An Integrated CyberSecurity Approach for HEP Grids and Clusters

William E. Johnston[a]*, Eugene Schultz*[b]*, Miron Livny*[c]*, Bart Miller*[c]*, Shane Canon*[b]*, Mike Helm*[d]*, Doug Olson*[b]*, Iwona Sakrejda*[b]*, and Brian Tierney*[b]

1 Motivation

A Security Incident Scenario

It's 8:45 a.m. Monday morning. As scientists arrive at a high energy physics research center, they soon find that something is wrong. The Linux hosts they use for Grid computing cannot connect to anything other than local mail and Web servers. They are unable to contact their site's network operations. Time passes and they are still unable to connect to anything but local systems. When they do reach the networking staff they learn that all incoming and outgoing traffic is being blocked at the network's external gateway at the direction of the site Chief Information Officer's (CIO). This is due to an as yet unexplained security problem and a massive investigation involving network operations and computer security. Later in the day they find that members of the center's computer security staff are on some of their systems. When asked what is going on, the security staff explains that these hosts have been connecting to other hosts within the center in a suspicious manner and have also been attempting to connect to such a large number of hosts at a large number of other sites during non-working hours. It appears that these hosts may have been compromised. The scientists wait some more, anxious to get something done, but are unable to do much more than read old email, since no new email has arrived since early that morning. Then the big news comes – the supercomputer used for simulations and data analysis has been shut down because it has been hacked. All scientific activity at the research center comes to a halt for the rest of the day.

Well into the night members of the computer security staff and system administrators go from system-to-system, trying to determine whether each has been compromised. They spend a disproportionate amount of time examining the supercomputer, however. They spot a two hour gap in the system logs from the previous Wednesday and assume that this computer was compromised somewhere during that time interval. From the logs are obtained the names of local hosts that have connected to the supercomputer. These are then examined to determine whether they, too, have been compromised. Shortly after midnight the security staff discovers that a tty (terminal) sniffer that captures all terminal input and output traffic through the supercomputer has been planted in the /tmp directory. They now have confirmation of their suspicion that attackers have captured the passwords of scores of machines that have been connected to the supercomputer and then have used these passwords to gain unauthorized access to other (mainly Linux) hosts. Worse yet, when the Linux hosts are examined, a variety of rootkits that hides the activity of the intruders and tty sniffers are found on the majority of the Linux systems that have been acting suspiciously.

[a] corresponding author, ESnet, Lawrence Berkeley National Laboratory, wej@es.net

[b] Lawrence Berkeley National Laboratory

[c] Univ. of Wisconsin

[d] ESnet, Lawrence Berkeley National Laboratory

Anxious to get the research center's compromised hosts back to normal mission status, system administrators work through the night to rebuild these systems and to restore user data that was on them. In some cases they find that a recent backup tape is available, but in the majority of the cases they find that the only available backup tapes are two or three months old. In the latter case, the system administrators try to save user data before they rebuild the systems. However, the question of the integrity of this data emerges. Although there is no indication that any attacker has altered any user data, this cannot be confirmed and the need for high integrity of scientific data dictates that many experiments and simulations will have to be re-run. Additionally, backup tapes cannot be found for at least a dozen systems, resulting in a complete loss of data. By the next morning some of the systems that have been rebuilt once again start showing possible signs of being compromised, but since getting the supercomputer back to normal operational status has been the highest priority, no one has time to look at the hosts that may be compromised once again. The supercomputer is back to normal by the afternoon of the fourth day, but too many Linux hosts are spewing suspicious traffic to warrant removing the blocking of all traffic at the external gateway. At the end of day four, computing at the research center is still at a virtual standstill. By the end of day five, most of the hosts appear to be functioning normally, although calls from other research centers indicate that some hosts are still succumbing to the attacks that are occurring. By the end of day six, things are back to normal, but the cost has been high. The CIO's office estimates that over 10,000 work hours have been lost due to the loss of computing capabilities, not counting the effort needed to analyze and restore systems and verify user data integrity. Many scientific projects will not be able to meet project milestones now; in some cases, ongoing projects will have to be restarted from scratch. The financial cost is estimated to be over $2 million, and the news of what has happened will be anything but favorable to the research center's ability to obtain additional research funding.

Reality

This scenario is reality. With variations, it has played out several times over the past two years as identity theft exploits have become more effective. Most spectacularly, this scenario was reality earlier this year when half a dozen US supercomputer centers were knocked off-line – some for weeks. And supercomputers are typically more carefully tended than Grid based clusters.

> *America's precious and powerful supercomputers are bound together by the "Grid/TeraGrid" which has now been proven to be extraordinarily vulnerable to intrusion. The recent hack of the Grid was most likely accomplished by a small group of young U.S. hackers ["crackers"]. [... etc.]*
> *http://www.rawstory.com/exclusives/koch/vulnerable_computer_Grid.htm*

In fact, these attacks had less to do with Grid software[e] than with the increasing connectivity in the scientific community computing infrastructure needed to support large-scale collaborations such as those required for the LHC data analysis. The Grid is just a tool in this process, and while there are current Grid vulnerabilities they were not involved in these successful attacks. However, the Grid will be successfully attacked in the future. One indicator that the Grid is on attackers' radar is a talk[f] entitled "Hacking the Grid" presented at a 'hacker[g] conference[h]' in New York City in summer 2004. The talk poses the question "Can this 'big iron' be hacked?"

[e] To be clear, "Grids" are collections of systems that use a common software approach such as the Globus Toolkit to provide a uniform distributed computing environment. Grids are not just any collection of associated systems.

[f] See, e.g., "Hacking the Grid," Greg Newby. http://www.the-fifth-hope.org/hoop/index.khtml (One should not visit hacker Web sites such as this one unless your system and browser are current with all patches, and JavaScript and ActiveX are turned, though as they go this is a fairly "tame" hacker site.)

In general, identity theft attacks are becoming more sophisticated. A Russian based attack[i] is currently (late June, 2004) in progress that simultaneously exploits three different vulnerabilities – one of which has no current fix – in Microsoft products to accomplish identity theft just by Web browsers visiting infected Web servers.

2 Implications of Identity Theft and Root Compromise

The current most difficult cybersecurity threat to deal with (and the one cited above) is identity theft: A set of real users unknowingly lose control of their "identity." Today this is equivalent to someone discovering your password and then "quietly" using your account(s) for various purposes – for example to set up distributed hacking environments called BotNets[j]. Things become much less quiet when the identity thief successfully applies a rootkit and gains root access to the system. In science clusters and supercomputers (both are environments where the same people/identities have access to many systems because of the associated large collaboration) this exploit typically spreads quickly, contaminating many systems.

Consider the consequences of the situation described in the incident scenario.

In the version of Unix currently used in the science community, having root access allows the circumvention of all access controls on all aspects of the system. Protected system directories can be written and modified; protected system files that provide all of the functions of the system, including the operating system's kernel itself can be modified in any way. All I/O devices, including all storage devices can be accessed and modified; all user data – files, data bases, archives – can be modified without any indication. (The system clock can be set to a time in the past corresponding to the last valid modification of that data, the data changed, and the clock restored to the present, making it appear that nothing has happened.) "Trojan Horses" can be installed (and usually are, in order to steal passwords, etc.) by crackers with root access. Full control of the system is allowed to remote parties as is common in BotNets, thus turning the system into an attack automaton controlled by remote crackers, or a password cracking engine, or – in principle, though not commonly at this point as far as we know – a remote execution engine for prohibited codes such as nuclear weapons design codes.

Today's firewall based security is essentially of no value in the sort of compromise described here because the attacker appears to be a legitimate user. A comprehensive approach to security must address vulnerability, attack, and compromise at all levels, and in ways that are practical for the community, and that produce a significant risk reduction. This approach must cover the entire life-cycle of the software used to manage and control the computing infrastructure of the HEP community – design, implementation, deployment, configuration and operation. And the approach must address recovery. Compromise will happen, and good recovery tools and procedures are necessary in order to mitigate the impact of compromise.

Authentication must be strengthened to reduce the success rate of identity theft. In order to be effective this must be done throughout a community that shares computing resources.

---

[g] In that paper we use the term "cracker" to denote malicious operatives.

[h] "The Fifth Hope", New York City, July 2004, http://www.the-fifth-hope.org/hoop/index.khtml

[i] See, e.g., "Web site virus attack blunted." http://news.com.com/Web+site+virus+attack+blunted--for+now/2100-7349_3-5248279.html

[j] Networks of computer systems using IRC or related capabilities for communication, command, and control typically for coordinated denial of service attacks. See "BotNets: Detection and Mitigation" at www.fedcirc.gov/library/documents/botNetsv32.doc

Tools must be put into place to detect the consequences of attacks that lead to identity theft, e.g. inappropriate uses of the system ranging from relatively harmless service theft, as in running user level IRC bots, to attempts to compromise the system by gaining root access. Once user identity theft is accomplished the security response relies on detection: Active network monitoring must be done in order to thwart the import of dangerous code such as rootkits that exploit system vulnerabilities to gain root access.

Systems must be monitored to detect the results of root compromise if the network detection fails, and root compromise is accomplished. System integrity must be constantly monitored to detect the actions of crackers as they modify the system once they have gained root access.

Significant system vulnerabilities must be patched – most consequential root exploits are based on well understood vulnerabilities that people know how to fix.

Provision must be made for recovery from (inevitable) security failure and root compromise of systems. Procedures must be established, documented, and put into place, and communities of rescue workers must be trained and equipped for system recovery.

And, finally, this whole process must be completely dynamic. It must undergo continual re-evaluation and update because it is the nature of vulnerabilities that new ones are never in short supply, and the crackers are continually developing new exploits.

An effective cybersecurity response must evaluate risk and establish procedures that will reduce the risk to an acceptable level. Risk is established by vulnerability analysis and threat analysis. Both are essential: many vulnerabilities without identified threats pose little (known) risk. On the other hand, the potential adverse consequences of some vulnerabilities are so great that they must be addressed even without established threats. Vulnerability reduction is expensive and must be balanced against the understood risks so that significant risks are reduced and that level of effort/cost is kept acceptable.

Computer security is an endless game of chess between more or less equally matched opponents in which the board continually changes and the advantage continually shifts. The goal of usable, cost effective cybersecurity for science is not permanent success – that will never happen. The goal is containment and management within the context of environments that are usable by the scientific community.

If we consider the threats illustrated in the scenario (which is fairly comprehensive) then we can build up a catalog of weaknesses and define a set of coordinated tools to reduce the risk arising from the weaknesses through various hardening approaches and various defensive approaches.

The issues that must be addressed include

- development of secure (low-vulnerability) software
- management of software weaknesses / vulnerabilities,
- identity management and secure authentication,
- host integrity management,
- weak / overly general authorization,
- data integrity,
- active detection of miscreant behavior,
- recovery,
- testbed and training environments, and
- continual risk assessment.

The issues are addressed individually below.

3 Goals

The goal of this whitepaper is to describe a starting point for defining comprehensive approach and an implementable set of tools and procedures that can systematically reduce the cybersecurity risks for the scientific environment while not disrupting scientific collaboration and the use of distributed computing and storage resources. Continual re-evaluation by independent observers is also needed in order to ensure the identification and adoption of new tools and techniques as the threat environment changes.

4 Approach, Milestones, and Cost Estimates

There is no one best approach, no “right” approach, and no foolproof approach. Effective security is like effective hygiene – it is reasonably comprehensive and is applied as consistently as possible without becoming more burdensome then the benefits warrant. And when it fails, effective recovery techniques must be available.

4.1 Development of Low-Vulnerability Software

Software life-cycle management for security includes design reviews, aggressive (even antagonistic) testing, operational monitoring, and continual improvement. This will not be uniformly applied. User applications are hard to incorporate into such a process. Operating systems and utilities are already written. However, Grid middleware is new enough, and potentially a sufficiently rich cracker target, that this approach should be applied.

Much more secure operating systems than are currently being used exist in the Linux milieu. An access control list (ACL) based authorization for everything from files to processes, which would help to significantly harden the environment, also exists. There are various projects that provide some of these protections. These include *grsecurity*, SE Linux enhancements, *stack guard*.

Technology exists to isolate code execution in tightly controlled environments that would preclude many cracker exploits – e.g. virtual machines.

However, none of this has been evaluated for use in scientific computing environments for effectiveness, usability, and deployability.

Example Milestones and Estimated Costs

- o Yr. 1: 1-2 FTE initial studies
- o Yr. 2: 2-3 FTEs pilot environment studies
- o Yr. 3: 2-4 FTEs deployment

4.2 Management of Software Weaknesses / Vulnerabilities

The US high performance computer system attacks referred to above have the appearance of an object lesson: none of the exploited vulnerabilities was newer than 6 months old. Systems where a reasonable level of patching had been done were not compromised.

Centralized administration of the diverse computing environments of science to ensure consistent, up-to-date patching is essentially impossible. However, making easily available standardized and validated system configurations that admit to a standardized way of applying patches (that are also made available in a standard and validated way) can be effective for cohesive science communities. Communities such as HEP have a commonality of goals that lead to at least some system environment similarities, thus making this practical. The definition and distribution of such standard configurations could be a service center type function.

The other aspect of this is identifying software vulnerabilities in the first place. For common applications like mail and Web clients, and for the operating system, a lot of people contribute to this work. This is not yet true for Grids software, and this must be addressed.

Technical Approach

(Miron Livny and Shane Canon)

Evaluation of the Risks of Vulnerabilities

A big problem is where to put the effort of repairing vulnerabilities. For common software literally dozens of patches a week may be available. It is usually impractical for science computing environments to deal with all of the available patches, and many are not of primary concern. A triage group consisting of security experts needs to periodically (e.g. weekly) evaluate the available patches against the current threat environment and prioritize the effort put into applying the patches. This approach has been successful at the site level.

Standardized Distributions

Once a patch is applied to all affected software components, a new distribution must be built, packaged, verified, tested, documented and released. Any delay in the deployment of the patched software extends vulnerabilities and reduces productivity. Given the complexity of the software stacks used by the HEP community and the heterogeneity of their computing environment, providing an easy to install patched release in a timely fashion requires a devoted professional team and a well managed and well equipped facility. The team will have to work closely with both the providers of the various software components and system administrators of the different sites. The facility will have to include all the OS/HW combinations that are deployed at the different sites. Effective scheduling tools and techniques must be employed to manage the build and test processes some of which may require significant resources. The Virtual Data Toolkit (VDT) team (http://www.cs.wisc.edu/vdt//) which leverages the build-and-test infrastructure of the NSF Middleware Initiative (NMI – http://www.nsf-middleware.org/) provides an example framework for this activity

Host hardening and protection

At the lowest layer of a layered approach to security is the host and at the deepest level is the kernel. Attackers are typically most interested in gaining elevated privileges on a system. This typically accomplished by exploiting bugs in privileged programs and processes. While patching is the first step in protecting against these approaches, this is only a part of the solution.

Another defense layer is to employ protections that prevent a vulnerability from being exploited or minimize the impact of a successful exploits. One example is protection methods that limit the effectiveness of stack smashing approaches. This helps prevent many buffer overflow exploits from working.

As described in the scenario, intruders typically employ root kits to cover their tracks. Often on Linux systems, these root kits attempt to hook directly into the kernel. This allows the root kit to effectively hide rogue behavior and makes detection of both the intruder and the root kit extremely difficult. One defense against this technique is to employ mechanisms inside the kernel to both help detect root kits and protect against them. One such approach is St. Michael (http://sourceforge.net/projects/stjude/) which is currently being used on the PDSF HEP cluster in production at LBNL. Future plans include extended deployment as well as enhancements. As is typical of security challenges, this module will need to be updated and maintained to insure that it can detect the evolving methods used by hackers.

Evaluation and Remedy of Grid Software Vulnerabilities

While middleware developers have been embracing security systems and standards, and developing security systems of their own, strong security requires an additional dimension: the white hat[k] evaluation. It is important that strong systems and standards are being used, but it is often in the interaction of components, the usage patterns of these components, or their interactions with the underlying operating system that security flaws appear. To seriously consider security, a software stack must go beyond design; it must undergo independent testing.

The first step would be to evaluate the security of existing middleware tools. The HEP community needs to find the holes and harden the tools that are currently deployed, and to increase confidence in them. This effort requires close coordination and collaboration with the middleware developers. The results of such an evaluation will feedback to the system designers and resulting in suggestions for improvements or changes in design and/or implementation.

The next step would be to encapsulate expertise and ad hoc experiences into systematic tools. Today, this kind of security testing is not turnkey, but as the HEP community and middleware developers work together to evaluate and test software stacks, they are bound to cobble together quick and dirty tools. The long term goal is to devote the resource required to make these tools usable by any group of middleware developers.

Software Weaknesses Example Milestones and Estimated Costs

- o Evaluation of the Risks of Vulnerabilities, 1-4 FTE/yr, on-going
- o Identification and on-going management and distribution of patched and standardly configured software for the HEP community - 3 FTE/yr, on-going
- o Evaluating and testing of system hardening techniques, followed by packaging and deployment in the community - 3 FTE/yr, on-going
- o Evaluation of Grid software vulnerabilities, and developing patches - 3 FTE/yr, on-going

4.3 Identity Management

One of the almost universal weaknesses in today's scientific computing environments (including Grids) is the use of single factor authentication – usually memorized user passwords. Theft of these passwords is easy and getting easier. Two-factor authentication – frequently something you know and something that you have – makes identity theft much more difficult. One-time passwords are a variant of two-factor authentication – you know your user name and you have a non-repeating token for a password, e.g. a list of one-time use passwords or a challenge-response crypto card (a hardware token), etc.

[k] "White hat describes a hacker (or, if you prefer, cracker) who identifies a security weakness in a computer system or network but, instead of taking malicious advantage of it, exposes the weakness in a way that will allow the system's owners to fix the breach before it is can be taken advantage by others (such as black hat hackers.)" (http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci550882,00.html)

Technical Approach

(Mike Helm)

Authentication needs to be re-engineered to provide a much more secure and robust environment. Four things have to happen: hardware tokens must be deployed; Grid middleware must be adapted to support new forms of authentication; a robust, secure authentication "fabric" must be created; federation of site authentication mechanisms must take place in the authentication fabric. One approach to this is to build a federated infrastructure based on the widely used RADIUS authentication service. Grid middleware developers must provide authentication capability beyond the form of X.509 certificate support available today; this capability will include RADIUS client support, either directly, or indirectly through standard Pluggable Authentication Modules (PAM). Trusted service organizations such as ESnet and iGOC can provide a secure RADIUS authentication infrastructure that can be deployed securely and in reliable (non-interruptible) configurations. Sites will deploy and manage the hardware token solution of their choice and will use the RADIUS fabric to provide common (federated) authentication.

This allows all software that uses the standard RADIUS approach (e.g. via PAM) to forward authentication requests to the federated RADIUS infrastructure and obtain user authentication based on site issued tokens. NCSA's Gridlogon is one such piece of software.

Example Milestones and Estimated Costs

- Year 1 – Prototype RADIUS fabric; prototype Gridlogon deployment
  - 4 FTE + $250 K
- Year 2 – Federation mechanisms, agreements, and management
  - 4 FTE + $250 K; OTP deployment costs unknown
- Year 3 – Production deployment with geographically distributed, replicated servers
  - 4 FTE
  - 1 FTE for ongoing federation and engineering support

4.4 Host Intregity Management

The lowest layer of layered approach to security is the host environment. While identity management and authorization attempt to keep unauthorized users off Grid resources, the host should be configured and secured in a way that limits the capability of an attacker and allows quick detection and recovery from an attack. In some cases, there are existing technologies that provide some of these capabilities. In other cases, effort is needed to enhance the existing solutions to meet science computing requirements.

Technical Approach

(Shane Canon)

Even with the most diligent effort, systems will be compromised. Consequently, tools to help minimize the impact of a system compromise are needed. These tools would include detection methods, root kit prevention, and detailed logging.

One familiar approach to host based intrusion detection are systems like TripWire that monitors and notifies about system changes. One weakness with many of these packages is few have been designed to run on large clusters. A framework for using these techniques on clusters must be developed.

Once a system has been compromised, understanding the scope of the attack allows administrators to focus effort where it is needed and quickly notify other affected sites. This not only helps minimize down time for the compromised system, but can help contain an attack.

Most operating systems offer some level of auditing, however what is offered is typically limited and is aimed at addressing issues other than security. This is true of the standard Linux kernels distributed with most Linux distributions. However, there are projects underway to address this and related issues (i.e.: syslog-ng). Approaches need to be assessed in order to select the most promising approach (or approaches) and effort put into these projects such that they meet our requirements. The auditing should capture as much detail as feasible. In addition to normal process information this would include information such as Grid identity and role information when possible. This information should be exported off the host to a central location to prevent attackers from removing the record of their activities.

The tasks to be performed for improving host integrity are:

- Enhancement of distributed "Trip-Wire" based approach.
- Testing and enhancement of root kit protection tools for Linux
- Evaluation enhancement of auditing tools for Linux
- Evaluation of Linux kernel enhancements for security
- Deployment of tools on test bed systems

Example Milestones and Estimated Costs

- We estimate this work will require 2 FTEs for the first two years.
  A single FTE in the 3rd and 4th year will be needed for on-going maintenance and support.

4.5 Authorization

Because of the collaborative nature of experimental physics, scientists must to share data files, programs, and computing work. Large experiments have many jobs and tasks to be performed and scientists in the collaboration take on various roles over time to contribute to accomplishing this work. The driving goal of the scientists is to get the job done, and if the authorization system prevents this or makes it cumbersome, scientists will find ways to circumvent the security systems by allowing too broad of read or write access, or by sharing credentials so that a particular job or role can be carried out using a single identity.

Authorization mechanisms that are finer-grained than currently available are being defined[l] and developed[m] to better match the requirements for security as well as end-user functionality. Today there is not a single, broadly accepted development project for authorization, but as time progresses there will be choices for deployment of software that enforces authorization decisions and other privilege adjustments based upon policy. It is important that these authorization systems are integrated into a comprehensive auditing and testing program. The example used

[l] https://forge. Gridforum.org/projects/authz-wg

[m] http://zuni.cs.vt.edu/publications/PRIMA-2003.pdf

below is to show how such integration should happen while realizing that the choices of what software to include will be an ongoing process in a real security program.

Technical Approach

(Doug Olson and Iwona Sakrejda)

The US Physics Grid Projects (PPDG, GriPhyN, iVDGL) and the participating experiments have deployed a persistent Grid starting in 2003, initially called Grid3.[n] These stakeholders have joined/formed the Open Science Grid Consortium in 2004 and the persistent Grid is evolving from Grid3 and in the future is called OSG. The fine-grained authorization and privilege systems mentioned above will be deployed and used on OSG, and for the purpose of this pre-proposal we will call that authorization and privilege system OSG-authZ. The work described here is about deploying the OSG-authZ system in testbed environment and integrating it with the other auditing, logging and diagnostic services.

On the testbed OSG-authZ will consist of a Linux server running the VO membership group and role attribute services (authorization policy), and the OSG-authZ policy enforcement services running on the Grid gatekeeper nodes. The tasks to be performed for each new version of the OSG-authZ software are:

1. Deploy OSG-authZ software on testbed (VO server and gatekeepers)
2. Populate databases with sample membership, attributes and policies
3. Integrate OSG-authZ software with general logging and auditing tools
4. Design and run challenge tests of the OSG-authZ service
5. Communicate status and results with OSG consortium and OSG-authZ developers

The milestones are to perform each of the 5 tasks following a release of the OSG-authZ software, and we anticipate a new version of the OSG-authZ software annually.

Example Milestones and Estimated Costs

- o Integrate OSG-authZ V1, integrate with logging/auditing, develop and run tests
  - – 1 FTE/yr, on-going + $3K for VO server

4.6 Data Integrity

In an experimental science environment data is of paramount importance. Data integrity must be carefully managed and monitored.

In scenarios where attackers may have malicious intent toward the science being conducted (either to gain some advantage, or more likely, just vandalism) monitoring data integrity is very important

Such monitoring is not supported in current operating systems and tools must perform such tasks as computing cryptographic hashes (strong checksums) of data and securely archive them

Techniques exist to do this, and are use, e.g., in the medical digital imaging environment. However, the scientific community has no experience deploying and using these tools.

Testing and evaluation is essential, especially as the consequences of false positives (incorrectly reporting corrupted data) are just as consequential as deliberate corruption.

[n] www.ivdgl.org/ Grid3

Example Milestones and Estimated Costs

- o Yr. 1: 1 FTE initial studies
- o Yr. 2: 2-3 FTEs pilot environment studies
- o Yr. 3: 2-4 FTEs deployment

4.7 Active Detection of Miscreant Behavior

It must be assumed that identity theft will, or already has, occurred on any given system. Therefore a continual monitoring for dangerous activity is essential. One example of this is detecting rootkits being transported onto science systems. Detection can be used to automatically and immediately terminate the transfer and/or isolate the target system from the network.

Technical Approach

(Brian Tierney)

Bro[o] is a stand-alone system for detecting network intruders in real-time by passively monitoring a network link over which the intruder's traffic transits. Bro targets high-speed (Gbps), high-volume intrusion detection. By judiciously leveraging packet filtering techniques, Bro is able to achieve the performance necessary to do so while running on commercially available PC hardware, and thus can serve as a cost effective means of monitoring a site's #1 source of attacks: its Internet connection.

Bro's use of a specialized policy language allows a site to tailor its operation, both as its policies evolve and as new attacks are discovered. For example, Bro can be used to detect hosts that are attempting to break in to a site, and send a message to the router to block that host.

Bro is currently being used to provide site-level intrusion detection, however there are a number of important additions needed to make Bro a more effective tool to protect Grid-based systems.

Example Milestones and Estimated Costs

- o Develop Bro analysis scripts and event engines for HEP Grid applications, including monitoring GSI, Radius, and one-time password systems: This will allow Bro to capture primary HEP Grid software handshakes, and flag any anomalous behavior. As Grid software continues to evolve, these scripts will also need to evolve. (1 FTE/year, ongoing)
- o Integrate host-based Grid security elements with Bro's network analysis, using the Bro client library: This will allow construction of Grid middleware that can coordinate with Bro, such as telling Bro when it is going to start a bulk data transfer and on what port, or learning from Bro that a given subnet is being block because of a virus. (1 FTE/year, ongoing)
- o Design and develop a version of Bro that will work at speeds beyond 10 Gbit/second Ethernet. This will require multiple Bro systems working in parallel and exchanging information with each other, each monitoring different types of data. This will also require working closely with router venders to incorporate data filtering in the router to send a portion of the traffic to each Bro system. (1 FTE/year, ongoing)
- o Design and develop "Grid Honey-nets" which will attract Grid hackers and can be used to generate Grid attack signatures. (1 FTE/year, ongoing)

[o] "Bro: A System for Detecting Network Intruders in Real-Time." http://www.icir.org/vern/bro-info.html

4.8 Recovery

Planning and procedure, together with a few useful services, are at the heart of intrusion recovery.

An important service that is needed is a way for system admins and security staff to confidentially communicate with other affected sites. "Confidentially" may mean privately with a trusted but very dynamic community of allies. And it may mean ephemeral. It may (and frequently is) the case that these communities of allies do not want their communications preserved for subsequent analysis or delivery to the local news media (or even to their own management). An effective security incident communication system must accomplish all of these.

Technical Approach

(Mike Helm)

Security officers dealing with an on-going computer security incident rely on an informal network of personal contacts and ad hoc tools to exchange information during the crisis period. Information must be exchanged quickly and securely, reaching only the right parties, and good judgment is required to make use of it.

A suite of information and communication services tuned to the needs of security officers and their partners, responding to an on-going incident is needed. Familiar services such as email, "instant messaging", and a weblog should be provided with additional security and authentication technology from trusted services such as ESnet's DOEGrids CA and One-Time Password initiatives, to support token-based authentication. Within the community (e.g. HEP) a standing committee should be formed to manage incidents and organize a response team; the response team will manage its access and membership rules in the provided services. Video, audio, and data conferencing services need to be part of this. "First responders" have often been unsure of the integrity of their home environments; conferencing services unaffected by this threat would improve the effectiveness of the response team and raise the level of trust each team member has for the rest of the team. Further, these teams must be able to be rapidly and automatically created to respond to incidents. Tools for direct integration of detection information into the responder's communication services may be needed.

Incidents span organizational, agency, and international boundaries, and similar efforts are known to be underway elsewhere (but with varying emphases and levels of maturity). In particular efforts at iVDGL, iGOC, and in Europe must be taken into account. For example, the DOEGrids service at ESnet has good liaison and coordinating relationships with many of these parties, and has been sponsoring an "Incident Response Committee". This sort of work needs to be extended to help coordinate the creation of an appropriate inter-agency board to facilitate communication between interested parties.

Incident response is another "arms race", and it is reasonable to expect the needs of responders and the community of interest to shift over time.

Example Milestones and Estimated Costs

- Year 1: Deployment of pilot service; inter-agency committee

3 FTE + $100k redundant servers, software licenses, training

- Year 2: Video/audio conferencing integration

3 FTE + $50k conferencing hardware and licenses, training2

- o 3Year 3: Second-generation: Bro integration

3 FTE + unknown

4.9 Testbed Environments

Usability is of paramount importance effective security, otherwise it will be circumvented. For system admins this means deployability and maintainability. For users this means non-interfering. The only way to get this right in a given environment is to do it and test it at scale. A realistic testbed environment is essential. The testbed must extend across multiple sites and include the different administrative domains and LAN topologies (firewalls, etc.) that are realistic in the HEP Grid community. This amounts to having 2 or 3 nodes at each of several sites as part of the testbed setup. These sites should be the HEP labs (FNAL, BNL, SLAC) and having them participating in the testbed will form a natural basis for them to collaborate with the testing and deployment of the technical solutions of this project. . Having a university site in the test bed will provide an important, different dimension to the testing activity. A natural candidate for such a site is the University of Wisconsin-Madison.

Technical Approach

(Shane Canon, Doug Olson)

One testbed for this work could be PDSF at LBNL. Since PDSF is somewhat unique in the breadth of HEP groups it services, it provides an ideal resource to explore the impact of these approaches on a variety of analysis models and frameworks. Furthermore, PDSF has demonstrated the ability to provide secure, yet effective computing for HEP experiments. As a result, PDSF provides valuable feedback on the feasibility and effectiveness of various approaches to security and their impact on HEP users.

In order for the testbed environment to support realistic Grid “security challenges” it should include a university site (for example Wisconsin) as well as the Grid or VO wide services not specific to a particular site.

The testbed environment also serves as the primary means of establishing and maintaining collaboration with DOE and university sites on security plans and developments so there should be a phased expansion of the testbed from the minimal lab+university+VO servers in the first year to include more DOE labs and universities. The schedule below reflects this expansion.

Example Milestones and Estimated Costs

- o Year1 – LBNL/PDSF (DOE lab cluster) + U. Wisconsin (univ. cluster) + VO servers (ESNet)
  - 2 FTE DOE lab (testbed support + running and diagnosing test results)
  - 1 FTE university (testbed support)
  - 1 FTE – VO servers (e.g. ESnet operating identity management and attribute servers)
  - $25K equipment - additional servers at 3 sites.
- o Year2 – Add FNAL and BNL to testbed environment
  - 4 FTE from year1 continues
  - 2 FTE (1 each for FNAL, BNL for testbed)
  - $25K equipment – additional servers at 2 new sites + additional VO servers

- Year3 – Add SLAC, JLab, and more university sites (Caltech, UFL, MIT, IU, UofC, UTA, OSU)
  - 1 FTE at each site (9, 10, …?)
  - $10K equipment at each site for new servers

4.10 Continual Risk Assessment

A key to effective security is to address all known issues and the fact that the vulnerabilities and the threats are constantly changing. No single group can effectively address this – they get too focused on the issues at hand. External advisory groups – including "friendly" crackers – are essential.

Example Milestones and Estimated Costs

- Identify experts who can serve an external advisory role
- Manage and support periodic meetings and communicate the results to the science cybersecurity community
- Effort – 1 FTE on-going, meeting support, and travel support

5 Conclusions

This white paper is an initial step in identifying the types of security that must be deployed and used in the scientific computing environment in order to prevent large-scale disruption in the future. The contributors represent computer scientists, physicists, and computer and network operations engineers. However, what should probably be done is to convene a workshop representing a broad, but select, cross-section of all of the involved communities. This workshop would take an approach similar to this white paper, but with a boarder scope, and produce a roadmap for getting to a "reasonably" secure scientific computing environment. (E.g. see "DOE Science Networking Roadmap Meeting, June 2003" at http://www.es.net/#research .)

Based on the gross estimates of effort in this paper, and using the assumption of a mix of the three types of contributors noted above, the order-of-magnitude effort is 63 FTE over three years, plus $750K in equipment and deployment costs, for a total of $13.5M.

The value of this work, is of course, not limited to any one science community, even though it's most immediate applicability is for HEP.