that the same article is not published simultaneously in other trade journals. The Journal enjoys a reputation of outstanding quality and value. We distribute the Journal to more than 30,000 registered CSIAC patrons free of charge and we publish it on our website where thousands of viewers read the articles each week.

FOR INVITED AUTHORS:

CSIAC typically allocates the author one month to prepare an initial draft. Then, upon receipt of an initial draft, CSIAC reviews the article and works with the author to create a final draft; we allow 2 to 3 weeks for this process. CSIAC expects to have a final draft of the article ready for publication no later than 2 months after the author accepts our initial invitation.

PREFERRED FORMATS:

- Articles must be submitted electronically.
- MS-Word, or Open Office equivalent (something that can be edited by CSIAC)

SIZE GUIDELINES:

- Minimum of 1,500 – 2,000 words (3-4 typed pages using Times New Roman 12 pt font) Maximum of 12 pages
- Authors have latitude to adjust the size as necessary to communicate their message

IMAGES:

- Graphics and Images are encouraged.
- Print quality, 300 or better DPI. JPG or PNG format preferred

Note: Please embed the graphic images into your article to clarify where they should go but send the graphics as separate files when you submit the final draft of the article. This makes it easier should the graphics need to be changed or resized.

CONTACT INFORMATION:

CSIAC
100 Seymour Road Suite C102
Utica, NY 13502
Phone: (800) 214-7921
Fax: 315-351-4209

John Dingman, Managing Editor
Email: jdingman@quanterion.com

Michael Weir, CSIAC Director
Email: mweir@quanterion.com

CSIAC is a DoD sponsored Information Analysis Center (IAC), administratively managed by the Defense Technical Information Center (DTIC), technically managed by the Air Force Research Laboratory (AFRL) in Rome, NY and operated by Quanterion Solutions Incorporated, Utica, NY.

ABOUT THE JOURNAL OF CYBER SECURITY AND INFORMATION SYSTEMS

CSIAC JOURNAL EDITORIAL BOARD

Distribution Statement
Unclassified and Unlimited

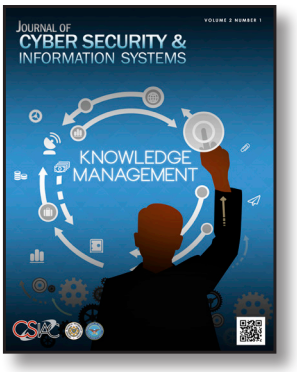
CSIAC
100 Seymour Road
Utica, NY 13502-1348
Phone: 800-214-7921 • **Fax:** 315-732-3261
E-mail: info@thecsiac.com
URL: http://www.thecsiac.com/

ABOUT THIS PUBLICATION

The Journal of Cyber Security and Information Systems is published quarterly by the Cyber Security and Information Systems Information Analysis Center (CSIAC). The CSIAC is a DoD sponsored Information Analysis Center (IAC), administratively managed by the Defense Technical Information Center (DTIC). The CSIAC is technically managed by Air Force Research Laboratory in Rome, NY and operated by Quanterion Solutions Incorporated in Utica, NY.

COVER DESIGN

Shelley Howard
Graphic Designer
Quanterion Solutions, CSIAC



ARTICLE REPRODUCTION

Images and information presented in these articles may be reproduced as long as the following message is noted:

"This article was originally published in the Journal of Cyber Security and Information Systems Vol.II, No 1"

In addition to this print message, we ask that you notify CSIAC regarding any document that references any article appearing in the *CSIAC Journal.*

Requests for copies of the referenced journal may be submitted to the following address:

Cyber Security and Information Systems
100 Seymour Road
Utica, NY 13502-1348

Phone: 800-214-7921
Fax: 315-732-3261
E-mail: info@thecsiac.com

An archive of past newsletters is available at **https://journal.thecsiac.com.**

Cyber Security and Information Systems Information Analysis Center
100 Seymour Road
Suite C-102
Utica, NY 13502

Return Service Requested

Journal of Cyber Security and Information Systems – Volume II Number 1
Knowledge Management

— IN THIS ISSUE —