LLCySA: Making Sense of Cyberspace

Scott M. Sawyer, Tamara H. Yu, Matthew L. Hubbell, and B. David O'Gwynn

Today's enterprises require new network monitoring systems to detect and mitigate advanced cyber threats. The Lincoln Laboratory Cyber Situational Awareness (LLCySA) system gives network operators insights into the current state of an enterprise-scale network and provides cyber analytics researchers a platform for algorithm development. LLCySA includes highly scalable data warehousing, real-time data ingestion, responsive querying for both human users and automated analytics, and the computational capabilities required by advanced algorithms.

» **Government and business operations** rely heavily on cyber infrastructure. Users require high availability for their day-to-day work, and organizations expect sensitive data, including valuable proprietary or classified material, to be protected from destruction or unauthorized disclosure. At the same time, malicious hackers, as well as threats within an organization, pose risks to network availability and data security. However, most enterprise networks are ill-equipped to prevent, detect, or respond to sophisticated threats. Lincoln Laboratory researchers are exploring data-driven approaches to network protection.

Imagine a cyber analyst navigating a three-dimensional (3D) game, walking down virtual office building hallways, inspecting network activity emanating from each office. Suddenly, flying in from another part of the building, a computer, engulfed in digital flames, alerts the analyst to anomalous activity. A quick inspection reveals that the problem is botnet activity: malware has infected this PC, and hackers are attempting to gain control. With a few deft keystrokes, the analyst quarantines the system and begins patching the infected operating system. The Lincoln Laboratory Cyber Situational Awareness (LLCySA) program is integrating the data, algorithms, and interfaces to help make this vision a reality.

The mission of cyber situational awareness (SA) is to provide detailed, current information about the state of a network and its path in arriving at that state. A key component of this mission is building and maintaining a real-time understanding of active devices and connections on the network. Situational awareness in cyberspace differs from SA in traditional domains because cyberspace lacks clear physical boundaries, situations evolve very quickly, and the quantity of actors and activity can be substantial. However, large amounts of information are available to address the problem—a fact making cyber SA both tractable and more challenging.

Current security tools fail to detect and prevent advanced attacks or to provide the general cyber SA necessary for resilient operations. Many enterprise networks use security information event management (SIEM) systems to monitor network events. However, enterprise network operations generally require several stove-piped tools to cover all requirements, and correlating data between the tools is challenging. Existing tools also have limited extensibility and scalability. New data sources and analytics often require contracting with the SIEM developer for system modifications. Additionally, the systems do not scale well to cluster computing environments that utilize parallel architectures for storage and processing.

Cyber SA serves two main functions: operations and analysis. Network operations require real-time representation of the network to assess defensive posture, help maintain network availability and performance, and identify potential systems of interest. On the other hand, network analysis is primarily forensic and involves responding to security incidents or investigating infected devices. Effective cyber SA enables operators to better secure their network, provides greater visibility of friends and foes, and facilitates the planning, commanding, and execution of missions.

Lincoln Laboratory has developed a system to provide robust SA on enterprise-scale networks using big data technologies. LLCySA's technical approach is to collect information from network sensors, such as server log files and monitoring services, and perform timely analysis of these datasets. LLCySA's main challenges involve achieving scalability while maintaining extensibility. The volume, velocity, and variety of incoming data require the use of a scalable distributed database system combined with a carefully designed storage scheme and parallel ingest processing. Data modeling and query processing enable analysts to construct complex queries and analytics that fuse and enrich information from disparate data sources to better address SA questions. Finally, a plug-and-play software architecture enables rapid insertion of new data sources and analytics, and promotes portability of system components to various networks.

Research and Development on a Real Network

LLCySA's research approach is based on bringing real data from Lincoln Laboratory's network into a research environment. Research and development occurs in the Lincoln Research Network Operations Center (LRNOC), where research groups can directly collaborate with the Laboratory's network operations and security departments. By talking to real operators and analysts, researchers gain a deeper understanding of their SA challenges and needs. To date, dozens of use cases have been identified as targets for analytic research and development.

Within the LRNOC, researchers experiment with operationally relevant datasets. LLCySA currently stores more than 50 event types, including server log files, aggregated traffic statistics, alerts from network appliances, and enterprise data. Data sources routinely include firewalls, routers, web proxy servers, email servers, name servers, network and host-based intrusion detection systems, enterprise applications, and other custom systems. The raw data could be in a structured, semistructured, or binary format; arrive streaming or in batches; and must be parsed and preprocessed before being inserted into the database. A data-enrichment process can create additional event types and fuse entities represented in multiple data sources.

Safeguarding Lincoln Laboratory's real network data both for security and privacy is a priority and a condition for data usage. Physical and logical controls have been put in place to protect access and assure security of the data. Additionally, all researchers using the data have signed specific data-handling agreements on what is acceptable use and handling of the data.

This research environment enables cyber security research staff, network operators, and software engineers to closely collaborate while analyzing real data. This approach results in rapid and agile innovation cycles, as tools can be prototyped with immediate feedback from end users. It also accelerates the timeline for research ideas to become mature technologies running on a real network.

Fighting Cyber Threats with Big Data Technology

Enterprise networks are large and complex, as an organization may have thousands of users and devices, dozens of servers, multiple physical locations, and many users connecting remotely. Achieving cyber SA on these networks involves dealing with massive quantities of data.

To support forensics and advanced analytics, the system must maintain a long history of event information, which in turn requires scalable, distributed storage at the terabyte to petabyte scale. The system must also keep pace with a data rate, engendered by various attached network sensors, that far exceeds the insertion rates of traditional database systems. Each sensor provides information of a different type, in both format and semantics, and these sources may change over time.

Big data challenges are characterized by data volume, velocity, and variety. Traditional servers and software architectures do not scale, and thus LLCySA relies on big data technologies and cluster computing to achieve true scalability. The centerpiece of our big data approach is Apache Accumulo [1], an open-source distributed database with origins in the intelligence community. Coupled with a parallel-processing framework, the system can ingest the data from various sources in real time while simultaneously computing key statistics for use by analytical algorithms. Support for big data is a key requirement for the LLCySA architecture, and we have accordingly designed LLCySA to deploy on any network without compromising scalability.

Systems that attempt to monitor network activity by capturing and processing full-packet data are fundamentally limited by the amount of data they can retain and the complexity of processing they can perform at line speed. Therefore, the primary information sources for the LLCySA system are events generated by network sensors rather than the complete data flow. This design decision results in the loss of some information but also enables a robust system capable of performing advanced analytics over extended time ranges.

Events broadly represent a record of something happening in the cyber environment and can include activity (a user requesting a web page), protocol events (a server assigning an address to a computer), and observations from monitoring systems (a network scanner detecting an operating system vulnerability on a particular computer). An event record ties together attributes of various types or formats, including the Internet protocol (IP) addresses, physical addresses, protocol names, domain names, and uniform resource identifiers. These attribute types often imply a particular string format (e.g., IP addresses consist of four numbers separated by periods) and some semantic meaning (e.g., a physical address uniquely identifies a real or virtual network interface). Event occurrences imply some sort of connection between attributes, and analysts and algorithms can exploit these connections to gain insight into network activity.

In addition to archiving network events, LLCySA has tools to enable analysts to explore activity. Thus, the system provides responsive and flexible data retrieval. However, combining the capacity and scalability of big data tools with the interactivity of more traditional databases is a particularly tough challenge. Lincoln Laboratory is a leader in adapting scalable database technology to meet the needs of analysts and algorithm developers, and we have used multiple techniques to meet data retrieval requirements.

Cyber Situational Awareness Architecture

The LLCySA system architecture comprises components for data ingest, storage, query, and analysis while providing cluster-computing resources for parallel processing. Figure 1 shows the system components. The system supports three types of users: the data engineer, analytics developer, and security analyst. The data engineer understands the raw network data feeds and writes applications for the ingest platform that parse and import the data sources into the system. The analytics developer writes applications that run on the analytics platform. These applications can request data through the query-processing subsystem and can insert new data to issue alerts or to enrich existing records. Some analytics applications are designed to interact with a user (such as browser-based or command-line event search tools), while others run as an uncued background process searching for particular types of network activity. The security analyst operates the various analytics applications and can issue queries for the purpose of network operations or forensic analysis. This three-pronged design enables the system to perform ingest, query, and analysis of the network data in an efficient and scalable way.

Data flow in the system starts with the various data sources being initially staged on the central file system. The data sources contain information about network events (authentication, web requests) involving entities on the network (machines, users). The parallel ingest platform handles the necessary parsing and preprocessing prior to ingesting the event records as rows in a database table. Once ingested, network events can be queried by applications running in the analytics platform. The

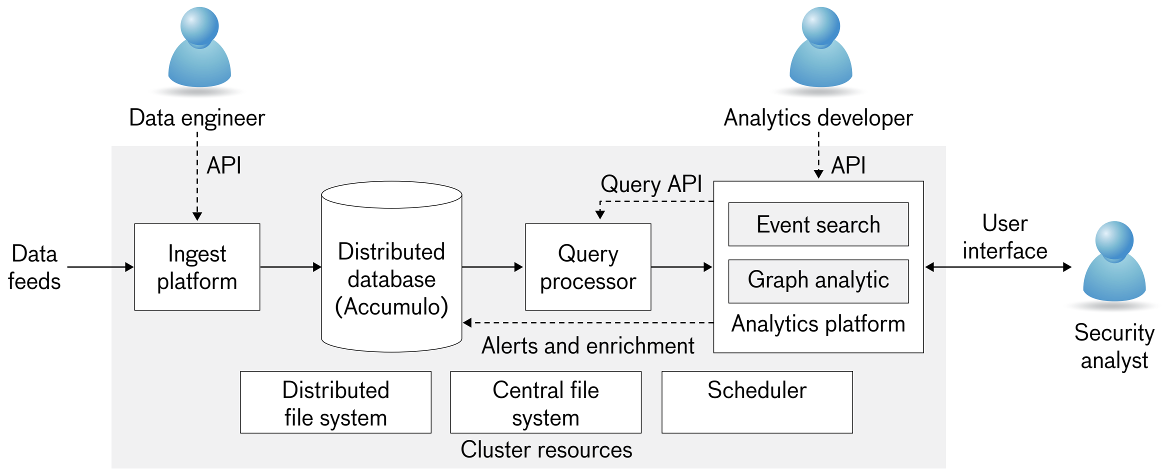


FIGURE 1. The LLCySA architecture enables developers and analysts to use high-performance computing resources to draw conclusions from massive datasets. The ingest platform adds data feeds to the distributed database, from which applications query data via the analytics platform. API is application programming interface.

query processor is designed to deliver responsive results to these applications. For some types of queries, parallel clients can be used to accelerate data retrieval. These processes all run within a cluster environment, as shown in Figure 2.

Scalable Storage and Ingest

The LLCySA system handles data volume with a multinode distributed Accumulo database instance, which is based on Google's BigTable design [2]. Our initial instance of the system uses eight server nodes for 60-terabyte capacity, but the database has been demonstrated to scale to more than 1000 nodes. Accumulo is a key-value store that achieves scalability by sorting records at the time of ingestion and that offers fast lookups along a single dimension [3]. Records in Accumulo are stored in lexicographical order by key—a row identifier and column identifier. This key structure encourages Accumulo's use as a tabular store in which each record is a row identifier, a column identifier, and a value. Sorting by key ensures that scans based on a single row or range of rows are very fast, regardless of how large the database grows.

For maximum retrieval performance, the keying scheme must be carefully designed such that the majority of common queries can be performed by simply selecting a range of rows. Unfortunately, given the wide variety of data sources being ingested and the need to support ad hoc exploratory queries, it is impossible to design any one row-key structure to make all potential queries highly efficient. However, one dimension stands out as applicable to all event data: a time stamp. Given the frequency with which users restrict queries by time range, we encode time into the row key to give time restrictions first-class support. Seen in Figure 3a, our row key is composed of three parts: a shard (or partition) number, the time stamp of the event occurrence (reversed with respect to the Unix epoch), and an 8-byte hash of the event to minimize the chance of key collisions in dense datasets.

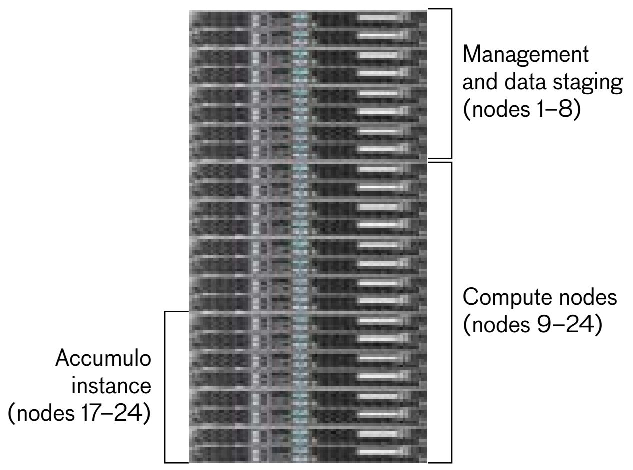
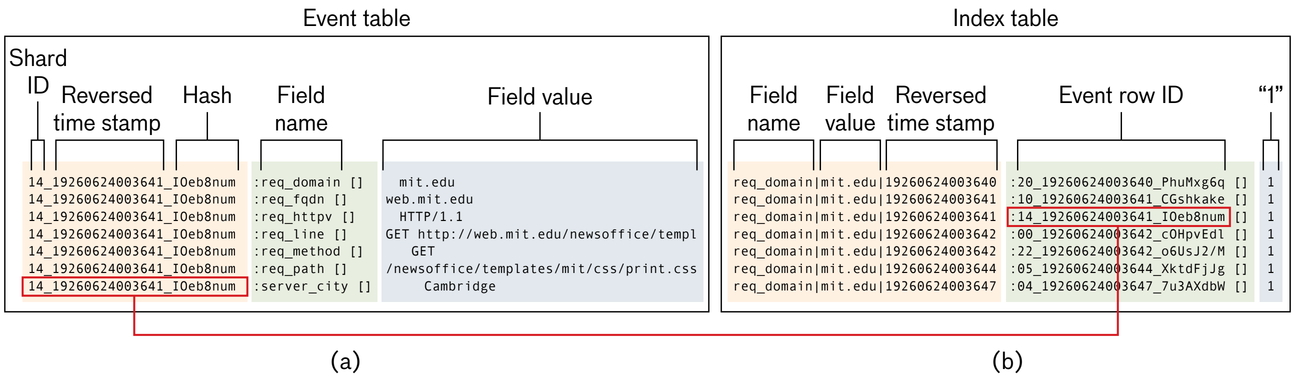

Management and data staging (nodes 1–8)
Compute nodes (nodes 9–24)
Accumulo instance (nodes 17–24)

FIGURE 2. TX-CYSA is a 24-node compute cluster that provides more than 200 terabytes of distributed storage and 576 processing cores for cyber analytic development and network monitoring.



```
14_19260624003641_IOeb8num :req_domain []      mit.edu
14_19260624003641_IOeb8num :req_fqdn []       web.mit.edu
14_19260624003641_IOeb8num :req_httpv []       HTTP/1.1
14_19260624003641_IOeb8num :req_line []       GET http://web.mit.edu/newsoffice/templ
14_19260624003641_IOeb8num :req_method []        GET
14_19260624003641_IOeb8num :req_path []       /newsoffice/templates/mit/css/print.css
14_19260624003641_IOeb8num :server_city []         Cambridge
```

```
req_domain|mit.edu|19260624003640 :20_19260624003640_PhuMxg6q [] 1
req_domain|mit.edu|19260624003641 :10_19260624003641_CGshkake [] 1
req_domain|mit.edu|19260624003641 :14_19260624003641_IOeb8num [] 1
req_domain|mit.edu|19260624003642 :00_19260624003642_cOHpvEdl [] 1
req_domain|mit.edu|19260624003642 :22_19260624003642_o6UsJ2/M [] 1
req_domain|mit.edu|19260624003644 :05_19260624003644_XktdFjJg [] 1
req_domain|mit.edu|19260624003647 :04_19260624003647_7u3AXdbW [] 1
```

FIGURE 3. The LLCySA storage schema uses two tables per event type and carefully designed keys to support efficient queries. The event table is the primary query table set in reversed time stamp order. The index table enables queries to quickly retrieve records containing a particular value (in field name or value) and links back to the corresponding event table row.

In database architectures, sharding is a horizontal partitioning of data across servers such that groups of rows are stored together. LLCySA achieves sharding by beginning the row identifier with a shard number. This number corresponds to a zero-padded integer string between 00 and 31, for a total of 32 shards. The shard count is chosen at design time and should be a multiple of the total number of database servers. The purpose of the shard is to maximize parallelism among the servers, both at query and ingest time. The shard of a particular event can be chosen in a number of ways. We experimented with various methods and found that beginning a row key with a random shard number ensures uniform data distribution across the tablet servers with the best ingest and retrieval performance.

The time stamp is included to provide for time-oriented query optimization. It is reversed such that recent events appear before older events when sorted. To calculate the reversed time stamp, the event time is converted to a Unix-style time stamp (i.e., the number of seconds since the Unix epoch), negated, and projected back to a Gregorian calendar and formatted without delimiters. In a real-time system, it is useful to be able to quickly query for the latest events. Our reversed time stamp ensures that any query for a particular time period will return the most recent events first.

The column key and value contain the field name and field value, respectively. This scheme is based on Lincoln Laboratory's Dynamic Distributed Dimensional Data Model (D4M) schema [4] with a slight modification to enable some performance improvements when computing projections of very long rows. The D4M computation system [5] can be readily adapted to accommodate this schema change.

Our keying scheme for the event table allows efficient time-based queries, but all other queries require an exhaustive search of row values within that time period, which would be prohibitively time-consuming. To help solve this issue, we employ an index table for a subset of event fields. The structure of these index entries is shown in Figure 3b. To preserve storage space, only the subset of fields likely to be included in query conditions is indexed. Given a query on a particular column-value pair (e.g., "select all entries where `src_ip` equals `10.10.10.87`"), a scan on the secondary index table is dispatched for all row identifiers starting with `src_ip|10.10.10.87`. This scan will quickly return all row keys in the event table where this condition is met. If necessary, these rows can then be retrieved from the event table to get the event occurrence's other fields.

The high-velocity network data are processed in parallel to support real-time preprocessing and ingest. Ingest is a batched process, with log files and network sources processed at regular intervals. The process is highly data parallel, as the raw data can be easily partitioned across the cluster for preprocessing and insertion into Accumulo. Preprocessing steps include parsing the raw data and transforming it into one of two common file formats: comma-separated value (CSV) or JavaScript object notation (JSON). Since both of these formats are supported by a wide array of development environments, integrating new data sources is straightforward. After preprocessing, ingestion of the outputted files is parallelized, so that rows in CSV files and objects in JSON files are transformed into Accumulo *writes* (sets of row, column, and value triples). Aggregate database ingest rates scale well with the number of clients, so this parallelization is efficient and can be implemented using well-known job schedulers such as Open Grid Engine.

We have demonstrated ingest scalability up to 256 processors on the TX-CYSA cluster [6]. Given the current cluster size and 10 Gbps network, data ingest has not yet fully utilized the network. However, as this application grows to larger clusters to handle petabyte-scale datasets, scalability will become limited by network bandwidth.

Query Processing

The LLCySA query processor provides composable operators for selecting, filtering, and aggregating entries [7]. The operator model is based on Accumulo's concept of entry iterators. In Accumulo, all server-side data operations are handled through a Java iterator framework. At scan time, entries are read off disk or from a memory cache into a tree of cascaded iterators, in which inputs and outputs are sets of key-value pairs. Accumulo server-side iterators are stateless between iterations and cannot initiate new scans (although seeking other entries within a row is possible). These restrictions enable trivial parallelism, but they preclude standard Structured Query Language capabilities, such as "distinct" and "join" operations. Such tasks, therefore, must be performed client-side.

Data retrieval in LLCySA occurs through queries composed of one or more operators. These operators run externally from the database servers, and the first operator always initiates Accumulo scans. Each query specifies a database table name, a list of fields (i.e., columns) to be retrieved, and a set of conditions. Other operators accomplish tasks such as aggregating results (e.g., forming histograms), limiting the number of results, or splitting the result set into pages. Unlike server-side iterators, LLCySA operators preserve states between iterations, enabling computationally trivial determination of aggregates such as histograms and counts of distinct values. Loosening the state preservation restriction complicates query parallelization, but several strategies exist for exploiting concurrency.

Query conditions are evaluated by filtering entries on the tablet servers or by using the index table. Generally, the Accumulo iterator framework operates on individual entries (i.e., single key-value pairs). However, the LLCySA query model applies restriction conditions to entire rows, such that a row is accepted or rejected by the iterator depending on whether that row's set of columns collectively satisfy query conditions.

Accumulo is designed for petabyte-scale data analysis and has been demonstrated to solve out-of-core graph traversals requiring 10 to 100 hours to complete. Thus, it is not tuned for low-latency performance on interactive queries. Scan results are sent to the client in batches of automatically determined size. This batching can impose a latency penalty of several seconds before the client receives the first result. Moreover, Accumulo has no built-in feature for limiting the number of scan results. Thus, the user may need to wait for a large number of results to batch on the tablet servers even if only a small number of results are desired.

Therefore, to improve interactive query performance, we have implemented an adaptive batching technique based on the time range. All queries specify a particular time range by which to restrict results. Instead of executing a query over the entire range at once, the LLCySA technique splits the time range into batches, which change in size adaptively depending on the number of results returned and the runtime of the query. Our query batching technique achieves improvement in query responsiveness of more than 10 times over other techniques as measured by the latency in receiving the first result.

Analytics Platform

The LLCySA data storage, retrieval, and processing resources form a platform on which developers create analytical and user-facing applications. These applications can request data through the query processor, insert new data, or interface directly with the Accumulo database or distributed file system. Rather than restricting developers to a particular programming language or application programming interface, LLCySA provides a flexible Linux cluster environment based on the LLGrid software stack [8]. This platform enables algorithm development in MATLAB/D4M, Python, R, and other high-level languages, while user interfaces can be implemented as web services using any software framework or library. In particular, we highlight two applications built on the LLCySA platform.

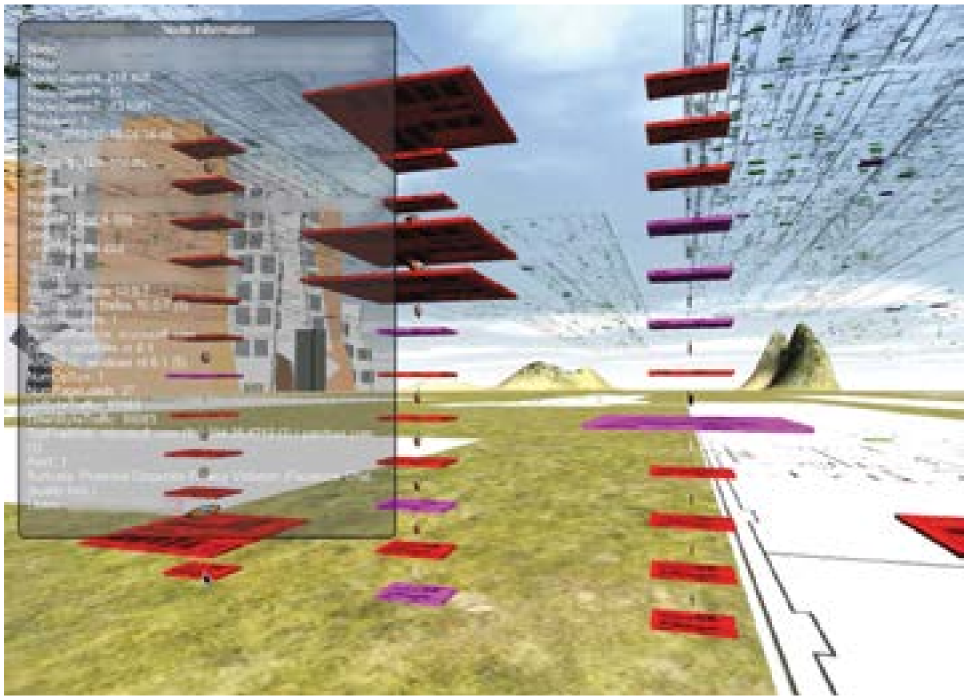


FIGURE 4. CYSA3D immerses a user in a 3D representation of the Laboratory network, allowing an analyst to quickly observe the state of the network and interrogate devices encountered along the way.

CYSA3D: Visualizing Cyberspace

Obtaining SA of network activity across an enterprise presents unique visualization challenges. Cyber operators and analysts are required to quickly gather and correlate large volumes of disparate data to identify the existence of anomalous behavior. Traditional network SA tools present volumes of logs and graphs of data in a variety of forms. Over time, the stream of logs and scatter plots, bar charts, pie charts, and graphs lose much of their meaning because of information overload.

One approach to this challenge leverages technology utilized in the 3D gaming industry. The video-game medium provides a platform for users to immerse themselves in a world in which the player is able to absorb a tremendous amount of environmental information rapidly and sustain it for a long time. Within LLCySA, the 3D environment employs a diverse set of cues to creatively depict a computer's behavior, state, and location. For example, a computer downloading an unusually large amount of data may be depicted as a computer icon spinning around, which indicates abnormal behavior. These tools can create a virtual world that accurately represents the physical. Accurate geolocation of the assets lets the user seamlessly identify the location of network assets operating on the network. The user is then able to obtain pertinent information about the assets of interest, thereby enabling accurate SA and even triggering a real-world response.

On the basis of this concept, researchers have developed CYSA3D (for cyber situational awareness in 3D), a real-time visualization of the Lincoln Laboratory network, shown in Figure 4. The application relies primarily on three data inputs. The first is the network access control data feed that contains events of hosts connecting and disconnecting from the network and profiles of those hosts, including connecting network ports, assigned IP addresses, media-access control addresses, operating systems, and owner information. Second, the application imports electronic drawings of the Laboratory's floor plans and extracts the locations of its nearly 10,000 network faceplates. The third input is a mapping of network ports to faceplates maintained by the Facility Services Department. By fusing these three data sources, the application produces a virtual representation of the Laboratory, in which network assets are geolocated onto building drawings.

CYSA3D allows a user to patrol through this virtual environment and interrogate devices encountered along the way. Devices are assigned visual cues (e.g., colors, icons, or simple animations) to show their properties.

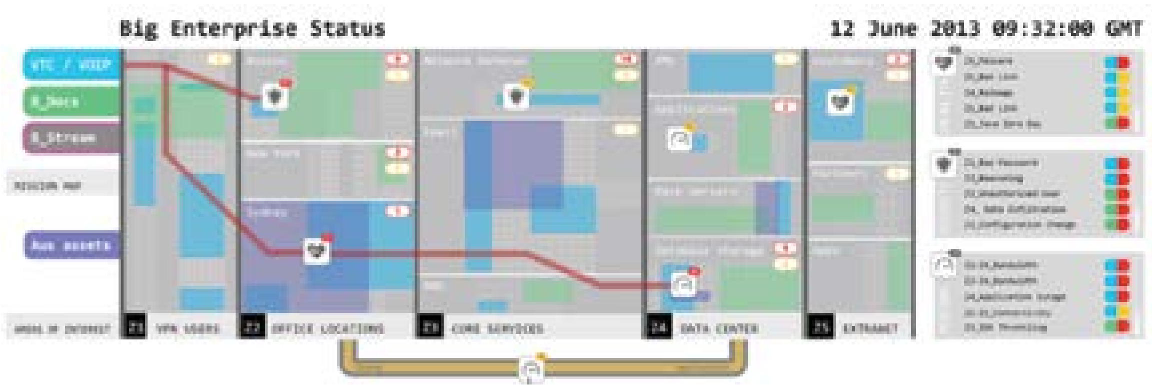


FIGURE 5. A multidisciplinary team led by Diane Staheli designed the "Big Enterprise BigBoard" to provide a concise visual representation of the state of the hypothetical enterprise network. For more information on this interface, see the corresponding LabNote on page 12.

If additional data feeds are available, such as web activity logs or alerts from intrusion detection systems, the application can augment the display of devices with such information. In addition to using the patrol mode of interaction, users can search based on arbitrary criteria and have the resulting devices fly directly into the field of view. For example, users may pull all devices that have been flagged by an intrusion detection system and triage all the alerts.

Graph-Based Botnet Detection

A key benefit of LLCySA is support for advanced analytical techniques that derive actionable intelligence from large datasets. One such technique considers the mathematical graph in which vertices represent internal computers and external web servers, and edges represent web requests between two computers. Researchers have demonstrated an algorithm capable of detecting the signature of malicious botnet traffic in the presence of normal web traffic and have integrated this technique into LLCySA as an uncued background analytic.

This graph analysis technique applies detection theory from conventional signal processing theory to find threats in a noisy background [9]. The approach formulates the web-connection graph as a sparse adjacency matrix and applies sparse matrix decomposition techniques to establish a model for normal activity and reasonable thresholds for anomalous activity. The anomalous subgraph detection analytic is implemented in D4M, which leverages the large web traffic archive stored in Accumulo.

Continuing Analytic and Application Development

The LLCySA program has demonstrated various capabilities on a pilot system and has already deployed capabilities on the Lincoln Laboratory network. Further development and technology transitions are planned for the Laboratory's network as well as for other tactical networks.

As the LLCySA program enters its final year, the team's focus is shifting to building additional applications, preparing the platform to transition into a support phase, and deploying the tools to the Laboratory's network operations. Following the conclusion of the program, the platform and the LRNOC will remain operational for future research efforts.

The research team continues to collaborate with the Laboratory's internal security departments to identify opportunities to design interfaces and applications that can directly aid network analysts and operators in their daily tasks. A cross-divisional Laboratory team designed a visual analytic for enterprise network SA [10], which has inspired ongoing LLCySA prototyping and development. Figure 5 shows a sample screen from this interface design.

Additionally, big data analytic development for cyber security continues as part of the LLCySA program and through several other Laboratory efforts. Researchers are applying the latest data science techniques and the Laboratory's cyber expertise to detect network threats more quickly and effectively. LLCySA serves as an important research platform for proof-of-concept development before deployment on the Laboratory's network and transition to tactical networks in the Department of Defense and intelligence community.

Acknowledgments

This article highlights the work of the entire LLCySA team: Ethan Aubin, Chansup Byun, Suresh Damodaran, Christopher Degni, Jack Fleischman, Jeffrey Gottschalk, Nwokedi Idika, Stephen Kelley, Jeremy Kepner, Peter Michaleas, Jeremy Mineweaser, Matthew Schmidt, Alexia Schulz, Diane Staheli, An Tran, Seth Webster, and James Will.

The team would also like to acknowledge the following stakeholders: from the Lincoln Research Network Operations Center (LRNOC), Joshua Haines, Chad Meiners, Maureen Hunter, Philip Marquardt, David Danico, and Ronald Hersey; from the Information Services Department (ISD), James Medeiros and Stephen Castellarin; and from the Security Services Department (SSD), John "Scott" Macdonald and Paul Burke. ■

References

1. Apache Accumolo^{Tm} website, http://accumulo.apache.org/
2. F. Chang, J. Dean, S. Ghemawat, W. Hsieh, D. Wallach, M. Burrows, T. Chandra, A. Fikes, R. Gruber. "Bigtable: A Distributed Storage System for Structured Data," *ACM Transactions on Computer Systems*, vol. 26, no. 2, 2008.
3. S. Patil, M. Polte, K. Ren, W. Tantisiriroj, L. Xiao, J. Lopez, G. Gibson, A. Fuchs, B. Rinaldi. "YCSB++: Benchmarking and Performance Debugging Advanced Features in Scalable Table Stores," *Proceedings of the 2nd ACM Symposium on Cloud Computing,* 2011.
4. J. Kepner, C. Anderson, W. Arcand, D. Bestor, W. Bergeron, C. Byun, M. Hubbell, P. Michaleas, J. Mullen, D. O'Gwynn, et al., "D4M 2.0 Schema: A General Purpose High Performance Schema for the Accumulo Database," *Proceedings of the 2013 IEEE High Performance Extreme Computing Conference,* 2013. Available online at http://ieee.hpec.org/2013/agenda.htm.
5. J. Kepner, W. Arcand, W. Bergeron, N. Bliss, R. Bond, C. Byun, G. Condon, K. Gregson, M. Hubbell, J. Kurz, et al., "Dynamic Distributed Dimensional Data Model (D4M) Database and Computation System," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),* 2012, pp. 5349–5352.
6. C. Byun, et al. "Driving Big Data with Big Compute." *Proceedings of the 2012 IEEE High Performance Extreme Computing Conference*, 2012. Available online at http://ieee.hpec.org/2012/agenda.htm.
7. S. Sawyer, D. O'Gwynn, and T. Yu, "Understanding Query Performance in Accumulo," *Proceedings of the 2013 IEEE High Performance Extreme Computing Conference*, 2013. Available online at http://ieee.hpec.org/2013/agenda.htm.
8. N.T. Bliss, R. Bond, J. Kepner, H. Kim, and A. Reuther, "Interactive Grid Computing at Lincoln Laboratory," *Lincoln Laboratory Journal*, vol. 16, no. 1, 2006, pp. 165–216.
9. B. Miller, N. Bliss, P. Wolfe, and M. Beard, "Detection Theory for Graphs," *Lincoln Laboratory Journal*, vol. 20, no. 1, 2013 pp. 10–30.
10. D. Staheli, A. Brennen, D. Danico, R. Harnasch, M. Hunter, R. Larkin, J. Mineweaser, K. Nam, D. O'Gwynn, H. Phan, A. Schulz, M. Snyder, and T. Yu, "Situational Awareness Display Design," VAST Challenge, 2013. Available as a paper, "A Novel Display for Situational Awareness at the Network Operations Center," at users.soe.uscs.edu/~pang/visweek/2013/vast/challenge.html.

About the Authors

Scott M. Sawyer is an associate member of the technical staff in the Computing and Analytics Group at Lincoln Laboratory. His diverse research activities have included characterizing the performance of big data technology, designing scalable software architectures, parallelizing computer vision techniques, and co-inventing a novel, photonically enabled multicore processor architecture. He joined the Laboratory in 2011 from Lockheed Martin, where he worked on naval radar in software, electronics and systems engineering roles and was awarded a patent for secure memory technology. He received his bachelor's degree from Villanova University in 2006, summa cum laude, and his master's degree from the University of Pennsylvania in 2010, both in electrical engineering.

Tamara H. Yu is currently a technical staff member of the Cyber Systems and Operations Group at Lincoln Laboratory. Her research focuses on data fusion and visualization for network security. Her current project aims to create an enterprise-scale data and analytic platform based on open standards and net-centric principles. Previously, she has developed a number of tools for visualizing security metrics and monitoring cyber threats. She has also contributed to the Lincoln Adaptable Real-time Information Assurance Testbed (LARIAT), focusing on test configuration and monitoring as well as desktop application actuation. She received bachelor's and master's degrees in computer science from the Massachusetts Institute of Technology in 2004.

B. David O'Gwynn is a member of the technical staff in the Cyber Systems and Operations Group. Since joining Lincoln Laboratory in 2011, his research activities have included cloud computing and storage, big data analytics and visualization, user-interface design, and the intersection of these areas with the national challenge of cyber security and defense. He received his bachelor's degree in mathematics from Belhaven University, his master's degree in computational engineering from Mississippi State University, and his doctorate in computer and information sciences from the University of Alabama at Birmingham.

Matthew L. Hubbell is a system engineer in the Computing and Analytics Group. Since joining Lincoln Laboratory in 2010, his activities have included building high-performance computing (HPC) systems as a member of the LLGrid team. He is currently deploying TX-Green, the next-generation LLGrid general-purpose Laboratory-wide HPC compute cluster located in the Holyoke, Mass., data center. His other activities include the development of immersive 3D environments leveraging video-game technology to provide a multiplayer platform to achieve situational awareness in diverse environments. He received a bachelor's degree in business management from Gettysburg College in 2002 and a master’s degree in business administration from the University of Massachusetts at Lowell in 2010.