Global Information Assurance Certification Paper

The Use and Administration of Shared Accounts

By: David J. Johnson

GSEC Practical Assignment, version 1.4b, Option 1

Table of Contents

Abstract

This paper will discuss the use and security of shared accounts. While shared accounts exist on other systems, this paper has been limited in scope to focus on UNIX- and Microsoft Windows-based systems, however the basic principles should be applicable to other systems as well. The paper will start by defining what shared accounts are, some of their uses, and some of associated risks of those uses. Following that will be a discussion about how to secure shared user accounts to help mitigate their associated risks. It will conclude by presenting information on how a shared account that had minimal security applied and was "in the wild" was secured.

Shared Accounts -- What They Are, Their Uses, and Their Risks

Shared accounts are, generally speaking, just that -- accounts that are shared by one or more users.

These accounts are different from user accounts that can be associated with a particular person. The sharing of accounts that are associated with a particular person will not be discussed in this paper except to say that it is a generally accepted policy and practice that accounts associated with a particular user "should NOT be shared" (Venner) as it invalidates the security and auditing applied to these accounts and their access. While it would be ideal that there never be a need for more than one user to access any account this is not practical for many companies, small and large, as there are benefits to sharing some accounts.

Attempting to discuss all possible uses and risks of shared accounts is too large of a topic to be discussed in a single paper of this nature. There are several ways to categorize accounts that are shared and, for the purpose of this paper, four categorizations will be used when discussing shared accounts. They are anonymous, or guest, accounts, accounts for temporary employees, administrative accounts, and accounts used in batch processing. Discussing them here is to illustrate that there are risks for using such accounts and familiarize readers prior to then focusing on how to secure a shared account when they must be used.

Anonymous/Guest Accounts

Anonymous, or guest, accounts are accounts that are setup for the use of visitors to your system. They are often used for allowing a user access to your system for a specific reason, such as to access to an external server so that they can download shareware or freeware programs. Most systems include at least one such account when installed and are often named "anonymous" or "guest". For example, Microsoft NT 4.0 servers included a default "guest" account that is "disabled by default… and… can't be deleted" (Strebe, p. 331)

In some cases, these accounts do not require a password. Perhaps one of the best known is the "IUSR_computername" (Strebe, p. 159) setup by default when installing Microsoft's Internet Information

Server, also called IIS. This account allows "anonymous Web and FTP" (Strebe, p. 331) access to IIS hosts. In other cases, the password is set to the same value as the account name. An example of this would be an "anonymous" login allowed with a password of "anonymous".

(FTP stands for “file transfer protocol” and is used for moving files between systems. Depending on how the system is configured, FTP may support anonymous logins, logins requiring username and password authentication, or both. There are many sources for additional information on FTP, including the official specifications of the protocol that is available from the Internet Engineering Task Force RFC. The URL for this RFC (RFC0959) is listed in the bibliography of this paper. Before allowing FTP to be used by shared accounts, you should have a strong working knowledge about this protocol.)

Some administrators setup anonymous accounts such that they request that the password sent be the e-mail address of the person logging in. This would include logins as "guest" with a password of "jdoe@email_provider.com". This does not really increase the level of security because often this is just a request and there is no validation that the e-mail address really belongs to the person using it.

The biggest risk of anonymous accounts is also the same reason that they are often used -- they allow anyone access to the system. In some cases, the default access granted to these accounts allows them to access data that they should not have access to view. It is not uncommon for administrators to overlook reviewing and/or restricting the access these accounts have.

Accounts for Temporary Employees

Another type of shared account is an account that might be setup for temporary employees with names such as "temp01" or "temp_9". The thought is that if accounts such as these exist, then the temporary employees will have an account setup when they arrive for the day. In theory, this works great because the administrator doesn’t have to setup a new account every time a new temporary employee is in the office, which may be as often as daily.

However, there is the risk that the new person will have access that they shouldn't. For example, was the account set up for use by someone working in system administration and then reused for some else that is filling in for a vacationing receptionist. Another risk is if the password is not changed after a temporary employee leaves. This would enable them to have access to the system(s) well after the time that they should have lost access.

Administrative Accounts

A third type of shared account is system administration accounts such as "Administrator" or "root". These accounts are often overlooked as being considered shared accounts because it is often, and should be always, that only the system administrators that have access to these accounts and because these accounts are required by the system. While

necessary to configure the systems and administer other accounts on the system, these accounts are also the crown jewel for those seeking unauthorized access to a system. This is because they can then create any other accounts they desire, change configuration settings, corrupt data, or launch attacks on other hosts, especially if the now compromised account is trusted by others hosts and given the same authorization.

Batch Processing Accounts

The final type of shared account discussed in this paper is accounts used for automated batch processing. Since such accounts are used for critical business support, they might actually be an "anonymous" type of account that does not require a password or they might require a "secret" password that is unique to the account. Even with a password, the acocutn might not be secure as a large percentage of the company's IT staff knows what it is in order to provide troubleshooting. In today's world, most companies do not run batch processing on a single host but on multiple hosts which means that if a batch processing account is not secured on one system correctly then access might be gained to another. Another issue occurs when a batch process on one system creates data for or needs data from another system, which means that anonymous or secured access must be granted to the other system. Obviously, secured access in the form of password authentication would be a good approach to take but this can lead to the "secure" password being hard coded into programs and scripts that are not secured resulting in the secured account being easily compromised.

General Security Risks of Shared Accounts

There are two major security risks with any type of shared accounts. The first is that multiple people access these accounts. This can make monitoring and creating an audit trail difficult, even more so if there are multiple logins to an account at the same time. There is also minimal accountability for shared accounts as users do what they want/need with the accounts but the account really isn't their account. As such, they often times don't feel accountable for the security of the account and will do things that they don't do with their own personal account, such as write the password down and leave it lying in plain view.

The second risk is password management. When you have a shared account, the password, if there is one, can not be changed without coordinating the change with everyone that uses the account so that his or her access is not impacted. This also requires that the password be distributed in a secure manner. This can mean a lot of work, especially if the password is widely known, and, therefore, the password can become static where it is not changed on any kind of regular basis, or at all. If the password is well known, or "in the wild", this can also mean that employees, or former employees, that no longer need access to this account may know the password and can still access the account.

Allowing Shared Accounts

The first step in securing shared accounts is to determine whether shared accounts are right for your environment. Do they fit your organization's established policies that govern account creation and access? Is there a business need to justify them? In the case of system administration accounts, the answer to these questions are usually easy to answer and that answer is typically "yes", as they are required by the systems. The answer for other shared accounts is usually more difficult and requires more discussion within your organization.

In terms of anonymous accounts, they might be appropriate for systems such as your external web server but not for your database server that can only be accessed by internal users. In determining if you want to allow anonymous accounts there are several questions that you should consider before determining if you need them or not. What are you trying to accomplish by creating or using these accounts? Is there a way to accomplish this without an anonymous login? Would it be better to require the use of a regular user account that is associated with an identifiable person and require password authentication? If, as previously stated with some web servers, like IIS, then answer as to the need of an anonymous account is "yes". However, just determining the need to have an anonymous account isn't the end of the process.

The determination of whether shared accounts are appropriate for your organization should not be left up to users that simply say "we need this" or denied by system administrators that feel shared accounts aren't a good idea. The decision should be made jointly by the information security team, the system administrators, IT management, and key users. Once a decision is made, it would be appropriate to document the decision in the form of a policy governing the use of shared accounts that outlines when they are appropriate to use as well as a guide stating when and how it should be accessed. This policy might also address the issue of what systems within your organization may have a shared account on them. For example, if the account is going to be used for internal batch processing it would be wise to not setup the account on your externally exposed servers as that could provide an attacker with a valid account name on your internal systems. A sample policy is provided in the appendix.

Securing Shared Accounts

Securing shared accounts is, in a lot of ways, similar to securing an individual user's account although the scope of some of the issues, such as password management, becomes more complex.

Securing Unneeded Shared Accounts

If you determine that your organization does not have a need for an anonymous account, then you have several options for securing them. First, you can not create them or delete them if an application or operating system does not require them. Another option is to disable

accounts if they must exist but are not going to be used. An example of this would be the guest account on Windows servers is not accessible by default and "it is a good idea to leave it that way since it can't be deleted." (Strebe, p. 331)

Determining Account Permissions

If, on the other hand, you determine that a shared account is needed then next you need to determine what access is needed by this account. If anonymous accounts are required by the system then one option is "to change the default anonymous user account name" (Strebe, p. 333) and instead "[u]se non-standard names" (Venner). For example, instead of 'anonymous' or 'guest' you could use 'unknownuser'. You might want to leave the common named account setup but just disable it so that you can audit it for attempted hacking attempts.

There is no simple and fast answer as to what you do to secure these accounts -- the answer is dependent on your environment and organizational needs. One basic rule of thumb for securing shared accounts, which should be applied to all accounts, is the principle of least privilege – give the account just enough access to do what it needs to do. There are several questions that should be asked when trying to determine how to apply this principle. Three of these questions are:

1) Should the account be a member of the default system groups? In the case of an UNIX system all accounts are, by default, members of the group "other" and in Windows accounts are a member of the group "Everyone".

2) Will the default permissions be enough to let the intended user access the data you want them to see? If not, you will need to add it to other groups that have the correct permissions or create a group that does.

3) Will the default permissions be too much? In some cases, data that you don't intend the shared account to access might be inappropriately secured itself. If there data of a sensitive or confidential nature that is accessible by the account then this will effect how you secure the account – or his discovery of inappropriately secured data might cause another project to be started that will secure the data appropriately.

Another suggestion would be to limit what servers the account can exist on. For example, an account used only by internal users to do work on internal servers should not be setup on your external servers where the account name could be more easily compromised. Your organization might decide to have a shared account on external and internal servers named differently with different user ids.

Account Access

After you have decided that a shared account is needed and what access it should have, a decision also needs to be made as to who should have access to the account and when the account should be used.

Hopefully, it should be straight forward because, simply put, if someone doesn't need access to the shared account then they should be not given the information necessary to access the account. In the case of 'root' or 'administrator' accounts, only the system administrators that work with the host in question should be given the ability to login to the account. If it is an account that will be used for batch production processing then it would probably be best to only let your production support staff access the account and not provide users or development staff to gain access. If it is the "IUSR_computername" account on a Microsoft IIS server, then access is normally unrestricted because the account is used by anonymous browsers of the website(s) hosted by the server.

Determining when a shared account should be used will probably take a bit longer to decide. A rule of thumb for this decision is that the account should only be used when a personal account can not be used. For example, if a user has access to review a file using their own accounts access then they should not use the shared account for this task.

Since there are multiple ways to access the account determining the best method for certain tasks should also be looked at.

Direct Logins

Should the shared account be accessed via a direct login to the account, which allows a user to work at the account itself? In the case of anonymous accounts, like "IUSR_computername", this is the purpose of the account – it allows a guest to access the system and perform certain tasks such as looking at your website. In the case of 'root' or 'administrator' access, there are times when a direct login is required such as when the system is being initially created. Your organization may decide that it is appropriate to restrict direct logins to the 'root' or 'administrator" accounts such that you must be at the system's console. For other types shared accounts, should a user be accessing the account directly? The risk in this is that there is minimal logging of who is accessed the account and what they did during their login session. There are alternatives to allowing direct logins.

Account Cloning

One way to minimize the number of direct logins to a shared account and to provide a better audit trail for who is accessing the account is the use of a cloned account instead of a shared account as much as possible. A cloned account has most, if not all, of the same permissions and access as the shared account but the cloned account is associated with a particular individual. Your organization might determine that this is an appropriate method for allowing administrative tasks to be completed on your server as it allows for better auditing of what person is doing what. For example, your organization might have several Windows-based servers (NT, 2000) that have an "administrator" account. Instead of using this account, you can create unique accounts

for each administrator that is a member of the “administrator” group on the server. This account should not be their normal user account but a separate account that has the higher level of permissions. If this is an option that you choose for your organization and the administrator’s name is Jane Doe then her regular user account (that she uses for e-mail, writing troubleshooting procedures, and non-system administrative tasks) might be “jane_doe” but her account for administrative tasks might be “jane_doe_a” or “jdoe”. It is best to not include “administrator”, “admin”, “super user”, or other such labels in her administrative account as this would allow someone with malicious purposes to easily identify “jane_doe_admin” as an account to try to access as it probably has more system access than standard account.

Switching Effective User IDs

On UNIX systems, there is a command called `su` that allows a user to login as themselves and then switch their effective user id to that of another account. The format of this command is “su [option] [user] [shell_args]” and “[i]If no *user* is specified … become a superuser”. (Robbins, 164)

This is good for administrators on these servers, because John Public could log in to the server as “jqpublic” and then use the command `su` or `su root` to become ‘root’ and perform administrative tasks. This also works well for non-administrative accounts such as an account used for batch production processing as it allows the system administrators to track what user is using, or attempting to use, a shared account.

The benefit of requiring the use of `su` is that the system maintains a log called /var/adm/sulog, by default. The benefit of this file will be discussed later as part of the auditing of the account access.

One time when a shared account must be accessed directly instead being accessed via an `su` login is when the password needs to be changed. This is necessary because “[t]he passwd command changes the password … associated with the user's login name” (Christias, passwd) and since the user’s login name is not the same as that of the account that is have been accessed via `su` the password can not be changed. This does not apply when the account accessed via `su` is ‘root’ due to the fact that this changes the effective user ID to zero and “[o]ne whose effective user ID is zero is called a super-user” and “super-users may change any password” (Christias, passwd).

Using `sudo`

Another option on UNIX hosts is the use of the `sudo` command. This command allows a user to execute commands as ‘root’ or another user based on specifications made in /etc/sudoers. This is a configuration file that specifies which user or group can submit a command and have it run as another account. This is useful as the user never gains access to the other account but is required to re-authenticate their identity by providing their own password. Depending on the setting

in the sudoers file, this authentication may be required for each `sudo` command ran or cached for a period of time. If you allow the authentication to be cached, you will want to ensure that this is only done for a reasonable period of time, such as five minutes, so that there is a limited period of time that `sudo` can be used before reauthentication occurs. This will help preclude someone from using the session to run `sudo` if the authenticated user leaves their login session unsecured.

/etc/sudoers is a formatted file that, basically, is a "list of which users may execute what" as either 'root' or the account specified in the list (University of Florida, "Sudoers"). This allows `sudo` to be very configurable as it allows you to create aliases of multiple users or multiple commands. For a listing of all of the options, a good reference is the Unix system manual (man) page for "sudoers". The man page clearly recommends that "[t]he *sudoers* should always be edited by the visudo command" (University of Florida, "Sudoers"). Using `visudo` prevents multiple users from editing the file at one time and also does syntax checking of entries that are being made. Another benefit of using `visudo` is that it "will not save the changes [to the sudoers file] if there is a syntax error." (University of Florida, "Visudo")

Account Ownership and Password Management

It is advisable to have a user or small group that is responsible for owning a shared account. This might be the system administrators or it might be a handful of trusted users. By having an owner associated with the account, you can make them accountable for the management of the accounts password. The owner(s) should be held accountable for tracking which users are authorized to access the account, changing the account's password, and distributing new passwords to authorized users. Only those considered the account owner should be allowed to distribute the password to authorized users and should be the only ones that provide the password to newly authorized users. This allows the owner to authenticate the identity of the person receiving the password and to remind the user of basic security procedures such as not leaving their desktop, terminal, or console logged in with the shared account – especially if they can't secure access to the session.

The account owners should maintain a list of users that have access to the shared account and publish this list at least the system administrators and the information security team. This listing should also include a list of which host(s) the account exists on. This serves to allow the account owners to ensure that they know which users need the password when it is changed and also provides a list that others can reference when auditing the accounts usage.

Password Changes

Passwords on shared accounts should be changed on a regular basis. The recommended frequency of these changes will vary depending on your organizational policies. It could be the same frequency as password changes for regular user accounts but might need

to be changed on a more frequent basis. In some cases, your organization might opt to change it less frequently because it is used on multiple hosts but should probably be done at least every 180-days. The password should also be changed immediately if someone gains unauthorized access to the account or someone with authorized access no longer has a need to access the account or leaves your organization. Failure to change the password to a shared account provides a backdoor entry point for disgruntled current and former employees.

New passwords for shared accounts should follow your organizations normal password policies, which hopefully require the use of strong passwords that contain mixed-case letter, numbers, and special characters. In some organization, this might allow for passwords such at "Happy_1" and in others it might require the use of a more cryptic password. When the password is distributed it should be done via a secure method such as distributed in person instead of having it sent via e-mail or shared over a phone or in the middle of the office.

.netrc Files

On UNIX systems, it is possible to create and utilize .netrc files. These files contain space-delimited records in the form of

machine *hostname* login *username* password *userpswd*

and are used to facilitate the automatic login to remote systems via FTP. This allows for users and batch processes to initiate FTP sessions with hosts listed in their .netrc file without being prompted for their username and password but are still authenticated to the server. These files should always be created with permissions of 600 (-rw-------) so that only the account that owns the file may view its contents. For shared accounts, this file needs to be handled very carefully to ensure that it is not incorrectly modified and that it only contains account information within it that everyone with access to the shared account is authorized to view. Due to the fact that .netrc files provide authentication information to other hosts, it is advised that .netrc files not be allowed on any servers that are exposed to and accessible from the Internet.

Account Auditing

All user accounts should be audited per the policies and procedures of your organization. This should also be done for shared accounts but the auditing should be more thorough due to the fact that multiple individuals can access the account.

The first piece of information that should be audited is direct login access of the account. On Microsoft Windows-based systems, this is accomplished automatically by the system with all logins being logged to the Security log. This log is viewable using the "Event Viewer". This log records such information as type of log entry ("Success Audit" or "Failure Audit"), the user, the event ID, the category (such as "Logon/Logoff", and the date and time of the event. The Event Viewer allows you to filter this information based on certain users, events, or other criteria. Microsoft's TechNet website provides good information on viewing and filtering the

Security Log using the “Event Viewer. To view logs with “Event Viewer”, you do not need to be an administrator which allows your organization to have users that are not administrators review system logs. Separation of duties is a good practice as those monitoring security should not necessarily be the same people that are configuring permissions. One nice feature of “Event Viewer” is that it allows you to review logs from remote systems, “[h]owever, you must be a member of the Administrators group on the target computer to open the Security log, or any log when targeting a remote computer.” (Microsoft, “View”.)

On UNIX systems, this information is likewise logged automatically. If a direct login is successful then this information is logged to the /var/adm/wmtpx file. Information from this file can be access using the command `last` which will display all logins by all users since the log was started or last emptied. For a display of direct logins by a particular user, the command can be ran using the form `last username` and then only logins by that user will be displayed. This will display the username, the terminal they that they connected from, the host that they connected from or a blank if it was from the console, the date of the login, the time of the login, and either the time of logout and duration of the connection or the text “still logged in”. Any user may run the `last` command to view all of the data within /var/adm/wmtpx or may filter it for a specific user by issuing the command in the form `last username`. It is also possible to have all UNIX system logs written to a central “loghost” in addition to or instead of the host being accessed. The location of the “loghost” is specified in the /etc/hosts file.

For UNIX systems, the second piece of information to audit is the /var/adm/sulog as this is the log for all `su` attempts to the account. sulog is a space-delimited file that includes the date (“in ‘mm/dd’ format”) , the time (“in ‘hh:mm’ format”), a single character success or failure indicator (“’+’ means it succeeded, ‘-‘ means it was disallowed”), the terminal that was used during the `su` attempt, and “the usernames involved” with a fomat of user1-user2 where user 1 is the “invoking username” and user2 is the user that user1 is becoming (Christias, sulog). This allows system administrators to not only track successful logins via `su` but to also identify users trying to access accounts via su that they might not be authorized to access.

Both the /var/adm/wmtpx and /var/adm/sulog files should be reviewed on a regular basis. First, this will allow you to verify that only authorized users are accessing the account and that they are only accessing it from proper locations. Also, this allows you to develop a pattern of normal usage so that you know if there are any anomalous logins that need to be investigated. For example, if Sally is accessing the account but you know she shouldn’t have the password then you should find out how she got the access and what she is doing with the account. Likewise, if Joe is an authorized user but is accessing the account from a host with a foreign hostname but has been at his desk in Indiana all week then you might have just found out that the shared account, Joe’s account, and your system are compromised.

Your organization might want to formalize the auditing of shared accounts by developing a policy governing the auditing of shared accounts. A sample policy for the auditing of shared accounts can be found in the appendix.

Taming a "Wild" Shared Account

What happens if you find out that a shared account exists and that it has not been secured? What about if you find out that the account has had the same password for years or that the password is widely known by people that have no need to ever access the account? What do you do if you find out that the password has been, for all practical purposes, essentially published to the world? The ideal answer is that you need to secure the account immediately and can do so with just a few quick key strokes or mouse clicks. In reality, this will probably not be the case.

This section is not intended to be step-by-step guide case study about how to secure a shared account but rather is included to provide you with some guidance on steps to take to successfully secure an existing account. The scenario is a generalization but is based on a real world project organized and implemented by the author.

Background

An organization, known as "The Company", had created a shared account to be used by their batch processing group to facilitate the running of automated processes on multiple platforms, including multiple UNIX servers with the transfer of data to Microsoft Windows-based hosts. This account was called "batch" and was intended to be used for batch production processing. All batch processes ran under this account and connected to other hosts when transferring files via FTP using this account. The Company felt that this was a beneficial and appropriate use of a shared account and that it worked well in their environment. They gave this account access to modify production data while restricting access to this data by regular user account. Additionally, this provided a way for processes to be run during off-hours without having to run the process as 'root' which meant that batch processing could run but that there would be less concern about a batch process doing actions that only superusers should do – such as rebooting or reconfiguring the system or modifying user accounts.

After a few years, the intended and actual purposes differed. The account name and password were known by all production support staff, system administrators, and developers. Developers we using the 'batch' account for testing new code or making modification to existing code by passing "The Company's" change control procedures. It was hard coded into batch processes that ran FTPs and, because batch processes were readable by everyone, everyone with access to the host could obtain the account's password. The password had never been changed from its initial setting years before resulting in fact that hundreds of current and former employees knew the password to the account.

Project Inception and Analysis

Every project must have a starting point and a project to secure a shared account is no different. The first step of a project to secure a shared account should be to gain buy-in on the project from your organization's information security team, internal IT auditors, system administrators and IT management. These are the groups that are responsible for ensuring the safety and security of your computing environment. By presenting them with both the benefits of and need to have a particular shared account as well as the risk and vulnerabilities of the account, you will normally receive the project buy-in and support that you need. This is especially the case if you have an account that is, for instance, critical to the batch processing of your financial systems but that people without the authorization to view the raw financial data could actually not only view the data but modify it as well. I found that this support was very helpful because when I encountered an individual that was resistant to securing the 'batch' account, I was able to suggest that they talk with one of these groups as this project was support by them. Often times, just showing that this wasn't something I was doing on my own was enough to lessen or end the resistance.

Once this buy-in has been achieved and you have been given the time to work on this project, you need to conduct a thorough analysis of the account. What is it being used for currently? What should it really be used for? Who has access to the account that shouldn't? Who should have access to it that doesn't? How is the account being accessed?

Project Execution

The next step was to inform the users of the account that a change was going to be made. This was a good time to explain the risks of the vulnerabilities that you have found. Often times, people are more willing to accept the change if they understand the reason behind the change. This was especially important if the person I was speaking with ws going to lose access to the account or have to modify their use of the account. If they understand why they shouldn't access the account then they usually agreed with the change. I found that by explaining the reasons for the change and showing that the loss of access to the account would not impact their ability to do their work, they were usually agreeable to the change.

As part of your analysis, you should have found out how the account is being accessed. If the account is only being accessed interactively by users then there is little risk that your batch processing would be effected. However, if the account is used by batch processes and those processes access other hosts, then you need to make sure that your change to secure the shared account won't negatively impact your batch processing. As stated earlier, I found batch processes that used FTP to move files between servers with the account name and password hard coded into the process – and the code was accessible for viewing by everyone with access to the system. In order to remove these hard coded passwords, I had to find an alternative that would allow batch

processing to not be impacted by the change. After discussing options with the project team, I setup .netrc files for the 'batch' account on the platforms where there were batch processes running FTP sessions. I then worked with the process batch development and support teams to remove the hard coded account information from the code. This proved to a very valuable and time saving step down the road as it made the actual password change easier as the password only needed to be updated in the .netrc files and not in several dozen pieces of code. Also, this eliminated the issue of everyone on the system being able to view the shared account's password.

Once all of the preparatory work has been completed, the next step was to change the password. This applied security to the account by preventing unauthorized users from accessing the account. When this was done at "The Company", dozens of people that had known the old password lost access to the account immediately but because of the steps taken before the password change there was not a single instance of an unauthorized user complaining about no longer being able to access the account.

Establishment of Policies and Procedures for Account

The project team also set up two policies concerning the 'batch' account. By setting up policies, we formalized our decisions and are able to reference those policies when issues are raised by someone that doesn't agree with how the account is secured and administered. At the time this was done, the policies were not documented in written form and distributed but merely agreements reached and supported by the different areas represented by the project team (system administration, IT management, the information security team, and the production support team).

The first policy that we set up governed access to the account. It stated that the only people that would access the account were those on the production support team. This was a group of about a dozen individuals – well below the very inappropriate number of individuals that previously had access to the 'batch' account. This group did not include developers that support the batch processes because they had the authorization as themselves to view all of the data and files that they needed to access without utilizing the 'batch' account. It also stated that the 'batch' accounts password would not be hard coded into any processes. The final statement in this policy was that the account would not be accessed via a direct login but rather would be accessed only by `su batch` after an authorized user had authenticated their identity by logging into their own account.

The second policy governed the administration of the account. While the production support team would be the owner of the 'batch' account, only three people within that team would be authorized to administer the account and manage its password. This provided accountability for changes to the account. This policy also specified that the password for the account would be changed on a regular basis and

also as needed due to such events as an authorized user leaving the company.

A document was also created that lists the steps necessary to change the accounts password. Such documentation is helpful in the event of someone new taking over support of the account as well as to help current account administrators to ensure that they have made all of the necessary changes to mitigate the risk of missing a minor task that could be forgotten.

Project Conclusion

Once a shared account is initially secured, I was able to officially close the project but in practice, the real work had just begun. There are the ongoing account administration tasks such as password changes. There are also times where new processes or new information lead to a review of the accounts security and create opportunities to further enhance the level of security on the account. If you find and secure one account, you might, perhaps, fine that there are other shared accounts in your organization that could be secured in a similar manner to reduce the number of users that access them. Or perhaps there are other methods for users to accomplish a task instead of using a shared account, for example can a task be accomplished by using `sudo` as previously discussed so that the user doesn't need to login to the shared account but could still have the task accomplished with the permissions of the shared account.

Conclusion

In conclusion, this paper has defined shared accounts and discussed the benefits and risks of allowing shared accounts to be used within an organization. Alternatives to shared accounts and methods for increasing security on shared accounts have also been discussed. The paper then discussed, as a high level, a real world project that secured a shared account and showed some of the issues involved in achieving a successful conclusion of such a project. It is hoped that this paper has provided you with information that will help you to better secure your organization as well as to have raised questions about your organization that you can research and address, if necessary.

Shared Account Policy

1.0 Purpose

The purpose of this policy is to help ensure the security of <Organization's Name>'s computer systems by outlining the use of shared accounts.

2.0 Scope

This policy applies to employees, contractors, consultants, temporaries, and other workers at <Organization's Name>, including all personnel affiliated with third parties. This policy applies to all equipment that is owned or leased by <Organization's Name>.

3.0 Policy

3.1 Anonymous/Guest Accounts

Anonymous/Guest accounts are not permitted to be setup on systems owned or leased by <Organization's Name> with the exception of accounts required by an operating system or third party application. Such accounts shall be secured to provide the minimum amount of access required for the operating system or third party application to function.

3.2 Temporary Employee Accounts

Temporary employees will have a unique account setup for their individual use be allowed access to systems owned or leased by <Organization's Name> such that they are given the minimum access required to perform their job functions. Such accounts will be used only by the temporary employee that the account was setup for, these accounts will not be reused, and they must have a password that meets the specifications outlined in the password policy of <Organization's Name>.

3.3 Administrative Accounts

Administrative, or superuser, accounts will be setup on systems owned or leased by <Organization's Name> as required by the operating system. Such accounts will be secured with a password that meets the specifications outlined in the password policy of <Organization's Name>.

3.4 Batch Processing Accounts

Batch processing accounts will be setup on systems owned or leased by <Organization's Name> as required by business needs with the approval of the <Production Support Team> and <Information Security Team>. Such accounts will be secured with a password that meets the specifications outlined in the password policy of <Organization's Name>.

3.5 Approval for Shared Accounts

The setup of a shared account shall be done only after the account

has been approved by <Name of individual(s) or teams(s) authorized to approve shared accounts – could include the InfoSec team, system administrators, and IT management)

4.0 Enforcement
Any employee found to have violated this policy may be subject to disciplinary action, up to and including termination of employment.

5.0 Revision History
Initial Revision Published on <DDMMMCCYY>. Written by <Author Name>. Publication Approved by <Manager Name>.

Shared Account Access Policy

1.0 Purpose

The purpose of this policy is to help ensure the security of <Organization's Name>'s computer systems by outlining the acceptable methods for users to access the shared account, <Account Name>

2.0 Scope

This policy applies to employees, contractors, consultants, temporaries, and other workers at <Organization's Name>, including all personnel affiliated with third parties. This policy applies to all equipment that is owned or leased by <Organization's Name>.

3.0 Policy

3.1 Authorized Users

Only authorized users shall access, or attempt to access, the shared account <Account Name>. An electronic list of users authorized to access the <Account Name> account is available at the following location <UNC location of file including path and file name>. This list shall be maintained by <List of Account Administrators or UNC location of electronic list>.

3.1.1 Loss of Authorization

An authorized user will lose authorization to this account in the event that they leave the employment of <Organization's Name> or move to a new position within <Organization's Name> where they no longer have a business need to access the account in performance of their work.

3.2 Unauthorized Users

Any user not expressly listed as an authorized user under section 3.1 of this policy is an unauthorized user of the <Account Name> account and is forbidden from accessing or attempting to access the <Account Name> account.

3.3 Authorized Access Methods

3.3.1 Interactive User Access

All interactive user access to the <Account Name> account shall be made in the following manner, <Statement of Acceptable Access Method, such as "on a Unix system, the user shall login to the system as themselves and then switch to the <Account Name> account by using the `su` command>.

3.3.2 Non-Interactive Access

Non-interactive account access, such as by a batch process, shall be made in the following manner, < Statement of Acceptable Access Method>.

4.0 Enforcement

Any employee found to have violated this policy may be subject to disciplinary action, up to and including termination of employment.

5.0 Revision History
Initial Revision Published on <DDMMMCCYY>. Written by <Author Name>. Publication Approved by <Manager Name>.

Shared Account Administration Policy

1.0 Purpose

The purpose of this policy is to help ensure the security of <Organization's Name>'s computer systems by outlining how the shared account <Account Name> is to be administered.

2.0 Scope

This policy applies to employees, contractors, consultants, temporaries, and other workers at <Organization's Name>, including all personnel affiliated with third parties. This policy applies to all equipment that is owned or leased by <Organization's Name>.

3.0 Policy

3.1 Authorized Administrators

Only authorized administer shall adjust the configuration of the shared account <Account Name> or change this account's password. An electronic list of users authorized to administer the <Account Name> account is available at the following location <UNC location of file including path and file name>. This list shall be maintained by <List of Account Administrators or UNC location of electronic list>.

3.2 Unauthorized Users

Any user not expressly listed as an authorized administrator under section 3.1 of this policy is an unauthorized administrator of the <Account Name> account and is forbidden from modifying the shared account <Account Name> or from changing this account's password.

3.3 Administrative Tasks

3.3.1 Account Configuration

Only authorized administrators of this account may modify the accounts settings, including but not limited to default printers, login shell, and environmental settings.

3.3.2 Password Changes

The administrator(s) of this account shall change the accounts password at least once ever <Number> days. Exceptions to this period will be granted only by the approval of <Name of Person(s) or Team(s) that can grant extension>. The account's password will also be changed in the event that the account is compromised and/or an authorized user loses authorization to access the account.

4.0 Enforcement

Any employee found to have violated this policy may be subject to disciplinary action, up to and including termination of employment.

5.0 Revision History

Initial Revision Published on <DDMMMCCYY>. Written by <Author

Name>. Publication Approved by <Manager Name>.

Shared Account Audit Policy

1.0 Purpose

The purpose of this policy is to help ensure the security of <Organization's Name>'s computer systems by outlining how access to the shared account <Account Name> is to be audited.

2.0 Scope

This policy applies to employees, contractors, consultants, temporaries, and other workers at <Organization's Name>, including all personnel affiliated with third parties. This policy applies to all equipment that is owned or leased by <Organization's Name>.

3.0 Policy

3.1 Authorized Auditors

Only authorized authorized auditors of the shared account <Account Name> shall audit the access of this account. An electronic list of users authorized to audit the <Account Name> account is available at the following location <UNC location of file including path and file name>. This list shall be maintained by <List of Account Administrators or UNC location of electronic list>.

3.2 Unauthorized Users

Any user not expressly listed as an authorized administrator under section 3.1 of this policy is an unauthorized user of the <Account Name> account and is forbidden from accessing or attempting to access the <Account Name> account.

3.3 Audit Tasks

Audit tasks include, but are not limited to, reviewing of account access via direct logins, FTP, and users that switched their effective user id to that of this account as well as the commands executed during such access.

3.4 Audit Frequency

The shared account <Account Name> shall have it's access and use audited on a regular basis of no more than every <Number> of days.

3.5 Audit Anomalies

Any anomaly found in the audit of the shared account <Account Name> shall be reported to <Name of Person(s) and/or Team(s)> and investigated. Any unauthorized access or use shall be addressed following the appropriate access and/or use policy in effect as <Organization's Name>.

4.0 Enforcement

Any employee found to have violated this policy may be subject to disciplinary action, up to and including termination of employment.

5.0 Revision History
Initial Revision Published on <DDMMMMCCYY>. Written by <Author Name>. Publication Approved by <Manager Name>.

Bibliography

____. Administrating Security for Solaris™ 2.x Operating Environments Student Guide, Volume 1, Revision B. Sun Microsystems. 2000. Pages 6-1 - 6-22.

*Christias, Panagiotis. "UNIX man pages : passwd (1)". 1994. As published by the Mississippi Center for Supercomputing Research and University of Mississippi. URL: http://www.mcsr.olemiss.edu/cgi-bin/man-cgi?passwd+1

*Christias, Panagiotis. "UNIX man pages : sulog (4)". 1994. As published by the Mississippi Center for Supercomputing Research and University of Mississippi. URL: http://www.mcsr.olemiss.edu/cgi-bin/man-cgi?sulog+4

Microsoft. "Filter Events". "Microsoft TechNet". 2003. URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/windowsserver2003/proddocs/standard/nt_filteringevents_how_ev.asp

Microsoft. "View an event log". "Microsoft TechNet". 2003. URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/windowsserver2003/proddocs/standard/nt_viewdetail_how_ev.asp

Orrell, Carolyn and Eric Maiwald. "HIPPAhelpdesk - Shared Network Logins" HIPAA Advosry Forums. URL: http://www.hipaadivsory.com/forum/messageview.cfm?catid=26&threadid=417

Robbins, Arnold. UNIX in a Nutshell: A Desktop Quick Reference for SVR4 and Solaris 7. Third Edition. O'Reilly and Associates, Inc. 1999. Page 164.

SANS Institute. "Acceptable Use Policy". The SANS Security Policy Project. URL: http://www.sans.org/resources/policies/Acceptable_Use_Policy.pdf

SANS Institute. "Audit Policy". The SANS Security Policy Project. URL: http://www.sans.org/resources/policies/Audit_Policy.pdf

Strebe, Matthew, and Charels Perkins. MCSE: Internet Information Server 4 Study Guide. 2nd Edition. Sybex, Inc. 1998. Pages 159, 331-333

Postel, J. and J. Reynolds. “File Transfer Protocol (FTP)”. The Internet Engineering Task Force. October 1985. URL: http://www.ietf.org/rfc/rfc0959.txt

Puschitz, Werner. "Securing Red Hat Linux 7.2, 7.3, and RedHat 2.1 Advanced Server". 23AUG2003. URL: http://www.puschitz.com/Security.shtml

*University of Florida. “Sudo Manual”. The Open Systems Group. URL: http://open-systems.ufl.edu/docs/sudo/sudo.html

*University of Florida. “Sudoers Manual”. The Open Systems Group. URL: http://open-systems.ufl.edu/docs/sudo/sudoers.html

*University of Florida. “Visudo Manual”. The Open Systems Group. URL: http://open-systems.ufl.edu/docs/sudo/visudo.html

Venner, Lorraine. "HP-UX Computer Security Checklist". 2001. http://www.idiom.com/securecheck.html

Watters, Paul. Solaris 8 Administrator’s Guide. O’Reilly & Associates. 2002. Pages 103-110

Yale University Information Security Office. "Network Security Checklist for Unix Workstations". URL: http://www.yale.edu/its/security/Procedures/Securing/Unix/

*If installed, all man pages should also be accessible on UNIX systems from the command line by typing `man *command*`, where *command* is the name of the command you are trying to find information about (i.e. `man passwd`).