Global Information Assurance Certification Paper

Implementing the "Just-enough Privilege" Security Model

Tom Martzahn
August 22, 2003

GSEC Version 1.4b

Abstract

This paper discusses some of the challenges associated with migrating a large, widely distributed Windows NT environment with widespread administrative access for the application and server support personnel to a native Windows 2000 environment which embraces the philosophy of the "Just-enough privilege" (JeP) security model to complete assigned job responsibilities. I'll define the concept of Just-enough Privilege within the scope of this migration, briefly describe the old environment, outline some perceived benefits of moving to this security model, discuss the challenges and roadblocks of implementing the JeP Security model, and provide some real-life examples of how to limit widespread administrative authority on Windows 2000 servers for tasks that are commonly perceived to require administrative privileges to complete. This paper depicts experiences with migrating from a Windows NT environment, but the strategies discussed to implement JeP can be applied to an existing Windows 2000 environment which doesn't enforce the JeP philosophy.

Definition of JeP

At a high level, it is pretty obvious that the Just-enough Privilege (JeP) Security Model is nothing more than another name for the principle of "least-privileged" access. In fact, throughout most of this implementation, it was called the least-privileged security model and it still is within some areas of the organization. The prevailing thought to rename the strategy at a high level is an attempt to eliminate the negative connotations associated with the word "least", and replace it with the more positive connotations associated with the phrase "Just-enough". To some, I'm sure it sounds like nothing more than semantics and nitpicking, is the glass half-empty or half-full? But as you will find out as you read on, one crucial aspect of a successful implementation of JeP is executive sponsorship. Without executive sponsorship, you will not implement JeP in your organization. Lessons learned through this implementation have demonstrated that this minor upgrade in semantics will lead to a smoother overall acceptance of JeP at all organizational levels.

The exact definition of the "Just-enough Privilege" Security Model, or JeP, can be pretty extensive depending on the context of that definition within a large organization. For our purposes here, let's define the JeP security model as:

To grant application and server support personnel the appropriate access rights, and only enough access rights, on a Windows 2000 server to complete specific task assignments as defined in requirements to support an application or server.

The word to remember in that definition as you move forward is requirements. Requirements definition coming from application and server support personnel is another crucial element to a successful JeP implementation, yet very difficult to attain in many cases due to a lack of understanding or training in the application or server technology and a misplaced understanding of task ownership within the IT support organization.

Implementing the JeP security model is just one of many facets of a Windows 2000 deployment and migration strategy. Out of context of this paper are the Active Directory design requirements of the Windows 2000 infrastructure, specific migration requirements, and workstation requirements.

Life before JeP Implementation

The network architecture at the outset of this project consisted of more than 20,000 NT 4.0 Enterprise Edition servers. All of this was contained within more than 20 NT account domains using a mutated multi-master domain model, multiple one and two-way trusts, and a couple of resource domains built into the mix. Further details would only exacerbate the obvious problems faced by a very large number of network and server administrators, application support personnel, and help desk staff.

The security model for the Windows NT server infrastructure provided administrative rights to just about anyone who needed to touch an NT server for support reasons, regardless of the tasks they needed to perform for this support. Task ownership and application support requirements were poorly defined and documented or non-existent in some cases.

This is not an all-inclusive list, but some of the major problems with this particular model included:

1. Our servers were configured, tested, and deployed with a well-tested but not very well documented base profile configuration. This base configuration included security settings, monitoring software, backup software, anti-virus software, and a host of additional hardware and software support tools. Application support folks were granted administrative privileges on servers hosting their application, and made

changes to this best-known configuration over time based on what suited them. When problems arose, getting back to a working and well-tested configuration was difficult without a reload.

2. Support personnel, acting in good faith, did what they felt needed to be done to 'fix' the problem they were assigned to fix. Fixing this problem sometimes led to subsequent problems to be fixed by someone else. In some cases, a shared admin account was being used, so there was no audit trail of who made previous changes to the server and what those changes consisted of. In some instances, the fix for problem 2 was to undo the changes made to fix problem 1, thus leading to another break/fix opportunity for problem 1.
3. Anti-virus software was being stopped without notice in efforts to improve application performance.
4. Default administrator account passwords were being changed and distributed to others. Generic administrator accounts were being created to 'make it easier' to support the server.
5. Application/Service accounts (most with administrator level privileges) had passwords that were not complex, were unchanging, and distributed somewhat freely.
6. Change management processes were being circumvented or ignored altogether.
7. Server audit policy was being customized or turned off.
8. Unapproved and/or unlicensed software was being installed.

What is at risk with this architecture? Outside of the potential legal issues surrounding the use of unlicensed software, with so many having full administrative privileges to mission-critical servers, it provides the potential to easily violate the most fundamental principles of any practical security program. Best stated in a white paper posted on ITsecurity.com[1], a text book definition of any practical security program should include:

> *A security program is the frame work that helps organizations to maintain the **Confidentiality**, **Integrity** and **Availability** of their information assets. Meeting the business objectives of the organization and its customer's needs should be the goal of every security program. :*
>
> ***Confidentiality**: The protection of information from unauthorized disclosure (i.e. computer, industrial espionage)*
>
> ***Integrity**: Protection from unauthorized modification 9i.e. to pricing, process mode, deletion of critical information etc.)*
>
> ***Availability**: Protection of the accessibility of information when required (i.e. denial of service)*

[1] http://www.itsecurity.com/papers/trinity7.htm

Also at risk with this architecture was the potential for a dramatic increase in the Total Cost of Ownership for the Windows server platform. Defining an accurate TCO formula is elusive, but the most simplest of formulas as documented in Anil Desai's book on Windows NT Network Management shows management and support costs being almost 60% of the total costs over a three year period of a project implementation[2] , far exceeding the hard costs of initial deployment in terms of hardware and software.

It was a difficult environment for applications to share servers, as changes introduced into a system could be made without proper or formal testing of all applications residing on the server. This resulted in an increase in the number and length of problems calls taken by help desk personnel as changes made to fix one app caused problems for other apps on the server. One of the answers to remedy this was to deploy every application on its own server, leading to higher hardware costs and increased staff needed to properly support growing server numbers.

In addition to this, one must assess their comfort level with the growing concerns related to "the insider threat" and how it relates to wide-spread administrative authority:

The most serious breaches resulting in financial losses occurred through unauthorized access to insiders. Insiders represent the greatest threat to computer security because they understand the organization's business and how the computer systems work. Therefore, an insider attack would be more successful at attacking the systems and extracting critical information. The insider also represents the greatest challenge in securing your network because they are authorized a level of access to your network and are granted a degree of trust.[3]

This statement most successfully encompasses the hundreds of studies and white papers in existence describing the insider threat. Some of the resources that support the need to take the insider threat seriously are:

http://www.nwfusion.com/news/2002/130577_03-04-2002.html
http://rf-web.tamu.edu/security/SECGUIDE/Treason/Infosys.htm
http://www.computerworld.com/securitytopics/security/story/0,10801,57889,00.html

The Problem Statement

[2] Desai, Anil Windows NT Network Management, page 4, copyright 1999 New Riders Publishing
[3] Boston, Terry. "The Insider Threat." SANS. October 24, 2000. WWW.SANS.ORG

It was decided early on through a research project to build a native Windows 2000 environment from the ground up in parallel to our existing NT infrastructure, and provide a methodology for all users and applications to migrate onto this infrastructure over time. When our organization decided the time was right to begin the migration from NT to Windows 2000, many projects were activated concurrently to provide solution paths to meet specific goals and requirements.

One project was specifically designed to build a Windows 2000 Security infrastructure from the ground up in an effort to address the following problems stemming from the current NT infrastructure:

1. Support costs associated with supporting a very large and complex NT domain environment.
2. Availability and stability concerns caused by the inability within the architecture to enforce change management guidelines.
3. Security configuration management guidelines too easy to modify and work around.
4. Wide-spread and potentially unneeded administrative authority.

Additional considerations, both internal and external, were emerging to expose the need to address potential new risks from a security perspective. Our company was expanding into businesses that are more directly affected by specific state and federal regulatory practices, as well as the recently enacted privacy legislation such as the Health Insurance Portability and Accountability Act of 1996 (HIPAA)[4], the Gramm-Leach-Bliley Act of 1999(GLB)[5], and the California Civil Code 1798.82 (SB 1386)[6] initiative. All of this legislation, designed to provide 'best practices' for how companies handle sensitive personal data considered 'owned' by each of us, appears to be signaling a new paradigm for Information Security Officers to examine. Is the day coming where legislation will demand companies document all employees who have access to this sensitive personal data and the reason for this access? Not knowing the answer, without JeP, companies will find it very difficult to comply should the answer be yes.

Life during JeP Implementation

As stated earlier, a group of projects were initiated to plan, design, deploy, and migrate the organization to Windows 2000. This project was initiated to build the Windows 2000 security infrastructure from the ground up. The project team's first step was to take the problem statement and translate that into the objectives of the project. In essence, the problem statement pointed out one simple fact, too many people had too much access to make too many changes. That led to a very simple conclusion: Remove their ability to make these changes.

[4] http://www.hhs.gov/ocr/hipaa/
[5] http://www.senate.gov/~banking/conf/
[6] http://www.dmv.ca.gov/pubs/vctop/appndxa/civil/civ1798_82.htm

To support this decision, the project team did some research on what the industry was recommending for Windows 2000 implementations.

Section 1.4 of the SANS (SysAdmin, Audit, Network, Security Institute) Securing Windows 2000 Step-by-Step Guide version 1.0c recommends this:

Enforce the "Least Privilege" Principle:
The least privilege principle states that each user should be given the minimum amount of privileges on networked computers necessary to do their job. A user with unrestricted access to the entire network can cause catastrophic damage by spreading a simple "Trojan horse" virus. A network that has no precautions is a time bomb waiting to go off. [7]

Page 14 of the Security Project Cookbook from The Burton Group recommends the implementation of the Principle of Least Privilege by stating this:

Users, and the processes that act on their behalf, should only be given access to information that is required to perform their job. Enforcing the "principle of least privilege" greatly decreases the possible exposure and vulnerability to the IT environment. By enforcing this principle, the threats of malicious insiders and accidental user errors are significantly decreased. [8]

Even Microsoft, as stated in their publication "Security Operations for Microsoft Windows 2000 Server", recommends:

In a Windows 2000-based environment you can control very precisely the administrative rights your users have. You should ensure that you tightly define the scope of administrative rights that should be available to each member of your IT staff. No member of your staff should have more administrative access than is strictly required for their job. [9]

One common interpretation of all this material was the suggestion to limit those with administrative privileges, and to implement the principle of "least-privilege". With this information in hand, a high-level objective of the project was defined: Enforce the principle of Least-Privileged. This objective, at least on the surface, would appear to address the entire problem statement.

This takes us to what we thought was an important first step to a successful implementation of JeP. Define the access rights for the teams that needed to support the base server profile. The base profile (delivered by another project) can be defined as all of the components, software, and services that are required to be loaded on all servers deployed within the organization. And based on the components that make up this base server profile, what tasks can be defined to

[7] Copyright 2001, the SANS Institute.
[8] Copyright 2001, the Burton Group.
[9] Copyright 2002, Microsoft Corporation.

support these components, who within the organization owns these tasks, and what access rights are needed on a server to complete these tasks.

To do this, we loaded a server with all base profile components, gave nobody administrative privileges to this server, and asked each component owner to deliver requirements on what they felt they needed to do on this server to properly support their component. This was a mistake. There were approximately 20 components (anti-virus software, monitoring software, hardware and software support tools, etc.) in our base profile above and beyond the operating system, and 19 of the component owners responded that they required administrative privileges to support their component.

The granting of administrative privileges under JeP cannot be considered a requirement to be met; it must be viewed as a potential solution to meet valid and defined requirements. Stating that one needs administrative privileges to support a component is not a requirement, it is their perception of what is needed to support their component. Ensuring least privilege requires identifying the specific tasks that each user (or group of users) must perform when supporting a server component and determining the minimum set of privileges required in performing those tasks. The tasks are the requirements, the set of privileges required to allow the users to perform those tasks is the security solution. Granting administrative privileges can be considered an appropriate security solution if and only if testing has demonstrated the successful completion of the task requires administrative privileges.

Obviously, this wasn't what we were looking for. We were looking for specific services, directories, and registry entries they needed access to in support of their component(s). Our communication to these component owners was ineffective, and our methodology to assist in determining their requirements was non-existent. On top of that, many of these component owners disagreed with the philosophy of implementing the least-privilege principle. They questioned the benefits of this model, the time it would take to test and implement this model, and ownership of the model in general. Many of the component owners initiated an escalation process to their management to protest the proposed implementation of least-privilege. The project team had to circle the wagons, regroup and come out with a better way of reaching its objective.

This led us to the first real step to a successful implementation of JeP. The project team had to gain executive sponsorship at a senior management level in order to successfully reach its objective. Senior management wanted to know this:

1.) What are the benefits of the 'least-privileged' model?
2.) What are the costs, in money and time?
3.) What is the return on security investment (ROSI)?

This is where the project team decided to change from 'least-privilege' to 'Just-enough Privilege.' We found, through experience, as forward thinking as senior level management folks are, management fails to understand the need for security[10]. The positive concept of providing access for someone to do their job (JeP) was perceived much better at this level than the negative approach, taking someone's access away (least-privilege).

The project team came up with the following answers to the questions posed by management. The perceived benefits of JeP are:

1.) For financial services institutions subject to state and federal regulations, JeP has the potential to lay the groundwork for meeting and exceeding all current and future guidelines set forth in legislation protecting the privacy of personal information. HIPAA and GLB are examples of federal legislation, and California Civil Code 1798.82 (SB 1386) is an example of state legislation which affects how companies need to address how it handles sensitive personal information. In thinking outside of the box, one can only anticipate activity in this specific legislative arena to increase in the next few years.
2.) Enhance our ability to meet the most fundamental principles of any security program, maintain the confidentiality, integrity, and availability of the data assets that reside on these servers, by restricting who has administrative authority, which means restricting who has full control to these data assets on these servers to only those who need it.
3.) Improve performance, stability, and reliability of these servers, reduce the trouble calls associated with these servers, and lower the Total Cost of Ownership by restricting changes on each server to the area accountable for the component. Unwise, untested, unnecessary, and unplanned changes would be reduced and/or eliminated.
4.) Reduce exposure to risk. Over 70% of computer incidents occur from insiders. Implementing a program to reduce their access to only those tasks they need to perform was considered an important compensating control.

The costs, it time and money, could only be anticipated in these terms:

1.) The requirements analysis, design, and testing phases for every project implementing a component on a Windows 2000 server would increase to meet JeP security requirements.
2.) Total Cost of Ownership could potentially increase if access to a server was restricted to the point where it interfered with legitimate troubleshooting.
3.) General organization costs would increase as initial resistance to the JeP security model was expected.

[10] Chastain, Stacy. "Management Fails to Understand Information Security." SANS. July 31, 2003. WWW.SANS.ORG

The return on security investment (ROSI) will be the most elusive and intangible concept to get down on paper, but in order to gain executive sponsorship, you must attempt to illustrate positive numbers. As Eric Korofsky states in his article entitled "Insight into Return on Security Investment" appearing in the Secure Business Quarterly: "A company needs to assess the investment versus the chance of something occurring multiplied by the severity of the problem" [11]

Nothing could be closer to the truth yet so difficult to quantify, as there are so many unknown factors to take into account. While the investment can be more easily quantified in project costs to implement security, the cost of not implementing security, with variables like "chance" and "severity" being unknown, makes any equation too intangible to solve.

For our project, we responded to the ROSI question with this:

1.) We turned this security expense into a strategic investment by showing research studies that ROSI can be up to 21% if engaged in the appropriate project phase. [12]
2.) Using the research depicting the risk of insider attacks, we effectively displayed how the JeP philosophy could mitigate this risk.

This completed the first and most important step of implementing JeP. We won executive sponsorship and senior level management acceptance of this philosophy, with two stipulations: 1.) create a steering group to facilitate communication, own the JeP migration process, and provide an escalation path for issue resolution and 2.) Update corporate security policies and IT architectural standards and guidelines to reflect best practices concerning JeP. The importance of these two stipulations will become clearer as we move further into the implementation.

Now that the project team had executive sponsorship and senior level management acceptance of JeP, the next step was to get the "ammunition" in place to move the rest of the organization forward into the JeP model. The steering group was created to 'govern' the migration to JeP. It included the major stakeholders of the organization responsible for supporting the Windows 2000 infrastructure, Active Directory, Security, Network Operating System support, DNS support, and Operations support. The project team created the charter for this group, which included items such as:

1.) Generic administrator accounts will not be allowed.
2.) Separate administrator accounts will be created for those with administrator privileges.
3.) A documented authorization process to become an administrator.
4.) Restricted memberships for Schema Admins, Enterprise Admins, Domain Admins, and Administrator groups.
5.) The "Just-enough Privilege" security model will be in place by default.

[11] Secure Business Quarterly, Volume 1, Issue 2, Fourth Quarter, 2001.
[12] Secure Business Quarterly, Volume 1, Issue 2, Fourth Quarter, 2001.

6.) Defined ownership of all security-related configuration tasks.
7.) Process definitions for supported ways to access the registry and services running on a server.
8.) The approved trust model and a process to authorize changes to this model.

This is just a subset of the charter, but these items give you a flavor of the project team's intent to document processes, guidelines, and standards and to communicate these to the organization. The role of this group was to sponsor updates as they became necessary, authorize exceptions based on a valid business cases, and to validate and assign task ownership based on the current organizational structure.

Validating and assigning task ownership is the most important function within the overall process of defining application and server support requirements during JeP migration. Application and server support folks, by nature, are fearless in their ability to do the right thing in conquering any application or server problem, so many consider themselves 'owner' of all tasks on the server. In our organization, nothing could be further from the truth. Our organizational structure had teams of people responsible for specific server disciplines, security, operating system, anti-virus software support, performance, disk management and backups, software installation, etc. This steering group became the owners of assigning tasks to specific teams within the organization.

Overall, the project team created eight Information Technology architectural standards and guidelines for all project implementations to consider when deploying an application onto the infrastructure. These guidelines and standards ranged from password recommendations for application/service accounts to MTS access control standards to workgroup configuration standards (in essence, not allowing workgroup servers on the network); to standard install directories for applications and services.

This completed steps two and three of implementing JeP. Armed with strategies, standards, and guidelines to support the implementation, along with executive sponsorship and senior management acceptance, it was time to revisit the base server load and try again. This time, we developed a more concise and effective communication about what we are looking for:

1.) What services and/or applications do you install on a server?
2.) What directories do you need access to in support of your service or application.
3.) What registry entries are created and changed by your application or service?
4.) Document the steps you take to troubleshoot your application or service today.

Many component owners were better able to respond, but had reservations based on the following two concerns:

1.) Why do they have to do this? Are they not trusted to be administrators?
2.) What happens if we can't identify all of our support requirements?

You need to practice and practice your answer to number 1. It is most definitely not a question of trust. All of these folks are trusted individuals within the organization, and the effort to restrict their access to only the tasks they need to properly support their application(s) has nothing to do with trust or anyone's lack of technical skills to be an administrator. This reinforced the need to have documented corporate security policies and architectural guidelines in place. You need to tie this back to these enterprise security policies as mandated by legislative initiatives, privacy concerns, data integrity concerns, recommended industry best practices, and stability and availability goals and objectives. Over and over again you need to communicate these policies and the reasons for these policies, thus the need for corporate security policies at an executive level. Please repeat, it has nothing to do with trust and is not an attempt to pass judgment on one's technical skills.

To answer number 2, a process was created to invoke temporary emergency administrator privileges for component owners to specific servers hosting their component(s) in the event of an access problem which was preventing the root cause of a problem from being identified and service being restored. This access was time-boxed in 48 hour increments with a document of understanding (DOU) in place outlining specific tasks which the component owner could not perform. Most of these 'taboo' tasks were security configuration tasks, like clearing out logs, changing memberships in security groups, creating local logon accounts, deleting directories and files (except those owned by the application), installing software, etc. Within the DOU was an agreement to document all changes made to the server. A follow up process was also developed to review the component owners permanent access rights and make adjustments to these access rights based on root cause of the problem so emergency administrator access wouldn't be needed the next time this same problem happened. If the DOU was not signed in advance, access was not granted. This is where executive sponsorship and senior level management acceptance of JeP were so important, in support of this concept.

One of the tools we created to facilitate our deployment to JeP was a risk document template outlining the potential exposures of not enforcing the JeP principle, and forcing signed acceptance of the risk by the management accountable for not implementing JeP. Each risk has to be evaluated on a case by case basis, and weighed against the potential to expose proprietary company assets to unauthorized disclosure and/or place the organization at risk of litigation based on non-compliancy with state and/or federal legislation.

For example, if a server was hosting a database containing personal information protected by California Civil Code 1798.82 (SB 1386), and another application was being implemented on that server which had an application control console that wouldn't run if the user was not a member of the local administrators group on the server, it actually queried the local SAM on the server when launching (non-disclosure agreements prevent me from naming all of the vendors who have software architected in this fashion), the management sponsoring the proposed purchase of this application had to accept knowledge of and the risk of placing the company in potential violation of this California statute should the folks running this application console use their privileges to access this protected data within the database. Instead of signing off on and accepting this risk, management, in most cases was prudent enough to seek out alternative software or work with the software owner to implement a more secure design.

Life after JeP Implementation

Now that we are well into the deployment and migration to a Windows 2000 infrastructure using JeP (100% of users logging onto Windows 2000, 50% of all server profiles migrated, application and support staff access rights implemented using JeP), the chance to sit back and review just how successful we've been in addressing some of the risks of the previous environment was in order.

This is a short list of some of the things that have made our infrastructure much more secure:

1.) The number of those with administrator accounts on Windows 2000 has been reduced by over 60%.
2.) There are no generic administrator accounts created anywhere on the Windows 2000 network.
3.) Application and service account passwords are known only to a select group of security analysts and are changed on a routine basis. Some application and service accounts are even configured to run without admin privileges even though the software documentation states this as a requirement.
4.) Call volume numbers related to deployment and configuration issues have been reduced by almost 40%.
5.) Audit policy is now enforced via Group Policy and does not change.

Some things that still need to be addressed:

1.) There is far too much software architected without JeP in mind for the privileges needed to successfully execute on a Windows server platform. Too many application vendors are taking the easy way out by documenting admin requirements for its software to execute.
2.) Upgrading the technical skills of application and support staff to make the process of defining support requirements under the JeP principle less painful and time-consuming.

3.) The dynamic and constantly changing interpretations of existing legislative requirements, the vagueness of these initiatives, and the growing concern of new legislative initiatives are placing all companies at risk of potential litigation for non-compliance with legislation dealing with privacy issues, so organizations need to take proactive steps to mitigate these risks. Articles such as this: http://www.computerworld.com/databasetopics/data/story/0,10801,67883,00.html emphasizes the need to implement JeP.

And most recently, even though it wasn't available to use to validate the need for our organization to implement JeP, the ominous and most disturbing advisory coming from the Department of Homeland Security specifically recommending an organization:

- Restrict roles and limit privileges to ensure individuals do not have access or control beyond what they need to perform their authorized functions.[13]

This advisory alone serves fair warning that computer security as we know it could be shifting to a model even much more secure than JeP.

Real-life solutions.

Here are a few real-life solutions for tasks that are commonly perceived to require administrative rights to perform. Some of these details are specific to the configuration of the server and would need to be translated to your specific server configuration.

1.) Running PerfMon.

Many application support personnel need the ability to run PerfMon in the day-to-day support of their application. Give them this access:

- Change access to the directory containing the performance logs created by PerfMon.
- Full control to the registry key: HKEY_LOCAL_MACHINE\SYSTEM\CURRENTCONTROLSET\SERVICES\SYSMONLOG.
- Read, start, stop, pause, write, and delete privileges to the Performance Logs and Alerts (SysMonLog) service.
- Terminal Service access.

If this is a SQL server, then add this access:

- Read access to the directory that contains SQL, usually called the MSSQL directory.

[13] Non-copyrighted material obtained from http://www.state.ar.us/insurance/pdf/advisory_071803.pdf

- Read access to the HKLM\System\Services\CurrentControlSet\Services\ MSSQLSERVER\Performance Registry key.

2.) Com Developer Access to IIS Web Servers.

Want to give a large number of component developers the ability to develop and test their components on a limited number of shared IIS web servers without giving every one of them full administrative privileges to 'tweak' the system and step on each others toes? Do this for each COM+ developer for each IIS server (use global or domain local groups if you want):

- Place them into the Administrators role of the MTSAdmin application in COM+.
- Add them to the Administrators role of the System application in COM+.
- Give each developer "Query Value", "Set Value", "Create Subkey", "Delete", and "Read Control" to the registry hive that contains the component information. This is specific to your server configuration.
- Give each developer "Query Value", "Set Value", "Create Subkey", "Delete", and "Read Control" to the HKLM\Software\Classes registry hive.
- Change access to the directory that will contain their code.
- Stop/start/read to the IISAdmin service.
- Stop/start/read to the W2SVC service.

3.) Access to Disk Administrator.

Need to give certain support personnel the ability to run Disk Management through Computer Management? Do this:

- Grant full control to the Logical Disk Manager Service.
- Grant full control to the Logical Disk Manager Administrative Service.

4.) Access to log on locally to a server via Terminal Services.

Want to give someone the ability to log onto a Windows 2000 server via Terminal Services in administrative mode without granting them administrative privileges to the server? Do this:

- Give them the Log on Locally right (preferably via group policy).
- Give them full control to the RDP-Tcp connection object using the Terminal Services Configuration MMC through Administrative Tools.

Executive Summary

Here is an executive outline of the step by step process to implement the Just-enough Privilege security access model.

1.) Gain executive sponsorship and senior level management acceptance by demonstrating the benefits of JeP.
2.) Update your corporate information security policies and guidelines manual to display the business reasons for moving to JeP.
3.) Update your IT architectural manual with standards and guidelines supporting the technical requirements for implementing JeP.
4.) Organize a team of stakeholders to serve as owners of the JeP security model for your organization and empower this group.
5.) Create a risk document for not implementing JeP and a process to introduce this document into each project implementation.
6.) Create a testing environment that can be used by application and server support personnel to define specific support requirements under JeP, not to be used for application testing.
7.) Update the technical skills of your IT support personnel to better understand the application(s) they support and the infrastructure that hosts these applications.
8.) Document and effectively and repeatedly communicate the benefits of JeP to all levels of the organization.
9.) Be firm, yet flexible, in timelines for defining requirements. After all, the business needs to move forward in deploying mission-critical applications which make the business run.

References

1.) The Encyclopedia of Computer Security. "Implementing a Security Programme for your Organisation." http://www.itsecurity.com/papers/trinity7.htm (21 April 2002).

2.) Desai, Anil. Windows NT Network Management. New Riders Publishing, 1999.

3.) Boston, Terry. "The Insider Threat." SysAdmin, Audit, Network, Security Institute, WWW.SANS.ORG, 24 October 2000.

4.) "Medical Privacy - National Standards to Protect the Privacy of Personal Health Information." http://www.hhs.gov/ocr/hipaa/ 23 July 2003.

5.) "Information Regarding the Gramm-Leach-Bliley Act of 1999." http://www.senate.gov/~banking/conf/ .

6.) "Civil Code 1798.82." California Department of Motor Vehicles, http://www.dmv.ca.gov/pubs/vctop/appndxa/civil/civ1798_82.htm , Copyright 2001 by California Department of Motor Vehicles.

7.) "Securing Windows 2000 Step-by-Step Guide." Version 1.0c, Section 1.4. Copyright 2001, the SANS Institute.

8.) "The Security Project Cookbook." Page 14, Copyright 2001, The Burton Group.

9.) "Security Operations for Microsoft Windows 2000 Server." Page 18, Copyright 2002, Microsoft Corporation.

10.) Chastain, Stacy. "Management Fails to Understand Information Security." SysAdmin, Audit, Network, Security Institute, WWW.SANS.ORG, 31 July 2003.

11.) Korofsky, Eric. "Insight Into Return on Security Investment." http://www.sbq.com/sbq/rosi/sbq_rosi_insight.pdf , Secure Business Quarterly, Volume 1, Issue 2. Fourth quarter, 2001.

12.) Korofsky, Eric. "Insight Into Return on Security Investment." http://www.sbq.com/sbq/rosi/sbq_rosi_insight.pdf , Secure Business Quarterly, Volume 1, Issue 2. Fourth quarter, 2001.

13.) "Al-Qaeda Intent to Conduct Computer Network Attacks Against Financial Institutions." http://www.state.ar.us/insurance/pdf/advisory_071803.pdf Advisory, Department of Homeland Security. 18 July 2003.

Last Updated: September 25th, 2017

Upcoming Training

SANS DFIR Prague Summit & Training 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
Community SANS Sacramento SEC401	Sacramento, CA	Oct 02, 2017 - Oct 07, 2017	Community SANS
SANS Phoenix-Mesa 2017	Mesa, AZ	Oct 09, 2017 - Oct 14, 2017	Live Event
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event
SANS Tysons Corner Fall 2017	McLean, VA	Oct 14, 2017 - Oct 21, 2017	Live Event
CCB Private SEC401 Oct 17	Brussels, Belgium	Oct 16, 2017 - Oct 21, 2017	
SANS Tokyo Autumn 2017	Tokyo, Japan	Oct 16, 2017 - Oct 28, 2017	Live Event
SANS vLive - SEC401: Security Essentials Bootcamp Style	SEC401 - 201710,	Oct 23, 2017 - Nov 29, 2017	vLive
SANS Seattle 2017	Seattle, WA	Oct 30, 2017 - Nov 04, 2017	Live Event
San Diego Fall 2017 - SEC401: Security Essentials Bootcamp Style	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	vLive
SANS San Diego 2017	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS Gulf Region 2017	Dubai, United Arab Emirates	Nov 04, 2017 - Nov 16, 2017	Live Event
SANS Miami 2017	Miami, FL	Nov 06, 2017 - Nov 11, 2017	Live Event
Community SANS Vancouver SEC401^	Vancouver, BC	Nov 06, 2017 - Nov 11, 2017	Community SANS
Community SANS Colorado Springs SEC401~	Colorado Springs, CO	Nov 06, 2017 - Nov 11, 2017	Community SANS
SANS Paris November 2017	Paris, France	Nov 13, 2017 - Nov 18, 2017	Live Event
SANS Sydney 2017	Sydney, Australia	Nov 13, 2017 - Nov 25, 2017	Live Event
SANS San Francisco Winter 2017	San Francisco, CA	Nov 27, 2017 - Dec 02, 2017	Live Event
Community SANS St. Louis SEC401	St Louis, MO	Nov 27, 2017 - Dec 02, 2017	Community SANS
Community SANS Portland SEC401	Portland, OR	Nov 27, 2017 - Dec 02, 2017	Community SANS
SANS London November 2017	London, United Kingdom	Nov 27, 2017 - Dec 02, 2017	Live Event
SANS Khobar 2017	Khobar, Saudi Arabia	Dec 02, 2017 - Dec 07, 2017	Live Event
SANS Munich December 2017	Munich, Germany	Dec 04, 2017 - Dec 09, 2017	Live Event
SANS Austin Winter 2017	Austin, TX	Dec 04, 2017 - Dec 09, 2017	Live Event
Community SANS Ottawa SEC401	Ottawa, ON	Dec 04, 2017 - Dec 09, 2017	Community SANS
SANS Bangalore 2017	Bangalore, India	Dec 11, 2017 - Dec 16, 2017	Live Event
SANS vLive - SEC401: Security Essentials Bootcamp Style	SEC401 - 201712,	Dec 11, 2017 - Jan 24, 2018	vLive
SANS Cyber Defense Initiative 2017	Washington, DC	Dec 12, 2017 - Dec 19, 2017	Live Event
SANS Cyber Defense Initiative 2017 - SEC401: Security Essentials Bootcamp Style	Washington, DC	Dec 14, 2017 - Dec 19, 2017	vLive
SANS Security East 2018	New Orleans, LA	Jan 08, 2018 - Jan 13, 2018	Live Event
SANS Amsterdam January 2018	Amsterdam, Netherlands	Jan 15, 2018 - Jan 20, 2018	Live Event