Applying Security Configurations to a Large Number of Windows NT Computers Without Visiting Each Machine

Alessandro Augusto

IC - UNICAMP
University of Campinas - Campinas, SP, Brazil
alaugusto@yahoo.com.br

Abstract

Security administration in large Windows NT sites is a very challenging task. Windows NT's environments has a reputation to be a system requiring hands-on administration, that is, it needs a manual by-hand work. It is necessary the administrator's physical presence in each one of the machines every time it needs to do some modification or configuration.

This paper presents a case study of a challenging task: a system administrator who wants to apply a security checklist on each network computer of a large Windows NT network without visiting each machine and using some scalably remote solution.

1. Introduction

With the increased proliferation of system networks, computer security has become an increasingly large problem for system administrators of large sites with several hundreds or more systems.

Among the operating systems with wide prominence and use in several environments, Windows NT gets the attention with its growing use and its user-friendly interface [4], [7].

Out of the box, Win32 machines lack the ability to be efficiently managed. Its up to an administrator to implement some form of system management. For many administrators with relatively small networks this means walking from machine to machine to complete the tasks. Even a simple task as discovering which computers have enough free hard drive space for a software upgrade can take many hours of an administration teams time [12]. On large networks, that is not so easy. With hundreds or thousands of computers on the network it becomes virtually impossible to physically visit each computer. Sometimes the computers are located far away from each others.

Windows NT's environments has a reputation to be a system requiring hands-on administration, that is, it needs a manual by-hand work [7]. Windows NT system administration often involves using programs designed for interactive use. Many such programs (`explorer.exe`) can not be placed into shell scripts. It is necessary the administrator's physical presence in each one of the machines every time it needs to do some modification or configuration. The associated costs increases as the amount of the network computer gets bigger.

There is an overwhelming perception in the UNIX community that Windows NT is impossible to be managed remotely and also that the NT administration is not scalable [8], [9]. A remote automated procedure should not require that system administrators visit each workstation. This is a problem in many environments where the workstations are located in different rooms, buildings and so on. Fixing each machine through physically visiting it requires a lot of manpower and can be error-prone; operator errors can lead to machines being configured erroneously, improperly, or not at all [1].

In organizations that have a considerably large Windows NT network, administrators always have a hard time when they need to apply some security configurations on each network machine [1].

Unlike many other types of system administration tasks, which can be done at a later time, delaying the installation of a security patch, could leave a site more vulnerable to an intruder attack. The worse mistake made by information technology people is: “Connecting systems to the network before hardening them” [13]. Nowadays, there are many system administrators who think that the process to apply security on a NT network is just to install the patches only on the servers. That is a big mistake. Security is a chain; it's only as strong as the weakest link [14]. A single weak link can break the entire network. Security must be applied to all network computers.

This paper presents a case study of a challenging task: a system administrator who wants to apply a security checklist on each network computer of a large Windows NT network without visiting each machine and using some scalably remote solution.

2. Case Study

The goal of this case study is to demostrate a practical example of how the administrator can apply or modify the security configurations of each Windows NT network computer without visiting each workstation.

In Windows NT, all configurations are stored centrally in one database called Registry, which is one of the most important system resources, especially when talking about security [10], [11]. The Registry contains all the information about hardware and software configuration: it stores information about user accounts, user groups, and information about installed hardware and software [10].

The Registry is developed in a hierarchical structure and each modification of a Registry’s value directly affects the configuration and the status of that computer.

A large portion of NT security requires modifying Registry values. For a practical example, Table 1 show four examples of a Windows NT security checklist [6]:

Recommendation 1
When a user successfully logs into an NT system, the username is shown by default the next time someone tries to log into the system. Knowing the usernames of a successful logon can help intruders perform dictionary or brute attacks. To prevent the username of the last logged on user from being displayed to the next user to logon to the device, use this setting.
Path: *HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon* Key Name: *DontDisplayLastUserName* Recommended Value: *1*

Recommendation 2
This recommendation will be used just as an example, to change some value under the HKEY_CURRENT_USER key.
Path: *HKEY_CURRENT_USER\Control Panel\example\* Key Name: *example_key* Recommended Value: *1*

Recommendation 3
Service packs are used to patch a wide range of vulnerabilities and bugs. The latest service pack that you have tested to work in your environment should always be applied after installing the operating system. Service packs are cumulative; you only need to install the latest Service Pack.
Recommended to install the lastest Service Pack

Recommendation 4
NT tend to run services under systems accounts instead of specific blocked user accounts, i.e., with minimum priorities. This could open future holes if bugs are found in the system services.
Disable unneeded services.

Table 1: example of a Windows NT security checklist

Suppose the following cenario: there is a large Windows NT network with hundreds of computers and only one administrator who wants to apply the security checklist from Table 1 as three steps:

(Step 1) apply the recommendations 1 and 2;

(Step 2) audit the service pack version that each computer is running (recommendation 3);

(Step 3) audit the service “*alerter*” and then stop it on the computers that are running it.

How can the administrator apply and audit all these configurations remotely? Also, it needs to be a scalable solution, i.e., the solution provided must be good for small or large networks. For a better understanding, Figure 1 shows a NT network. The `Latin America` computer is the server (primary dommain controller), and the computers `Argentina, Brazil, Paraguai` are some computers of a the large network.

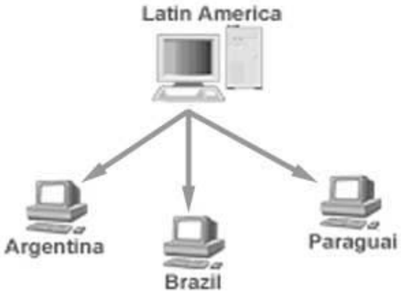

Figure 1: cenario

2.1 Design Goals

One of the keys to administering large networks is join together a set of good tools or to write your own tools to handle as many common tasks as possible. The main point of a good administration is the automation of tasks for any network size. It doesn't matter how many computers there are on the network, the administrator must be able to remote execute the same task on each computer without visiting and with the less interaction as possible.

Faced with the NT weaknesses, the desirable solution for our case study must be: simple, centralized, scalable, inexpensive and espeacilly minimal human interaction. When trying to figure all these desirable properties in a single solution, the author developed a new remote system administration tool, called DoIt4Me ("do it for me") [1], [2], [3]. But before present this great solution, the paper shows a related work and its deficiencies.

2.2 Related Work

Harlan Carvey presents a framework of a few administrative scripts that had some similar goals to our project's [5]. One of his scripts, called '`regkeys.pl`', is devised to collect Registry values from a remote NT system. However, these scripts have some weaknesses: they are not scalable to a large NT network.

In our cenario, if the administrator wants to collect the value of the key '`DontDisplay-LastUserName`' of all workstations (`recommendation 1`), it could be done with the script presented by Harlan, but the administrator will spend a lot of time and effort. Harlan's script requires human intervention for each audited machine. The administrator will have to execute the script for each computer. The script audits only one computer at a time. The effort to compare the results after all executions is really hard too.

It is clear that his approach is unable to handle a large number of machines. Also, once the system administrator audited the computers and discover which workstations are not in compliance with security policies, there is no script able to configure the machines with new values, i.e. to act upon.

However, these scripts also have their strength, since they show how to do the remote collection for a single machine, and as such can be used as a building block to achieve our goals.

2.3 Proposed Solution

DoIt4Me is an automated and remote tool that solves the general administration problem of Windows NT networks, however it solves an unusually large number of problems in the system administration arena [1], [2], [3]. The main goal of DoIt4Me is to turn the security administration tasks, such as applying security configurations to each computers, easier.

Due to it's remarkable abilities and reasonable price (free), DoIt4Me was developed in Perl. Perl is well known as the Swiss army knife of languages. It is almost as powerful as C++ but has a much faster development cycle. And if the language has any limitations they can be eliminated with Perl extensions. Another reason to use Perl is the vast number of modules and extensions that have already been created for it [12].

DoIt4Me was developed to help administrators solve common administration tasks found in Windows NT 4.0 networks, although most of the functionality can also be used on Windows 2000 networks.

By installing DoIt4Me on the primary domain controller of the NT network (PDC), the administrator can remotely control any subset of workstations served by the DC. On the Figure 1, DoIt4Me should only be installed on the `Latin America` computer. The administrator will control the network from this machine. The workstations does not need any modification on its configurations. It is a centralized solution.

DoIt4Me has a lot of features such as: (1) the ability to scan the entire network for especifics Registry keys, (2) the ability to remote configure the Registry of each network computer, (3) it can check the status of the services presented in each computer and also, start or stop it, etc. The features and options of DoIt4Me is presented on Figure 2.

Before demonstrate how to execute the first step, it is necessary to explain that there are some limitations. Some management tasks cannot be performed remotely because of Windows NT limitations, such as full remotely access to the Registry. Windows does not export the whole Registry, only two of the five Registry keys can be accessed remotely: the HKEY_LOCAL_MACHINE and the HKEY_USERS [11]. But there are ways to by- pass these weaknesses.

```
C:\> DoIt4Me.pl
_________________________________________________________________
                            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

DoIt4Me, Copyright (C) 2001 Alessandro Augusto
DoIt4Me comes with ABSOLUTELY NO WARRANTY; for details
see option <8>.  This is free software, and you are welcome
to redistribute it under certain conditions; see DoIt4Me license.
_________________________________________________________________
```

Figure 2: DoIt4Me Interface

2.3.1 Applying recommendations 1 and 2

On the first step, the administrator wants to apply the recommendations 1 and 2. Note that on the recommendation 2, it is not a real key, the reason for that is to show a real example of how to configure one key that is not exported by Windows.

To apply the recommendation 1, it is necessary that the administrator configure the file called "`regconfig.cfg`", stored on the folder `cfg`, which is a subfolder of doit4me. Besides that, it is also necessary to especify which computers will be configured. This configuration is made on the `pclist.cfg` file. The syntax of the `regconfig.cfg` file is: the key name; the key path; the new value to be applied (it is necessary to use ";" as separation). Table 2 and Table 3 show example of the syntax of both files.

File: ***doit4me/cfg/regconfig.cfg***
DontDisplayLastUserName; HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\Current-Version\Winlogon; 1

Table 2: Regconfig.cfg

File: ***doit4me/cfg/pclist.cfg***

argentina;
brazil;
paraguai;

Table 3: especifying the subset of computers

After configure the files, it is necessary to execute doit4me with the option 2, see Figure 2.

For the recommendation 2, it can not be executed by the same way because Windows limitation. To configure a key that is not exported, it is necessary to use the schedule technique [4].

This technique requires that the system administrator uses the schedule service to apply a package. The schedule service allows the system administrator to schedule tasks remotely at a given time. The package to be applyed is a file created by the administrator, which will contains all the configurations that he wants to apply. The ideia is simple: create the package that contains the configurations, start the schedule service in each computer to apply this package, then schedule at a given time a task to connect each computer to the server, where the package is and apply it.

The goal of the packaging, process to create the file, is to discover what changed in the NT system of the computer after any configuration and to create a file that just contends these modifications. It is necessary some application to do a "sweeping" of the file system and the Registry. The application used here to sweep the system and to generate the package is called *sysdiff.exe* [4].

To create the package, the administrator must choose one computer and install *Sysdiff.exe*. Then, execute the application *Sysdiff.exe* with the option */snap* to take a snapshot of that computer before any modification. This snapshot will contain all the configuration of the current NT system. This step should be done before the administrator configure any security setting. Soon after, the administrator should accomplish the change wanted in the machine, e.g. accomplishes the security recommendations, such as *recommendation 2* from Table 1. After accomplishing the configurations, execute the application *Sysdiff.exe* again but with the option */diff* to find the differences happened in that computer and to create an installation package. To create the final package, *Sysdiff.exe* receives as entrance the snapshot took before the changes and it supplies as result the generated package, which contains all the changes that were done on that machine by the administrator.

This package will be stored in some network server, and a task that will connect each computer to this server will be remotely scheduled by the administrator.

In our case study, suppose the administrator created the package. This package will be stored at the *Latin America* computer. Figure 3 shows the whole sequence.

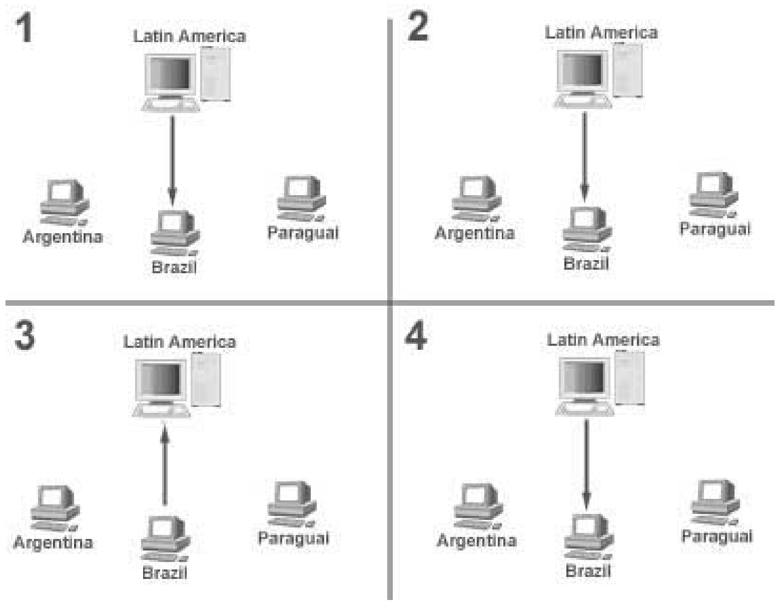

Figure 3: Sequence

On slide 1, the administrator uses DoIt4Me to start the schedule service in network computer from the `Latin America` machine. Note that the package created is stored in this server. Also, to start the service, the administrator must configure the file `pclist.cfg` with the computer's name, and the `serviceconfig.cfg` file with the service name and the new status (1 to start or 0 to stop), see Section 2.3.3.

On slide 2, from the `Latin America` computer the administrator schedule a task on the workstations to apply the package. This task is just a simple script.

The step 3 shows the exacly time that the task is executed on the workstations. This task will connect each workstation to the server. And the last step is the package being applyed on each network machine.

2.3.2 Auditing the Service Pack Versions

The second step is to audit the service pack version that each computer is running. This task is very easy with DoIt4Me, the administrator needs to configure the `regaudit.cfg` file as shown on Table 4 and also configure the `pclist.cfg` file with the name of the computers that will be audited. Then, execute DoIt4Me option 1.

File: *doit4me/cfg/regaudit.cfg*
`CSDVersion; HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion`

Table 4: example of regaudit.cfg

One problem became very apparent during the implementation. The output produced should be in a format fit for human consumption. The reports enable the administrator to identify quickly and easily, any problems related to the machines, ranging from a workstation being down to reporting a subset of machines that are not in compliance with security policies and standards. Figure 4 shows an audit report from step 2.

The `recommendation 3` advises the administrator to install the lastest version of the service pack. By the time of this writing, the lastest version is 6. With the report presented on Figure 4, it is simple to see that the computer `paraguai` is not in compliance with this recommendation.

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

Figure 4: DoIt4Me Audit Report

2.3.3 Auditing and Stopping the `Alerter` Service

It is also possible to change the status of any service. DoIt4Me permits the system administrator to start or stop any service in any subset of computers.

In our case study, the administrator wants to audit the service `alerter` and then stop it on the computers that are running it. To audit the service, it is necessary to define the subset of computers (`pclist.cfg`). Also, the administrator must specify the subset of services to be audited, in this case only the `alerter` service, such as Table 5.

File: *doit4me/cfg/pclist.cfg*

argentina;
 brazil;
 paraguai;

File: *doit4me/cfg/serviceaudit.cfg*
alerter;

Table 5: configuring the pclist.cfg and the servicegaudit.cfg files

Figure 5 shows an example of output produced by this option. In this report, the administrator can see that the service alerter is being executed on the machine *brazil* and *paraguai*. After discovered which computers are running the service, the administrator wants to stop it.

```
C:\> DoIt4Me.pl  4
-----------------------------------------
          Services Status
-----------------------------------------
alerter
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
COMPUTER                     STATUS
-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
argentina                   [Stopped]
brazil                      [Started]
paraguai                    [Started]
```

Figure 5: Services Status Report

To stop the service on this two computers, the administrator must configure the file *pclist.cfg* again but only with the name of both computers, and also, it is necessary to configure the *regconfig.cfg* file with the service and the new status (1 to start or 0 to stop). Table 6 shows an example of these configurations. To complete the task, the administrator needs to execute DoIt4Me with option 5.

File: *doit4me/cfg/pclist.cfg*

brazil;
 paraguai;

File: *doit4me/cfg/serviceconfig.cfg*
alerter; 0

Table 6: Stopping the Alerter Service

3. Conclusion

Security administration in large NT sites is a challenging task. It is imperative to automate as much tasks as possible. Most of the tools and checklists available today focus only the identification of security vulnerabilities, not the its correction.

By automating the tasks in an scalable fashion, the system administrator eliminates the possibility of error-prone, besides that the tasks are done in a timely fashion saving computer and staff resources. DoIt4Me was developed for system administrators who wants to audit or to apply security checklist to each NT computer without visiting each machine.

4. References

[1] AUGUSTO, Alessandro; 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.

[2] AUGUSTO, Alessandro; GUIMARAES, Celio and GEUS, Paulo Licio. *DoIt4Me*. Proceedings of SBRC’ 2001: 19th Brazilian Symposium on Computer Networks. 1st Tool Demonstration. Florianópolis, SC, Brazil, may, 2001.

[3] AUGUSTO, Alessandro; GUIMARAES, Celio and GEUS, Paulo Licio. *DoIt4Me: a Tool for Automating Administrative Tasks on Windows NT Networks*. Proceedings of WSEG'2001: Workshop of Computer Security, Florianópolis, SC, Brazil, March, 2001.

[4] AUGUSTO, Alessandro and GUIMARAES, Celio and GEUS, Paulo Licio. *Administration Techniques for Implementing Security on Large Windows NT Networks*. Proceedings of SSI'2000: 2nd Symposium on Informatics Security. São José dos Campos, SP, Brazil, October, 2000.

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

[6] CERT. *Windows NT Configuration Guidelines*. 23/07/2001. http://www.cert.org/tech_tips/win_configuration_guidelines.html

[7] FULMER, Robert & LEVINE, Alex. *AutoInstall for NT: Complete NT Installation Over the Network*. In: Proceedings of the Large Installation System Administration of Windows NT Conference, USENIX LISA, Seattle, Washington, USA, 1998.

[8] GOMBERG, Michail; EVARD, Rémy and STACEY, Craig. *A Comparison of Large-Scale Software Installation Methods on NT and UNIX*. Proceedings of USENIX LISA-NT, the Large Installation System Administration of Windows NT Conference, Seattle, Washington, USA, 1998.

[9] GOMBERG, Michail; 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, Seattle, Washington, USA, 1999.

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

[11] MICROSOFT Windows NT Workstation Resource Kit: *Comprehensive Resource Guide and Utilities for Windows NT Workstation Version 4.0*. Microsoft Press, 1996.

[12] ROTH, Dave. *A Networked Machine Management System*. Proceedings of USENIX LISA-NT, the 2nd Large Installation System Administration of Windows NT Conference, Seattle, Washington, USA, 1999.

[13] SANS. *Mistakes People Make that Lead to Security Breaches*. 25/07/2001. http://www.sans.org/mistakes.htm

[14] SCHNEIER, Bruce. *Why Cryptography Is Harder Than It Looks*. 24/07/2001. http://www.counterpane.com/whycrypto.html