Building Secure, Resilient Architectures for Cyber Mission Assurance

Harriet G. Goldman

> *"You are going to be attacked; your computers are going to be attacked, and the question is, how do you fight through the attack? How do you maintain your operations?"* – Lt Gen Ted Bowlds, Commander, ESC, 28 Jan 09

Motivation/Background

Today's information technology (IT) environments are increasingly subject to escalating cyber attacks. Cyber threats vary widely in sophistication, intent, and the consequences to the targeted systems and networks. The range of attackers extends from users who unintentionally damage systems to hackers, to cyber criminals, to full-scale cyber spies and cyber warriors; their intentions span from annoying vandalism to economic threats to taking out the electric grid or defeating armed forces. Similarly, the target of the attacks can vary from a single computer or router to an entire on-line banking system, business enterprise, or global supply chain. At the same time, our missions and businesses fall along a spectrum of criticality—from desirable to necessary, essential, and mission or safety critical. Given the broad spectrums of threat, intent, and consequence to mission-critical functions, determining exactly where our mission systems lie in this continuum of dimensions is vital to determine the appropriate level of investment and response.

The notion that we can achieve 100% protection is not only unrealistic but also results in a false sense of security that puts our missions and businesses at serious risk. Consequently, we must compensate for our inability to achieve full protection by ensuring that we can accomplish our missions despite cyber attacks. The cyber defenses generally available today help address the low-end threats against our less essential systems, but are often ineffective against most forms of cyber attacks targeting our most mission-critical systems. It is at the high end of the continuum that architecture resilience will matter most—to enable continuity of mission critical operations and support rapid reconstitution of existing or minimal essential capabilities or the deployment of alternative means of accomplishing the mission.

This paper offers ideas along the full spectrum of cyber security, but concentrates on architectural resilience against the upper end of the spectrum, where the stakes are high, the mission or business is critical, and the adversary is sophisticated, motivated, and persistent. However, many of the same techniques are valuable at the low to medium levels of threats and consequences because they can significantly reduce the operational impact and cost of cleanup after an attack. Even if the intentions and consequences of the threat are currently not very serious, we must keep in mind that today's massive data thefts or passive reconnaissance can quickly escalate into data and system modification, surreptitious commandeering of control, or denial of essential services with far more dire mission impact in the future.

The cyber adversary continues to have an asymmetric advantage as we fruitlessly play Whac-A-Mole in response to individual attacks. To reduce the adversary's advantage, we must proactively rearchitect our systems to impede or neutralize attacks and diminish their impact and consequences. While we cannot stop all attacks or make them totally ineffective, rearchitecting for resilience will make adversaries' attacks less likely to succeed, will minimize consequences to critical operations when they do succeed, will increase the adversary's cost and uncertainty, and may act as a deterrent against future attacks.

Recent events demonstrate that Government is increasing its attention to resilience. While these actions clearly indicate senior leaders understand the importance of resilience, we are just beginning to understand what it means to turn the concept into practice. Much work is needed to define and validate resilience: techniques and strategies; policies to promote operational and system resilience; risk decision methodologies, analytic processes, and acquisition guidance; and metrics for measuring resilience improvements and evaluating progress. Moreover, funding must be aligned to budget cycles to reflect these needs and build momentum.

Recent Government actions addressing resilience:

- *The National Infrastructure Protection Plan: Partnering to Enhance Protection and Resiliency*, produced by DHS last year, placed greater emphasis on the importance of building resilience into critical systems, especially IT systems.
- Darrell Darnell was appointed to a post on the White House National Security Staff in the newly created Office on Resilience in October 2009.
- Resilience was listed as one of the five homeland security missions in the recently published *Quadrennial Homeland Security Review*.
- Goal 4 of the *OSD/NII DoD IA Strategic Plan* is "Prepare for and operate through cyber degradation or attack."
- The FAA just initiated a Commercial Space Transportation Grant Program to ensure the resilience of the United States space transportation infrastructure.

However, game-changing technologies, techniques, and strategies can make transformational improvements in the resilience of our critical systems. This paper explores the art of the possible from which to begin evaluating the viability of promising strategies and techniques for resilience, singularly and in combination, to determine which are the most cost-effective to pursue. Some of the suggestions are already commonly embraced and in practice, whereas other notions are new and speculative. Often these new approaches cost more, but sometimes they reduce costs or improve reliability and thus may be part of the business justification. Decisions on how to proceed must weigh the cost and impact of failed critical operations against the cost and benefits of incorporating resilience. More important, in today's environment of sharply increasing cyber threats, these approaches can make the difference between success and failure, between life and death. To do nothing is to accept defeat and pay the price in terms of failed missions and business objectives.

Defining Resilient Architectures

Goals and Objectives

The term *resilience* has many definitions depending on the context and application. For a computing paradigm, the simple definition from the University of Kansas's ResiliNets Project proves most useful: "Resilience is the ability to provide and maintain an acceptable level of service in the face of faults and challenges to normal operation."[1] Resilience is related to survivability, which builds on the disciplines of security, fault tolerance, safety, reliability, and performance. This paper focuses on how to achieve resilience in our mission-critical computing environments against specific patterns of cyber attacks. It covers recommendations for how critical processing systems should be designed, deployed, and operated to support adaptation, scaling, replacement,

[1]University of Kansas ResiliNets Wiki and Wikipedia, "Resilience is the ability to provide and maintain an acceptable level of service in the face of faults and challenges to normal operation." https://wiki.ittc.ku.edu/resilinets_wiki/index.php/Definitions (accessed on 11 August 2010).

reconfiguration, and recovery in the event of an unexpected disruption, degradation, or compromise of critical data, system components, or services.

Improved technology, architectural advances in modularity, integration, standards, and service-orientation, and new distributed processing paradigms facilitate the creation of resilient architectures. At the same time, resilience is challenged by the last decade of unconstrained connectivity and business cost-cutting measures, which have resulted in extensive use of homogeneous, commercial-off-the-shelf (COTS) hardware and software, unknown interdependencies and connections, and reliance on outsourced services and network infrastructures whose pedigree, development life-cycle, and systems management and operations are out of an organization's control. Therefore, the question of how to achieve resilience is part of a larger question: how to construct resilient systems from components whose resilience characteristics may be limited, unknown, and possibly unknowable; and how to express resilience characteristics so that we can evaluate and measure them in desired capabilities throughout the system development life-cycle.

While this paper focuses on architectural strategies, attention to operator resilience is crucial to mission success given all the unknown unknowns. We need to improve our woefully inadequate training approaches and to exercise under realistic cyber intrusions. Traditional techniques derived from continuity of operations (COOP), disaster recovery (DR), operator training, and red team exercises remain vital to ensure that we are prepared to respond during a failure or natural crisis. They also play a crucial role in validating technical and operational resilience of the architecture to such events. However, as implemented and practiced, these techniques do not address current cyber attacks and must be revisited.

Similarly, implementing best security practices is insufficient. We must change our current philosophy of assuming we can either keep adversaries out or detect their breaches of our first-line defenses. Instead, we must assume some adversary success. Too often we do not know the cause of anomalous system behavior when we detect it or are not even aware that an attack is in progress. Consequently, we need to design systems that are more agile and adaptive AND more resistant and resilient to compromise.

> *"It is not the strongest species that survive, nor the most intelligent, but the ones most responsive to change" –* Charles Darwin

The goal for improving the resilience of our architectures is not to seek perfect protection against advanced threats—a goal that is elusive at best. Instead, it is to adopt and implement design strategies and techniques that support a balanced combination of protections, detections, and adaptive technical and operational responses that dynamically evolve in response to current and future cyber events.

The objectives fall into two categories: to make architectures more resistant to unintentional incidents and targeted attacks, and to make them more resilient to initial and subsequent compromise. The first category creates a more secure and resistant foundation and helps inform and trigger operational responses. This category of objectives can be met by techniques to protect systems and deter attackers, and to detect compromises when possible. In many cases we may already be addressing these objectives to some extent, but we must either do this more or do it better. The second category focuses on altering the environment in order to constrain or isolate critical capabilities and data, to reduce consequences, and to support agility and adaptive responses. Collectively the following five objectives can help achieve architecture resilience.

1. **PROTECT/DETER** by disincentivizing the adversary and raising the protection level of our systems. We cannot expect inadequately protected systems to be resilient, any more than we can hope to achieve

functional assurance from spaghetti code. First-order actions are to protect our critical infrastructure applications, services, and data as best we can against both known and possible cyber threats by:

> *"[w]e may improve deterrence if we…ensure resiliency and continuity of service. If opponents believe …attacks will have little effect, they will be less likely to launch them."* – Securing Cyberspace for the 44th Presidency (December 2008)

- Incorporating security technology and operational best practices (for confidentiality, availability, and integrity)
- Reducing the number and severity of vulnerabilities
- Making our IT systems more trustworthy by applying high-assurance techniques
- Adhering to fundamental security principles of policy enforcement**,** least privilege, and simplicity and modularity for trust analysis
- Designing for high availability, integrity, confidentiality, safety, agility, and scalability

Disincentives to attackers can be introduced by incorporating design strategies and impediments that:

- Make the attacker's task more difficult, costly, time-consuming, or uncertain;
- Increase the likelihood that the attacker will be detected;
- Disrupt or neutralize attacks; and
- Confuse the attacker and introduce deception and unpredictability.

2. **DETECT/MONITOR** those attacks and abnormalities that can be discovered to reveal intrusions and gain situational awareness (SA) to inform our responsive operational strategies. While we cannot always detect advanced exploitations, we can improve our capabilities and continue to extend them on the basis of after-the-fact forensic analysis. Recognizing degradations, faults, intrusions, etc., or observing changes or compromises can become a trigger to invoke contingency procedures and strategies. A fundamental prerequisite is deliberate placement of sensors that can provide in-depth coverage across the environment to monitor critical processing and flows.

   Many forms of monitoring are needed across the layers of the architecture (e.g., event monitoring, traffic analysis, identification of user and system anomalous behavior, audit analysis, and forensics). Analyzing and correlating operational measurements of performance, capacity, response time, latency, and processor health metrics, in combination with monitoring of misuse, abuse, attacks, exfiltration, malicious code, and modifications, will improve SA. Subtle abnormal changes are important because we have no assurance that our protective measures can fully deter the cyber adversary.

   Because we cannot predict the adversary's changing tactics, techniques, and procedures (TTPs), we need dynamic sensor deployment models that support active sensor tuning and deployment adjustments in real time based on early warning alerts or an incident. Monitoring adversarial action, rather than shutting it down, can provide an opportunity to learn the attacker's TTPs. Collecting data during an attack for later forensic analysis can enlighten us about new or improved detection, protection, or deterrent capabilities to develop for the future. The challenge is to do so undetected while still containing the damage as the attack is happening.

3. **CONSTRAIN/ISOLATE** interfaces and functional capabilities. System developers must apply design approaches that separate functions, data, and network segments in order to isolate critical assets and

problems, minimize damage and impact propagation, and allow better forensics and analysis. Isolation should ensure that some portions of the system continue to function even if others do not. Some examples are:

- Separate critical from non-critical processing and data
- Isolate an intranet from an extranet from the Internet
- Partition processing, access points, data, and network segments
- Separate inbound from outbound traffic
- Separate requests from responses
- Isolate faults
- Constrain propagation when attacks succeed

The concept of separation is not new. It constitutes the basis of a security kernel used to separate the security policy enforcement portion of the operating system (OS) from the rest of the OS. In an extreme case, this can be equated to a standalone system in a physically protected data center or isolated computing enclaves for executing highly sensitive work. For the power grid, isolation can take the form of *islanding*, a term used to denote a situation where the distributed generation generator continues to power a location even though the electric utility is not functioning. In computer networks, separation may equate to segmentation or to a hardware switch that delivers just-in-time connectivity of limited duration when needed.

4. **MAINTAIN AND RECOVER** operations for minimal essential capabilities. Maintaining critical operations means first distinguishing essential from non-essential capabilities, understanding dependencies among components, and performing contingency planning activities. Planning should address possible degradation in capacity and performance, denial of service, and corruption of data, hardware, and software processing. Good planning also calls for building fine-grained, adaptive manageability and configurability into system designs and supporting alternative operational capabilities and/or functionality for times when normal critical processing capabilities are under attack.

   Highly modular architectures, boundary devices, and administrative and management interfaces form the underpinnings for dynamic contingency operations. For contingency planning, administrative and network operational capabilities must support rapid and automated replication, scaling, failover, reconfiguration, recovery, reconstitution, replacement, relocation, and initiation of critical services or alternative services in a distributed environment. When attacks succeed despite COOP and contingency operations, rapid, dynamic discovery and composition of alternative services/capabilities that are interoperable with existing capabilities may be needed, in addition to the ability to return to a trusted state within a reasonable timeframe.

5. **ADAPT** continuously in response to escalating attacks and changing threats or risk posture, and as a proactive step to foil exploits. By introducing technical, defensive, and operational change management into the system, system designs can potentially foil an attacker's exploit and/or confuse the adversary by adding an element of surprise, uncertainty, or unpredictability. For example, continuous change through randomization, adaptive computing, self-healing, and other adaptive techniques, or simply leveraging emerging technologies or new computing paradigms as an early adopter or in novel ways, can provide covertness, unpredictability, and resilience for a period of time.

While autonomic adaptive responses have the advantage of reacting at computing speeds, sophisticated attackers can also use them against us. Ultimately it is the operator's ability to understand the mission and situation and determine the best response in the heat of a crisis that will save the day. There is no substitute for trained, experienced operators who can readily adapt and respond to an unexpected situation.

Characteristics

To remain resilient while under cyber attack, our architectures must be evolved or at times radically redesigned to exhibit many of the functional and technical characteristics and properties summarized in the table below.

Objective	Resilient Architectural Characteristics and Properties
Protect/Deter	Implements security best practices
	Minimizes the loss or corruption of services or data
	Tolerates some failure, faults, intrusions, degradation, and loss
	Minimizes and simplifies the system's minimal essential functions to enable successful operations and management of the resulting system in a crisis situation
	Supports offensive abilities to react and, in some cases, fight back in a contested cyberspace
	Possesses hardware, software, services and data replication, redundancy, and diversity
	Provides assurance mechanisms for correctness and integrity of software and hardware functions for essential functions
Detect/Monitor	Detects anomalies, the symptoms of failures and/or attacks, and signs of problems in neighbors
	Monitors its operating condition; possesses self-awareness of state of health, performance, availability.
	Collects information for later forensic analysis.
Constrain/Isolate	Enables configurability, continuity of operations (COOP), and disaster recovery (DR) to support rapid, predictable reconfiguration or restoration of capability
	Integrates safeguards (e.g., segmentation and stops) to contain the spread of damage and propagation of failures
Maintain/Recover	Degrades gracefully, when necessary
	Fails in a known good way, when necessary
	Returns to its nominal operating condition as quickly as circumstances permit

Adapt	Operates adaptively during normal operations and in response to changes in external and internal situations • Sometimes in a predictable way and sometimes more randomly for the purpose of unpredictability for an adversary. • Is self-learning, agile, adaptive, reconfigurable and extensible.
	Leverages hardware, software, data, and processing diversity and distribution in a random way
	Returns relatively quickly to its level of trust prior to the anomaly or to an acceptable level of trust
	Adapts to introduce randomness, deception and unpredictability to confuse the adversary

Getting Started

Apply a Risk Management Approach

A one-size-fits-all approach to designing resilient architectures is neither practical nor appropriate. We should choose a carefully balanced combination of protection mechanisms, detection capabilities, and adaptive technical and operational responses based on mission or business needs, processing environment, risk tolerance level, and critical operational scenarios.

Achieving this balance depends on first applying a risk analysis methodology to determine which critical capabilities must be resilient, to what level, and against which threats. This risk analysis forms the basis for deciding the adequacy of existing mechanisms and procedures, identifying any gaps, and determining the trade space of alternative technology approaches and courses of action. Risk analyses and dependency modeling can help identify the best locations to place additional safeguards to monitor and limit attack propagation, and drive the design of failover partitions and snapshots to recover to a secure state. The risk management process should:

- Identify mission- or business-critical capabilities, use cases, and assets (aka crown jewels);
- Map crown jewel dependencies to one another and to the underlying IT infrastructure, people, and processes;
- Weight relative priority in terms of criticality and minimal essential capabilities;
- Analyze the current or planned architecture's susceptibility to known and probable attacker TTPs; and
- Evaluate alternative mitigation strategies at the nexus of security protections, business continuity disciplines, and network and computer network defense (CND) operations.

Once we understand the risk posture and constraints of our critical processing and operational environments and the trade space and criteria for making risk decisions, we can begin to evaluate a variety of game-changing technologies, strategies, and techniques for improving resilience.

Virtualize the Infrastructure for Agility

Using virtualization technologies to build in the agility needed to change mission systems easily can facilitate cost-effective adoption and greater impact of the approaches described above. Virtualization is a recent computing paradigm shift. Businesses are rapidly adopting and deploying virtualization to consolidate data centers for efficiency and to reduce costs (space, resources and power consumption savings), as well as to provide cost-effective redundancy and improved provisioning, recovery, and security. This single disruptive technology supports all the objectives for resilience and therefore serves as a keystone to building secure, resilient architectures.

Specifically, virtualization can be used to implement techniques for isolation, non-persistence, and replication and scaling for availability. Virtual machines (VMs) offer a cost-effective approach to diversity and randomness because they can incorporate diversity in hardware platforms, chip sets, operating systems, applications and services, and randomness in deployment practices. These features make virtualization a fundamental enabler of resilience. VMs can be created, replicated, reconstituted, and deployed in milliseconds, thereby providing scalability, manageability, and agility to create, deploy, and move critical processing capability at will if the system is under attack. Many of the approaches presented in this paper become more effective in a virtualized environment.

Cloud computing represents the most recent instance of this paradigm. It is a model for enabling self-service, on-demand access to shared computing resources (e.g., networks, servers, storage, applications, and services) in an Internet-based environment. Shared, managed services can be rapidly provisioned, deployed, and scaled with minimal service provider interaction. In this paper we do not distinguish virtualization techniques from cloud computing. The diversity and distributedness of cloud computing services raise virtualization and deployment to a higher level, but also introduce significant security, governance, and control issues that must be addressed.

Virtualization is already an important part of high-availability strategies, but has not yet been generally applied to promote resilience. To date, disaster recovery (DR) and high availability (HA) are achieved primarily through redundancy, capacity planning, and backup and restoration to achieve COOP. Virtualization offers the key benefit of reducing the time and cost of recovering from a backup. An entire VM can be backed up and restored in less time than it takes to save files to backups, reinstall the operating system, and restore data. Of course, operators must plan appropriately for such uses of virtualization to ensure the availability of the resources it will consume (e.g., memory, CPU cycles, and disk space).

Administrators can use virtualization to create a master image of how to configure servers in a given data center. This makes it easy to create cookie-cutter systems and to recover if a server is compromised, since the administrator can quickly reset the system to the original or different template. VMs can be reinstalled on different physical servers when portions of a system are inaccessible, corrupted, or under attack or when the nature of an ongoing attack warrants a different or more stringent security environment.

When a denial of service attack is in progress, we can make critical processing resilient by replicating, reconstituting, or deploying additional VMs to increase processing capacity. These VMs can be run on the same hardware, on different hardware that is known not to be under attack, or simply on a platform that supports alternative technologies, stricter security constraints, or is managed by a different service provider.

At the same time as we pursue applications of virtualization for resilience, we must address the security of the hypervisor and VMs to ensure that they cannot be easily compromised and that viruses and exploits cannot hop from one VM to another faster than they can move on physical machines. In addition, operators must evaluate the potential transient and movable nature underlying the range of deployment options to prevent introduction of new vulnerabilities or attack opportunities and to adapt scanning and monitoring for both on-line and off-line platforms and VMs.

Key Resilience Techniques and Mechanisms

The remaining sections of this paper describe how system designers might apply the following techniques and strategies individually and in combination to create architectures with greater resilience against particular cyber threats.

- Diversity
- Redundancy
- Integrity
- Isolation/segmentation/containment
- Detection/monitoring
- Least privilege
- Non-persistence
- Distributedness and moving target defense
- Adaptive management and response
- Randomness and unpredictability
- Deception

Given operational constraints and lifecycles, not every technique applies in all environments. At a minimum, however, designers should consider these ideas when developing new systems. Legacy systems will pose a greater challenge and system designers will need to evaluate which techniques have the greatest potential value and how best to introduce them.

Diversity and Redundancy

The initial steps are simply to introduce diverse and redundant technologies and capabilities into the system architecture. Over the last decade, cost reductions, economies of scale, and business efficiency have driven the current overreliance on a single vendor for OSs, applications, software utilities, network components, and other capabilities. This has resulted in creating single target points of attack that enable adversaries to cause pervasive impact: a single exploit with no diversity can succeed in damaging all systems based on that vendor's products. Too often organizations fail to buy redundant or back-up capabilities for critical processing until a disaster strikes, even though redundancy enables availability during unexpected peak loads, faults, and failovers.

Introducing diverse COTS and proprietary technology can minimize the impact of technology-specific attacks and raise the bar for adversaries by forcing them to attack the alternative means as well. Simple replication in a homogeneous technology environment is efficient when no technology-specific attacks exist against that

technology. However, in the face of zero-day attacks against a specific technology, heterogeneity may offer the best option for permitting continuing operations in at least those portions of the system that are not affected at that moment.

Without diversity, failover and reconstitution capabilities will be similarly impaired, since re-creating the same technology that was just compromised merely invites another attack as soon as the system reconstitutes or fails over. Consequently, diversity and redundancy should be designed into essential hardware, software, data, and network components identified through the risk analysis process. Determining the optimal level of redundancy and diversity requires balancing the increase in total cost of ownership, management, and complexity against the benefits of higher assurance that failover and reconstitution of capabilities on alternative hardware and/or software will succeed.

Another homogeneity trend to counter is the increasing preference for implementing all services over IP. This can create a single point of failure that is susceptible to the frequent and common exploits of Internet-based services. One mitigation approach may consist of identifying alternative protocols to provide critical functions and services for trusted communications or contingency processing capability that ensures continuity of operations while under attack.

Similarly, we should plan diverse processing paths and operational TTPs. For example, we can design out-of-band approaches to communications that take advantage of wireless, satellite, or telephone communications when networks are compromised or under attack, or deploy out-of-band communications using two separate networks working simultaneously to authenticate a user or authorize a bank transaction. We should identify alternative ways to achieve the same operational outcomes when specific applications, services, and data are unavailable or compromised. This might include using a less capable legacy means of processing or at times even resorting to manual procedures to complete essential tasks.

Virtualization can help to achieve cost-effective diversity. For example, if a zero-day cyber attack were executed against a VM running Windows, an organization could constitute a predefined template for a VM running Linux that might not be susceptible to the same attack. Other techniques for achieving synthetic and design diversity can also be developed.

After performing the risk analysis to guide decisions about where to implement diversity, we can decide how best to deploy diversity: whether as part of normal operations or as a reserve reconstitution capability when all else fails. The threat level and length of time a defensive measure can be expected to remain effective will drive the choice of the best approach. Diversity that an attacker can very quickly understand will become useless, which requires that we have other resilience techniques ready on hand.

Integrity

The most insidious cyber attacks target system integrity, rather than denying service. Attacks on integrity cause improper modification by inserting, deleting, or changing existing information, code, or functions. It is important to understand that if the integrity of a critical component or system is compromised, traditional COOP and DR support for resilience may actually worsen the situation. Replication, failover, or reconstitution services that simply move an execution environment or data repository to another location can exacerbate the problem by propagating corrupted services and data.

A resilient architecture should provide assurance of correctness and integrity of essential software and hardware functions and data wherever possible. In the face of an integrity threat, it will be important to reconstitute images, VMs, applications, or databases (DBs) from a known good image instead of replicating the running VM, DB, or backup. Reconstitution may occur when an integrity compromise is detected, or be performed periodically as a preventive measure to help foil an attacker's attempt to target and establish a foothold.

Beyond traditional integrity checks such as encrypted checksums, digital signatures, or MD5 or SHA-2 hashes, another approach to validating the integrity of security functions and configurations is to measure the health and correct functioning of critical components (e.g., boot functions, intrusion detection software, and anti-virus software) to ensure that protection mechanisms are intact and functioning properly. Such trusted attestation measurements may be accomplished by leveraging trusted components such as trusted platform modules (TPMs). However, such solutions are not commercially available and most self-validation mechanisms inherent in COTS products could be subverted.

In such instances, one approach is to introduce a polling mechanism on systems that have implemented redundant processing or have data residing on distributed nodes. Polling could determine the likelihood of compromise on the basis of measurements and calculations taken at multiple points in the network, or by comparing the operational measurements to an isolated benchmarked system. The assumption that underlies polling is that diversity and other techniques will prevent compromise of all nodes in the system. This notion can be modeled after a neighborhood watch program. The determination of whether a majority poll provides sufficient assurance based on a common community view will depend on the environment. Similarly, we can validate the integrity of critical information by querying multiple independent sources for critical data and updates and comparing the values.

Separation/Isolation

Today's cyber incidents result directly from connecting formerly standalone or private systems and applications to the Internet and to partner networks. Pervasive interconnectivity and consolidation have exposed systems in ways not originally imagined. A key foundation for building resilient architectures is to partition off components of dubious pedigree from those we trust to reduce the attack surface and limit the damage and spread of exploits when they do occur. We can reduce attacks on critical processing and data by separating them from non-critical data and processing. This requires us to modify system architectures, either to create physically distinct entities or to virtualize computing enclaves to achieve the desired separation of sensitive processing and data repositories. Fine-grained separation of applications and services also allows greater precision in the placement of sensors for monitoring mission-critical functions.

Separation requirements should implement the principle of least privilege. We should not combine applications and services that require different levels of privilege, as they will rise to the high watermark for privilege and access and could result in undue exposure if attacked. The goal should be to decouple capabilities in order to prevent ripple effects that can contaminate large portions of our systems as the result of a single attack or failure.

Virtualization supports easy creation of trusted, isolated computing enclaves for conducting sensitive processing because of the inherent reduction in the amount of physical assets required. Segregation should be implemented at the network level with controlled interfaces between segments, as needed, both within an

enterprise and at its boundaries. This helps limit LAN contamination and flow-through, and with proper sensor deployment can provide better SA and more precisely targeted CND. VPN and VLAN approaches can also help achieve the desired logical separation depending on the acceptability of the differing levels of security and assurance.

Highly critical functions may call for stronger boundaries than logical separation or virtualization can offer, and should rely on physically separate hardware and networks. For example, the CND network should be isolated from critical processing networks to prevent the adversary from learning our intrusion analysis and forensic capabilities. If monitoring occurs out of band, adversaries cannot easily disable the monitoring functions when they compromise functional capabilities in the operational network. Similarly, critical functions such as key distribution should be performed on physically separate networks; we should also consider establishing a separate network for degraded operations. By pairing separation with diversity techniques, we can further bolster the value of the separation by forcing attackers to compromise both forms of communications, thus adding a deterrent to make their task more difficult. Over time we should move toward a highly stratified architecture that isolates most security enforcement mechanisms from the functional layers of the architecture.

Imposing separation at the network level (e.g., Internet from Intranet and DMZ segmentation) and at the application and data levels (e.g., non-sensitive from sensitive) segregates risky traffic and processing from critical traffic, processing, and data. These tactics help reduce complexity by removing extraneous data and transactions and promote more effective intrusion detection, packet capture analytics, and other anomaly detection capabilities because anomalous behavior cannot easily "hide in the noise."

It is nearly impossible to spot data exfiltration given the volume of traffic, numbers of protocols, and the mixture of inbound and outbound packets. Finding extraneous traffic could be simplified if we also implemented separation principles by isolating asynchronous communications, analyzing and correlating request-response traffic, and isolating different protocols to look for anomalous traffic.

Vast numbers of cyber attacks target untrustworthy end-user client systems connecting to the Internet, where the exposure to attacks and compromise is highest. Keeping these systems fully patched and securely configured is the bane of system managers and is beyond the ability of many end users. In addition, many computers in industrial control systems cannot be patched for extended periods of time because the computer systems cannot be taken out of service while the facility operates. This situation has resulted in widespread data loss, viral spread of malware, and the hijacking and control of systems to function as zombie computers within botnets.

Thin clients, secure browsers, virtualized clients, and diskless nodes are architectural approaches to mitigating these risks. Secure browsers and virtualized clients can be used to sandbox risky processing from critical processing. Combined with non-persistence, they can terminate the browser session and re-instantiate a new "good" session automatically to ensure that any malware introduced does not remain or spread.

Thin clients and diskless nodes isolate critical data and services and prevent them from being stored or run on untrusted end user devices. In a thin client configuration, if an end user system is compromised, the attacker must resort to screen scraping to gather the data. This activity is further constrained by user privileges that are better managed and enforced on the server than on the client. Thin client approaches do place a greater burden on communications to and from the server, so planning must address the increased network traffic and the usability issues associated with working in a disconnected mode.

Another beneficial application of isolation techniques is to sequester and contain replicating malware. For example, at the server side we can virtualize and encapsulate the OS and/or applications and data separate from the underlying hardware. Encapsulation provides better security separation and helps prevent attacks in one VM from affecting the applications and OSs running in other VMs without having to invest in separate hardware. This isolates critical processing from non-essential or risky processing and thus avoids contamination and propagation of exploits. Extremely high security risks may warrant isolation implemented in hardware rather than virtualization separation.

Detection/Monitoring

In general, monitoring adversarial actions affords us the opportunity to learn attackers' TTPs so that we can evolve and implement new protections and countermeasures in the future. Detecting attacks, or, better yet, gaining early warning during the attacker's pre-attack reconnaissance stage can inform and trigger resilience responses. Because systems are not fully resistant to cyber attackers, this depends on detection of misuse, abuse, dynamic attacks, and malicious code as well as observable changes in the operational state of the systems.

Understanding normal or expected user and system behavior is fundamental to developing real-time global SA across all layers of the architecture and the surrounding environments. Creating global SA, in turn, requires judicious, in-depth placement of anomaly sensors across the environment: at network segment boundaries, gateways, end systems, and servers, as well as for applications and data, not just at the perimeter as is commonly done today. Given our reliance on infrastructure and outsourced service providers, this also requires cooperation and arrangements with network and service providers.

Correlating detections into meaningful threat indicators and trends and then fusing this knowledge with other observed changes to the health of the network or to the system's run time can inform our responsive strategies when operations are degraded. Detecting compromises to the integrity of the runtime environment via memory analysis, as well as monitoring network file systems and system registries, can support forensic analysis and enable reconstruction of the system environment to its state prior to an incident or attack. Because we cannot predict attackers' changing TTPs, this calls for dynamic deployment of detectors and the ability to adjust the sensitivity and location of sensors in real time based on early warning alerts or during an incident. In addition, we need specialized sensors and deployment mechanisms to understand and develop metrics for measuring resilience.

We can leverage VM isolation support to permit sandboxing for detection, adversary monitoring, and forensic purposes. The hypervisor provides a central point from which to monitor and detect compromises: instead of placing sensors or software monitoring agents on every system, designers can emplace agents to run in the hypervisor, where they can be more easily protected and provide more centralized monitoring. VM introspection can supply detailed knowledge about the running state of any machine to help detect anomalous activity.

Honeyclients constructed using virtualization technology can serve to detect malware. Honeyclients are state based; they detect attacks on clients by monitoring files, registries, and processes. By taking a measurement snapshot before and after a processing event (e.g., a browser click on a URL) the honeyclient can determine unexpected changes occurred, suggesting the possibility of malware. The snapshot can later be forensically

analyzed to determine the purpose and impact of the malware without allowing execution that would compromise the real system.

By periodically taking comprehensive snapshots of volatile memory and capturing full packets, we can also run these images and packet flows retroactively against newer versions of malware and intrusion detection analysis tools. This can help determine whether we were previously attacked by malicious code only recently uncovered, and to establish forensically if the new attacks resemble previous ones. In addition, after an attack is detected, we can replay snapshots to determine when the attack happened or whether the adversary attempted to cover his tracks by deleting files in audit trails, which might otherwise cause the attack to remain undetected. This will help us assess damage and determine how to recover to a trusted state.

Non-persistence

As previously discussed, refreshing to a known good image is an example of applying a non-persistence strategy. The goal is to set the periodicity so that refreshes occur frequently enough to prevent the spread or intended impact of an attack, but not so frequently that it makes the system unstable. An even better approach is to implement aperiodic refreshes, because such randomization hinders an attacker's ability to predict the window of opportunity in which to launch an attack, and increases the risk that the attack will fail or be detected.

Non-persistence techniques can be applied for access to data, applications, and connectivity when continuous access is nonessential. For example, if critical data or applications are only needed during a particular phase of an operation, we could ensure that they are not accessible at other times. This will reduce the exposure of the data and applications, as well as the opportunity for the adversary to analyze our vulnerabilities or gain a stronghold and maintain a persistent presence.

Non-persistence based on virtualization techniques can be used to stand up a capability on demand and tear it down when it is not needed. Used in this way, non-persistence reduces the window of opportunity for attacks and can impede the adversary's reconnaissance. Defining retention requirements can ensure that data and critical applications are not stored online longer than necessary–serving only to allow attackers to study our TTPs or infer the evolution of capability. These actions can help reduce attackers' window of opportunity and can possibly remove their stronghold in our systems if the location where they become entrenched is not permanent.

Non-persistence techniques can protect systems from the long-term effects of risky behaviors. For example, virtualization on the client side can create non-persistent VMs that allow users to perform risky Internet surfing and transactions from a VM and then simply discard the VM afterwards. Even on VMs running trusted applications, we can periodically push down or refresh to a known "good image" to ensure that the current VM complies with the most secure configuration. During an attack, security managers could create and push a gold image with more restrictive security settings or simply prevent those who are not running the latest image from connecting. After an incident, recovery could be eased by pushing a new patched image. Network bandwidth and latency considerations must be considered to achieve the right balance with this approach.

Non-persistence also provides operational provisioning and management benefits. Often new exploits benefit from a long interval between the time patches become available and the point where all systems are fully patched. Instead of patching hundreds of thousands of end user systems we could instead push a new image

from a central management function when a user connects each day, at some fixed interval, or on every Patch Tuesday.

Distributedness and Moving Target Defense

Another trend driven by cost reduction and efficiency is centralization. Centralization allows us to manage and control our systems more tightly, but centralizing data and processing can also create a single point of attack for the adversary. At times it may be more prudent to decentralize processing and data across multiple physical locations and ensure some level of redundancy or overlap. By distributing critical processing and data across different hardware and physical locations, we create multiple points that attackers would have to compromise in order to defeat critical operations.

Replication and synchronization technologies can copy and distribute data, transactions, and objects from one database to another and synchronize them to maintain consistency. Data can be distributed to different geographical locations, hardware, and user devices over a variety of media such as local and wide area networks, dial-up connections, wireless connections, and the Internet. A decentralized model can be designed to support diversity of communications and access as well. We must recognize that distributed computing also demands greater reliance on the network in terms of availability, connectivity, bandwidth, and performance, and must plan accordingly. Caching strategies to address availability, latencies, and performance considerations are also useful to promote data resilience.

When distributed processing is paired with dynamic relocation for deployment and possibly with different enforced security policies at the different locations, we can create the ability to confuse adversaries and possibly limit their advantage. By no longer passively waiting for adversaries to launch their attacks at the time and place of their choosing, as happens today with static deployments, we can reverse the game. Our crown jewels become the mole, and we can confuse adversaries by increasing the unpredictability of where they are located or what security policy has been enforced. This may drive attackers to act in more detectable ways as they attempt to target crown jewels on the move.

To create an effective moving target, we must ensure that the movement itself is not predictable. The more random and unpredictable the movement, the more likely it will actually disrupt adversarial action. Given the ease of moving virtualized services while they are running, a moving target serves as a realistic countermeasure to attacks, since it can disrupt the attacker in the planning stage and act as a deterrent if the outcome of an attack is not predictable or assured. Of course the randomness must be hidden from the adversary in order to reap the benefit of surprise.

Incorporating diversity in conjunction with randomness further increases the adversary's difficulties and can make an exploit more detectable. If we know we are under attack, we can move processes to a similar environment in another location, to an environment based on different technology, or to one with a more stringent defensive posture that features added sensors and controls. Another option is to move our active defenses, similar to a moving sentry guard. Once again, if the movement is unpredictable and the defenses change, attackers can never be quite certain what defenses they will encounter. Pervasive persistence will likely be more detectable than hiding in a few discrete locations.

Adaptive Management and Response

To achieve adaptive management responses for resilience, the system must be able to monitor its operating condition and gain self-awareness of its state of health, performance, integrity, availability, etc. These techniques to a large extent depend on capabilities to detect anomalies, failures, and other symptoms of abnormal conditions or attacks. A system must also understand the SA of its surrounding environment and its alternative processing capabilities to achieve transparent service deployments, scaling, redeployment, etc. More important, operators must be able to measure, quantify, and set thresholds that specify acceptable levels of system degradation, the constraints that may be placed on the timing of changes, and the priority or order of changes before alternative actions are evaluated, selected, and invoked.

The effectiveness and maturity of existing management functions will in part drive responses. For example, if a robust reconstitution capability exists, we may prefer triggering a failure in order to reconstitute rather than allowing the system to degrade further and being forced to develop operational capabilities for a degraded mode. Continuity of operations plans may necessitate that we temporarily shut down a portion of the system when we believe the adversary has gained a foothold. This would allow us to close the attacker's command and control channel while we complete critical operations in another portion of the system.

Consequently, it will be important to make the underlying indications and warning resilient. Detectors and diagnostic sensors that are pre-positioned in a network should have redundant paths for reporting. They may also have to be moved to ensure availability and integrity based on awareness of what portions of the system are being compromised.

Because damage is unpredictable, the damage assessment (DA) capability must itself be resilient. Today DA agents typically assess and report damage at the local level (e.g., host functions, local communications, and data integrity). To ensure availability, we will need to report assessments to distributed collection points. We must also fuse and analyze these inputs to create an integrated DA that is then correlated to the mission risk assessment and an assessment of the readiness of critical processing.

Armed with this information, operators and decision makers can develop alternative courses of action and rules for dynamic management. For example, during a denial of service attack, the best course of action might be to dynamically provision additional processing capability. This can be done on the same or different hardware or software and in the same or different locations. If the attack comes from the outside, we could reconfigure boundary protections and security policies. During a failure or degradation, we could simply choose to execute COOP plans for failover, shut down nonessential functions, and initialize minimized alternative capabilities to execute critical processing. Alternatively, we could reconfigure the existing capability to a more trusted minimized state with a stricter security policy. Integrity compromises would yield other operational responses. Under extreme conditions, we could dynamically compose new capabilities on demand depending on our ability to discover alternative capabilities offered by cloud service providers.

We should continuously learn from each new incident and evolve or develop new TTPs and response capabilities. For instance, after a system compromise, if we can retrospectively identify observable changes and measurements that serve as indicators, we can use this information to develop new detectors and designate possible courses of action. The complexity, the different types of information needed to understand the cyber situation, and the spectrum of possible options demand visualization tools and decision support systems to aid operators in selecting the best course of action at a given point in time.

Despite the best laid plans and TTPs, the myriad unknown unknowns mean that operator resilience will remain paramount for success. Contingency planning, training, and red team exercising against inevitable attacks will continue to be critical to evolving our operational procedures and preparing our operators.

Randomness/Deception/Unpredictability

Confusing an attacker or adding the element of surprise may possibly foil an exploit, introduce uncertainty into the results, put the attacker at risk of being detected or exposing tradecraft, or buy us time when systems are under attack. Sophisticated attackers may quickly discern even covert randomness, deception, and unpredictability, but these measures can remain effective until detected, or they may only be exposed slowly over a period of time. The more stealthy and realistic a deception, the longer it can remain effective.

While techniques based on these strategies are perhaps the least mature in practice, they have the greatest potential to reverse the attacker's advantage. We must evaluate where and when to deploy deception and unpredictability for any given environment; this approach may not be appropriate for some specialized environments (e.g., workflow or industrial control systems).

Randomness was previously discussed in terms of introducing unpredictability and of its added value when combined with other resilience strategies, such as heterogeneity. Deceptions can be introduced simply as misdirections, modifying data through transformations, and hiding critical data and processing functions, or more subtly by designing system responses to give the illusion that the system is something it is not. If we can deceive adversaries about the exact components of our system as they attempt to map out our technologies and configurations during the reconnaissance phase, we can increase the probability an exploit will fail against the actual system. For example, we can set up honeypots that mimic a production environment to lure the adversary. A honeypot can provide a safe haven that allows us to monitor and analyze the attacker's intent, tools and techniques. If the honeypot becomes compromised, it serves as an early warning of attacks targeted at the real system. This can buy time for an operator to bolster security, reconfigure, or institute new safeguards and mitigation strategies in the actual system and thereby prevent the attack from penetrating the intended target.

Another decoy practice involves retaining an obsolete capability and ensuring extraneous use of it to project realism. This could confuse attackers or act as a decoy to catch them. Cluttering a system can make it difficult to find the crown jewels and obfuscates the real targets. Similarly, if we insert phony records in a database and apply steganography or watermarks to objects in files, we can perhaps detect usage or exfiltration and thus understand if an adversary has established a stronghold in our databases.

Additional deception strategies to consider include:

- Code and data obfuscation
- Self-modifying code
- Self-protecting data
- Customized and proprietary protocols, extensions, code, solutions, etc.
- Entrapment techniques
- Misinformation
- Responses based on varying CNO response actions

Summary/Conclusion

To reverse the asymmetric advantage of the cyber attacker and minimize the impact on our critical mission capabilities, we must be proactive in building secure and resilient systems. By promoting resilience against escalating cyber attacks, we can simultaneously achieve resilience against acts of nature, loss of physical network elements, and other threats. While it is not realistic to assume we can stop all cyber attacks or make them totally ineffective, redesigning architectures for resilience will make attacks less likely to succeed, will minimize the consequences when they do succeed, will increase adversary cost and uncertainty, and may act as a deterrent against future attacks. Improving resilience will also increase system reliability.

Game-changing technologies, design techniques, and operational strategies are available today to get us started. This paper has presented various possible approaches to improving resilience, but not all approaches are feasible in all situations. To begin building resilience into systems, we must decide which promising strategies and techniques are most appropriate for our environments and critical missions. The next step is to experiment with these techniques and strategies in laboratories and pilots to (1) demonstrate the feasibility and effectiveness of different approaches in different environments and operational scenarios and (2) identify the usability, cost, performance, and other operational considerations that we must assess. During these evaluations, we can start to develop resilience metrics and to measure the tangible benefits, readiness, and residual issues we must address in order to proceed with deploying these techniques and strategies.

Doing nothing is not an option. We must act now to reverse the adversary's advantage and ensure that we can rely on our mission-critical capabilities to be available and trustworthy when we need them most.

