# SECURITY MANAGEMENT OF WINDOWS 2000 NETWORKS WITH DOIT4ME AND IPSEC

ABSTRACT

*The areas of remote machine security configuration and maintenance have been frequent areas of work in recent years. But on Windows-based networks this task is arduous and challenging. The success of the administration process requires automated mechanisms for auditing and configuring, through a security communication between workstations and servers.This paper shows the necessity and the advantages brought by implementing a system management tool, which is designed to reduce the complexity of the administration of a large Windows 2000 networks. The paper also advises to use this tool with IPSEC. This combination automates administrative tasks without the risk of eavesdropping and spoofing. There is a case study of a high-risk vulnerability that even installing the last operating system patch, the network can still be vulnerable to attacks.*

## 1 - INTRODUCTION AND MOTIVATION

Local and wide area computer networks have changed the landscape of computing forever. Almost gone are the days when each computer was separate and distinct. Today, networks allow people across a room or across the globe to exchange electronic messages, share resources or even use each other's computers. Networks have become such an indispensable part of so many people's lives that one can hardly imagine using modern computers without them [9].

But networks have also brought with them their share of security problems, precisely because of their power to let users easily share information and resources. Networks allow people from anywhere to remotely do anything that is possible to do locally. It created almost as many risks as they have created opportunities [9].

For good conditions, each organization should have a system administrator team with available time, staff and information to plan network growth, management and security. But this cenario usually does not happen. System administrators are often responsible for a large number of tasks that keeps them permanently busy, i.e. with no available time to manage the computers adequately or to apply a good level of security in each computer.

Most network security strategies have focused on preventing attacks from outside the organization's network. Firewalls, secure routers, and token authentication of dial-up access are examples of management attempts to defend against external threats. But hardening a network's perimeter does nothing to protect against attacks mounted from within.

In a list of the top ten worst security mistakes information technology people make, the number one is: "Connecting systems to the network before hardening them" [16]. Many system administrators still think that the process to apply security on a network is just to install the last operating system patch (Service Pack), and many of these administrators think that is only necessary to install these patches on the servers, what is a big mistake.

System administrators must concern themselves that security must be maintained not only on the servers, but also on each workstation, i.e., every computer on the network must be as secure as possible.

However it seems easy, but on Windows environments it is an extremelly complex task. The Windows environments have a reputation to require hands-on, i.e. manual administration. The administrator's physical presence in each machine is necessary every time if configuration is needed. In organizations that have a considerably large Windows network, administrators always have a hard time when they need to manage the whole network. Especially to apply security configurations on each machine in the network. These hardships imply on high monetary costs to maintain a group of system administrators in service and normally take many hours of work.

In the last several years, there has been a large number papers published on Windows security [6] [7] [8] [10] [17]; nevertheless, Windows networks lack efficient remote administration and management of large sites.

All the Windows configurations are stored centrally in one database called Registry. The Registry contains hardware and software information, besides all the security settings [12]. Each modification of a Registry's value directly affects the configuration and the status of that computer. A large portion of configuring security on Windows is modifying Registry values.

The goal of this paper is to demonstrate the necessity to have a good system management tool that automates the security administration tasks to apply security on each network machine, when determined by the administrator. Suppose a security checklist which contains a set of Registry values that should be changed to improve security, the problem is: how can the administrator automate the configuration of all these security settings without sitting in front of each computer and do the same job on each machine? How can the administrator execute the whole task with one command line? How can a system administrator effectively audit and maintain compliance to security standards (which often changes) on a large Windows 2000 network? Besides automate the tasks, how to do it without eavesdropping?

The remainder of this paper is organized as follows. Section 2 presents a case study of the risk caused by one vulnerability left on networks that don't implement security on all its computers. The developed system management tool, called DoIt4Me, is presented on section 3. Section 4 shows the necessity to implement a management tool using IPSEC, which provides secure connections between the network computers, providing the protections of integrity, authentication, and confidentiality. Finally, the paper makes some concluding remarks about using DoIt4Me and IPSEC on section 5.

## 2 - A CASE STUDY

The vulnerability exploited in this case study demonstrates the possibility for a malicious user to place a program named `Explorer.exe` in the `C:\` folder (the root of drive C) that will run in place of the standard Windows shell program.

Substituting it for a code of his choice, such code would run automatically whenever a user subsequently logged onto the same machine, and could take any action the user had privileges to take on the machine [18].

The permissions on the `C:\` folder are set to `Everyone Full Access` even after applying the lastest service pack. Anyone who can interactively log onto this computer or who has access to this share, either locally or through a network connection, can place a program there that is run before the `explorer.exe` shell [18]. This stunt could be remotely exploited too, when the administrator on the machine had shared the root directory of the machine to the malicious user, and given him write privileges to it. Another way to exploit it remotely would be sending by electronic mail a trojan with the bogus `explorer.exe` inside of it.

This vulnerability results because the name of the executable that provides the Windows Shell functionality is specified by a relative path (`explorer.exe`), rather than absolute path (`%SystemRoot%\Explorer.exe`), in the Registry key that Windows consults during startup. The permissions on `%SystemDrive%` (`C:\` folder) also allow all interactive users to have full control. The vulnerability has nothing to do with Explorer.exe per se. It has to do with the search order that occurs when the system starts up.

After placing the bogus `explorer.exe`, the only way to prevent it from running when some user logs on that computer is removing the bogus file.

### *2.1 - Demonstration*

The author of this paper created a bogus `explorer.exe` that gives any user of the network, a command prompt of the remote victim machine. Every time the victim logs on interactively, the trojan explorer will run with the user rights. Imagine the risks if someone installs the trojan on the administrator's computer of a large Windows network.

The trojan will:

1. Execute Netcat in background leaving at least one open port for any connection (this step is invisible. The victim does not know that the explorer is a trojan horse).
2. Execute the real Windows `explorer.exe`

A brief review of Netcat [14] is necessary: Netcat is a system utility which reads and writes data across network connections, using either the TCP or UDP protocols. Netcat is designed to be a reliable "backend" tool. It has proved to be an extremely versatile on Unix and Windows NT platforms. Netcat listens on a particular port for a connection and when the connection is made, it executes any program, connecting the program output (stdout) to the network connection. In this demonstration, the trojan will execute Netcat listening to any port defined when the trojan is installed, and when a client connects it to it will spawn a command shell (`cmd.exe`) giving the malicious user full control of the remote computer. This option lets Netcat to bypass firewall rules, for example, running on port 23, 53.

The trojan looks exactly like the original `explorer.exe`, even the icon is the same. But how about the Windows task manager, where the user can list all running process and see the suspicious Netcat process. To cheat this, it is necessary some extra tricks:

1. rename `Netcat.exe` as `NTLDW.sys`, and also put this file on the `C:\` folder
2. then set this file as a hidden file, so the user can only list it, if he sets the option to show hidden files on the real explorer

With the above trick, it is difficult for the user to discover that the process `NTLDW.sys` on the task manager is in fact a backdoor[1].

On a large network, it is only necessary to place a trojan on any computer, that the entire network will be compromised. Note the risk, if the trojan is placed on the network server, where the administrator usually logs on, and spawns a command shell

1. A backdoor is simply a way back into a system that not only bypasses existing security to regain access, but also defeats any additional security enhancements added onto a system.

(`cmd.exe`). A malicious user just has to wait for the administrator to log on that machine, then telnet to the specific port, and do whatever he wants, such as copying secret files to public areas, adding new users to the server, etc. Figure 1 shows what can happen if the explorer.exe trojan is installed in the network server, in this case it was installed on the computer `Latin America`. When any of the clients (`Brazil and Argentina`) connect to the especific port on the server, it will spawn a command prompt (dot lines).

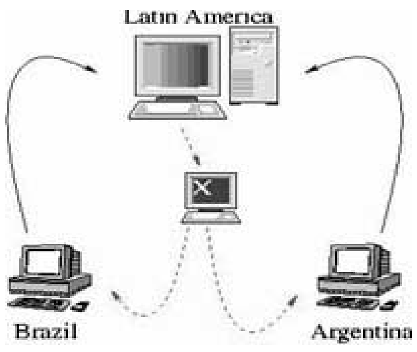

**Figure 1: Remote prompt**

*2.2 - Resolution to the "Search Path Vulnerability"*

Microsoft has confirmed this to be a problem in Windows NT 4 and Windows 2000 [18], and has delivered a patch that eliminates this vulnerability.

There are other recommendations that can avoid similar problems. In [7] [17] [18], the recommendations for the access control lists (ACL) on the root folder (`C:\`) is:

| User | Permissions |
|---|---|
| Administrators | Change |
| Everyone | Read |

**Table 1: resolution**

## 3 - SYSTEM MANAGEMENT TOOL (*DOIT4ME*)

In order to automatically manage a large network, it was necessary to cover the Windows NT and W2K deficiency of tools for remote automation of administrative tasks, and to scale whatever solution one finds to large numbers of machines [2] [3] [4]. This had to be done with a large amount of configuration flexibility (so it could be tailored to the needs of different machines and administration methods) in a way as automatic as possible.

The developed system management tool should have properties such as: simple use and maintenance; be centralized and scalable; be configurable in order to meet specific user needs; capable of enforcing compliance with security policies and standards; reduced overall cost of administration; and require minimal human interaction. It should also scale to a network of any size.

To solve all requirements above, it was implemented a new tool, called DoIt4Me [2] [3] [4]. There is one related work but it is not a scalable solution, i.e., it needs a lot of human interaction to automate the administration of a large NT site [6].

System administrator can customize the DoIt4Me code at any time, because it is implemented in Perl [1]. Its interface has a simple unified syntax and is used through the Windows command line interpreter. The DoIt4Me current options include, but are not limited to:

1. Perform remote auditing of a subset of Registry settings. The administrator only needs to specify what Registry settings he or she wants to audit.
2. Remotely configure a subset of Registry settings. The administrator can modify the Registry settings specifying the new value of each Registry key.
3. Perform service status auditing. The administrator can configure DoIt4Me to audit the status of either all or a set of services. Auditing of specific services are also contemplated, such as "which machines are running the service "schedule"?"
4. Start or stop remote services. The administrator can start or stop any subset of services. For this purpose, he or she needs only to specify the service name and the action (to start or to stop it), and the subset of computers to apply these configurations on.
5. Reboot or Shutdown. There is an option where the administrator can reboot or shutdown a subset of workstations. In this option, the administrator can configure the grace period before rebooting, the message to send before rebooting, and the subset of machines to be rebooted.
6. Apply permissions on files, folders and Registry keys (ACLs); (module under construction)
7. Ping a list of computers.
8. And any other automated task. DoIt4Me is developed in Perl, and as such is open source. The administrator can add or modify DoIt4Me modules any time he or she wants.

All the above functions can be executed once with any subset of computers. As a practical example one can collect Registry values of any subset of workstations and apply new configuration values to this subset. To solve problems such as our case study (section 2), DoIt4Me will permit the administrator to specify new ACL permissions to a subset of workstations.

### *3.1 - Installation, configuration, reporting*

To manage a Windows network with DoIt4Me, is only necessary to install it on the domain controller (DC). There is no DoIt4Me clients running on the workstations. With this system management tool, the administrator can remotely control any subset of machines served by the DC. All global security policy changes are made only on the configuration files, which are stored on the server. This model works well for complying with changing security policies.

The current version of DoIt4Me has eight configuration files. All these files are located on the "`doit4me/cfg`" folder. In this folder there are files such as the `pclist.cfg`, which contains the subset of machines that DoIt4Me will scan or configure. Another file, `srvnewstatus.cfg`, is used to change services status. The syntax of this file is: on each line is specified the service name followed by its new status, i.e., 1 to start the service or 0 to stop the service.

The output produced should be in a format fit for human consumption. The reports enable the system administrator to identify quickly and easily, any problems related to the machines, ranging from a client being down to reporting a subset of machines that are not complying with security policies and standards. Below, Figure 2 shows DoIt4Me interface, Figure 3 shows a Registry audit report and Figure 4 presents the service audit report.

```
                        DoIt4Me
      Automate NT Administrative Tasks Remotely

Usage: doit4me.pl <option>
Option: <1> Audit Registry keys
        <2> Configure the Registry
        <3> Check the status of ALL NT services
        <4> Check the status of a subset of NT services
        <5> Change NT services Status (Start/Stop)
        <6> Ping a subset of workstations
        <7> Reboot a subset of workstations
        <8> Help
```

**Figure 2: DoIt4Me Interface**

DoIt4Me uses TCP/IP packets to communicate the server with the workstations. The packets are not encrypted, so one problem became apparent during the implementation: eavesdropping. To guarantee security during the communications, DoIt4Me could be used with IPSEC (IP Security Protocol) [13].

```
C:\> DoIt4Me.pl  1
------------------------------------------------------------------------
                        Auditing Report
------------------------------------------------------------------------
COMPUTER              KEY                  VALUE
------------------    --------------       --------------
argentina             CSDVersion           Service Pack 6
brazil                CSDVersion           Service Pack 6
paraguai              CSDVersion           Service Pack 5

argentina             DontDisplayLastUserName     0
brazil                DontDisplayLastUserName     1
paraguai              DontDisplayLastUserName     0
```

**Figure 3: DoIt4Me Registry Audit Report**

```
C:\> DoIt4Me.pl  4
--------------------------------------------------------------
                     Services Status
--------------------------------------------------------------

schedule
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
COMPUTER                        STATUS
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
argentina                       [Started]
brazil                          [Started]
paraguai                        [Stopped]
```

**Figure 4: DoIt4Me Services Audit Report**

## 4 - IPSEC

Throughout a few decades, computer on the internet were subject to many individual attacks. The solution to these attacks was relatively simple: encourage users to choose good passwords, prevent users from sharing accounts with each other. But this infrastructure has come under attack:

- Network sniffers have captured the packets passing through the networks as they are transmitted.
- IP spoofing attacks have been used by attackers to break into hosts.
- Data spoofing has been used by attackers on a network to insert data into an ongoing communication between two other hosts. Data spoofing has been demonstrated as an effective means of compromising the integrity of programs executed over the network.

IPv4 is designed to get packets from one computer to another computer; the protocol makes no promise as to whether or not other computers on the same network will be able to intercept, read or modify those packets in real time. Such interception is called eavesdropping. The only way to protect against eavesdropping in these networks is by using encryption.

The need for IP based network security is already great and is growing. The challenge for network administrators is to ensure that the traffic is:

- safe from data modification while enroute (data integrity);
- safe from interception, viewing, or copying (confidentiality);
- safe from being accessed by unauthenticated parties (authentication).

Designed by the Internet Engineering Task Force (IETF) for the Internet Protocol, IPSEC supports network-level authentication, data integrity, and encryption [11].

Because IPSEC is deployed below the transport level, network managers (and software vendors) are spared the trouble and expense of trying to deploy and coordinate security one application at a time. No user training is required. By deploying the networks, network managers provide a strong layer of protection for the entire network, with applications automatically inheriting the safeguards [11].

Network administrators and managers benefit from the integration of IPSEC in their networks for a number of reasons, including:

- Transparency: IPSEC exist below the transport layer, making it transparent to applications and users, meaning there is no need to change network applications on a user's desktop.
- Authentication: Strong authentication services prevent the interception of data by using falsely claimed identities.
- Confidentiality: prevent unauthorized access to sensitive data as it passes between communicating parties.
- Data Integrity: IP authentication headers and variations of hash message authentication code ensure data integrity during communications.
- Flexibility: the flexibility of IPSEC allows policies to apply enterprise-wide or to a single workstation.

One of the great benefits of IPSEC is the ability to protect against both internal and external attacks. Again, this is done transparently, imposing no effort or additional overhead on individual users. Besides this benefit we can list the cost savings, where organizations have had to strike a difficult balance between the desire to protect their data communications and high costs of establishing and the maintaining that protection. Security can impose costs that exceed the hardware cost of the network.

## 5 - CONCLUSIONS

Security management of Windows environments is a challenging task and it is imperative to automate it as much as possible. It requires a combination of auditing, correction, security and automation mechanisms.

One of the most important tasks of a system administrator is to keep the most current patches for an operating system and installed software. Many of these patches fix security vulnerabilities that are well known to intruders. Unfortunately "Windows systems are not secure by only installing the last Service Pack". Also, in a large network, system administrators must not only apply security on the servers , but also on each workstation.

There are several tools available to assess the security of networks, but few freely available solutions for fixing the problems on each network machine. DoIt4Me is a must-have automated management tool for Windows network administration.

The current version of DoIt4Me addresses security weaknesses and eases standardization and adherence to Windows network security policies. Our experience has shown that it is possible to remotely manage a large NT and W2K network in a scalable way with DoIt4Me.

And at a time when network security is increasingly vital, IPSEC makes it easy for network managers to provide a strong layer of protection to their organization's information resources.

Combining DoIt4Me and IPSEC, the system administrator provides network managers with a critically important line of security. The facility of DoIt4Me permits the administrator to automate hard administrative tasks with one command line. On the other hand, the flexibility of IPSEC permits easily the network managers to be able to create custom security policies and filters, based on user, work group, or other criteria.

## 6 - REFERENCES

[1] ActiveState WebSite. http://www.activestate.com/

[2] AUGUSTO, Alessandro and GUIMARAES, Célio and GEUS, Paulo Lício. ``Administration of Large Windows NT Network with DoIt4Me". Proceedings of SANS 2001: The 10th International Conference on System Administration, Networking and Security, Baltimore, MD, USA, May, 2001. (in English)

[3] AUGUSTO, Alessandro and GUIMARAES, Célio and GEUS, Paulo Lício. ``DoIt4Me". Accepted to the 1st Tools Demonstration. Proceedings of SBRC 2001: the 19th Brazilian Symposium on Computers Networks, Florianópolis, SC, Brazil, May, 2001.

[4] AUGUSTO, Alessandro and GUIMARAES, Célio and GEUS, Paulo Lício. ``DoIt4Me: a tool for automating administrative tasks on Windows NT Networks". Proceedings of WSEG'2001: Workshop of Computer Security, Florianópolis, SC, BRAZIL, March, 2001. (in English)

[5] AUGUSTO, Alessandro and GUIMARAES, Célio and GEUS, Paulo Lício. ``Administration Techniques for Implementing Security on Large Windows NT Networks". Proceedings of SSI'2000: Symposium on Informatics Security, São José dos Campos, SP, BRAZIL, October, 2000. (in English)

[6] CARVEY, Harlan, ``System Security Administration for NT``. Proceedings of USENIX LISA-NT: The 2nd Large Installation System Administration of Windows NT Conference, USA, 1999.

[7] CERT. Windows NT Configuration Guidelines. April, 2000. http://www.cert.org/tech_tips/

[8] DALY, Gregg and BUHRMASTER, Gary and CAMPBELL, Matthew and CHAN, Andrea and COWLES, Robert and DENYS, Ernest and HANCOX, Patrick and JOHNSON, Bill and LEUNG, David and LWIN, Jeff. ``NT Security in an Open Academy Environment". Proceedings of USENIX LISA-NT: The 2nd Large Installation System Administration of Windows NT Conference, USA, 1999.

[9] GARFINKEL, Simson and SPAFFORD, Gene. "Practical UNIX & Internet Security". O´Reilly & Associates, Inc. 1996.

[10] GOMBERG, Michail and STACEY, Craig and Sayre, Janet. ``Scalable, Remote Administration of Windows NT". Proceedings of USENIX LISA-NT: The 2nd Large Installation System Administration of Windows NT Conference, USA, 1999.

[11] Internet Engineering Task Force (IETF). Url: http://www.ietf.org/html.charters/ipsec-charter.html

[12] KIRCH, John. "Troubleshooting and Configuring the Windows 95/NT Registry". Macmillan Computer Publishing, 1999.

[13] Microsoft Windows 2000 Server. "IP Security for Microsoft Windows 2000 Server". White Paper, 2000.

[14] NETCAT WebSite. http://www.l0pht.com/~weld/netcat/

[15] ROTH, Dave, ``A Networked Machine Management System". Proceedings of USENIX LISA-NT: The 2nd Large Installation System Administration of Windows NT Conference, USA, 1999.

[16] SANS. ``Mistakes People Make that Lead to Security Breaches". 2000. http://www.sans.org/mistakes.htm

[17] Trusted Systems Services. ``NSA Windows NT Security Guidelines. Considerations & Guidelines for Securely Configuring Windows NT in Multiple Environments". June, 1999. http://www.trustedsystems.com

[18] Windows Knowledge Base. ``Registry-Invoked programs use standard search path". Article Q269049. http://support.microsoft.com/support/kb/articles/q269/0/49.asp