

# **BO2K: Use, Function, and Security Considerations**

**Greg M. Bednarski**

**January 2004  
Carnegie Mellon University**

## Introduction

---

Back Orifice 2000 (or “BO2K” as it has been renamed), is, according to its development community “... the most powerful network administration tool available for the Microsoft environment, bar none.”<sup>1</sup> The tool puts network administrators in control of client computers by giving them access to the registry, settings, passwords, file systems and processes of each client running the server software. Although the developers of this ‘remote administration tool’ claim its existence as legitimate for network management, many major computer security firms classify it as a Trojan or virus, or more specifically a Remote Administration Trojan/Tool (R.A.T.). Symantec Security Response has even created a detailed threat assessment of this particular tool, along with removal instructions and definitions for their associated virus scanning tool<sup>2</sup>.

Back Orifice was introduced to the public at DefCon7 in 1998, a Las Vegas convention of hackers, crackers, security analysts, and members of the law enforcement community. Its name was coined by Sir Dystic, a member of the Cult of the Dead Cow, as a play on Microsoft Corporation’s Back Office suite of server applications. Since its release by the Cult of the Dead Cow (cDc), a somewhat controversial hacking group who claim to have given Ronald Reagan Alzheimer’s disease, it has been used by less-than-scrupulous individuals to annoy and sometimes harm internet-connect computers. “Annoy” has taken the form of keystroke logging, registry editing, rebooting, redirection of network connections, and general invasion of privacy on the victim’s computer. From security analysts, corporations, to Uri Geller (a motivational speaker who claimed he could bend spoons with his mind) writing an article in *Computer Active* scolding cDc for creating the tool, and Microsoft for allowing it to happen<sup>3</sup> (although BO2K doesn’t attack any specific Microsoft weakness), the networked computing community at large was stunned at its release and use.

Although an entire paper could be written about the background and motivations of its colorful creators and critics, this paper aims to give a detailed analysis of the BO2K tool. For background information, we will look at other similar tools and their uses as well as the specific similarities to BO2K, and understand the reason for examining this particular tool. Secondly, we will discuss the apparent uses of this tool, both legitimate and otherwise, before exploring the behavior during actual use and execution. Finally, we will discuss any publicized uses of this tool, known and potential defenses, possible derivatives, and potential policy concerns for both IT organizations and large corporations. For our purposes, and to maintain as much of an objective analysis as possible, we’ll refer to the owner/operator of the BO2K controlling client as the “client” and the unsuspecting compromised host as the “server”.

## Section 1 – Why research this tool? What other tools are similar, and in what ways?

---

This tool poses an interesting threat to not only corporate and professional IT users, but also to the general networked home or small office computer user. Whereas the listings of worms (such as Slammer, Ramen, and NIMDA) carry a mostly benign payload – denial of service attacks, clogging network routes and defacement of web pages – and DDoS tools (such as Trin00 and Stacheldraht) have a single-minded purpose of reducing or completely preventing the availability of services on a network, BO2K and its equivalents offer a much more flexible and nefarious purpose. Potentially controlling every aspect of operation of victim’s computer, via registry and service modification, as well as the ability to copy and modify private files, offers an intruder an out-of-the-box solution to all their malicious aspirations. Couple this with the fact that BO2K is open-source; a more technical savvy user could easily write code additions to carry out data

---

<sup>1</sup> BO2K: Open Source Remote Administration Tool. “What is BO2K?”

<http://www.bo2k.com/whatis.html>

<sup>2</sup> Symantec Security Response. “Information on Back Orifice and NetBus”

<http://securityresponse.symantec.com/avcenter/venc/data/backorifice.html>

<sup>3</sup> Uri Geller. “Computer Active Columns”. <http://www.uri-geller.com/compacy.htm>

destruction routines, or use the tool as a method for installing and launching new worms and viruses from others' computers.

The most similar tools offered to BO2K are NetBus (and NetBus Pro) and SubSeven. While SubSeven also claims to be a legitimate remote administration tool, their website makes no bones about its uses as an application for controlling the computers of unsuspecting individuals (their disclaimer, prominently displayed on the first webpage of their mirrors, claims they use "hacker jargon and role-playing scenarios when describing its activities... any discussion of activities which could be considered illegal by either Federal or State laws are purely fictional..."<sup>4</sup>) Indeed, the SubSeven tutorial created by "Mobman" even refers to the system running the server as the "victim". Whereas BO2K and NetBus do not come pre-packaged with these options, SubSeven allows the client to spy on instant messaging communications, flip the computer's display up-side-down, change the screen saver, hide buttons, and other childish pranks.

NetBus, written by Swedish programmer Carl-Fredrik Neikter, has been described as Back Orifice's older cousin. NetBus allows the client to perform many of the same functions available with BO2K (such as registry editing, starting/stopping of services, and control via a graphical user interface), but adds the functionality to open and close the server's CD-ROM tray(s), was the first to add eavesdrop functionality via the server's attached microphone and/or web-camera (if any), and send interactive dialogs to chat with the user of the server<sup>5</sup>. This added functionality comes at a cost, as Back Orifice implements a much smaller server than NetBus (122KB versus 612KB). Table 1 highlights more similarities and differences between these three remote administration tools.

	BO2K	NetBus/Pro	SubSeven
Plug-in Extensibility	Yes	Yes	No
Session Logging	Yes	Yes	No
Keystroke Logging	Yes	Yes	Yes
File Transfer	Yes (via HTTP)	Yes	Yes (FTP or HTTP)
Registry Editing	Yes	Yes	Yes
Remote Upgrading	Yes	No	No
Network Redirection	Yes (TCP/IP)	Yes	Yes
GUI Messages	Yes	Yes	Yes
Process Control	Yes	Yes	Yes
Remote Reboot	Yes	Yes	Yes
Comm. Encryption	3DES	No	No
Childish Pranks	No	No	Yes
Remote Desktop Control	Yes	Yes	Yes
Server Size	122KB	612KB	372KB

Table 1

## Section 2 – BO2K's Apparent Purpose

---

As mentioned above, BO2K's core developers refer to the tool as the ultimate remote administration tool. While the functions offered could be useful to the most autocratic of system administrators, it has a significant number of similarities with tool designed for compromising an unsuspecting user's computer system. Additionally, the fact it was authored by a somewhat fringe hacking group, and released during a hacking convention tend to lend it to less than legitimate uses.

<sup>4</sup> Sub7.net, "Disclaimer and Terms of Use". <http://hackpr.net/~sub7/main.shtml>

<sup>5</sup> Internet Security Systems. "NetBus Backdoor Attack". <http://www.irchelp.org/irchelp/security/netbus.html>

Focusing on those types of uses, let's explore for a moment the apparent purpose of this R.A.T. The general implication of using this tool is for total control over a server's operations. This includes control of the services and applications running, shutdown commands, redirecting connections, and installing software from a remote location. On the privacy side, BO2K has the built-in functionality to log every keystroke entered on the server, capturing passwords (which may otherwise be stored in an encrypted form), personal communications, and private data (such as social security numbers, bank accounts, etc.). Additionally, the client can browse through the file system of the compromised computer and transfer any files to a remote location. The client can also modify the target's registry, dump passwords, access command-line functions, and control mouse and keyboard commands.

Again, one can argue the legitimacy of these functions (albeit taking some liberties), but one has to ask some obvious questions: why would encryption be required for administering other computers? Is there real value in re-directing TCP/IP connections through a computer (therefore making it a pseudo-proxy)? Does the built-in key logger lend itself to too much opportunity for abuse?

Once control is taken of a server, the client can then use all these tools available to corrupt the system, steal or destroy data, configure the computer to send spam or launch denial of service attacks – anything that the legitimate owner of the compromised system could do. This could, of course, be said of any commercial remote administration tool on the market<sup>6</sup>, but taken in context with its initial audience and creators, it's not hard to see the underlying purpose of becoming an expert in use of this tool (other than for defensive purposes...).

### **Section 3 – Detailed Behavior during Execution**

---

BO2K currently runs on all Windows operating systems, including Windows 95, 98, ME, NT, 2000, and XP, and conducts communications between the server and client via TCP and UDP ports. Below we will examine the detailed functions and behaviors by setting up a client-server relationship on a single computer: a desktop running Windows XP Professional, acting as the client as well as the server (in effect we've infected the controlling machine). This machine is connected to a private wired LAN connection, and not exposed to the internet or any other networks. Virus detection and firewall protection was disabled on both systems for this trial.

#### *Configuration*

The binary version of BO2K comes with three main files: the server configuration utility, the server file to be run on the target computer, and the client GUI, to control the server. We start by configuring the server we plan to plant on a target system (customizing the Trojan horse?), by opening a server file with the configuration utility. Here we can set the specific port which the server computer will listen for commands, install plug-ins (such as key logging, file transfer commands, and encryption), enable automatic start-up, and modify the way the server software is viewed by the user (if viewed at all). Figure 1 shows us the configuration menu for our trial server. We've installed multiple plug-ins: *srv\_getfile.dll*, for retrieving files from the target computer, *srv\_controll.dll*, for controlling various aspects of the server, and *srv\_interface.dll*, giving us various interface options such as key logging. Additionally, we've set the host process name to "BO2K" so we can easily identify its memory and CPU usage on our target machine (the default setting was EXPLORER). We've configured our server to listen for our commands on TCP port 31337<sup>7</sup>, and save keystrokes into a file named KEYLOG.LOG in the root directory of our target's hard drive.

---

<sup>6</sup> Ruppenthal, Eric. "The Back Orifice 2000 Controversy". [http://sourceforge.net/docman/display\\_doc.php?docid=7791&group\\_id=4487](http://sourceforge.net/docman/display_doc.php?docid=7791&group_id=4487)

<sup>7</sup> 31337: "Hackers" sometimes substitute numbers and other characters for letters; this would translate into ELEET, or "Elite". For more information regarding this subculture, pick up *Hacker Culture* by Douglas Thomas, University of Minnesota Press, 2002.

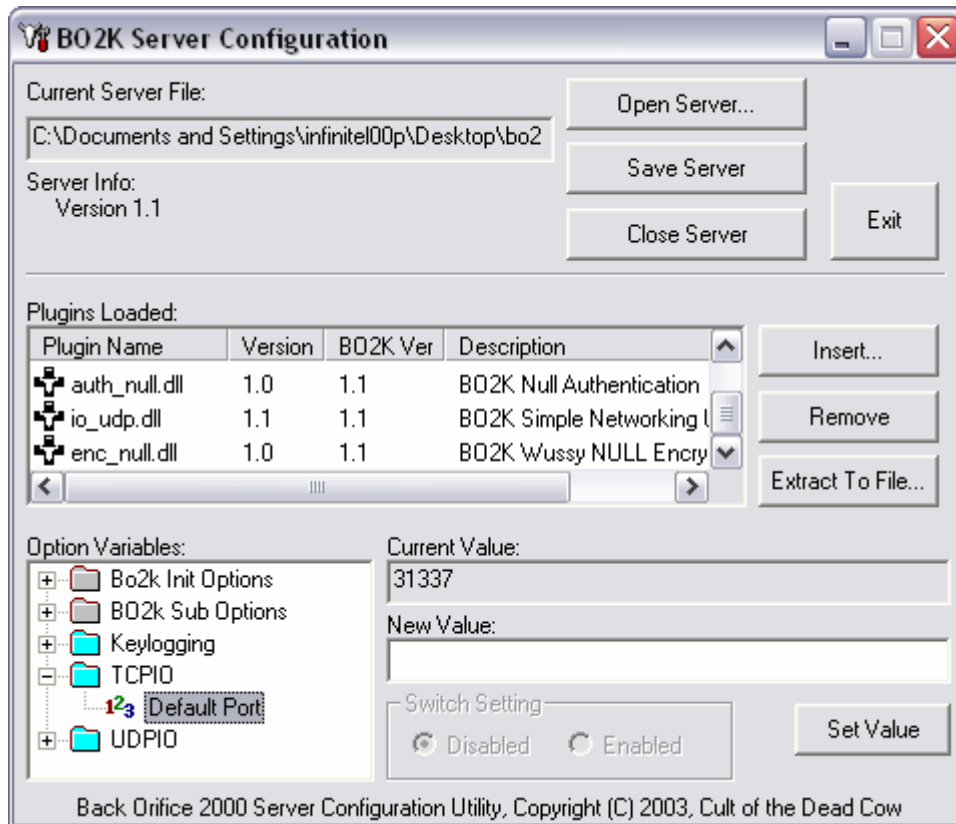


Figure 1

Now that we've completed our configuration, we save the server file (in our case "BO2K copy.exe"), and prepare to copy it to our target machine.

#### *The Plant*

The server file we've just saved needs to be run on the target computer. The toolset doesn't automatically seek out and compromise systems, so the executable must be received via email, disk, download, instant message, or any other method of file transfer, and manually run. A potential tactic useful in this case is to 'bind' the malicious file with another legitimate-looking file. When the file is accessed, the Trojan installs itself.

#### *Listening for the Client*

Once the Trojan has successfully installed itself on the target computer, it opens a TCP or UDP port and listens for any communication from the client. The communication can be encrypted, thus rendering the examination of the contents of single packets to and from a port useless. Using Sysinternals' TCPView port mapping application, figures 2a and 2b detail the state of all TCP and UDP connections on our experimental host machine before and after the server is installed, respectively.

The screenshot shows the TCPView application window with a table of listening ports. The table has five columns: Process, Protocol, Local Address, Remote Address, and State. The processes listed include alg.exe, lsass.exe, svchost.exe, and System. The states are either LISTENING or CLOSE\_WAIT.

Process	Protocol	Local Address	Remote Address	State
alg.exe:204	TCP	mfusion:3001	mfusion:0	LISTENING
lsass.exe:680	UDP	mfusion:isakmp	.*	
svchost.exe:1144	UDP	mfusion:3007	.*	
svchost.exe:1272	TCP	mfusion:5000	mfusion:0	LISTENING
svchost.exe:1272	UDP	mfusion:1900	.*	
svchost.exe:1272	UDP	mfusion:1900	.*	
svchost.exe:856	TCP	mfusion:epmap	mfusion:0	LISTENING
svchost.exe:960	TCP	mfusion:1025	mfusion:0	LISTENING
svchost.exe:960	TCP	mfusion:3014	mfusion:0	LISTENING
svchost.exe:960	TCP	mfusion:3019	mfusion:0	LISTENING
svchost.exe:960	TCP	mfusion:3022	mfusion:0	LISTENING
svchost.exe:960	TCP	mfusion:3002	mfusion:0	LISTENING
svchost.exe:960	TCP	mfusion:3003	mfusion:0	LISTENING
svchost.exe:960	TCP	mfusion:3022	unknown.level3.net:http	CLOSE_WAIT
svchost.exe:960	UDP	mfusion:ntp	.*	
svchost.exe:960	UDP	mfusion:ntp	.*	
System:4	TCP	mfusion:microsoft-ds	mfusion:0	LISTENING
System:4	TCP	mfusion:1025	mfusion:0	LISTENING
System:4	TCP	mfusion:netbios-ssn	mfusion:0	LISTENING
System:4	UDP	mfusion:microsoft-ds	.*	
System:4	UDP	mfusion:netbios-ns	.*	
System:4	UDP	mfusion:netbios-dgm	.*	

Figure 2a

Notice only system services are currently listening on ports, and most are listening locally (that is, waiting to talk to each other). After the BO2K server has installed itself silently, we can now see the listening state on port 31337.

This screenshot is identical to Figure 2a, but with an additional row at the bottom. The new row, highlighted with a red box, shows a process named UMGR32.EXE:1088 listening on TCP port mfusion:31337 to mfusion:0.

Process	Protocol	Local Address	Remote Address	State
alg.exe:204	TCP	mfusion:3001	mfusion:0	LISTENING
lsass.exe:680	UDP	mfusion:isakmp	.*	
svchost.exe:1144	UDP	mfusion:3007	.*	
svchost.exe:1272	TCP	mfusion:5000	mfusion:0	LISTENING
svchost.exe:1272	UDP	mfusion:1900	.*	
svchost.exe:1272	UDP	mfusion:1900	.*	
svchost.exe:856	TCP	mfusion:epmap	mfusion:0	LISTENING
svchost.exe:960	TCP	mfusion:1025	mfusion:0	LISTENING
svchost.exe:960	TCP	mfusion:3014	mfusion:0	LISTENING
svchost.exe:960	TCP	mfusion:3019	mfusion:0	LISTENING
svchost.exe:960	TCP	mfusion:3022	mfusion:0	LISTENING
svchost.exe:960	TCP	mfusion:3002	mfusion:0	LISTENING
svchost.exe:960	TCP	mfusion:3003	mfusion:0	LISTENING
svchost.exe:960	TCP	mfusion:3022	unknown.level3.net:http	CLOSE_WAIT
svchost.exe:960	UDP	mfusion:ntp	.*	
svchost.exe:960	UDP	mfusion:ntp	.*	
System:4	TCP	mfusion:microsoft-ds	mfusion:0	LISTENING
System:4	TCP	mfusion:1025	mfusion:0	LISTENING
System:4	TCP	mfusion:netbios-ssn	mfusion:0	LISTENING
System:4	UDP	mfusion:microsoft-ds	.*	
System:4	UDP	mfusion:netbios-ns	.*	
System:4	UDP	mfusion:netbios-dgm	.*	
UMGR32.EXE:1088	TCP	mfusion:31337	mfusion:0	LISTENING

Figure 2b

### Controlling the Server

Now that we've established our R.A.T., we connect to it using the client application. For this, we need to know the port that we've set up our server to listen on, as well as the IP address of our unsuspecting host machine. Since IP addresses are typically dynamic, port scanning could be employed to find infected machines in a block of addresses to re-connect after the plant. Additionally, a password (configured by us in the above step) is needed so as to authenticate the

client with the server. Using this authentication clients without access rights cannot control a server they haven't set up.

To demonstrate the basic functionality of our newly planted R.A.T., let's take a brief look at the key-logging functionality. Figure 3 shows the control window from the client's perspective, and allows the user to control the starting and stopping of multiple features, including saving all keystrokes to a file. After we've started the logging, captured some keystrokes, and stopped the logging function, we can view the results from within the client, or via the saved text file.

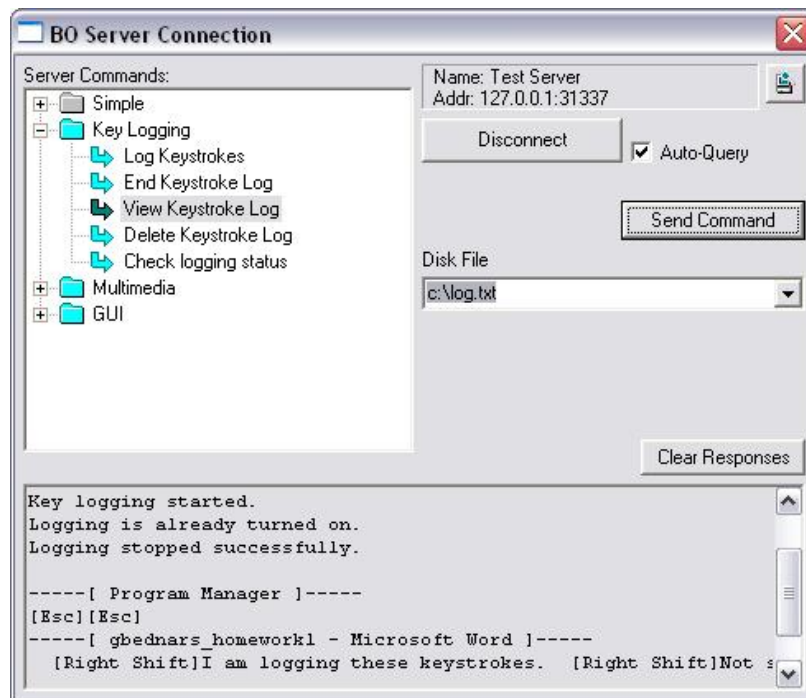


Figure 3

An interesting (and undocumented) feature using the key-logger is that it defaults to setting the 'hidden' attribute to true for the key log file on the server's machine. When searching for the logged keystroke file from the above example, I had to un-hide any hidden files before I could view it.

We can further use the client to send commands to the server to transfer files between machines, open chat dialogs, take screen captures, and a multitude of other functions as added by plug-ins.

#### Section 4 – Published Uses and Derivative Tools

---

The nature of the BO2K program does not lend itself to large published uses (aside from analysis and criticism). As it is not a virus, it does not replicate itself into other programs, nor does it spread automatically across networks as a worm does. This being said, the vast majority of uses of this program happen from user-to-user, and is largely overlooked by the popular press. Additionally, most anti-virus software will detect and remove the BO2K software if found.

Derivative tools, as such, have come in two separate forms: first, the BO2K platform is open-source. This gives the ability to any deft programmer with an understanding of C++ the ability to add functionality via the plug-in utilities. Some of the current add-ins provide functionality such as emailing the server's IP address to a pre-defined mailbox, broadcasting the server's IP address to an IRC channel (you'll note that port scanning a block of IP addresses is now not necessary), or adding packet-sniffing functions useful to further network foot-printing and data capture activities.

Secondly, the ease of use and range of functionality of BO2K has given rise to many similar tools. As detailed in section 1, NetBus and SubSeven are two closely related tools. All give you the ability to control almost every aspect of a target computer's functionality, although some come pre-packaged with more insidious tools for prank or destructive uses.

## **Section 5 – Potential Defense**

---

Back Orifice has three major drawbacks in its usefulness as a potential Trojan horse. First, it is well known and documented by all major computer security and anti-virus firms. Second, relies on specific port communication (TCP or UDP) to communicate. And finally, requires a user to actively run the server file (assuming the potential intruder does not have access to the target host already).

Since all major anti-virus vendors consider this application suite a Trojan horse, their software will identify and remove any running instances. Additionally, these companies recommend turning off unnecessary services, watching your listening TCP and UDP connections for anything suspicious, and keeping your operating system patched.

BO2K's reliance on TCP and/or UDP ports offers a second prescription for defense: firewalls. If a network's firewall is properly configured to only allow traffic on specific ports (those that are commonly used and authorized, such as SMTP and WWW) the server and client will not be able to communicate. Additionally, ingress and egress filtering can assure that only authorized connections are allowed to be created between the host computer and any other host in the wild.

Finally, the need for someone to activate the server can be defended on two fronts: employee training and email security policy. In order to entice users to run the server file, an attacker may bind it to another program (as mentioned above) or use simple social engineering techniques to trick the user into launching the file. Training employees, or staying vigilant yourself, against suspicious looking email attachments would dramatically reduce the potential for the back-door to be installed. Since the typical delivery method is email, servers could block or quarantine files ending with an EXE extension before they reach the user. Currently, Microsoft Outlook considers all files with this extension as "Level 1", or unsafe, and blocks them from being opened, although there is a lengthy process available to remove this function<sup>8</sup>.

## **Section 6 – Policy Concerns for Information Technology Organizations**

---

Information technology organizations, for our intents and purposes, can be considered professional groups that aide others in the creation, deployment, and management of networked information systems. These groups are typically responsible for the initial security level of the systems they create, in accordance with the policy delivered by their customers. The Back Orifice tool, or any other rouge R.A.T. for that matter, poses some specific challenges. BO2K uses deception and positioning as its main strategies in compromising a host, and the IT organization must take appropriate steps to shore their defenses against these tactics.

The major concern for an IT group is this: how do you create and maintain an information infrastructure that can guard against this threat? The strategy must revolve around the needs and requests of the customer or parent organization, but implement this strategy in the placement and configuring of technology.

As briefly described above, BO2K's reliance on deception is, at the same time, one of its greatest weaknesses and greatest strengths. It disguises itself as an application it is not, hides its presence once installed, and becomes persistent, running itself every time the host starts their

---

<sup>8</sup> Microsoft Knowledge Base Article 290497.  
<http://support.microsoft.com/default.aspx?scid=kb;EN-US;290497>

computer, without their knowledge<sup>9</sup>. The R.A.T. conceals its intent and extent of attack, hoping the target will defend the wrong assets (do you feel so secure that you chose not to encrypt your password files?) or build defenses of the wrong magnitude (does a firewall that does not block inbound UDP connections to a dynamic port really protect you?). For an IT organization to successfully deploy a host or a network of hosts for a customer, they must take a number of steps. Firstly, blocking EXE files coming into the network via email attachments would stop the most obvious attack route for the R.A.T. Additionally, since these files can be compressed into a different format, or bound to other software, the group should have virus and Trojan identification software installed and ready to clean files at the server level, so as to never allow the malicious code to arrive in a user's inbox. Secondly, to combat other methods of compromise, the group should institute a policy to disallow communication over ports other than what is explicitly defined as necessary. Although this may cut off customers from services such as instant messaging across the entire Internet, it will also block further avenues of attack.

The positioning poses an interesting threat to strategic development of a network, especially in the case of remote "administration" tools. One of the most challenging aspects of developing a network is to create it in a fashion where a R.A.T. can only (if at all) infect an area of the network where it will do little damage. The damage posed by BO2K could be the direct theft of password files, intellectual property, or personal documents. Using the notion of network "high points" and "low points", the architects of the network should aim to keep the Trojan in a low point, where it can access no sensitive data, by compartmentalizing storage areas and enforcing strict access rights (i.e. layered security). By keeping secure documents encrypted, or using authentication methods such as one-time passwords, the effectiveness key logging and document theft would be greatly reduced.

Finally, another policy concern for an IT group is to make its customer(s) aware of the risks posed by this particular suite of software. Being the subject matter experts on all things IT-related, the group should be able to develop information relevant to the customer's business and convey the dangerous nature of having a non-authorized R.A.T. present on their systems. The customer may be contracting this IT group because it recognizes that infrastructure design and deployment is not their forte, but the IT organization must realize that this does not give them carte blanche authority over blocking services and imposing policy on their customer.

## **Section 7 - Policy Concerns for Management of Large Corporations**

---

Large corporations are the flip-side of the IT Organization issue: they must contend with the on-going use and policy development for their information infrastructure after the contracted group has completed their installation. Management must stay apprised of new threats, and take appropriate actions to combat them. Mainly, management of large a corporation is responsible for defining critical assets, the training their users, as well as specific organization challenges, such as mergers, spin-offs, and cross-divisional policy differences.

The first and foremost policy concern is the definition of critical assets. What can the corporation afford to have compromised, if anything? Assets such as database information on intellectual property, human resource data, or financial information may be the most critical. Perhaps survivability of their online e-commerce application is paramount. Either way, the management must be able to definitively rank these assets, so defenses can be placed and infrastructure built to most effectively protect them. Assuming something like BO2K is received and installed by a network user, separating these critical areas from open-access by network users (or perhaps imposing a comprehensive access policy) they can help ensure safety from theft or destruction.

Very closely related is the matter of employee education. Continuing on the fact that BO2K employs deception as its main route to compromise, users need to be trained to not open

---

<sup>9</sup> Privacy Software Corporation. "Back Orifice 2000 Trojan Horse Program". <http://www.nsclean.com/psc-bo2k.html>

attachments unless they expect them (and are from trusted sources). Even attachments from trusted sources need to be scanned for viruses. Employees of a large corporation should be aware of a direct contact route to local system administrators or information security officers to report suspicious or infected files, and management should create a policy as such. Regardless of how hardened the exterior of the network is, exposing a soft user base can undermine the security of an entire infrastructure, with dire consequences.

Lastly, corporations may go through various organizational changes throughout their existence. As management is typically at the head of these changes, they need to fully consider the implications to security. If a corporation merges with or acquires another company, that company's security problems are now theirs, too. When two information networks are merged, what new exposure is created? Additionally, large corporations are typically segmented into multiple divisions, occasionally maintaining their own information systems. This leads to two problems: first, these divisions may maintain different levels of openness and secured layers. This creates the same issue as one company acquiring another with a foreign IT system. Secondly, the managers of the various divisions may resist change or implementation of various security measures due to cost, usability, and other factors. To allay these issues, top management should create a combined policy to uniformly manage all divisions from a central point, or at the very least ensure the security practices and definition of critical assets is consistent across the company. Again, the weakest link in the chain (network) could expose the entire infrastructure to compromise.

## **Conclusion**

---

BO2K is an excellent example of very powerful administration tool. As such, the opportunity for abuse of those powers creates an overwhelming security issue for both single hosts and large-scale network. Even though the keepers of this project have been touting its legitimate uses and surrounding themselves with people who agree with them (as those whose conscious conflicts with their actions typically do), the implication of compromise is obvious: data theft, loss of network control, identity theft, and more. The good news, though, is that this particular tool is well defended against by the application of some basic policies and technologies. With the proper deployment of a strategic infrastructure, training of employees, and regular review of security policy (including network audits), you can maintain privacy, integrity, and availability of your assets.