Whitelisting ground systems

Julio Vivero*
GMV, Barcelona, Spain, 08006

Ground systems have a very specific function, defined at design phase, and are not used for anything else. Hence, the processes, libraries, files, network connections and other resources used are clearly specified and stable unless an approved change takes place. Indeed, a sudden unexpected behavior of a ground system server should, ideally, raise suspicion if detected and trigger an inquiry to analyze the cause and decide corrective measures. Also, ground systems usually have an operational environment, a backup or redundant environment that takes over if the operational ones fails, a test environment to validate changes before they are actually implemented and even a development environment for development testing and integration. Even though this is a recommended best practice, if the environments are not correctly isolated and/or distinguished commands can be unintentionally executed in the wrong environment potentially impacting the service. Being the most likely cause of incidents, human errors are usually detected and contained fast, minimizing their impacts. On the other hand, intentional attacks or malware infections, though less likely, will cause a much bigger impact on the operational facilities and service availability. The stuxnet attack has been an eye-opening for many showing that network isolation and traditional antimalware solutions are not effective for targeted attacks. The fact that ground system functions are clear, specific and controlled can be leveraged to significantly increase their resilience, not only to intentional attacks but also to operational mistakes. The paper presents a solution for learning and implementing whitelists of allowed processes for each server and the resources permitted to each process including network connections, files and folders, USB drives and others. Allowed process can be user dependent and vary per ground system environment (i.e operational, test or development). Whitelists are then centrally managed and activated or deactivated on demand by authorized personnel.

I. Introduction

UNTIL not too long ago space systems (both ground and onboard) where completely isolated from external networks and security incidents where scarce and mostly accidental in nature, such as the ISS worm infection in 2008†. This situation is changing slowly but without pause. On the one hand, each passing day networks, even space ones, are more interconnected (with partners, customers, employees for teleworking, and others) but this increase in their exposure has rarely being accompanied with more stringent security safeguards. On the other hand, interest in space systems is also growing within hostile communities. Cyber crooks, hostile governments, hacktivists and any other kind of cyber fauna may find nowadays different motivations for attempting to compromise space systems.

There are three main motivations for attackers: revenge, political interest or financial interest. Revenge is usually more closely linked with internal threats while the other two are, usually, closer to external attackers and threats.

Public and private commercial satellite organizations operate assets (i.e. satellites) which might be attractive for attackers moved by the three types of motivations. Revenge is linked with the human nature. Political interest is clear for defense satellites, but even earth observation missions might generate political interest. Finally, financial interest is related with the costs of the satellites: if someone threatens to destroy an investment of several million dollars, how much money will you pay to avoid it?

* Head of Consulting Area, GMV Northeastern Region, Balmes 270 5th floor - 08006 Barcelona.
† http://nakedsecurity.sophos.com/2008/08/27/computer-worm-strikes-international-space-station/

The Stuxnet attack has been an eye-opening for many demonstrating two main things: first that theoretically secure environments are exposed if enough effort and talent is dedicated to it; and, second that the interest of attackers is not limited to credit cards and financial fraud.

This trend has been exemplified in the last year with security incidents reaching the media in several satellite agencies NASA[‡,§], ESA[**] and JAXA[††].

In addition to these external tensional factors on the security of space systems, internal factors have to be taken into account as well. The operation of Space Systems is more and more automated and dependent on technology to accommodate new functionalities in the one hand and to reduce operational costs on the other. However, this introduces more complexity and volume in space systems and networks. Additionally, more and more services are offered to partners and customers, exchanging more data and interconnecting more networks. Interconnectivity capabilities of devices have expanded from purely wired possibilities to local wireless connections, to simple broadband 3G interconnectivity. Software complexity is also continuously increasing to support more capabilities and functionality, but also introducing more bugs and vulnerabilities. All these factors demonstrate that protecting complex infrastructures such as space ones is a really complex task with new challenges being faced day after day.

The reader familiar with space systems design, implementation, operation and maintenance might be thinking, at this point of the reading, that still, security incidents are not the main cause of concern in the daily life of space operations. Human mistakes are the most common source of unavailability in space systems, though usually with lower impact than an intentional attack, particularly in the media. However, mistakes are also linked with the above factors in two ways. The first because more complexity in systems usually implies a higher likelihood of human mistakes. The second is that, human mistakes are usually exploited by attackers to compromise systems. In any case, it is unarguable that actions need to be taken to try to minimize human mistakes at least as much as intentional attacks.

II. Problem

Space systems, both ground and onboard, have several particularities which preclude the use of traditional IT security safeguards such as traditional user authentication approaches, patching or antivirus software because in such environment they are at least inefficient if not even problematic.

One example of such particularity is the strict change management procedures applied over space systems. Availability is an essential requirement to those systems and, as such, changes are closely controlled to avoid the introduction of inadequately tested software and to reduce the chances of human errors. In such an environment every change has an associated management cost which is significantly higher than in other IT environments. Additionally, any change, even though carefully tested introduces some level of risk. These two factors cause software patches (and particularly security patches) not to be deployed unless there is a strong business reason to do it.

Another typical characteristic of space systems is their strong perimeter security, if not complete network isolation. Incoming and outgoing connections, if allowed, are permitted only between some systems with strong monitoring and filtering safeguards. This factor, combined with the strong availability requirements, do not permit the introduction of antivirus software commonly used in other IT sectors. On the one hand, antivirus signatures updates will introduce unnecessary external network connections and, on the other hand, the risk of a false positive disrupting a particular service is simply too high for space system standards.

Finally, a last example (there are some others) of space system particularity is the need, on some of them, of been continuously logged on, since they must display on a 24/7 basis the status and health of systems to operators. In these systems the introduction of traditional access control mechanisms such as user and password is not feasible.

Based on all the above it is clear that many traditional IT security solutions do not fit well on space systems. Up to now, the most common posture to this situation has been just not to introduce these security safeguards and accept the risks of doing so.

However, as justified above, this risk that has to be assumed is continuously increasing as the interest on critical infrastructures is growing both by private parties and rival countries. Space organizations are no longer able to maintain the current situation and need to introduce more security safeguards. As an example, NASA declared in the

[‡] http://nakedsecurity.sophos.com/2011/11/16/nasa-hacker-arrested-perhaps-it-is-time-for-some-defense/
[§] http://www.bloomberg.com/news/2011-10-27/chinese-military-suspected-in-hacker-attacks-on-u-s-satellites.html
[**] http://www.zdnet.co.uk/news/security-threats/2011/04/19/esa-hack-did-not-breach-internal-network-40092569/
[††] http://www.jaxa.jp/press/2012/03/20120327_security_e.html

US congress that security incidents have cost $7 million in 2010 and 2011. The key to solve this problem is introducing safeguards that fit well with the particularities of space systems.

III. System Whitelisting

Space equipment has very specific functionality and behavior which does not change significantly during the mission lifecycle. Resources required by each server and workstation are well known and stable. In such scenario, not only hardening of both servers and workstations can be easily performed but also a whitelisting of what processes and libraries are allowed to execute. Whitelisting will permit only the execution of known good files, flipping the current concept in antivirus and HIDS solutions of 'default allow' to a 'default deny'. Executable files are identified based on their cryptographic hash value, so that modified files will be equally blocked.

The application whitelisting concept[1] can be further extended to whitelist not only executable files or java classes, but also whitelisting the system resources (e.g. files, folders, incoming or outgoing connections, I/O ports such as USBs, etc.) to which these executable files are allowed to access.

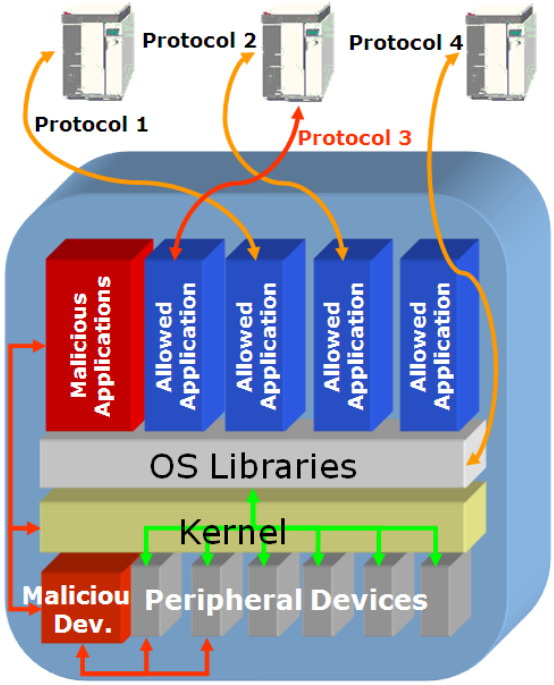

Figure-1- Malicious applications and resources

Permitted executable files and resources allowed are represented in policies. Privileges that are controlled and represented within the policies are:

1) Read or write privileges to registry branches (in windows boxes)
2) Read or write privileges to files and folders
3) Libraries allowed to be accessed by the executable file
4) Creation of listening ports by executable files
5) Incoming and outgoing connections from and to specific IPs and ports
6) Java classes or jar files which are allowed to run within a JVM
7) Access to USB ports

Policies are created in the central management console and enforced by low resource consumption agents (i.e. the whitelisting agent). The whitelisting agent receives and enforces policies defined in the server, and sends back to the server events of forbidden actions performed in the system.

The central management console allows the users to:

1) Edit policies and create different versions
2) Learn from agents events to refine policies.
3) Distribute policies to systems
4) Activate/deactivate the agent in systems

5) See the status of the agents within the network and filter monitoring events from agents.

A particularly useful functionality of the process whitelisting solution proposed is its learning capability. Systems are normally operated and run for a period of time which is used to learn what is really needed for the day-to-day operation of the system and therefore create the appropriate policies. Different policies can be created for different systems and even for different phases, for example, a system may have an operational policy and a maintenance policy. Policies can even define different allowed processes and resources for different users or different environments (e.g. operational and validation), thus forbidding in operations commands which might be allowed in the validation environment and hence, reducing the possibility for operational human mistakes.

IV. Implementation

Typically, the implementation of a system hardening and whitelisting solution starts from the unprotected operational systems, that is from systems containing all necessary software and hardware appropriately configured to work in an operational environment, but which have not yet being hardened and whitelisted.

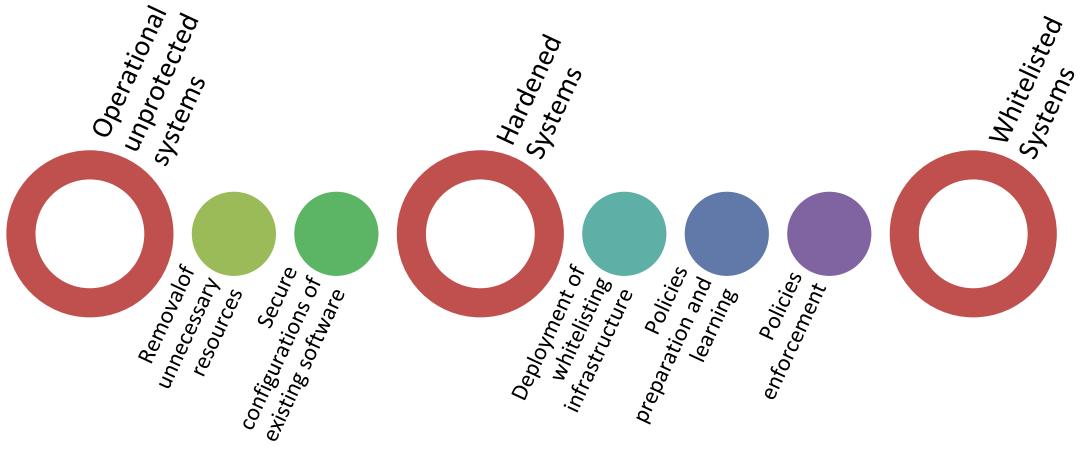

Figure-2- Implementation methodology

A. System Hardening

System hardening is the first step in the process. It consists of two main steps:

1) Removal of unnecessary resources
2) Secure configuration of existing software

Unnecessary applications, modules, drivers and services shall be identified and removed from the system. In some cases, even physical hardware removal (e.g. USB ports, card reading ports, Bluetooth, wi-fi, etc.) might be considered.

Remaining software, including operating system and applications, is then configured with the most restrictive parameters which permit the normal operation of the system. Usually, software manufacturers provide guidelines for secure configuration of their applications which might be very helpful in the process.

Nonetheless, the secure configuration step takes into consideration additional actions to be executed in addition to the pure hardening of system applications themselves. The system BIOS shall be configured securely as well, introducing a configuration password that forbids unauthorized persons from modifying the BIOS. Among configurations to take into account at BIOS level are: boot sequence, disabling optical drives or USB ports, Wake on LAN configurations and others. Also, user accounts shall be carefully secured so that, privileged accounts are strictly controlled, also sudoers privileges in Linux systems, no default passwords are left and the password policy requires minimum password complexity, unnecessary accounts are removed from the system, and user accounts have the minimum privileges required. Audit and application logs shall be configured to generate the appropriate events, protect them and process them accurately. Finally, other configurations might be considered such as configuring static hostnames or mac addresses to prevent network attacks from dns poisoning[2] or arp spoofing[3].

Finally, physical access protection for servers and workstation considerations are also advised.

System hardening provides several benefits to the overall protection of the solution. First, it reduces the complexity of the policy creation and learning process since system complexity has been reduced. Second, it enhances system performance by avoiding resource consumption by unneeded software. Third, and most important, it reduces attack possibilities which might be aimed at disabling security controls by, for example, booting the system from a live CD to modify system files which would otherwise be protected by the whitelisting policy.

B. Deployment of whitelisting infrastructure

Depending on the number of systems managed, the central management console infrastructure can be as simple as one machine running application servers and database (two for high availability purposes) or an infrastructure as the one shown in the figure below.

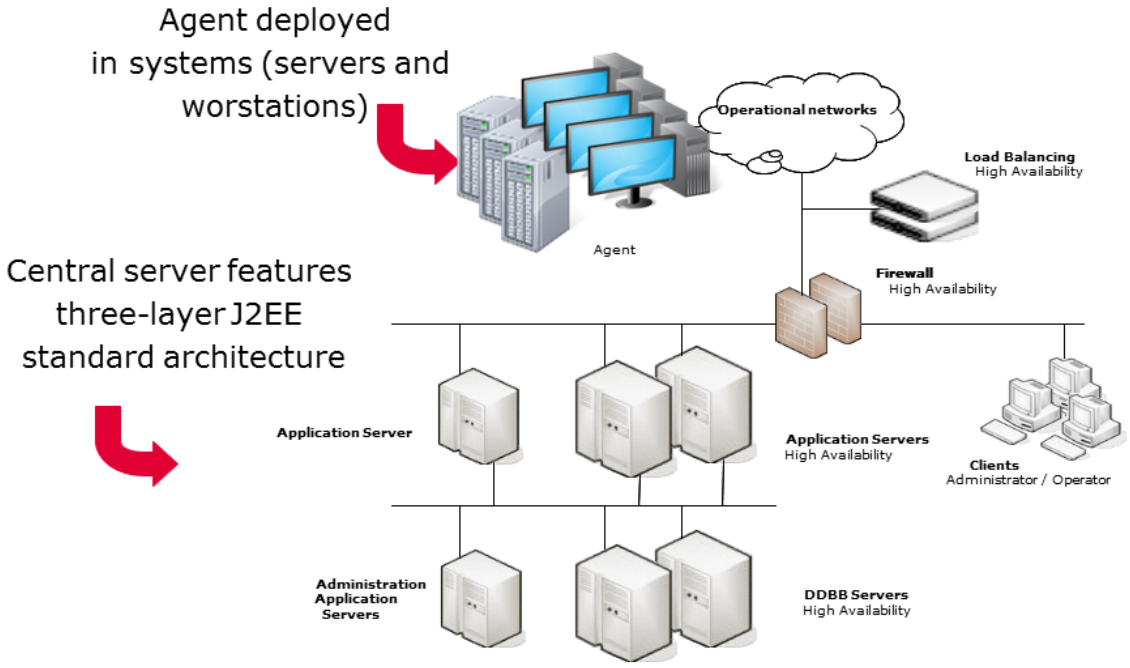

Figure-3- Whitelisting infrastructure

The deployment process starts with the deployment of the central administration console modules (application servers and database) in one or more systems and its configuration including user profiles registration.

Once the central administration console is correctly installed and functioning whitelisting agents are installed on servers and workstations. The central administration console will automatically identify and register installed agents. From this point onwards agents are ready to receive policies and send events to start the policy creation process.

C. Policy definition and enforcement

Whitelisting agents have three possible operational modes: disabled, learning and active. In disabled mode, the agent does not enforce the whitelisting policy and does not generate any events. In active mode, the agent does enforce the whitelisting policy forbidding any unauthorized action or resource access in the system and sending the corresponding alerts when unauthorized actions are attempted. The most interesting working mode though, from the policy edition point of view is the learning mode. In learning mode, the whitelisting agent does not enforce the whitelisting policy, but it monitors all actions and resources and generates events when unauthorized actions are executed even though they are not forbidden in the system.

The learning mode is very useful to create operational policies in a short period of time and with the confidence that nothing required for the system to work has been left out. At the central management console events are received and used to make enhancement suggestions to the current policy. If the event reflects an action which shall be enabled it can be added to the policy with a single click, otherwise discarded as easily.

The normal implementation process would be to set all whitelisting agents in learning mode and send “basic” (i.e. containing the basic processes for the operating system to work) policies to them. At this point in time, the

central management console is ready to learn the executable files and resources required for the system to work in an operational environment. All operational procedures can then be tested so that the policy can be refined to allow the required resources and nothing else.

Depending on the particularities of each organization different policies can be created for different system status (operational or maintenance; active or standby, etc.) or different permitted actions can be created for different users (operators, controllers, administrators, developers, etc.). Also, individual policies can be created for each particular system, or a single policy can be defined for a set of systems with share several commonalities.

Once the whitelisting policies are fully defined, agents can be switched to active mode and monitor their correct behavior.

V. Operation and Maintenance

A. Operation

In nominal operations, systems shall be running normally. When exceptional actions have to be executed over a system or workstation, the whitelisting agent for that element can be switched to learning mode from the management console. In that way, these exceptional actions would not be forbidden but each process executed and resource accessed will be monitored and events will be sent to the central management console.

An operator can periodically review agents alerts to detect potential intrusion attempts that might cause the creation of a security incident, or instead, the refinement of the policy to accommodate a new legitimate action.

Different operators can be defined for different sets of systems, as well as different user profiles can be created with different privileges on the management console. Typically, policy edition, whitelisting agent mode modification (i.e. disabled, learning, active), and alarm monitoring privileges will be assigned to different roles within the organization.

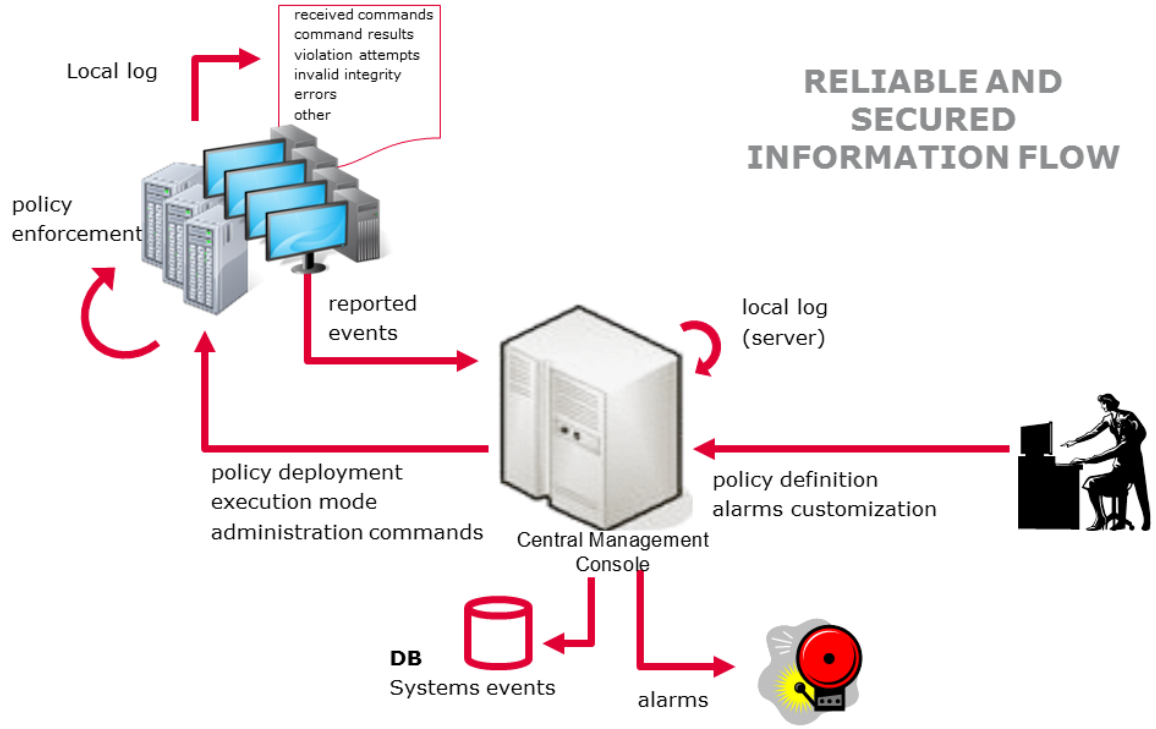

Figure-4- Operational whitelist management information flows

B. Maintenance

Even though space systems are usually quite stable in nature, maintenance operations are normal, well planned and controlled.

Change management procedures are strictly enforced. With a system whitelisting solution these procedures shall be updated to accommodate the policy adaptation linked with each change. Figure 5 shows the normal policy lifecycle linked with system changes.

The new policy version will be derived from the current valid policy, in this way there is full configuration control over the policies and roll-back actions are simple.

Once a new version has been created, which inherits all permissions from its parent, the known change modifications are introduced in the policy: old processes/resources that are no longer used or new actions that shall be permitted on the system.

As it is quite frequent that not all details are included in the change information the next step is to test the new policy within a validation system with the change implemented. The whitelisting agent will work in learning mode to provide to the central management console the learning events to close the policy adaptation and fully customize it to the changes introduced.

Once the policy is finalized it can be distributed to operational systems alongside the change deployment.

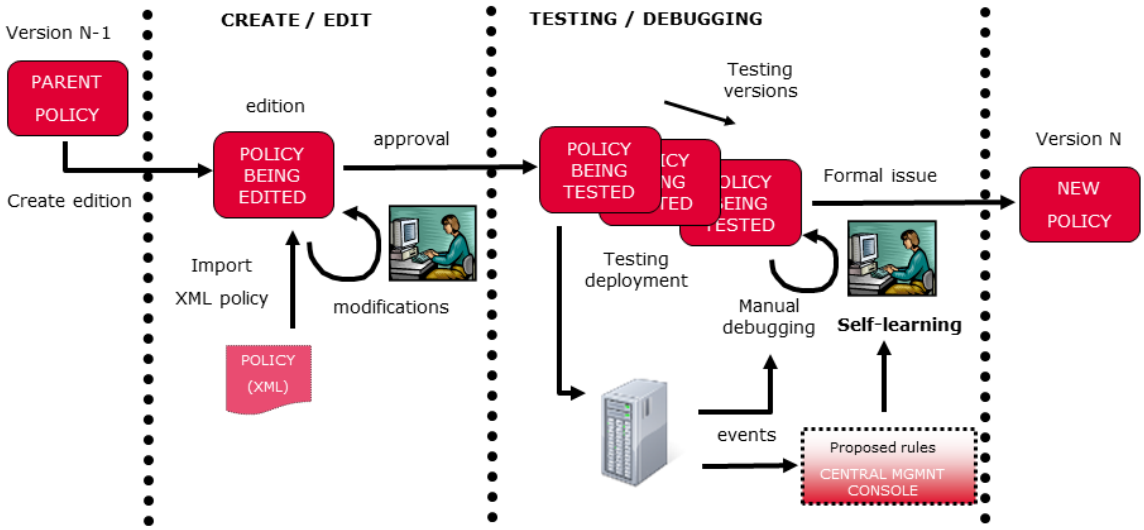

Figure-5- Policy lifecycle

VI. Conclusions

Sever hardening combined with process whitelisting fit well to space systems particularities removing the drawbacks of other typical security safeguards, namely:

1) It removes the drawbacks of unavailability risks due to changes since no patches or up-dates are required. The reason is that much less services potentially vulnerable will be running on systems, and even if they are compromised the attacker will be completely tied in the actions he or she can execute in the system (only those permitted by the process whitelisting policy).
2) It removes the need of external connections because no signatures or rules need to be downloaded from external providers.

However, this solution does not only remove these drawbacks but also introduce some additional benefits:

1) They not only protect the infrastructure against malicious activities but also from unintended operational mistakes such as inadvertently telecommanding a satellite from a testing environment, since the process whitelisting software will filter to which IPs each system is allowed to connect and which IPs are allowed to connect to concrete services within the system.
2) Also, operational mistakes will be significantly reduced because wrong processes or system commands will simply not be allowed on operational systems.

3) Antivirus, antispyware or any other kind of malware protection software would no longer be required in your environment simply because, independently of the infection vector and independently of the malware sample (no matter whether new or old) will not be allowed to run within space systems protected with a process whitelisting solution.
4) Server hardening will significantly reduce the exposure of systems by removing unnecessary services and introducing secure configurations for applications and the operating system itself.
5) Finally, but no less important, system performance and availability properties is also enhanced because no resources are being consumed by unnecessary services and the number of anomalies caused by unpredicted software interactions are reduced.

References

[1]Department of Defense, Intelligence and Security, Australian Government, "Application Whitelisting Explained", June 2011
[2]Son, S. and Shmatikov, V., "The Hitchhiker's Guide to DNS Cache Poisoning", SecureComm 2010
[3]Whalen, S., "An Introduction to ARP Spoofing", http://www.rootsecure.net/content/downloads/pdf/arp_spoofing_intro.pdf, Apr. 2001.