Achieving Cyber Survivability in a Contested Environment Using a Cyber Moving Target

Dr. Hamed Okhravi
Member, Technical Staff
Cyber Systems and Technology Group
MIT Lincoln Laboratory
Lexington, Massachusetts

Mr. Joshua W. Haines
Assistant Group Leader
Cyber Systems and Technology Group
MIT Lincoln Laboratory
Lexington, Massachusetts

Mr. Kyle Ingols
Member, Technical Staff
Cyber Systems and Technology Group
MIT Lincoln Laboratory
Lexington, Massachusetts

Evolving cyber threats in a contested environment provide a challenge in protecting operations and critical assets. Traditional cyber protection mechanisms can prove ineffective when facing a motivated, well-resourced adversary. As a result, many mission critical systems remain vulnerable to advanced, targeted cyber attacks despite the significant amount of effort and resources used to secure them. Complex systems and commercial off-the-shelf components often exacerbate the problem.

Although protecting the mission critical systems is a priority, recent cyber incidents and alerts have shown that we cannot rely completely on hardening individual components.[1, 2] As a result, new attention has been given to game-changing technologies to achieve mission continuity in a contested environment. In fact, the Air Force chief scientist's report on technology horizons mentions the need for "a fundamental shift in emphasis from 'cyber protection' to 'maintaining mission effectiveness' in the presence of cyber threats" as a way to build inherently intrusion-resilient cyber systems.[3] Moreover, the White House National Security Council's progress report mentions a "moving target (systems that move in multiple dimensions to disadvantage the attacker and increase resiliency)"[4] as one of the administration's three key themes for cyber security research and development strategy.

Our approach to developing the necessary survivability involves a combination of research, prototyping, architectural development, and evaluation. We have researched architectural ideas that make it difficult for adversaries to impact mission critical systems and prototyped an architectural component that provides platform heterogeneity as a proof-of-concept. We have also developed an analysis and assessment tool that can evaluate the attack paths into a system and support the architectural component in determining the appropriate orientation based on the current threat level. We are in the process of developing analysis and experimentation frameworks to thoroughly measure the effectiveness and protection offered by the components discuss in this work; we leave them as the future work here.

We describe two components for achieving cyber survivability in a contested environment: an architectural component that provides heterogeneous computing platforms and an assessment technology that complements the architectural component by analyzing the threat space and triggering reorientation based on the evolving threat level. Together, these technologies provide a cyber moving target that dynamically changes the properties of the system to disadvantage the adversary and provide resiliency and survivability.[5]

Trusted dynamic logical heterogeneity system (TALENT),[6] the architectural component, provides a framework to migrate, in real-time, mission critical applications across heterogeneous platforms. We hypothesize that in critical warfighting systems, the mission itself is the top priority, not individual instances of the subsystems. By live-migrating the critical application from one platform to another, TALENT can thwart cyber attacks and provide resiliency. This means the information collected by the attacker about the platform during the reconnaissance phase becomes ineffective at the time of attack.

TALENT provides heterogeneity at the hardware and operating system levels while it preserves the state of the mission critical application.[7, 8] This means we should be able to run the application on top of processors with different instruction sets.

By accurately measuring risk for mission critical networks, attack graphs allow network defenders to understand the most critical threats and select the most effective countermeasures. Network Security Planning Architecture (NetSPA),[9] the assessment component, analyzes critical networks against the current threat level using attack graphs and reachability analysis. NetSPA assesses the effects of known and zero-day attacks, computes the impact of possible compromises, and proposes countermeasures.

By integrating the architectural and assessment components, a critical warfighting system can achieve cyber survivability against aggressive cyber attacks. NetSPA assesses the potential compromises and reacts to changes in the current threat level by triggering reorientation. TALENT then performs reorientation by dynamically changing the platform of the critical applications to the platform recommended by NetSPA. Together they implement a polymorphic system that can operate through aggressive compromises in a contested environment.

Architectural Component

The architectural component of a cyber survivable system

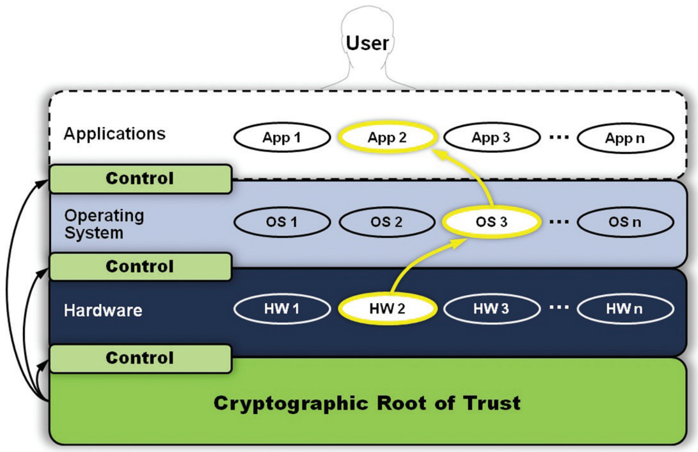

Figure 1. A dynamically composable platform of heterogeneous components.

must be able to dynamically provide heterogeneity. Figure 1 illustrates a dynamically composable platform of heterogeneous components. Heterogeneity at different levels can mitigate various attacks. Application-level heterogeneity mitigates architecture specific exploits and malicious compilers. Operating system (OS)-level heterogeneity mitigates kernel specific attacks, OS-specific malwares, and OS persistent attacks (rootkits). Hardware heterogeneity can thwart supply chain attacks, malicious/faulty hardware, and architecture specific attacks. TALENT is not a complete defense against all these attacks; it can, however, provide survivability in the presence of platform specific attacks by means of dynamic heterogeneity.

In order to provide dynamic heterogeneity, TALENT migrates both the environment (e.g., files and network connections) and the state of a critical application across different platforms. Figure 2 illustrates a heterogeneous migration process. To address the challenge involved in using heterogeneous platforms including binary incompatibility and loss of state and environment, TALENT uses two key ideas: OS-level virtualization and portable checkpoint compilation.

Environment Migration

An important goal of the architectural component is to preserve the environment of a mission critical application including the filesystem, configuration files, open files, network connections, and open devices. Many of the environment parameters can be preserved using virtual machine (VM) migration, but VM migration is only viable with homogeneous OS and hardware. Because we want to *change* the OS and hardware while migrating a live application, VM migration is not applicable. TALENT uses OS-level virtualization to sandbox an application and migrate the environment.

OS-level virtualization is a method in which a kernel allows for multiple isolated user-level instances. Each instance is called a container (jail or virtual environment). The method was originally designed to support fair resource sharing, load balancing, and cluster computing application. OS-level virtualization provides an environment in which all resources (devices, filesystem, memory, sockets, etc.) are virtualized.

Note that the major difference between OS-level virtualization and hardware-level (e.g., Xen and VMWare) is the semantic level at which the entities are virtualized. Hardware-level hypervisors virtualize disk blocks, memory pages, hardware devices, and central processing unit cycles, whereas OS-level virtualization works at the level of file systems, memory regions, sockets, and kernel objects (e.g., inter-process communication [IPC] memory segments and network buffers.) Hence, the semantic information often lost in hardware virtualization is readily available in OS-level virtualization. This makes OS-level virtualization a good choice for use cases where this information is needed like monitoring or sandboxing at the application level.

TALENT uses OS-level virtualization to migrate the environment of a critical application. When reorientation is requested by the assessment component, TALENT migrates the container of the application from the source machine to the destination machine. This is done by synchronizing the filesystem of the destination container with the source container. The OS keeps track of open files and the same files are opened in the destination.

To preserve network connections during migration, the internet protocol (IP) address of the container's virtual network interface is migrated to the new container. Then the state of each transmission control protocol socket is transferred to the destination. The network migration is seamless to the application, and the application can continue sending and receiving packets on its sockets. Many OS-level virtualization frameworks also support IPC and signal migration. In each case, the states of IPC and signals are extracted from the kernel data structures and migrated to the destination. TALENT supports these features by utilizing the underlying virtualization layer.

Application Migration

Migrating the environment is only one step in backing up the system because the state of running critical applications must also be migrated. To do this, a method to checkpoint (store the state of) running applications must be implemented. Once all checkpointed program states are saved in checkpoint files, the state can be migrated by simply mirroring the file system. TALENT uses a portable checkpoint compiler (PCC) to preserve the state of a running application and provide application migration.[10]

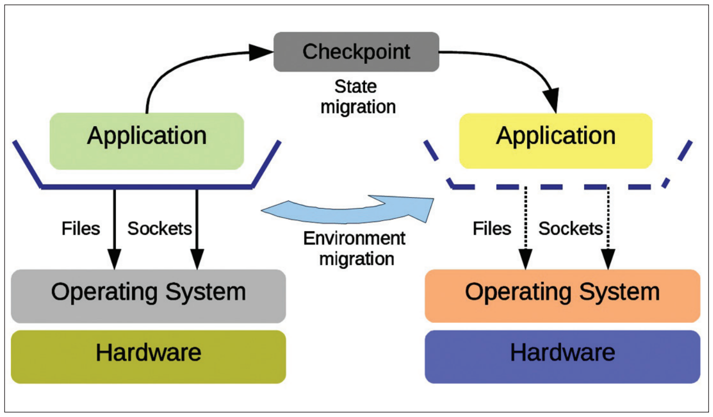

Figure 2. A heterogeneous migration process.

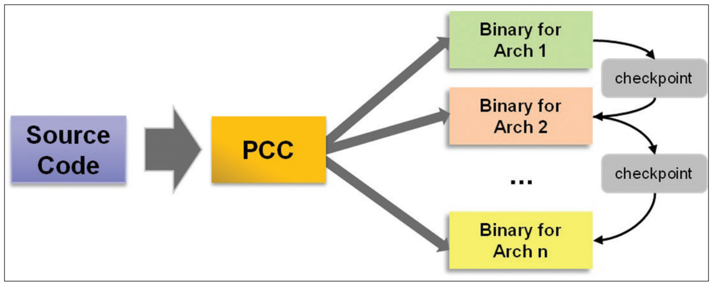

Figure 3. A portable checkpoint compilation process.

Figure 3 illustrates a portable checkpoint compilation process. PCC allows compilation to occur independently on various operating system/hardware pairs. The resulting executable program, including the inserted checkpointing code, functions properly on each platform on which it was compiled.

Using PCC, TALENT achieves both transparency and scalability. Transparency is obtained by performing automatic code analysis and checkpoint insertion. This prevents the end user from modifying code to indicate where checkpointing should be performed or what specific data should be checkpointed. Scalability is obtained in two ways. First, the frequency of checkpointing bottlenecks in the checkpointing process can be controlled. Second, through the use of compressed checkpoint file formats, the checkpoints themselves remain small as the amount of data processed by the program increases.

In order to achieve portability across heterogeneous platforms, the checkpoint file must have a portable format. Storing the checkpoint in a simple binary file can cause incompatibility if the destination platform has different bit instruction size (32 vs. 64 bits) or endianness (little vs. big). Thus the checkpoint file format has to be portable. We use the HDF5 format in TALENT.[11] HDF5 is an open, versatile data model that can represent complex data objects. It is a portable data model that can represent various bit-ness and endianness. Like Extensible Markup Language (XML),[12] HDF5 is self-describing. Unlike XML, HDF5 uses a binary format which allows efficient parsing of the data.

Analysis and Assessment Component

NetSPA, our attack-graph generation and reachability analysis tool, provides the assessment component of a cyber survivable system. By analyzing the impact of possible attacks and reachability of mission critical systems in a network, NetSPA facilitates platform reorientation based on the evolving threat level.

Data Collection

NetSPA's network model supposes that an individual host possesses one or more interfaces which have listening addresses. These interfaces have zero or more open ports, accepting connections from other hosts. A host and its interfaces may have rules that dictate how network traffic may flow to, and through, the host. A port has zero or more vulnerability instances, particular flaws or configuration choices which may be exploitable by an attacker. Each interface on a host is connected to a link, representing some combination of hubs and switches connecting a set of interfaces together. An attacker is able to obtain one of four access levels on a host: "root" or administrator access, "user" or guest access, "DoS" or denial-of-service, or "other," a confidentiality and/or integrity loss. The combination of a host and an access level is an attacker state. An attacker obtains a host's reachability if "root" or "user" access is achieved. Reachability and credentials serve as prerequisites to exploitation of a vulnerability instance.

NetSPA requires only a few core pieces of knowledge to build an attack graph. For a given host, the tool must know which credentials can be acquired at a given access level on the host and which ports the host can reach. For a given port, NetSPA requires knowledge of the port's vulnerabilities. For each instance of a vulnerability, the tool must know what is required to exploit it and what is gained by exploiting it.

NetSPA requires a large amount of data to compute the needed information, but system administrators often collect this data as a matter of course. The core pieces are network topology, vulnerability information, and credentials. NetSPA itself runs offline using the provided data, minimizing the risk of an attacker obtaining the source data or resultant graph. NetSPA can collect the raw information from Nessus scans, firewall rulesets, Open Vulnerability Assessment Language (OVAL)-based scanners, and vulnerability databases such as the National Vulnerability Database and Bugtraq.[13, 14]

Computing Reachability

Computing reachability is crucial in determining how to react to aggressive cyber attacks. In a contested environment, critical warfighting applications must be migrated to portions of the network that are not easily reachable from the external network through a zero-day attack.

A straightforward method to compute reachability is to try to reach every known target IP address and port from every host in the network. Such an approach would generate a reachability matrix, where a row represents a source interface on a host, a column represents a target port, and each cell indicates whether or not the source can reach the target. This is correct, but it scales poorly in terms of both space and time.

We have made three improvements to the straightforward approach. We collapse sections of the matrix into reachability groups, saving large amounts of both time and memory. Filtering rulesets are collapsed into binary decision diagrams,[15] allowing the reachability system to traverse a set of filtering rules in constant time. We also hypothesize a "generic attacker" by selecting a link on which the attacker will begin and allowing the attacker to use the most advantageous source IPs.

Reachability groups identify redundancies in the reachability matrix and collapse submatrices into single subrows before computing the contents, saving both time and space. First, intra-subnet reachability which is not influenced by any filtering devices can be collapsed into a single subrow, because every source interface within the subnet will have the same reachability to all ports within that same subnet. Second, inter-subnet reachability can be collapsed by identifying sets of interfaces within a subnet which are treated identically by the filtering devices on the network. If the source IP addresses of a set of interfaces on the same subnet match in the same set of filtering rules on the network, the interfaces are grouped together and reachability is computed for only one of them.

Building Attack Graphs

An attack graph specifies the sequences of vulnerabilities an adversary can exploit in order to gain access to a critical system. It is a valuable tool for understanding the impact of known and zero-day (unknown) vulnerabilities and studying the consequences of stepping-stone attacks on a mission critical system. NetSPA can generate attack graphs using the reachability analysis and the information collected about the current environment. As the threat evolves in a contested environment, attack graphs can help us understand which subnets are secure or which platforms can survive the current threat level. The assessment capability provided by NetSPA supports the reorientation of the critical platforms.

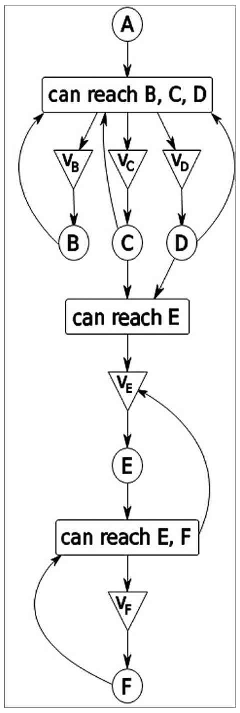

Figure 4. A multiple-prerequisite graph.

In order to build the attack graphs, NetSPA uses a data structure called a multiple-prerequisite (MP) graph. The MP graph also shows all hosts which can be compromised from any host the attacker has compromised. In figure 4 for example, host F is capable of compromising host E. The MP graph uses the following three node types:

- State nodes represent an attacker's level of access on a particular host. Outbound edges from state nodes point to the prerequisites they are able to provide to an attacker. In figure 4, state nodes are circles.
- Prerequisite nodes represent either a reachability group or a credential. Outbound edges from prerequisite nodes point to the vulnerability instances that require the prerequisite for successful exploitation. In figure 4, prerequisite nodes are rectangles.
- Vulnerability instance nodes represent a particular vulnerability on a specific port. Outbound edges from vulnerability instance nodes point to the single state that the attacker can reach by exploiting the vulnerability. In figure 4, vulnerability instance nodes are triangles.

Attack graphs for all but the smallest networks are too large for hand evaluation. NetSPA uses two approaches to this problem: automatic graph simplification and automatic recommendation generation. The former aims to reduce the size of the graph by collapsing similar nodes together. The latter treats the attack graph as an intermediate structure, not a final product, and extracts useful information from the graph for presentation to the user.

Building Recommendations

Even visually simplified attack graphs can be large and unwieldy. The core information from the graph should be extracted by the tool and presented in a more immediately useful form.

Often an attacker must compromise a directly accessible host through a filtering device in order to attack a group of hosts behind the filtering device. Attack graphs can be used to identify these bottlenecks and produce a list of the critical vulnerabilities which allow the attacker to compromise the bottleneck hosts. Defenders can then migrate the critical applications to subnets and platforms not affected by these vulnerabilities.

We form recommendations by computing, for each individual prerequisite in the graph, which vulnerability instances need to be removed in order to prevent the attacker from reaching the prerequisite, and which states the attacker cannot reach with the prerequisite absent.

We accomplish this by rebuilding the MP graph for each potential recommendation, noting which vulnerability instances are actually necessary to reach the selected prerequisite and which states are no longer achievable. Some prerequisites may yield identical recommendations. We discard duplicates in these cases.

We weight recommendations based on the number of critical hosts denied the attacker. A user could supply per-host "asset values" or weights to prioritize steps that protect critical servers.

Cyber Moving Target

The architectural and assessment components provide a system that dynamically moves in multiple dimensions in order to survive in a contested environment (see figure 5). The reachability analysis and attack graphs provided by NetSPA assess the cyber threats to a mission critical application in a hostile environment and facilitate reconfiguration and reorientation when facing a new threat. The assessment includes both known and zero-day (hypothetical) vulnerabilities so that the impact of previously unknown weaknesses can also be analyzed. Upon detecting a change in the threat level (as a result of a new vulnerability being discovered or an actual attack detection event), TALENT, the architectural component, facilitates the reconfiguration and reorientation of the critical applications by dynamically changing their platform and subnet to a survivable combination based on the recommendations provided by the assessment tool. In essence, the two technologies implement

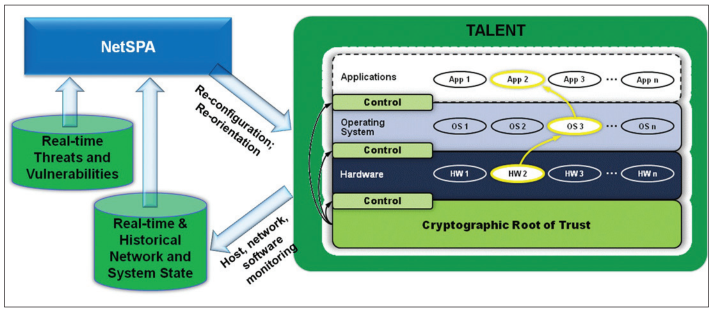
NetSPA
Real-time Threats and Vulnerabilities
Real-time & Historical Network and System State
Re-configuration; Re-orientation
Host, network, software monitoring
TALENT
Applications
App 1
App 2
App 3
App n
Control
Operating System
OS 1
OS 2
OS 3
OS n
Control
Hardware
HW 1
HW 2
HW 3
HW n
Control
Cryptographic Root of Trust

Figure 5. The architectural and assessment components provide a system that dynamically moves in multiple dimensions in order to survive in a contested environment.

an OODA loop (observe, orient, decide, and act) in which the observe and decide phases are provided by NetSPA and the orient and act phases are implemented by TALENT.

A system that deploys these components can provide a cyber moving target (also known as a polymorphic system) that changes its properties in a hostile environment. Cyber exploits are often feasible against a set of platforms (hardware/operating systems). By dynamically changing platform properties, a cyber moving target can mitigate platform specific exploits. The destination platform and subnet are chosen intelligently based on the result of reachability and attack graph analysis to resist the current threat level.

A Cyber Survivable Future

We believe cyber security must shift focus from cyber protection technologies to cyber survivability. Security incidents in highly protected environments have illustrated the fact that a motivated adversary can compromise even the most secure systems. We believe that new cyber security paradigms must leverage architectural and assessment technologies to create a cyber moving target that disadvantages potent adversaries and facilitates recovery and mission survivability after a successful compromise. It is imperative that we clearly understand the mission impact of potential security breaches and implement operate through capabilities in order to continue the mission objectives of the critical applications during and after a cyber incident rather than deploying protections and hoping that they can resist attacks.

In a contested environment with motivated, well resourced adversaries, it is likely that traditional cyber protection technologies will be bypassed or disabled as a result of previously unknown attacks. Achieving cyber survivability for critical warfighting systems necessitates the use of game-changing technologies that can get ahead of the adversaries.

In this work we described architectural ideas that can help improve the survivability of a mission critical system against cyber adversaries and prototyped an architectural component to demonstrate their feasibility. We also described an evaluation tool that analyzes the possible attack paths to a system and supports the architectural component. We are in the process of developing analysis and experimentation techniques to quantify the effectiveness and protection offered by these components which we leave as the future work.

Notes:

[1] Lolita C. Baldor, "New Threat: Hackers Look to Take Over Power Plants," *The Associated Press*, August 2010, http://abcnews.go.com/Business/wireStory?id=11316203.

[2] Rodney H. Brown, "Stuxnet worm causes industry concern for security firms," *masshightech.com*, October 2010, http://www.masshightech.com/stories/2010/10/18/daily19-Stuxnet-worm-causes-industry-concern-for-security-firms.html.

[3] "Report on Technology Horizons: A Vision for Air Force Science & Technology During 2010–2030," AFST-TR-10-01-PR, US Air Force Chief Scientist, May 2010, http://www.af.mil/shared/media/document/AFD-100727-053.pdf.

[4] Cybersecurity Progress after President Obama's Address, The White House National Security Council, July 2010, http://www.whitehouse.gov/administration/eop/nsc/cybersecurity/progressreports/july2012.

[5] Anup Ghosh, Ivan Arce, "Guest Editors' Introduction: In Cloud Computing We Trust - But Should We?," *IEEE Security & Privacy* 8, no. 6, (November–December 2010): 14-16.

[6] Hamed Okhravi et al., "TALENT: Dynamic Platform Heterogeneity for Cyber Survivability of Mission Critical Applications," Secure and Resilient Cyber Architecture Conference (SRCA'10), McLean, Virgina, 29 October 2010, https://register.mitre.org/sr/papers1/TALENT.pdf

[7] Daniel Geer et al., "Cyber*In*security: The Cost of Monopoly," September 2003, http://cryptome.org/cyberinsecurity.htm.

[8] D. Williams et al., "Security through Diversity: Leveraging Virtual Machine Technology," *IEEE Security & Privacy* 7, no. 1 (January–February 2009): 26-33.

[9] Kyle Ingols et al., "Modeling Modern Network Attacks and Countermeasures Using Attack Graphs," Annual Computer Security Applications Conference (ACSAC '09), 7-11 December 2009, 117-126.

[10] Gabriel Rodríguez et al., "CPPC: a compiler-assisted tool for portable checkpointing of message-passing applications," *Concurrency and Computation: Practice and Experience* 22, issue 6 (April 2010): 749-766.

[11] HDF4 Reference Manual, The HDF Group, February 2010, http://www.hdfgroup.org/release4/HDF4_RM_html/RM_Front.html.

[12] Mirella M. Moro et al., XML: Some Papers in a Haystack, *SIGMOD Record* 38, no. 2 (October 2009): 29-34.

[13] P. Meil et al., National Vulnerability Database (NVD), http://nvd.nist.gov.

[14] Bugtraq vulnerability database, *SecurityFocus.com*, http://www.securityfocus.com/archive/.

[15] Randal E. Bryant, "Graph-Based Algorithms for Boolean Function Manipulation," *IEEE Transactions on Computers* 35, issue 8 (August 1986): 677-691.

Dr. Hamed Okhravi (MS and PhD, Electrical and Computer Engineering, University of Illinois at Urbana-Champaign [UIUC]) is a member of technical staff at the Cyber Systems and Technology Group of MIT Lincoln Laboratory, where he conducts research in the area of cyber survivability and security architectures.

Currently, Dr. Okhravi is developing cyber-attack survivable systems and networks. The current effort focuses on creating a cyber moving target using platform heterogeneity.

Mr. Joshua W. Haines (BS, Electrical Engineering, Union College; MS, Electrical and Computer Engineering, University of Massachusetts at Amherst) is an assistant group leader in the Cyber Systems and Technology Group at MIT Lincoln Laboratory. He is responsible for managing research and development of technology and systems in support of national cyber missions including computer network defense, attack, and exploitation. Focus areas include system analysis, architecture engineering for robustness and security, development of network-centric cyber systems, automated analysis of network vulnerabilities, red-teaming of Department of Defense programs, and development and deployment of traffic generation and test development for test range environments.

Mr. Kyle Ingols (SB and M.Eng., Electrical Engineering and Computer Science, MIT) is a member of technical staff in the Cyber Systems and Technology Group of MIT Lincoln Laboratory. His current work covers a broad spectrum of cyber security, including secure system design, scalable aggregation of network defender data, and protection against physical access to cyber systems.