Digital Flight Plans for Server Access Control:

Restricting anomalous activity with path-based declarations of intentions

Ronald Loui
Dept. of Computer Science
University of Illinois Springfield
r.p.loui@gmail.com

Lucinda Caughey
Dept. of Computer Science
University of Illinois Springfield
lcaug2@uis.edu

***Abstract* — In response to increasing threats of malicious activity and data loss on servers, we propose a different and practical strategy for access control modeled after flight plans for pilots, which mixes existing role-based, object-based, and intention-based access models; it supports much finer grained, real-time, sequence-oriented anomaly detection. Users are required to declare their intended "flight path" in advance, a sketch of resource use: this may vary in detail, but could include database tables, file system directories, byte and bandwidth limits, use of encryption and archive creation, command sets, connection time, number and origin of connections, and ports. Sequence information provides especially strong constraint, even if it incomplete. We find an important place for active, on-line human sampling of flight plans, as well as pre-authorization for non-standard paths, and alerts for deviation from path. We also find a place for improved user profiling and a paradigm shift from ex-post log-based reconstruction of user activity to ex-ante declaration.**

***Keywords* — server access, access control, data loss prevention, intention recognition, privileges, computer security, specification, route, waypoints, flight plan, sequential constraint, anomaly detection**

I. Introduction

Computer access control has in the past been conceived in terms of authenticating the user at the perimeter, so that the user may claim rightful capabilities and privileges. But capability and privilege to do what?

Updated access models continue to be proposed, especially as it becomes clear that the insider threat and massive data breaches are becoming acute concerns in computer security and cyberwarfare.

We usually speak in terms of the subject's right to access an object, or the determination that a user has a role or belongs to a group, which confers privileges. Sometimes there is finer grained rule-based access control where a situation and conditions permit a range of activity. In the past, there has been little attempt to model what the user will do with the permissions. Machine learning and datamining have been aimed at modeling anomaly and normalcy rather than intention and deviation, and there are important differences. A recent paper [2] has suggested intention-based access control, according to which the user's intent is divined, and control applied based on this inferred intent or mal-intent. In similar spirit, we take a larger step and require that the users declare their intentions. This is a shift of paradigm because it requires a language in which to express what they plan to do, and it places an added burden of specification on the user prior to connection and execution. However, the security benefits are plentiful, and a dramatic shift may be warranted in an era of heightened concern over data loss and a greater commitment to layered intrusion detection/mitigation.

Our model is based on the flight plan that a pilot is required to file before taking off and landing within a well regulated (national) air space. Flight plans are useful not only for the orderly allocation of scarce physical resources, and for security against malicious behavior, but also for the safety of flights and real-time diagnosis of anomaly. Airspace has always been considered important enough to regulate in this way, if only because there are a limited number of places to take off and land, and because it can be hard to find a plane that has had an accident. Computer servers and databases, likewise, usually contain data and provide infrastructure that are too important to regulate like an automobile highway or pedestrian plaza. Servers deserve more regulation. Enough user activity has been moved to the client side that it may make sense now to increase the burden significantly on those who require access to the servers.

II. What does a flight plan look like?

In the aviation world, a flight plan usually specifies a few things:

A) departure and arrival points

B) estimated flight time en route

C) alternate/backup airports for emergencies

D) pilot data and passenger count

E) instrumentation

F) route/sequence

978-1-5090-1072-1/16/$31.00 ©2016 IEEE

i. pre-defined path

ii. waypoints/checkpoints

iii. bearing-directed or GPS-directed pt-to-pt

In addition, there are notions of paint markings, flight min and max altitude level, airspeed, holding pattern, landing approach, communication modes, horizontal and vertical separation, restricted (special use or military) airspaces, and intercept protocol.

Some flight plans require authorization in advance (e.g., takeoff clearance, or foreign airspace approval), some require voice authorization en route to enter a restricted airspace segment (e.g., approaching a temporarily restricted flight area, TFR, or simply approaching D.C., a national monument or nuclear power station), and some (designated "class G" airspace, low altitude unrestricted space) require no prior clearance at all and would not gain anyone's attention unless there were an investigation.

Some segments of authorization, taken by themselves as the right to "fly to the moon and back" at a particular time and in a particular location, can look like role-based or access-list control. However, a crucial idea in a flight plan is that a route will be followed in sequence, with a declared intention (usually a destination), without unnecessary repetition and with a fairly limited time en route (usually enforced by a fuel limit). It is true that a pilot could circle in some area once clearance has been given for no reason whatsoever (one author has circled the St. Louis Arch in a private plane just for the view), but this could invite interrogation, if only to check that the pilot is conscious and has sufficient fuel to complete the flight plan.

Another distinctive idea is that there is an Air Traffic Control (ATC) or many towers (ATCTs) that may inquire, authorize, restrict, command, or monitor some flights at some arbitrary times, and all flights at some mandatory critical moments (such an ATC did inquire about flying near the Arch).

A specific route declaration might look like "KSPI to KPDX, 454 knots, 30,000 ft, 03:47:00, UIN KP63E KD69Y KD75U KU81Q DBS PDT J16 KOTAA JKNOX" or "SYR RODAN V84 BEEPS V501 V119 BFDV72 ROD I77," or simply, "VFR DIRECT STL to ORD," which pertains to locations and flight guidance method (though not speed and altitude over each individual segments).

We aim to recreate these features for "digital flight plans" when users connect to servers.

The simplest analogy would be to specify a ceiling on bytes transferred, written, or changed, much like an altitude ceiling is given. Even more direct analogy would ask for the connect time, like a flight time, or the port numbers and IP addresses, like airfield names; services would be analogous to gates. But the deeper analogy gives the sequence of significant data locations and operations, like a sequence of waypoints.

III. What would a Digital Flight Plan look like?

A digital flight plan might differ for a database access, a ssh session, a file upload or download, routine data feed or mailserver dump. While each may have different details, they all share a few declarations such as connection information and data footprint expressed in size and character class characteristics. Each additionally makes typical progress in its own way. Three examples are given: ssh, sftp, and mysql "flight" details.

A. SSH

The ssh user is possibly the most general user because the path taken can be highly contingent, exploratory, interactive, and generally unplanned. However, users still have intentions, which they can state in advance:

1. number of connections from IP address (to prevent piggybacking or spoofing by third parties)
2. connection time (usually one can estimate hours, if not minutes)
3. byte limit in and out (usually one can estimate whether the exchange will be 10s, 100s, or 1k MB or more)
4. percentage of binary characters in and out (usually there would be a low percentage of non-printing characters)
5. file system areas to access, especially system or shared user areas
6. number of files/directories to create/revise/destroy
7. special commands that will be used (especially those with reach or volume)

The hard part is to get the user to specify something that is sequential. It is the serial aspect of a path that distinguishes a digital flight path from a fine-grained set of permissions or session limits. Probably 5, 6, and 7 could be given as waypoints for specific sessions, even if non-exhaustive:

a) "I intend to cd to cgi-bin, create a backup directory, then cp -r a directory, creating less than 100MB and no more than 5000 files."

b) "I intend to look at contents of files in public_html, do some wgets creating no more than 10 files, edit some files in public_html, then clean up my temp files."

c) "I intend to develop a python program in my ~/bin, with an edit/run debug cycle."

d) "I intend to install a new version of apache using apt-get, and configure it, then test it in a user directory."

We later consider ways to make these declarations more formal, while maintaining discretionary allowance.

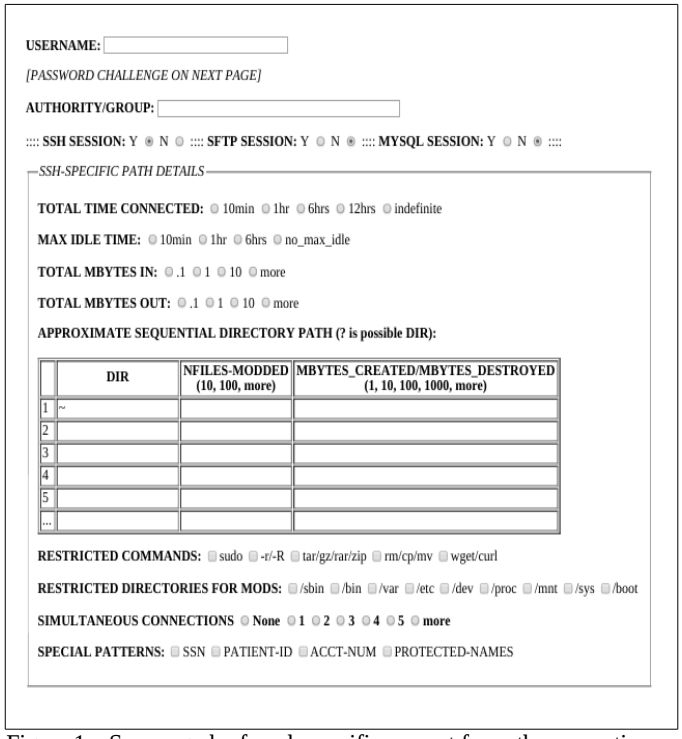

Figure 1. Screen grab of a ssh-specific request for path, connection details, and special-use commands, directories, and patterns of data. Not all fields are required.

Each of these session plans implies a sequential use of files in specific locations and a sequence of salient commands. A user may deviate in a minor way, or the formal specification may be so vague as to permit much freedom interpolating waypoints, but there is a general description that can be matched to each intention. For example, in (a), the directory sequence might be ~, *cgi-bin*, *cgi-bin/backup*, and the major motion and destructive command sequence might be *cd*, *mkdir*, *cp -r*. There may be considerable leniency for the user to interpolate those commands with *ls*, *find*, *less*, *grep*, *cat*, even *cd public_html*, but any considerable time spent in an undeclared directory, or any command used that creates, destroys, or modifies a significant undeclared number of bytes would trigger an alarm as deviation from plan.

For data loss prevention, one would particularly want to know about a user's plans to do anything recursively or any descending through file hierarchies, to do any monitoring of network traffic, or to do any accessing of known target data files. For malware mitigation, one would want to know what executables will be run, and what libraries and system files or configuration files might be altered.

These considerations are familiar to existing layered defensive tacticians who have automated intrusion detection, but the novelty is significant: instead of discerning user activity as an intention recognition problem, the user must declare in advance an intention that matches the manifest activity and must remain consistent with the partial, skeletal, or sketched specification. In this way, the decision is not binary, whether the activity is normal/abnormal, or admissible/inadmissible. Rather, the interaction takes the form of contractual obligation, with varied response, from "how may we help you with your change of plans?" to "what exactly are you doing now?"

B. SFTP

In contrast with an interactive shell, file upload and download present the clearest examples of a simple digital flight plan. The user almost always knows in advance exactly what files will be delivered or taken, sometimes a single file to be transferred, generally how large the files are, and where they are located for download or will be located after upload. Sometimes there is interaction, such as when merging and overwriting, but location, number, size, and type are the major concerns. Users should state in advance:

1. number of connections from IP address
2. connection time (bandwidth-dependent estimate)
3. byte limit in and out
4. percentage of binary characters in and out or actual file/mime type
5. file system areas to access, especially system or shared user areas
6. number of files/directories to create/revise/destroy
7. special restricted signatures that may appear in data that would set off alarms or firewall restrictions

The sequential aspect appears when there are multiple files in multiple places. A user may in an unregulated sftp interaction have chosen an arbitrary sequence of independent gets and puts, in different places. Under a digital sftp flight plan, the user should specify the order of those directories visited, and the sizes and types related to each place, e.g., "I plan to put about four ~1MB jpegs in the public_html/images directory, then two css files and three html files in public_html."

Perhaps sequence is not as important as extent (depth, cover, reach), especially across sessions.

C. MYSQL

In a database session, the language shifts to the tables and the size of the results, or impact of the insertions and other destructive revisions. Once again, users should state in advance:

1. number of connections from IP address
2. connection time
3. byte limit in and out
4. percentage of binary characters in and out
5. tables to be accessed

6. columns and attributes to be used in select statements; reported fields
7. order of magnitude estimate of number of records or rows to create/revise/destroy in persistent/non-temporary tables
8. special restricted functions of scripting to be used
9. special restricted tables or records that may be touched

The sequential aspect should be strong and easily characterized for a database user. Certainly many sessions will have an exploratory aspect prior to data gathering or data insertion, but the main impact of the session should be known in advance: "I have ~1000 new records to insert in the patient history table," or "I want to run a report on outcomes selected and sorted by year, generating ~1M results," or "I have ~8 columns to create in the existing billing table."

In some ways, multiple independent major acts would be better done in short, distinctive, more focused sessions. Much of the sequencing of a session results from a user wanting to do a few independent things, where there is no sequential dependence or constraint. Rather than force the user to declare an arbitrary order of independent tasks, everyone might be happier just asking the sessions to be shorter, and asking the user to file multiple digital flight plans, with each plan potentially very simple.

In fact, cogency and coherence of user activity is the main advantage of declared sequential detail. Intentions are more easily recognized, and deception is more difficult.

IV. Dynamic Port Assignment + Window whitelisting

One impetus for advance declaration of user intent is a related system and method of server management we are reporting currently reporting elsewhere [10]. The idea is to make the search space for port scanning with IP-spoofing much larger, and less dense, by using port numbers in a non-standard way. This includes rewriting HTML page URLs so each stage in a dialogue appears on its own novel port. It also includes shifting ssh, sftp, and mysql traffic, among other services, to different ports (virtually, in the easiest implementation), on an hourly or daily basis. The added, significant, advantage of segregating traffic in a finer grained manner is that byte-frequency restrictions, signature-based restrictions, and pattern-based (e.g., regexp) restrictions (e.g., for intelligent deep packet firewalls) are more specific and have more bite; they can be considerably more constraining when traffic is disaggregated and thus differentiated.

Implementation of this dynamic port assignment and windowed whitelisting strategy requires a registration stage, where users engage in a standard (perhaps layered) authentication protocol, exchanging IP address and credentials for the temporary whitelisting on an assigned port (whether opening a new port for a service, or revealing where a service currently is being provided at the nonstandard temporary port, or in some strategies, perhaps even a different host IP address). During this exchange, we realized that the user could be asked to provide estimates of connection time, data footprint, data types, and even patterns of i/o. This was the genesis of the current idea to require an entire digital flight plan, expressed as a user-declared, specific sequence of activity, for each session.

V. Planning Utilization of Resources

Another advantage of users declaring their session details in advance is that resource utilization can be planned. In aviation, this may mean avoiding congestion in airways, on runways, and at terminals. It may even mean clearing paths and pre-staging response teams for precarious or high priority flights, such as Air Force One and other military air movements. The analogy on computing servers is the temporary granting of permissions, often manually. A digital flight plan might be so disruptive that it might trigger a preemptive backup prior to "takeoff."

Obviously, with some notice, planning and scheduling algorithms can be used to determine potential bottlenecks, predict cpu, memory, and secondary storage shortfalls, or anticipate network traffic. There are some security pitfalls here too, as the attack on Pearl Harbor was famously mistaken as an incoming flight of bombers being repositioned to local airbases; hackers can still hide their activity with some luck, or if they can intercept digital flight plans.

VI. Profiling Users without Forensic Log Analysis

Another advantage of asking users to file digital flight plans is that a profile of each user emerges naturally. Hackers currently take data in many small parcels rather than using exhaustive single shipment. Much of this activity can be hidden by using a number of user ids, so the activity is not exposed by a high frequency of any one user connecting. However, if user activity is easily analyzed, for example, if a cadre or equivalence class of users seems to be taking data according to a recognizable pattern, the hack can be exposed. Of course, the adversary will respond, possibly by filing variations of flight plans, but pattern analysis can be powerful when the right data are made available. User profiles can be reconstructed from logs ex post, but usually only during an investigation, and usually non exhaustively for all users. Whenever a digital flight plan is filed, it presumably would be put into a database and would expose the features of activity at exactly the level of abstraction that supports pattern discovery.

VII. An Active Digital Traffic Controller

The most controversial part of this proposal is the designation of an online, active, human counterpart of the aviation world's air traffic controller, making real-time decisions, monitoring activity, and applying control manually. The immediate question is whether security is worth the allocation of such a costly resource, and each enterprise organization will have to make that decision.

A few points: first, the traffic controller need not share the same skills as the sysadmin superuser. Monitoring, directing, and granting privileges is a distinct set of requirements from the generation of precise data processing control at the shell or

database command prompt. The concepts are shared, but not the technical proficiencies.

Second, an active digital traffic controller could be shared across multiple servers, in the same way that an air traffic controller's responsibility covers many airlines, airways, and classes of air traffic in a region. Like many security guarantees, the oversight is probabilistic: not all sessions or flight plans will receive the same amount of scrutiny. In fact, one might be satisfied with a small sampling of sessions, and a large fraction of alerts, earning human attention. In some sense, sampling makes the digital traffic controller more like a telephone operator than an air traffic controller.

Finally, it is often assumed that computing almost always permits the generation of traffic so voluminous that it is beyond human controllability. But this is a misperception in the case of connections to a server for large file operations or nontrivial administrative acts. Except for automated data transactions, our experience as sysadmins is that major operations occur only a few times a day, if that frequently. The data itself may be beyond human comprehension, in the same way that the tonnage of air transport or even the passenger manifests may be daunting, but the actual events that deserve human attention are few.

A related question is whether the insider threat is exacerbated by adding a controller who can grant privileges and direct other sysadmins. Some may consider this "control tower" a vulnerability. Since the controller presumably may not actually be in a position to fly any planes, or interrogate the contents of user files, the relationship may be more like a paired-sysadmin, or a sysadmin-watcher. In most air traffic control situations, multiple controllers are able to provide some redundancy and mutual correctness checking.

VIII. Automatic Detection of Deviation from Declared Path and Clearance For Escalation By Order of Magnitude

We presuppose effective automatic detection of deviation. The language of path specification should be precise enough that deviations can trigger alerts. The question here is how large a deviation is required to require new clearance?

File and directory location deviation may be hard to control, but like the situation in aviation, any incursion into protected air space should automatically trigger a request for new clearance. Control of executables may be even more subtle, though control of process can often be exercised through control of data. Long-lived process time, high utilization, and protected library use can all be analogized to protected air space.

But numerical estimates about file counts, row and column counts, and byte-footprints should be susceptible to automatic alert generation as well. We propose an order-of-magnitude test as a default strategy for automatically triggering a requirement for new human clearance, or at least a newly detailed digital flight plan. For one who estimates creating 10 new files, 100 files are the hard limit; for an estimated download of 100MB, 1GB is a hard limit. Since estimates are probably already generous upper bounds with plenty of headroom, it may make more sense to implement powers of two: for someone who says two tables will be altered, altering four tables requires new explanation.

IX. A Language of Locations and Sizes

Formally, we might consider that a digital flight plan is a sequence s of triples, of moderate length n: $s = \{ <location_i, t/f\ touched_i, t/f\ altered_i, t/f\ removed_i, t/f\ created_i, bytes\ in_i, bytes\ out_i, main\text{-}command\text{-}form_i \mid i=1, \ldots, n \}$ and a summary of connection information, <*ip-connection-addresses*, *duration*, *network-bytes-in*, *network-bytes-out*, *charset-info*, *firewall-special-requests*>, where each *t/f* is a table or file depending on connection. Perhaps row limits can be inferred from table names, and row sizing, or one may require additional detail for highly regulated databases. For ssh connections, *f* might be a directory or even the parent node of the directories to be used (e.g., ~, but not /). *IP-connection-addresses* is an unordered set, possibly permitting prefixes and wildcards. Special-requests might be minimized to a set of pre-defined or customizable identifiers, such as *will-sudo*, or *will-reboot*. These sequences and summaries are filed in advance of connection and may in fact be used to whitelist ip addresses. They are logged and analyzed for any manual, non-automatic clearance needs. Like FAA flight plans, digital flight plans should probably contain specific contact information revealing aspects of physical identity.

X. Conclusion

In the "old days" a well-secured infrastructure resource could be accessed only by one, or a few systems administrators. Someone who had good reason to log on to a server or connect to a database, as a temporary administrator, would ask for verbal permission, possibly receiving a temporary password in exchange for an explanation or description of intent. While there may not have been a language for specifying details of the session that could support automatic deviation detection, oversight might have included checking periodically that the session consisted of the expected activity, no more and no less. We argue for a formalization of this informal arrangement.

In this space we have only sketched a language for declaring users' intentions in digital flight plans and there should probably be organic development through community discussion rather than an immediate attempt at specification standards. However, it should be clear that such development of specifications is possible, especially for very specific tasks, and desirable. The language that is actually used by pilots and aviation authorities is amazingly simple, though it refers to the important technical aspects that are of interest. So too will be a successful language of users of computer servers.

The server evolved from the specialist JCL operator on the mainframe, through the general purpose desktop computer for amateur users, then back to the special purpose computing service infrastructure, now often virtualized and removed to third party clouds. Had there never been the middle step, in the hands of amateurs, the professionalism of the systems

administrator and careful restriction of activity on the server would have evolved more fully by now. Other professional pilots, whether in air, on land, or at sea, are required to specify what they plan to do before they do it. Other areas of human activity, even voluminous activity, are considered important enough to be monitored, at least in part, by dedicated professionals in real-time. Given the immediate challenges presented by data loss, insider threats, and misappropriation, it may be time to adopt a regimen that has been successful for decades in the management of other critical infrastructure.

XI. ACKNOWLEDGEMENTS

We thank Rogelio Salvador, Mohammad Ghasemisharif, Kranthi Polimetla, Karan Kanwar, Chaitanya Kusurkar, Prem Sagar Bhamidipati, and Ehtesham Syed for their useful input.

REFERENCES

[1] M. Abadi and C. Fournet, "Access control based on execution history," *NDSS 3*, 2003, pp. 107-121.

[2] A. Almehmadi and K. El-Khatib, "On the possibility of insider threat prevention using intent-based access control (IBAC)," *IEEE Systems Journal 99*, 2015, pp. 1-12.

[3] E. Biermann, E. Cloete, and L. M. Venter, "A comaparison of intrusion detection systems," *Computers and Security 20*:8, 2001, pp. 676-683.

[4] M. Bondada and S. M. S. Bhanu, "Analyzing user behavior using keystroke dynamics to protect cloud from malicious insiders," *IEEE Intl. Conf. on Cloud Computing in Emerging Markets*, 2014, pp. 1-8.

[5] Pau-Chen Cheng, Pankaj Rohatgi, Claudia Keser, and Josyula R. Rao, "Risk adaptive information flow based access control," US Patent 8650623, 2014.

[6] Jason Crampton and Michael Huth, "Towards an access-control framework for countering insider threats," *Advances in Information Security: Insider Threats in Cyber Security, 49*, 2010, pp. 173-195.

[7] John D'Arcy and Anat Hovav, "Deterring internal information systems misuse," *CACM 50*:10, 2007, pp. 113-117.

[8] T. Jaeger, "Challenges in making access control sensitive to the right contexts," *Proc. 20th ACM Symp. on Access Control Models and Technologies*, 2015, p. 111.

[9] James Joshi, E. Bertino, U. Latif, and A. Ghafoor, "A generalized temporal role-based access control model," *IEEE Transactions on Knowledge and Data Engineering 17*:1, 2005, pp. 4-23.

[10] Loui, Ronald, Lucinda Caughey, Mohammad Ghasemisharif, and Rogelio Salvador. "Virtualized Dynamic Port Assignment and Windowed Whitelisting for Securing Infrastructure Servers," *Proc. IEEE EIT* 2016.

[11] Miltiadis Kandias, Alexios Mylonas, Nikos Virvilis, Marianthi Theoharidou, and Dimitris Gritzalis, "An insider threat prediction model," *Lecture Notes in CS: Trust, Privacy, and Security in Digital Business 6264*, 2010, pp. 26-37.

[12] Miltiadis Kandias, Nikos Vivrilis, and Dimitris Gritzalis, "The insider threat in cloud computing," *Lecture Notes in CS: Critical Information Infrastructure Security 6983*, 2013, pp. 93-103.

[13] G. Kim, S, Lee, and S Kim, "A novel hybrid intrusion detection method integrating anomaly detection with misuse detection," *Expert Systems with Applications 41*:4, 2014, pp. 1690-1700.

[14] G. Magklaras, S. M. Furnell, and P. J. Brooke, "Towards an insider threat prediction specification language," *Information Management and Computer Security 14*:4, 2006, pp. 361-381.

[15] Patrick McDaniel and Atul Prakash, "Method and system for determining and enforcing security policy in a communication session," US Patent US20030126464, 2003.

[16] M. Mondal, P. Druschel, K. P. Gummadi, and A. Mislove, "Beyond access control: managing online privacy via exposure," in *Workshop on Usable Security 14*, 2014.

[17] J. Ng, D. Joshi, and S. M. Banik, "Applying data mining techniques to intrusion detection," *IEEE Intl. Conf. on Information Technology New Generations*, 2015, pp. 800-801.

[18] Sejong Oh and Seog Park, "Task-role-based access control model," *Information Systems 28*:6, 2003, pp. 533-562.

[19] J. Park and J. Giordano, "Role-based profile analysis for scalable and accurate insider-anomaly detection," *25th Intl. Conf. on Performance, Computing, and Communications*, 2006.

[20] J. Park and R. Sandhu, "The UCON-ABC usage control model," *ACM Transactions on Information and System Security 7*:1, 2004, pp. 129-174.

[21] Animesh Patcha and Jung-Min Park, "An overview of anomaly detection techniques: existing solutions and latest technological trends," *Computer Networks 51*:12, 2007, PP. 3448-3470.

[22] F. Roesner, T. Kohno, A, Moshchuk, and B. Parno, "User-driven access control: rethinking permission granting in modern operating systems," *IEEE Symposium on Security and Privacy*, 2012, pp. 224-238.

[23] T. Ryutov, C. Neuman, Kim Dongho, and Zhou Li, "Integrated access control and intrusion detection for web servers," *IEEE Transactions on Parallel and Distributed Systems 14*:9, 2003, pp. 841-850.

[24] Ravi Sandhu, Edward J. Coyne, Hal L. Feinstein, and Charles E. Youman, "Role-based access control models," *IEEE Computer 29*:2, 1996.

[25] E. Schultz, "A framework for understanding and predicting insider attacks," *Computers and Security 21*:6, 2002, pp. 526-531.

[26] Jeffrey Stanton, Kathryn R. Stam, P. Mastrangelo, and Jeffrey Jolton, "Analysis of end user security behaviors," *Computers and Society 24*:2, 2005, pp. 124-133.