ARGUS SYSTEMS GROUP

PitBull LX

Revolutionary Security For Unix & Linux

White Paper
October 2001

COPYRIGHT

DISCLAIMER

THIS PUBLICATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

NOTICE TO USER

This publication could include technical inaccuracies or typographical errors. Changes are periodically added to the information herein. These changes will be incorporated in new editions of the publication. Argus Systems Group, Inc. may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time.

TRADEMARKS

PitBull is a registered trademark of Argus Systems Group, Inc. Argus Systems Group, PitBull Foundation, PitBull.comPack, Gibraltar, and their respective logos are trademarks of Argus Systems Group, Inc.

All brand names and product names used in this document are trade names, service marks, trademarks, or registered trademarks of their respective owners.

Table of Contents

THE THREATS ARE REAL 3

COMMON SECURITY PRACTICES DON'T MEASURE UP........ 3

THE NEED FOR APPLICATION SECURITY 3

PITBULL LX DESIGN........ 4

FUNCTIONALITY 4

TECHNICAL DESCRIPTION........ 5

FILE ACCESS DOMAINS........ 5
NETWORK ACCESS DOMAINS........ 5
DOMAIN SETS 6
FILE EXECUTION DOMAINS 6
FILE SECURITY FLAGS........ 7
ASG_AWARE........ 7
ASG_INHERIT_DOM 7
ASG_MASK_EXEC 7
ASG_COMB_EXEC........ 8
FILE EXECUTION FLAGS 9
ASG_AWARE........ 9
ASG_SEC_ACC_FS........ 10
ASG_SEC_ACC_NET........ 10
ASG_RES_SYS........ 10
ASG_RES_MNT........ 10
ASG_RES_RBT........ 10
ASG_RES_DAC_OVR 10
ASG_RES_SETID........ 11

APPLYING THE TECHNOLOGY 11

SECURING A BASE SYSTEM........ 11
SECURING A USER 11
SECURING THE NETWORK........ 12
SECURING A WEB SERVER 12

FOR MORE INFORMATION........ 13

The Threats are Real

Every day sophisticated hackers and malicious insiders launch attacks against sensitive e-commerce servers. Although many successful attacks are reported, the vast majority of serious attacks never make the headlines. To make matters worse, professional hackers can be so adept at covering their tracks that many companies simply have no way of knowing if their systems have been compromised. Despite the increased attention and spending levels given to e-security, the problems are escalating. Security holes and system vulnerabilities are being discovered at an alarming rate. And often, before the vulnerability is even made public, a sophisticated and readily available hacker tool has already been developed to exploit it.

Common Security Practices Don't Measure Up

So what is the answer to these serious and growing security threats – adding more and more costly layers of perimeter security such as firewalls and intrusion detection systems? Experience tells us that a new, more effective approach is needed – not at the perimeter, but at the server level. If your mission-critical servers are not built upon a secure foundation, your entire enterprise is vulnerable, regardless of the security mechanisms deployed at the perimeter. The only way to achieve this essential foundation-level security is to secure the operating systems of your e-business servers.

Fact: If an attacker (external or internal) gains control of the operating system, all traditional security mechanisms are totally ineffective – your e-business is at the mercy of the attacker.

The Need for Application Security

PitBull® LX™ is an application security solution designed to stop attacks from both outsiders and people inside an organization. Unlike common perimeter-based security products such as firewalls and intrusion detection systems, PitBull LX delivers security where it is needed most – within the operating systems that control the most sensitive servers in your IT infrastructure. PitBull LX from Argus is an ideal software foundation for securing web servers, application servers, DNS servers, mail servers, shell servers, database servers and e-commerce servers.

PitBull LX Design

PitBull LX was designed from the ground up to meet the security requirements of commercial network-based systems. PitBull LX is a secure operating system enhancement solution for AIX, Solaris, and a wide variety of Linux distributions. LX software is in the form of a kernel module which enhances the security of the base operating system. PitBull LX can only be applied only to the areas of the system that require it, whereas the remainder of the system will behave just like a normal Unix or Linux environment.

PitBull LX was designed to be easier to understand, configure, and work with on a day-to-day basis than other secure OS solutions. It is an intuitive security technology that lends itself to widespread adoption. At the heart of PitBull LX functionality is a compartmentalization mechanism that allows an administrator to completely isolate processes and subsystems from each other on the same system. This mechanism, known as Access Domains, is not tied to the user ID or user name and thus prevents compromise of other subsystems if a process is hacked and an attacker gains a different user ID. PitBull LX implements a fine-grained network access control policy similar to a firewall, except that it extends down to protecting the processes themselves.

Functionality

- PitBull LX provides the ability to isolate processes from system resources in a way that cannot be circumvented, even in the event of an attacker becoming superuser. System resources protected include file system access, networking, IPC mechanisms, and signals.
- Isolation mechanisms are fine grained and similar to common, easily-understood protection concepts such as the ability to read, write, and execute.
- Network isolation mechanisms are similar to firewall-filtering behavior and can be used to control or prevent communications between specific hosts, networks and processes.
- PitBull LX security is only active on the areas of a system that an administrator decides to protect.
- PitBull LX can limit the access a root user (superuser) will have on the system. Normal users can be restricted from ever obtaining the special privileges of superuser.
- PAM (pluggable authentication module) is supported, allowing configurable security settings in a variety of environments.
- Extending the standard Unix permissions concept through access domains makes PitBull LX very understandable and easy to use for current Unix and Linux users and administrators.

- PitBull LX installs on top of a running Unix or Linux system, allowing security upgrades to existing installations.

Technical Description

In order to understand PitBull LX technology it is necessary to gain an understanding of its functional components.

File Access Domains

A file access domain is a set of three bits used to control access to filesystem objects. The three bits of the file access domain represent read access, write access and execute (search) access, respectfully. These will be represented as *domain_name*(r,w,x), where *domain_name* is the symbolic name for the domain and the grouping (r,w,x) represents the type of access the domain can restrict: r for read access, w for write access, and x for execute (search) access.

Examples:	sys(x)	- execute (search) sys domain
	html(r,x)	- read and execute (search) html domain
	myuser(r,w,x)	- read, write, and execute (search) myuser domain

File access domains can be found on both file objects and on processes. In order for a process to access a file object, the process must possess the access domain that is required by the file object for that type of access.

File access domains are extremely useful for isolating files from certain processes or users. For example, few users on a system ever need to have write access to library files and other executables on the system. By placing a PitBull LX file access domain with write (w) restrictions on these files and never giving a user process this file access domain, it is possible to prevent a user from ever writing to these files. This holds true even if the user changes users IDs to that of superuser by sending a buffer overflow attack to a setuid root program.

Network Access Domains

A network access domain is a bit used to control access to a network object. The network access domain represents all types of access to a network object. These will be represented as *domain_name*(net), where *domain_name* is the symbolic name for the domain and (net) signifies that it applies to network objects.

Examples:	eth0(net)	- access to eth0 domain
	port80(net)	- access to port80 domain

Network access domains determine whether two processes can talk to each other through the local network or through other IPC mechanisms such as UNIX domain sockets. In order for two processes to talk to each other they simply need to have <u>one</u> network access domain in common. Network access domains are also used on network interfaces to restrict a processes access to the network. Firewall-style rules are used to configure the network and can limit or completely prevent processes from accessing the network, even in the event of a process gaining superuser status.

Domain Sets

A domain set is a logical grouping of related access domains for an object. There are four types of access domain sets: read domain set, write domain set, execute domain set, and network domain set. Each domain set contains all of its respective domains for an object. These will be represented as *domain_set: dom1, dom2…. domain_set* is one of: read, write, execute, or network. *dom1, dom2…* is the list of domains associated with the domain set.

Example:	object’s domains:	sys(r,x) myuser(r,w,x) html(r,x) eth0(net)
	objects’ domain sets:	read: sys, myuser, html
		write: myuser
		execute: sys, myuser, html
		network: eth0

DOMAIN SETS

		Read	Write	Execute	Network
ACCESS DOMAINS	sys	R		X	
	html	R		X	
	myuser	R	W	X	
	eth0				N
	port80				

File Execution Domains

File execution domains are exactly the same as file access domains except that they are used only on executables. These domains determine what type of PitBull LX file access domains a process that launches the executable program will receive. These domains can be used to force a non-PitBull LX user into LX mode, further restrict the access of a PitBull LX user, or temporarily grant a user additional file access domains. An example of a program that temporarily grants additional file access domains is the passwd command that is used for modifying a user's password and thus requires special access to the password database file.

File Security Flags

File security flags are special flags that can be placed on files to determine certain behaviors. These behaviors can tell the PitBull LX kernel module whether PitBull LX is protecting a file, whether PitBull LX security should automatically be placed on files that are created in a directory, as well as cause special security actions to be taken when a file is executed. Examples of security flags appear below.

ASG_AWARE

The ASG_AWARE flag indicates that PitBull LX is protecting a filesystem object. This flag will be set if any access domains or other flags are set.

ASG_INHERIT_DOM

The ASG_INHERIT_DOM flag, when set on a directory, indicates that all new files created in the directory will inherit attributes of the directory. This flag may be set on all objects, but only has meaning on directories.

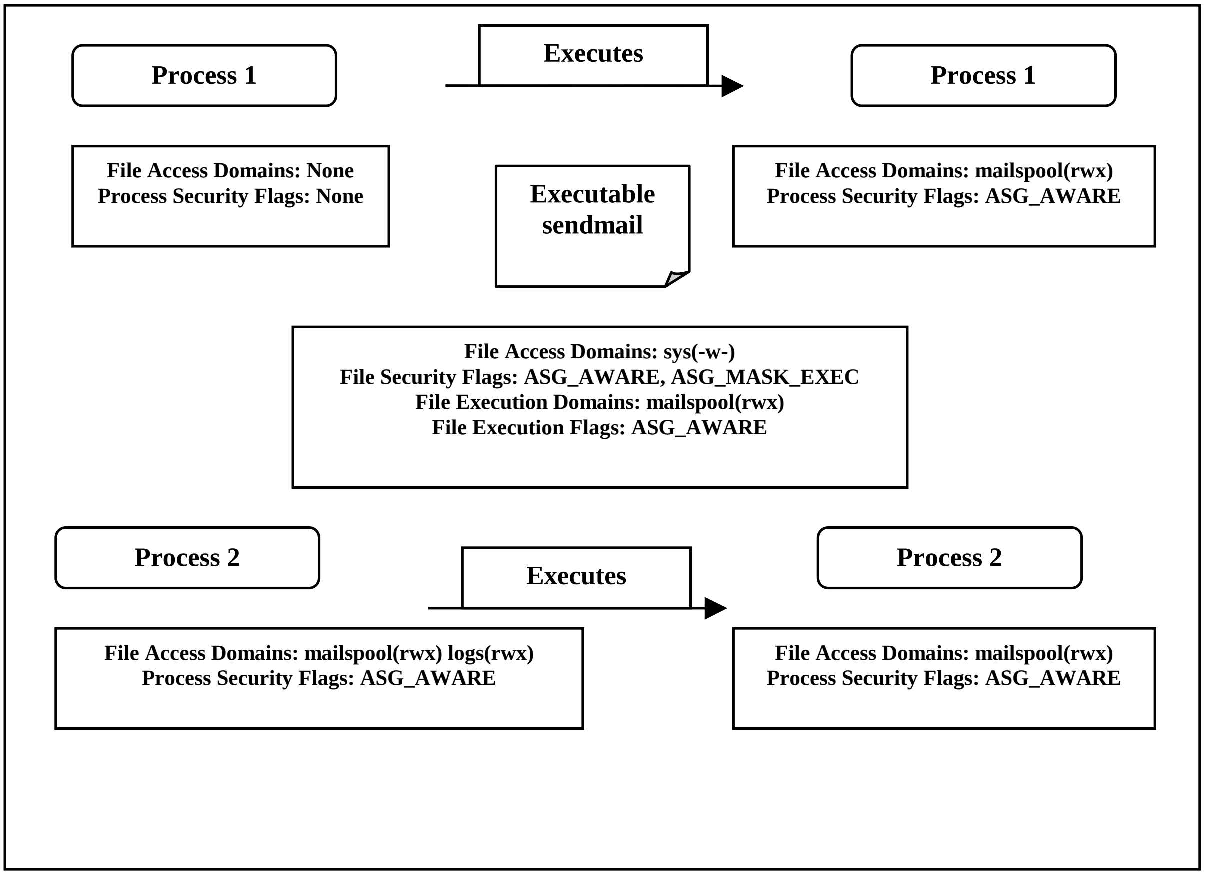
This flag is useful for simplifying the day-to-day operation of PitBull LX. For example, if /etc is protected with sys(-w-) (sys is a common PitBull LX file access domain used to protect system files) and if the ASG_INHERIT_DOM flag is set, then a file created in /etc by a non-LX administrator would automatically be protected with sys(-w-). This simple configuration helps to prevent operator error when dealing with security relevant files.

ASG_MASK_EXEC

The ASG_MASK_EXEC flag, when set on an executable, indicates that a process running the executable may have its PitBull LX attributes altered. This flag may be set on all objects, but only has meaning on executable binaries. The ASG_MASK_EXEC flag is used for the following purposes.

- It causes the executed program to receive PitBull LX file access domains that are a mask (or intersection) of the current process' file access domains and the file execute domains of the executable.
- It increases the process security flags on the executed process by setting them to the union of the current process' process security flags and the file execution flags on the executable's binary file.
- It is used for switching a user or process into LX aware mode and thereby picking up the appropriate file access domains as well as security flags. Once in this mode, the process is locked into its security settings.

- It is used for reducing the set of file access domains and increasing the process security flags on a process that is already LX aware.


Process 1
Executes
Process 1
File Access Domains: None
Process Security Flags: None
Executable sendmail
File Access Domains: mailspool(rwx)
Process Security Flags: ASG_AWARE
File Access Domains: sys(-w-)
File Security Flags: ASG_AWARE, ASG_MASK_EXEC
File Execution Domains: mailspool(rwx)
File Execution Flags: ASG_AWARE
Process 2
Executes
Process 2
File Access Domains: mailspool(rwx) logs(rwx)
Process Security Flags: ASG_AWARE
File Access Domains: mailspool(rwx)
Process Security Flags: ASG_AWARE

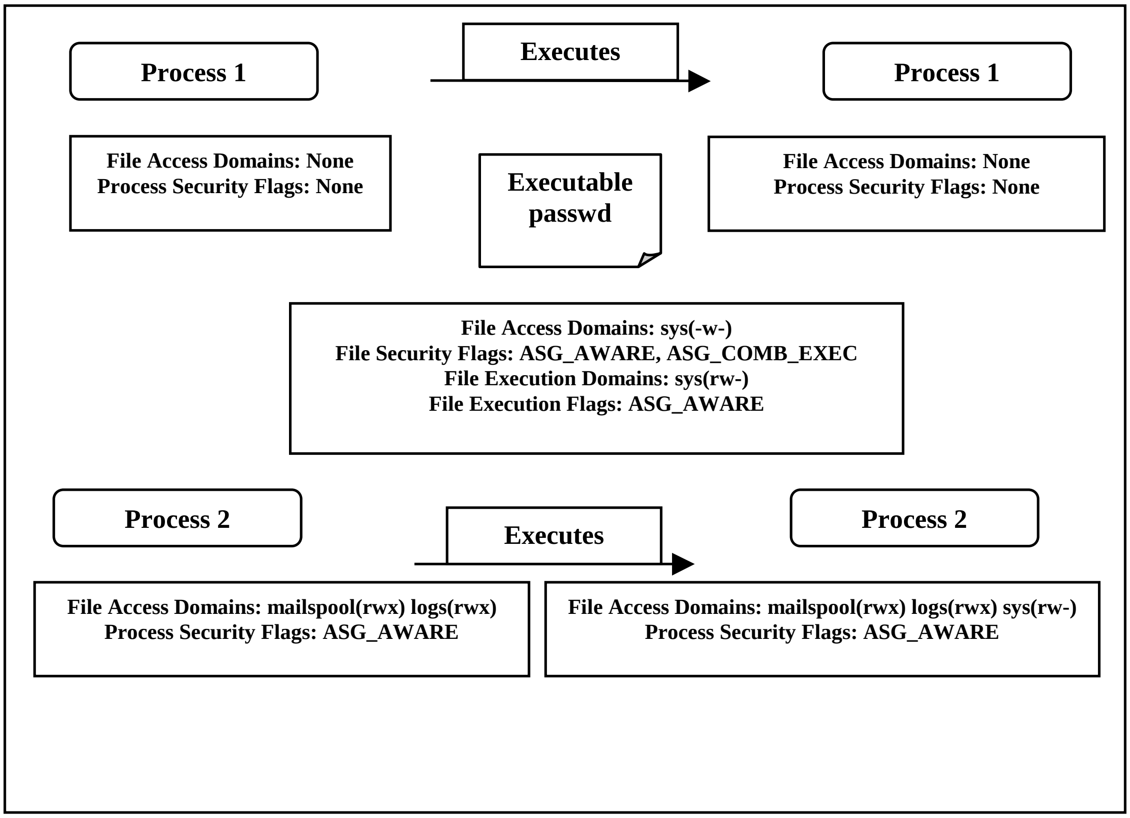
Note: The above example illustrates how two processes with different PitBull LX security settings interact with an executable that has the ASG_MASK_EXEC flag.

ASG_COMB_EXEC

The ASG_COMB_EXEC flag, when set on an executable, indicates that a process running the executable may have its PitBull LX attributes altered. This flag may be set on all objects, but only has meaning on executable binaries. The ASG_COMB_EXEC flag is used for the following purposes.

- It causes the executed program to receive PitBull LX file access domains that are a combination (or union) of the current process' file access domains and the file execute domains on the executable that is being run.
- It increases the process security flags on the executed process by setting them to the intersection of the current process' process security flags and the file execution flags on the executable's binary file.

- It is used for increasing the set of file access domains and decreasing the security flags on a process that is already LX aware.


Process 1
Executes
Process 1
File Access Domains: None
Process Security Flags: None
Executable passwd
File Access Domains: None
Process Security Flags: None
File Access Domains: sys(-w-)
File Security Flags: ASG_AWARE, ASG_COMB_EXEC
File Execution Domains: sys(rw-)
File Execution Flags: ASG_AWARE
Process 2
Executes
Process 2
File Access Domains: mailspool(rwx) logs(rwx)
Process Security Flags: ASG_AWARE
File Access Domains: mailspool(rwx) logs(rwx) sys(rw-)
Process Security Flags: ASG_AWARE

File Execution Flags

File execution flags are similar to file execution domains as they are used to determine what type of process security flags an executed program will receive. However, unlike file execution domains these flags are different than the file security flags.

ASG_AWARE

The LX_AWARE flag indicates that the process is subject to PitBull LX access restrictions. This flag will be set if any access domains or other flags are set.

ASG_SEC_ACC_FS

The ASG_SEC_ACC_FS flag indicates that the process may not interact with filesystem objects that are not protected with PitBull LX attributes.

This flag is used to limit a process from having access to any file that it has not been given explicit access to through its file access domain set. This prevents a user or process from gaining access to a file that a non-LX aware user may have created without PitBull LX security settings.

ASG_SEC_ACC_NET

The ASG_SEC_ACC_NET flag indicates that the process may not interact with network objects that are not protected with PitBull LX attributes.

The ASG_SEC_ACC_NET flag is used to limit a process from having access to any network resource that it has not been given explicit access to through its network access domain set. This prevents a user or process from gaining access to another process or running a daemon that is not LX aware.

ASG_RES_SYS

The ASG_RES_SYS flag indicates that the process may not perform certain actions on the system. The actions limited by this flag are those actions which may enable a process to bypass or modify the PitBull LX system.

The entire set of ASG_RES flags prevents a user or process from ever obtaining the ability to perform certain security relevant actions, even if they become superuser. These flags are applied to processes or users that will never require these abilities, which greatly reduces the potential problems a hacked process or rogue user can cause.

ASG_RES_MNT

The ASG_RES_MNT flag indicates that the process may not perform actions which may mount or unmount a filesystem.

ASG_RES_RBT

The ASG_RES_RBT flag indicates that the process may not perform actions which may reboot or halt the system.

ASG_RES_DAC_OVR

The ASG_RES_DAC_OVR flag indicates that the process may not override discretionary access control (permission bits / ownership), even if it is running as superuser.

ASG_RES_SETID

The ASG_RES_SETID flag indicates that the process may not change its uids/gids through system calls.

Applying the Technology

Securing a Base System

In order to begin securing a system using PitBull LX it is necessary to first construct a starting set of file access domains. The first and most important file access domain used is usually called ‘sys’. ‘sys’ is used as a file access domain to protect all system files from modification. No user on the system will ever be given this file access domain in his process’ file access domain set. After creating this bit, it needs to be applied to all system files. A good setting is to place sys(-w-) on all libraries, executables, and security-relevant files. This setting will allow all LX-aware processes to read and execute these files, but not allow them to write to them, ever. This leads us to the next configuration example.

Securing a User

In order to secure a user on the system you simply need to place them into PitBull LX- aware mode. This is easily done, using the PitBull LX PAM extensions. By creating a default entry in the /etc/argus/users database, it is trivial to force all users into a restricted environment where they can only modify their files. A good example set up is:

```
root:console::
ARGUS_DEFAULT:any:net_user(net):ASG_AWARE,ASG_SEC_ACC_FS,ASG_SEC_ACC_NET,ASG_RES_MNT,ASG_RES_SYS,ASG_RES_RBT,ASG_RES_DAC_OVR
```

This simple setup places all users into a PitBull LX aware mode with process security flag restrictions that will prevent them from performing any administrative tasks, even if they become root. The users have also not been given any file access domains so they will only have write access to unprotected LX files (files that are ASG_AWARE but have no file access domain restrictions for the access the user wants). Finally, the users have been given the net_user network access domain that will allow them to communicate with other standard user processes that have the net_user network access domain but not with any other processes. The users will also only be able to establish external connections that are specifically allowed for the net_user network access domain as specified in the network rules. This leads us to the next section.

Securing the Network

Securing a network is also easy with PitBull LX. The first step is to configure the network access domain net_user on the system in /etc/argus/ndommap. One other network access domain called net_denied should also be added. This network access domain is a network access bit to be used in a ‘netrule’ which completely disallows a particular connection, regardless of the circumstances. Once these network access domains are in the file, the netrules network security rules database can be configured.

A simple example of a netrules database is:

```
proto:icmp
domain:net_user
domain:net_denied
```

The above rule set will allow all users to ping out of the network using the ICMP protocol. All other connections will be denied as they will match the default rule at the bottom of the rule set and receive the net_denied network access domain that user processes will not have on them, thus preventing other network communication. This is a fairly restrictive set of rules. Users will be able to telnet into the system, as the inetd service will be running in a non-LX aware mode and thus be able to receive connections. This works because PitBull LX‘s network rules only apply to LX-aware processes.

It is a good practice to place your network services into PitBull LX-aware mode so that they only have access to what they need, and so that a security breach in one service will not lead to a system-wide penetration. This is one of the greatest strengths of PitBull LX.

Securing a Web Server

Securing an application into its own isolated environment is fairly easy using PitBull LX. In order to illustrate this, it is worth looking at a practical example of a commonly-used service such as a web server. In particular, this example will take a look at securing the Apache web server.

The first step in securing an application is to determine what it uses. A web server like Apache needs to have read access to its web pages and write access to its logs. In order to set up a web server that supports these functions you would configure the files in the following way:

Httpd:

File Access Domains: sys(rwx) <- Causes only non-LX aware users to be able to launch the web server

File Security Flags: ASG_AWARE,ASG_MASK_EXEC,ASG_COMB_EXEC

File Execution Domains: httpd(r-x), httpd_logs(rwx)

File Execution Flags: ASG_AWARE,ASG_SEC_ACC_FS,ASG_RES_SYS,
ASG_RES_MNT, ASG_RES_RBT, ASG_RES_DAC_OVR

All binaries, configuration files, and web pages:

File Access Domains: httpd(rwx)

File Security Flags: ASG_AWARE

Log file directory:

File Access Domains: httpd_logs(rwx)

File Execution Flags: ASG_AWARE, ASG_INHERIT_DOM

Log files:

File Access Domains: httpd_logs(rwx)

File Execution Flags: ASG_AWARE

The web server when launched now runs with httpd(r-x), httpd_logs(rwx) file access domains and with a very restrictive set of process security flags. The web server is able to read all of its web pages with httpd read access but is unable to write to them. The web server may also create and modify its log files in the log directory with the httpd_logs(rwx) domain set. Since the log directory has the ASG_INHERIT_DOM flag set, all files created in the directory will be created with httpd_logs(rwx) only.

For more information

For more information about Argus Systems Group and PitBull LX:

• Email:	info@argus-systems.com
• Website:	www.argus-systems.com
• Phone:	(217) 355-6308
• Fax:	(217) 355-1433

- Address:

Argus Systems Group, Inc.
1809 Woodfield Drive
Savoy, IL 61874 USA

WPPLX10.01