Global Information Assurance Certification Paper

Insecure Default Usage of Tabular Data Stream

GIAC Certified
Incident Handler

Practical Assignment

Version 3.00

by Chuck Gerhardt

Attended Online Hacker
Techniques, Exploits,
and Incident Handling
Course April 23, 2004

Assignment Submitted
October 4, 2004

Table of Contents

List of Figures

Abstract

This paper is to partially fulfill the requirements of the GIAC Certified Incident Handler certification. The purpose is to illustrate a vulnerable network protocol used by an enterprise database that is easily exploited by an insider. The vulnerability is described and analyzed. Next, a hypothetical insider attack is described. Finally, the incident handling process is described in the context of the hypothetical victim organization.

Document Conventions

When you read this practical assignment, you will see that certain words are represented in different fonts and typefaces. The types of words that are represented this way include the following:

`command`	Operating system commands are represented in this font style. This style indicates a command that is entered at a command prompt or shell.
`filename`	Filenames, paths, and directory names are represented in this style.
`computer output`	The results of a command and other computer output are in this style
URL	Web URL's are shown in this style.
Quotation	A citation or quotation from a book or web site is in this style.

Statement of Purpose

Many organizations have perimeter defenses which they consider satisfactory, or even have isolated networks. In such environments, internal security is frequently held to a much lower standard than perimeter security. Such networks give a false sense of security because although they are hard on the outside, they are soft on the inside.

Database systems in such environments commonly operate with default security configurations for convenience. The Sybase Tabular Data Stream protocol is one such example. Although Sybase added Secure Sockets Layer capability to Tabular Data Stream in 2001[1] and provided encrypted login capability long before, many customers continue to use the unencrypted configuration for Sybase Adaptive Server Enterprise and Sybase Open Client. This creates a vulnerability in which the even the most sensitive function, administrator login, is transmitted in clear text.

The United States Secret Service and the Carnegie Mellon University Software Engineering Institute's CERT Coordination Center announced in August 2004 the results of an Insider Threat Study[2]. In 87% of the cases studied, authorized personnel used legitimate commands to carry out attacks. The majority of attacks were planned in advance for the purpose of financial gain.

In "Security is Harder than You Think," John Viega and Matt Messier pointed out that insider attacks are more challenging to address than purely technological exploits. The "can't happen here" attitude is a key factor.[3]

The objective of this paper is to describe a hypothetical environment and a hypothetical scenario that illustrates the confluence of the circumstances described above. A simulated insider attack will be executed on a fictional organization using a simulated network.

Background Information

It is helpful to have some background information on the hypothetical environment and the characters within.

Site X is a contracted system provider for a custom analytic application that is critical to certain functions at Organization Y. Site X is a very small company

[1] http://sybooks.sybase.com/onlinebooks/group-as/asg1250e/whatsnew/@Generic__BookTextView/859
[2] http://www.secretservice.gov/ntac/its_report_040820.pdf
[3] Viega and Messier, p.62.

with a unique tool. Organization Y is much larger than Site X. The two entities have agreed to a long term contract that provides for service level guarantees and penalties for system outages during hours of service. Both cost and security are concerns for Organization Y and the two entities have agreed to a number of technical and operational compromises to meet mandatory financial constraints. System improvements, staff training, pay increases, disaster recovery, and external audits were dramatically curtailed.

The Site X technical staff is a small team composed of web/database programmers, desktop/network administrators, a UNIX administrator, a database administrator, and a technical manager.

Web/Database programmers	Alice, Bob
Desktop/Network administrators	Cindy, Dale
UNIX/Network administrator	Saul
Database administrator	Debra
Technical manager	Manuel

Figure 1: Site X Technical Staff

The Site X technical staff gets a salary review every year on the anniversary of their start date. Dale has just received his annual review and is deeply unhappy with it. His work was good, but Site X wants to reduce costs.

The Exploit

The default installation of Sybase ASE and Sybase Open Client uses an insecure network protocol configuration. There is no specific Common Vulnerabilities and Exposures (CVE) number for this vulnerability. Awareness of the problem is evident in the inclusion of a command line option to encrypt logins for Sybase utilities. For example, the Sybase 12.0 Utility Programs for UNIX Platforms manual from 1999[4] documents the "`isql`" utility. It includes a "`-X`" parameter for "`isql`" which causes the tool to send an encrypted password across the network instead of using clear text. The "`-X`" can be found in the "`isql`" utility from 1995. Since many customers permit clear text passwords to be transmitted on their internal networks, they are vulnerable to observation by network analyzers (also known as sniffers) such as Ethereal.

Ethereal, is a free network traffic analyzer, commonly used by network administrators for troubleshooting. Because it can capture network packets and because many network protocols fail to encrypt passwords, it can also be used maliciously to uncover sensitive information that travels across the network. Many sites have an acceptable use policy that prohibits network analyzers,

[4] http://sybooks.sybase.com/onlinebooks/group-as/asp1200e/uxutil/@Generic__BookTextView/15469;hf=0;pt=15469

except in cases where they are specifically authorized for network administrators. Since network administrators are trusted to use network analyzers, they are able to uncover passwords for systems that they are not ordinarily authorized to access.

This paper describes the security misadventure of Site X in which a trusted employee takes revenge after an unsatisfactory salary review. In this incident, a disgruntled employee uses social engineering to persuade a database administrator to access a compromised system and then uses the `ethereal` tool to uncover a privileged account password for a sensitive Sybase database by inspecting a Tabular Data Stream login packet. Finally, the perpetrator uses the acquired information to steal data, sabotage the system, and extort a "severance package".

Exploit Name

Tabular Data Stream transmits password in clear with default configuration

Two CVEs (Common Vulnerabilities and Exposures) refer to the Tabular Data Stream protocol, but none of them describe the default install vulnerability or clear text password vulnerability. CVE-1999-0999[5] describes a remote denial of service attack against SQL Server 7.0. CAN-2003-0327[6] describes a denial of service attack on Sybase.

Operating System

The vulnerability exists when a customer configures the Sybase product in an insecure manner. This is independent of the operating system in use. Therefore, the operating systems affected by this vulnerability include any operating system that can function as a Sybase client or as a Sybase server. Sybase documentation indicates that the currently supported versions of Windows are NT4 Service Pack 6a, Windows 2000 Service Pack 3 or later, Windows 2003 Service Pack 1 or later, and Windows XP.[7] Sybase documentation indicates that the currently supported versions of UNIX are HP Tru64 UNIX version 5.0a or later, HP 9000/800 version 11.0 or later, HP-UX Itanium 11.23 or later, IBM RS/6000 AIX version 4.3.3 and 5.1, Sun Solaris version 2.8 or later (SPARC), Linux, Linux Itanium, and SGI IRIX 6.5.18 or later. [8]

[5] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0999
[6] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0327 and http://www.rapid7.com/advisories/R7-0016.html
[7] http://sybooks.sybase.com/onlinebooks/group-cn/cnp1251e/ocsinst
[8] http://sybooks.sybase.com/onlinebooks/group-cn/cnp1251e/ocsinunx

Protocols/Services/Applications

Founded in 1984, Sybase was a pioneer of client-server computing.[9] Sybase client-server computing includes a server component which manages data storage and executes database requests on behalf of clients, as well as a client component which makes requests to the database server and makes the results available to the user. This architecture allows the client and server components to work together even when they are located on different computers.[10] The foundation of Sybase client-server communication is the Tabular Data Stream protocol.

The Sybase Tabular Data Stream (TDS) protocol is a database communication layer built on top of a network layer -- in this case TCP/IP. The Transmission Control Protocol (TCP) is built on top of Internet Protocol (IP). A Sybase client needs to specify an IP address and a TCP port to send a Tabular Data Stream packet to a Sybase server.

The Internet Protocol is a network-layer protocol that contains addressing information which allows IP packets to be directed to their intended destination.[11] The IP packet contains the source IP address and the destination IP address. The Transmission Control Protocol packet is contained in the Data portion of the IP packet.

The Transmission Control Protocol is a transport-layer protocol that provides for reliable transmission of blocks of data by using sequence numbers and acknowledgement numbers.[12] The TCP packet contains the source port and the destination port. The Tabular Data Stream packet is contained in the Data portion of the TCP packet.

The communication sequence between the client and Sybase server begins with a TDS login packet. The beginning of a Tabular Data Stream login packet with default security configuration has the format shown below.[13]

byte	**field**
0	packet type, 1 byte, 0x02 indicates a TDS4.2 or TDS5.0 login packet
1	last packet indicator, 1 byte
2	packet size, 2 bytes
4	channel, 2 bytes
6	packet number, 1 byte

[9] http://www.sybase.com/about_sybase
[10] http://en.wikipedia.org/wiki/Client/server
[11] http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/ip.htm #xtocid2
[12] http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/ip.htm#xtocid16
[13] http://www.freetds.org/tds.html

7	window, 1 byte
8	client host name, 30 bytes
38	host name length, 1 byte
39	user name, 30 bytes
69	user name length, 1 byte
70	password, 30 bytes
100	password length, 1 byte
...	(remainder omitted)

Figure 2: Tabular Data Stream Login Packet

The Tabular Data Stream has additional complexity and functionality, but this information is sufficient for observing a Sybase client login using the default authentication scheme. There are more sophisticated authentication schemes available, such as PAM, Kerberos, and LDAP. Many Sybase customers continue to use default authentication.

The standard Sybase utility for connecting to the database server is "`isql`", the interactive SQL client.[14] The "`isql`" utility has a variety of parameters that influence the behavior of the tool or specify the details of the connection.

`isql`	Interactive SQL parser
`-U username`	Specifies the login id to use when connecting to the Sybase server
`-S servername`	Specifies the server name to access
`-P password`	Specifies the server password, if omitted then isql will prompt for a password

Figure 3: isql Command Syntax

The "`isql`" utility has additional parameters that modify behavior, formatting, and security configuration. The worst common security practice among Sybase customers, providing the password on the command line, is addressed at the end of this paper.

Transact-SQL is the dialect of SQL provided by Sybase. It has the standard SQL capabilities as well as some extensions that are specific to Sybase. When used with "`isql`", a SQL query, data modification, command, or stored procedure must be followed with a command terminator, "`go`", on a line by itself. A SQL query is used to extract data from the database. A data modification statement is used to add, change, or delete data. A SQL command is used for administration. A stored procedure is a compiled collection of Transact-SQL syntax that can perform complex business functions or administrative tasks.

[14] http://sybooks.sybase.com/onlinebooks/group-as/asg1250e/util/@Generic__BookTextView/15937;pt=15937

Exploit Variants

Sybase Tabular Data Stream can communicate over a variety of network protocols, including TCP/IP, IPX/SPX, Named Pipes, and DECnet. By default, the Tabular Data Stream connection is not encrypted. Consequently, any appropriately configured network sniffer is capable of capturing a login packet with a plain text password. While the Sybase product has the ability to connect over SSL and has the ability to encrypt passwords before they are sent to the server, many customers simply do not take advantage of these capabilities.

This simulation uses the "`isql`" utility to illustrate the Tabular Data Stream vulnerability. Any Open Client program would exhibit the same vulnerability when using the default configuration. It is possible to encrypt the password at the client side before logging in to the Sybase ASE server. Utilities with this capability include "`isql`", "`bcp`", and "`defncopy`". The Sybase "`optdiag`" utility does not have the ability to encrypt the password before sending it across the network.

The password sniffing attack could be conducted from a variety of machines. It could be executed on either the server or a specific client without placing the network interface in promiscuous mode. If the network interface is placed in promiscuous mode, then any machine that shares the same network signals with either the server or a client could be used to sniff packets.

In this simulation, a graphical network sniffer is run on the targeted client and the network interface is placed in promiscuous mode. This is the default operation of "`ethereal`". It is also possible to run a command-line version, "`tethereal`", without exposing a user interface. Additionally, both the graphical and command line versions of "`ethereal`" can be executed with promiscuous mode turned off; however this limits the packets that can be captured to those that are intended for the machine or packets that are intended for sets of machines that include it. Running "`tethereal`" with promiscuous mode disabled would be much more stealthy than the easily observed attack described here. The critical feature of this attack is not the tool, which is regularly used, but rather the intent of the user, which is malicious.

Description and Exploit Analysis

A Sybase Windows client connects to a Sybase server in five steps[15]:
1. Requests a connection to a specific server
2. Looks up connection specification in a configuration file (`sql.ini`)

[15] http://sybooks.sybase.com/onlinebooks/group-cnarc/cnp1003e/w95_supp/@Generic__BookTextView/305;pt=305

3. Matches connection specification to network library specification in another configuration file (`libtcl.cfg`)
4. Loads the corresponding network driver
5. Connects to the server using the connection specification

When the client initiates the connection to the server, it begins with the TCP "three-way handshake".[16] It sends synchronization (SYN) request to the IP address and port specified for the desired server, as determined in step 2 above. The SYN request contains an initial sequence number and a flag to indicate a synchronization request. The server responds with a combined synchronization/acknowledgment (SYN/ACK) response to the client. The response contains its own initial sequence number (the SYN part) and the client's initial sequence number plus one (the ACK part). Finally the client sends an acknowledgment response to the server's SYN, which contains the server's initial sequence number plus one.

Next, the client sends the Tabular Data Stream login packet. When Sybase is configured with default connection parameters, and the client is making a simple login without encryption or network authentication servers, then the Tabular Data Stream login packet will contain the user name and password in clear text. At this point the password can be observed by a network sniffer. The same protocol is used for unprivileged and privileged accounts, so that the system administrator account, "`sa`", is just as vulnerable to password sniffing as any other login.

Once authenticated, the client sends an additional login packet containing information related to environment configuration. This is omitted for clarity as it is not part of the vulnerability.

The Ethereal utility is a powerful tool for network traffic analysis which is primarily intended for troubleshooting network problems.[17] It can capture network packets and display very detailed protocol information. It provides a framework for packet analysis, allowing dissectors to analyze the specific protocol of each known packet type, such as Tabular Data Stream. The packet capture capability is built on top of "`libpcap`" (WinPcap on Windows). This modular architecture has allowed a large number of people to contribute to the project so that a wide variety of network protocols can be examined.

In the default mode, Ethereal opens the network interface in promiscuous mode. This allows the computer to detect all signals on the network interface, not just the ones addressed to that machine. The packets are then displayed on the screen. Some of the information is decoded for deeper analysis. The data portion is displayed in the bottom pane of the window. Because the default Sybase configuration does not provide for encrypted SSL connections, the Tabular Data Stream login packet contains the user name and password in clear text.

[16] http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/ip.htm#xtocid17
[17] http://www.ethereal.com/docs/user-guide/

If an attacker is running a sniffer on a machine that can see the login packet, either because the interface is in promiscuous mode or because the login packet is from (or to) the machine, then the Tabular Data Stream login packet can be captured. Once the login packet has been captured, the attacker only needs to look at bytes 39 through 68 to see the user name and to look at bytes 70 through 99 to see the password in clear text.

Exploit/Attack Signatures

Network packet capture can generate very large log files. If a machine suddenly uses a lot of disk space for no apparent reason, a packet sniffer may be in use.

Another symptom of an active packet sniffer is that the network interface for a machine is in promiscuous mode. However, if a network interface is not in promiscuous mode that does not mean there is no sniffer in use.

In the simulated attack documented below, the "`ethereal`" process could have been observed visually. Had "`tethereal`" been used, it would have been less obvious, but still could have been detected with Windows Task Manager.

Platforms/Environments

The Site X network is physically self-contained. There are no external connections, except by tape or compact disc. The users in Organization Y are physically co-located with the Site X staff and equipment.

Victim's Platform

The Site X analytic application accepts data input from media. It is processed by a collection of generic NT4 workstations. The results are stored in a Sybase database on a Sun Ultra Enterprise E3500. The Organization Y users get web reports from an Apache server on another Sun Ultra Enterprise E3500.

Source Network (Attacker)

Generic x86 Processing Node 1 (attacker machine)
NT4 sp6, build 1381
Sybase Open Client 10.0.3/P/PC Intel/Windows NT 3.5/2/1/OPT/Nov 13 1993 00:22:32
Ethereal 0.10.4
WinPcap 3.0

Target Network

Sun Database Server, dbsrv
SunOS Release 5.10 Version s10_63 32-bit
Sybase Adaptive Server Enterprise/12.5.2/ EBF 11955 ESD#1/P/Solaris Intel/OS 5.9/ase1252/1838/32-bit/OPT/Sat May 29 01:59:38 2004

Sun Web Server, websrv
SunOS Release 5.10 Version s10_63 32-bit
Apache Jakarta Tomcat 5.0.28
Sybase jConnect for JDBC 5.5

Generic x86 Processing Nodes
NT4 sp6, build 1381
Sybase Open Client 10.0.3/P/PC Intel/Windows NT 3.5/2/1/OPT/Nov 13 1993 00:22:32

Generic x86 External Access Machine
Windows XP sp2

Norton Internet Security 2005
America Online 9.0

Network Hardware
3Com SuperStack II Dual Speed Hub 500
Cisco 1720 Modular Access Router

Network Diagram

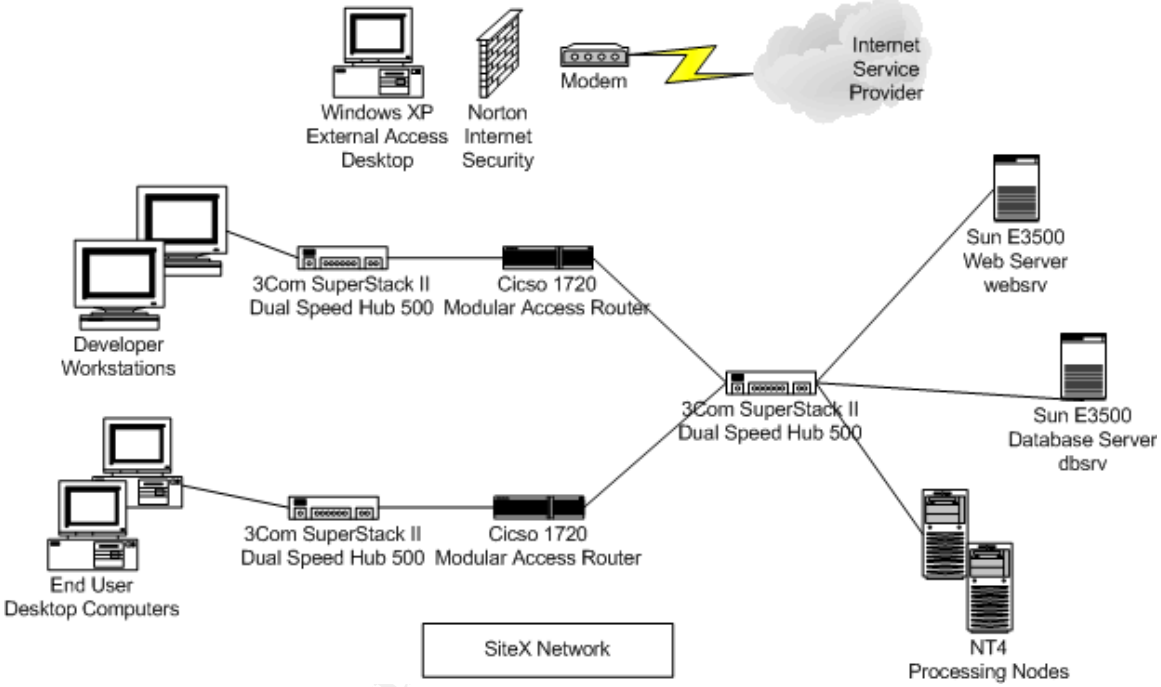

Figure 4: Site X Network Diagram

Stages of the Attack

The attack process has 5 phases -- reconnaissance, scanning, exploiting the system, keeping access, and covering tracks. The disgruntled Site X employee, Dale, will be used to illustrate the attack process.

Reconnaissance

In the reconnaissance phase an attacker gains as much information as possible about the victim. Knowledge of policies, procedures, patterns, and technologies are valuable.

As part of a new budget tightening initiative, Dale was given a token bonus and no raise. Dale began to plot revenge on his employer. At first, he considered it nothing more than an amusing fantasy. Eventually he put his fantasy plan into action when he realized he could use it to get the bonus he felt he deserved.

Dale began by observing the operations of Site X more closely, with a view to causing the company to lose a lot of money and perhaps collapse. He noted the times that people were in the office, the times security guards checked the suite, the data load schedule, the system backup schedule, the tape rotation cycle, the network topology, the phone list, database access, and the patterns of password ownership.

Dale observed that he had full administrative rights on all of the NT4 processing nodes. Those machines interacted with the Sybase database. That database was the key information asset for Site X and Organization Y.

Scanning

In the scanning phase an attacker makes a detailed search for technological vulnerabilities. In particular, the network is examined for weak entry points.

Dale already knew the network infrastructure at Site X since he had helped construct it. He considered that destroying the Sybase data could result in Site X losing the contract with Organization Y. He soon realized that the short weekly tape rotation cycle, only two tapes to conserve expensive media, was a key weakness. If he stole the backup tapes, he would be able to extract a large payment from Site X.

Dale began by learning what he could about Sybase. He read manuals and talked with Alice, Bob, and Debra about the application and Sybase whenever he

could. From them, he learned that the Site X application had a Sybase client on each NT4 processing node. Each node processed data and posted it to the Sybase database server across the network. His coworkers noticed his sudden interest in their tasks, but were not alarmed.

Dale examined the "`SQL.INI`" file on processing node 1 and determined that the server was located at IP Address 172.16.54.129 and that it was listening on port 5000. He also examined the "`SITEX.INI`" file and discovered that it included a login id of "`sitexappl`" and password of "`sitex123`" that he could use to connect to the Sybase server.

Dale soon learned that the Sybase login id for the NT4 processing nodes was quite limited. It could only execute certain stored procedures. Stored procedures are compiled database commands that ordinary users cannot change. He realized that he would need a more powerful account if he wanted to damage the database.

Dale installed a copy of WinPcap and Ethereal on processing node 1. He removed the icon from the desktop. He opened a command window, changed to the "`ethereal`" directory, and started "`ethereal`".

```
> ethereal -k -w sybtrace.pcap
```

The "`-k`" parameter tells the Ethereal tool to immediately start capturing packets. The "`-w sybtrace.pcap`" parameter tells Ethereal to write the captured packets in a file named "`sybtrace.pcap`."[18]

Dale minimized the window and opened a new command window. After logging in to the database with the application password, Dale stopped the packet sniffer and then loaded the file into Ethereal for analysis. He selected "`Find Packet`" from the "`Edit`" menu. The he entered a display filter:[19]

```
ip.addr==172.16.54.129 and tcp.flags.syn
```

He then scrolled down the display until he located the TDS Login Packet. He easily located the application password by simply looking at the login packet. The password was a set of letters and numbers surrounded by dots in the bottom pane of the "`ethereal`" window.

He closed the "`ethereal`" window, deleted the "`sybtrace.pcap`" file, and then emptied the Recycle Bin. He made a mental note of the command so he could quickly restart "`ethereal`" when he was ready to spring his trap.

[18] http://www.ethereal.com/docs/user-guide/ChCustCommandLine.html
[19] http://www.ethereal.com/docs/user-guide/ChWorkFindPacketSection.html

Dale's sudden interest in Sybase could have served as a warning signal.[20] While there is no clear profile of a malicious insider, and technical staff is involved in these crimes, management could have been more vigilant. A sudden interest in a technology outside Dale's area of responsibility in combination with the pay dispute should have raised suspicions.

The "`ethereal`" program is a graphical application. Dale simply minimized the window so that it blended in with the other programs on the toolbar. Any observant person could have seen a minimized window marked "`ethereal`".

Exploiting the System

In the exploiting phase an attacker gains access to the target system. Whether from the inside or the outside, the goal is to elevate privileges to achieve the desired level of control.

The next step in Dale's plan included a bit of social engineering. He needed to get the "`sa`" password from an unwitting accomplice. He waited for Debra, the database administrator, to make her weekly visit to the Site X office. Once she was there he restarted "`ethereal`" and minimized the window and then he asked her to investigate a problem with NT4 processing node 1. Working from her desk, Debra found no connectivity issues and reported that the application code on processing node 1 was successfully connected to the Sybase server. When she reported that there was no apparent problem, he insisted that she needed to verify the connection between the processing node and the database. Dale was concerned that Debra would inquire about the sniffer, but he was prepared to tell her he was using it to investigate the trouble on processing node 1. They went to the console of processing node 1 and she logged in as "`sa`", being careful to provide only the login id and server name, so that the "`isql`" utility would prompt her for the password. Once again she reported that nothing seemed amiss. Then she logged out and left Dale to continue troubleshooting. Relieved, he quickly terminated the packet sniffer and hastily checked for the "`sa`" password.

Debra had two opportunities to foil the attack at this point. She could have observed the "`ethereal`" process running and refused to log in while it was capturing packets. She could have logged in using "`isql -X`" to connect with an encrypted password.

[20] http://www.secretservice.gov/ntac/its_report_040820.pdf

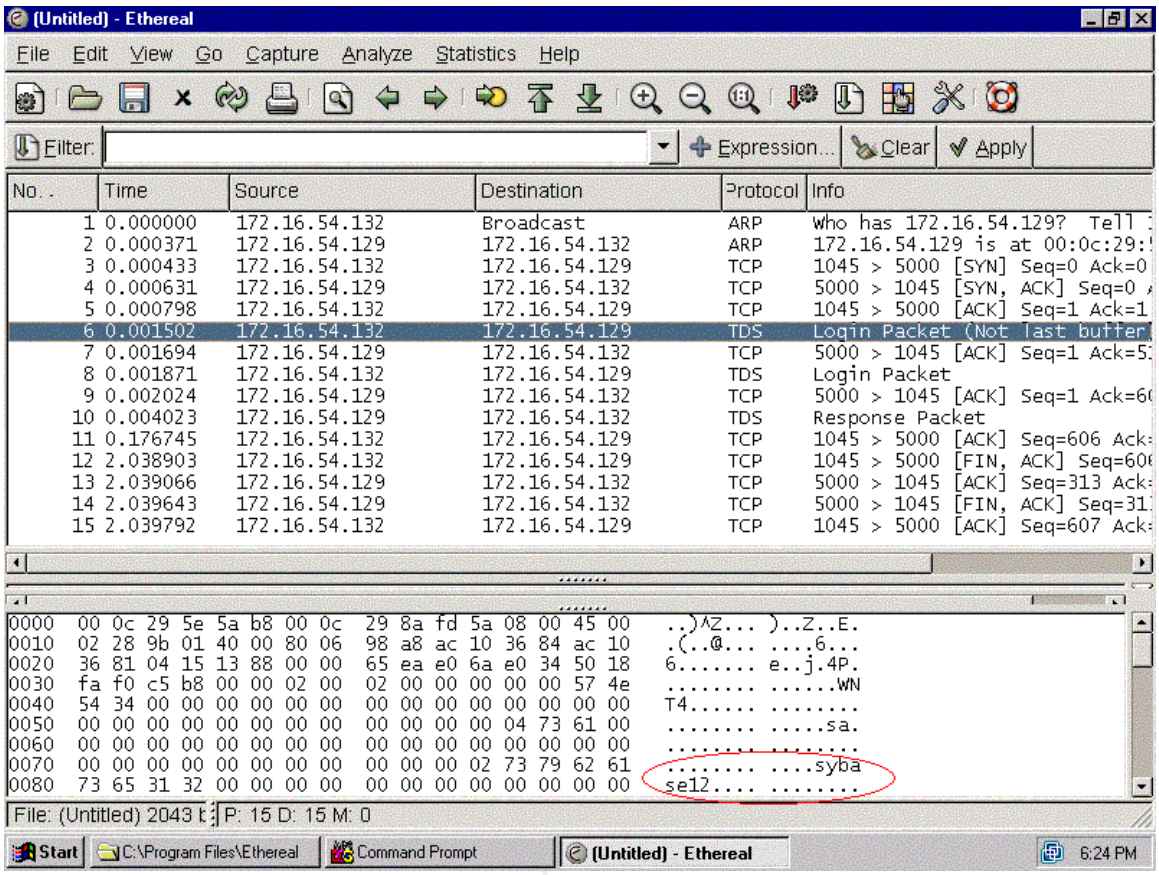

Figure 5: Clear Text sa Password

Knowing he could be interrupted at any moment, Dale wrote the password on a scrap of paper and started to delete the "`ethereal`" directory and the log file. By mistake he sent the files to the Recycle Bin without deleting them. Dale was so excited that he did not notice his error.

The sniffer trace file used a substantial amount of disk space. If Site X had a monitoring infrastructure in place, it could have detected the increased disk usage on the tainted NT4 processing node 1. If the sniffer trace had been allowed to run much longer, it would have filled the hard drive and caused the application to fail.

Keeping Access

In the keeping access phase an attacker creates alternative ways to gain privileged access to the target system.

Dale watched Debra closely for the rest of the day. He wanted to be sure she would not change the password. As soon as she left her desk for a meeting, Dale returned to the computer room. He logged in as "`sa`" on the NT4 processing

node. Then, using instructions from the Sybase manual, he created a privileged account for himself.

```
$ isql -U sa -S ts1252 -P sybase12
1> sp_addlogin 'sitexapp1', 'ur0wned'
2> go
1> grant role sa_role to sitexapp1
2> go
```

The first Sybase Trasact-SQL statement above used a Sybase stored procedure to create a database login account named "`sitexapp1`" (where the last character is a one instead of a lowercase L) with an initial password of "`ur0wned`". The "`go`" keyword is required to submit the statement for processing on the Sybase server. The second statement assigned system administrator privileges to the newly created "`sitexapp1`" account. This provided Dale with a way to log in with administrative privilege, even if somebody changed the "`sa`" password.

Sybase stores login ids in a database table called "`syslogins`". If Site X had a report that identified new accounts on a regular basis, this back door account would have been detected. It might have escaped detection though, since he was careful to create an account that looked similar to an existing application account.

Covering Tracks

In the covering tracks phase an attacker takes steps to remove evidence of wrongdoing to delay or prevent detection. In this scenario, the disgruntled employee made little effort to cover his tracks. Instead he sabotaged the application database.

Dale logged out of the Sybase server and looked to see if anyone was watching. Relieved, he checked the NT4 processing node to make sure that the ethereal directory had been deleted. Once again, he failed to check the Recycle Bin. He also failed to notice that the disk drive was nearly full.

The final step in Dale's plan had to wait until Friday. It was the week before a long holiday weekend. Dale offered to switch Friday tape duty with Alice so she could go home early. That night, he removed the returning tape from the secure storage container. Instead of placing the old tape on the shelf, he placed it in his backpack. He waited for the weekly backup to complete. Then he removed the new tape and placed it in his backpack. He logged on to the Sybase database as "`sa`" and issued another dump to overwrite the application dump file.

```
$ isql -U sa -S ts1252 -P sybase12
1> dump database model to '/tmp/sitexappl.dmp'
```

```
2> go
```

The Transact-SQL statement above[21] backed up the "`model`" database to a file named "`/tmp/sitexappl.dmp`". The model database is used to create all new databases on a Sybase server. It is the smallest possible database. By backing up the "`model`" database on top of the real "`/tmp/sitexappl.dmp`" dump file, Dale erased the production application database dump and replaced it with a database dump that would create an 'empty' database if loaded.

Then he manually transferred the dump files to a fresh tape using the UNIX "`tar`" utility.

```
$ tar cf /dev/rmt/0 /tmp/*.dmp
```

The "`$`" is the UNIX command prompt. The tar utility is the Tape ARchiver for UNIX. The "`c`" parameter tells it to create an archive. The "`f`" parameter tells it to take the next work on the command line as the output file. The "`/dev/rmt/0`" parameter specifies the file name for a tape drive. The "`/tmp/*.dmp`" parameter specifies that all files in the "`/tmp`" directory with a name that ends with "`.dmp`" should be included in the archive.

After carefully labeling and logging the false tape, he placed it in the secure storage container and left the building with the current backup tape as well as the previous week's backup tape.

As he left, he realized that he had not given any serious consideration to what he would do next or the consequences if he was caught. Shrugging, he got into his car and started driving. He was sure he could get immunity from prosecution in exchange for the tapes. He would call Site X the following week.

The tape switch could have been prevented if Site X had required two people to handle the tapes. The damage could have been reduced if more than two tapes had been used in the weekly tape rotation.

[21] Garbus, Chang, Garbus, and Tyrrell, p. 154-157.

The Incident Handling Process

The incident handling process has 6 phases -- the preparation phase, the identification phase, the containment phase, the eradication phase, and the recovery phase. The phases, and their application at Site X, are detailed below.

Preparation Phase

The preparation phase of incident handling is critical to success. The old saw that failure to plan is planning to fail certainly holds true in this regard. A particularly insidious planning failure is the failure to imagine a contingency.

As this simulation illustrates, the Site X team imagined system failures and planned accordingly. They failed to consider the possibility of an insider attack and thus made no plans for such a contingency. Financial pressure, complacency, and inexperience made them especially vulnerable.

To manage costs, Site X hired inexperienced staff and trained them internally in their procedures. All internal procedures, including security practices, had been defined at the beginning of the contract with Organization Y and were occasionally updated as the application code changed. External consultants provided deeper technical skills on a part time basis. Historically, staff was encouraged to pursue self study through a book reimbursement policy, but this policy had been discontinued to reduce expenses.

Existing Incident Handling Procedures

The incident handling approach at Site X was oriented toward equipment failures and disaster recovery. The focus was to maintain system availability during service hours. Failure to provide service during contracted service times resulted in fines to Site X. A call tree was published and maintained. A disaster recovery document was written, but never reviewed or tested. The system administrators were responsible for maintaining, recovering, and repairing the systems.

Site X had a clear understanding of the contractual penalties associated with system outages and data leaks as well as the profit margins associated with those contracts. Accordingly, they scheduled hardware maintenance regularly and validated their database integrity every week. Additionally, they stored failed hard drives and tapes in an on-site container. Every server was recorded in an equipment inventory log book.

The Site X incident handling plan was entirely oriented to maintaining system availability during service hours and notifying management when there was a problem of some kind. In the event of an emergency, the responsible technical staff member was to notify the technical manager Manuel. They would identify the nature of the problem and then the necessary parties would make the repair and restore service. The greatest concern was a hardware problem on one of the Sun machines. They were configured with enough resources to be able to run both the web server and the database. Additionally, enough disk drives were kept in storage to completely replace the disks for one entire machine. They did not feel the need to stock other spare Sun components. They also maintained two complete processing nodes that could be installed on short notice.

The emergency communication plan at Site X consisted of notifying the Site X technical manager, Manuel, if anything happened. The technical manager would communicate with the Organization Y contract contact. A phone list was maintained for all technical staff, the technical manager, and the Organization Y contract contact. This list was distributed to all technical staff. A copy was placed in each locked storage container that was sent off site with backup media.

Existing Countermeasures

Site X anticipated external network attacks, natural disasters, and equipment failures. To address external network threats, the systems were not attached to an external network and wireless access was prohibited; the servers were checked for OS updates and patches every quarter; and minimal services were enabled on the NT4 processing nodes. To address disasters, the systems were backed up quarterly and weekly; installation and configuration logs were maintained; dump tapes and copies of installation media were stored in an off site location. To address equipment failures, the servers were rebooted quarterly; logs were examined weekly; warning lamps were checked daily. On Friday nights, one staff member would collect the backup tape and put them in a secure storage container for offsite storage.

The only outside access allowed at the site was on an isolated computer that could connect to an Internet Service Provider and print messages. It was also used to download vendor software updates, which were burned to compact disc. This machine was not allowed on the network. Because it had a modem connection to the outside world, it was considered the greatest risk to the network. Norton Internet Security was installed on the computer. Manuel insisted that the machine be powered off unless needed for a specific task. When used there would always be two people working, with one operating the computer and one observing the operator. Manuel carefully logged all access to the machine and reported it to Organization Y every month.

Each group maintained their own password store. The UNIX administrator, Saul, never documented the "`root`" password, but only shared it with the technical manager verbally. Since it was based on specific letters from a memorable phrase they felt they had no risk of forgetting it. The database administrator, Debra, gave a copy of the "`sa`" password in a sealed envelope to the technical manager, Manuel. He kept it in a small safe in the Site X office. The network administrators kept a list of "`administrator`" passwords, a different one for each machine, on a sheet of paper which they photocopied regularly and kept in a binder on a shelf.

Incident Handling Team

Site X approached incident handling as an exception to ordinary processing and thus used ad hoc working teams to handle incidents. The members of the team were assigned according to the nature and severity of the incident.

Manuel, the technical manager, always handled the liaison role. In general, a problem with a processing node would be handled by the network staff. A software problem with the application would be handled by the developers. A more severe or complex problem might also require the database administrator, the UNIX administrator, or both to be added to the team. In this simulation, the severity of the incident required all staff members.

The Site X technical manager functioned as the point of contact during any system outage. The Organization Y contract contact and every technical staff member had the technical manager's cell phone number. They occasionally joked about him being hit by a bus, but no steps were taken to mitigate the risk due to security, cost, and relationship management concerns.

The Site X network administrators, Cindy and Dale, maintained a help desk phone number which was communicated to all staff. Their primary task was to keep the NT4 processing nodes running. They also provided application support and had a small toolkit of spare disks, cables, recordable compact disks, and installation media.

The Site X developers, Alice and Bob, were responsible for application development. When there were system problems, they were consulted so that Site X could avoid the expense of contacting either the database administrator or the UNIX administrator, both of whom were contractors.

Policy

Site X required all staff and consultants to sign nondisclosure agreements and acceptable use policies when they joined. Every login session included a banner

reminding of acceptable use policies and monitoring practices. Every user and developer saw a statement that they could expect no privacy every time they logged on to their computer.

Site X had a written policy against weak passwords. Every user was told to use a password containing numbers and letters, although no steps were taken to enforce this policy. Network administrators checked monitors and keyboards for sticky notes with apparent passwords and publicly confronted users when they were found.

Site X did not use any intrusion detection tools. Their network was physically self-contained. The network administrators manually scanned for wireless devices and modems on a regular basis. Remote access of any kind was not permitted. Because of this network isolation, Site X had no policy for peer notification and no partner network agreements.

Identification Phase

The identification phase is the beginning of the active portion of incident handling. An organization must determine if an observed anomaly is in fact an adverse occurrence. It is necessary to carefully assess the situation to determine the actual cause of the event.

Incident Timeline

The table below summarized the incident handling timeline.

08:30am	Cindy sees that Site X application is not working
09:00am	Manuel, Alice, Bob, and Cindy perform initial investigation
09:30am	Debra investigates database
10:00am	Saul investigates UNIX
10:15am	Debra determines database dump is invalid
10:30am	Debra stops database, Saul backs up UNIX
11:00am	Team develops plan to restore service
12:00pm	Team creates alternate database server instance
03:00pm	Debra loads alternate database from very old dump
04:30pm	Team examines dumps from database server restore set
05:00pm	Team examines log files from database server restore set, concludes outage is likely result of deliberate action
06:00pm	Team considers possible scope of attack
07:00pm	Team decides to change all passwords
08:00pm	Team begins reconstructing processing nodes

Figure 6: Incident Handling Timeline

Identification Phase Narrative

At 8:30am on Tuesday morning after a holiday weekend, Cindy arrived at work, logged in to her computer, and attempted to access the Site X application. She was unable to get past the login page of the application. To verify that the web server was functioning, she attempted to connect to a static web page. The page was served immediately, so she suspected a possible problem with the database server.

Cindy called the technical manager, Manuel, on his cell phone to let him know of a possible outage. He notified the Organization Y contract contact and promised to provide a status update as soon as possible. Then Manuel called the contract UNIX administrator, Saul, and the contract database administrator, Debra, to let them know their services may be needed that day.

While waiting for the other technical team members to arrive, Cindy verified that the UNIX machine that hosted the database was available. She examined all the equipment in the computer room for fault lights. She logged on to the UNIX machine with a limited privilege account, executed a process status and observed that the Sybase database server was running.

```
$ ps -fu sybase
     UID   PID  PPID   C    STIME TTY          TIME CMD
  sybase  2042     1   0 20:53:11 pts/4       0:00 /bin/sh
./RUN_ts1252
  sybase  2043  2042   1 20:53:11 ?           0:31
/opt/sybase/12.5.2/ASE-12_5/b
in/dataserver -d/export/home/sybase/data/master.da
```

The "`ps`" UNIX command shows the status of processes. The "-f" parameter indicates that the process status should be reported in 'full' format. The pair of parameters, "`-u sybase`", indicate that only processes owned by the "`sybase`" user should be reported.

Thinking that one of the developers might have introduced a bug in the previous week's release, Cindy started filling out a Software Error Recovery Log form.

Manuel, Alice, and Bob arrived at the office at 9:00. Manuel reviewed Cindy's initial results and directed Bob to check the database server. Bob checked the log files for the batch job they had implemented on Friday. It indicated two successful runs on Friday and one failed run Monday at 6:30 PM. He connected to the database server with "`isql`". The connection was successful. He attempted to change to the application database and examine the new table they had created on Friday, but the command to change to the "`sitexappl`" database failed. He abandoned his plan to check the new table.

```
$ isql -U sitexappl -S ts1252
Password:
1> use sitexappl
2> go
Msg 911, Level 11, State2:
Server 'ts1252', Line 1:
Attempt to locate entry in sysdatabases for database
'sitexappl' by name failed - no entry found under that
name. Make sure that name is entered properly.
```

The application database was not found. Bob and Alice queried the server and verified that the application database did not exist. Bob immediately notified Manuel, who advised the Organization Y contact that the outage was likely to last more than one hour. Manuel called Debbie and asked her to come in to Site X for the day. Alice contacted the offsite storage vendor to retrieve the Friday backup tapes.

Manuel questioned Alice and Bob about the Friday release. They examined the test logs, the batch logs, and the source code. Cindy took notes, keeping a log of each hypothesis and evidence. The group agreed that the application code could only have caused a problem if something was wrong with the database server or the UNIX server. Manuel called Saul and asked him to come in to Site X for the day.

At 9:30am Debra arrived at Site X and was immediately asked to check the Sybase server for problems because it had lost the application database. Debra connected to the server and issued a database consistency check command for the master database which controls the server.

```
$ isql -U sa -S ts1252
Password:
1> dbcc checkcatalog(master)
2> go
1> dbcc checkdb(master)
2> go
```

The "`dbcc checkcatalog(master)`" command validates the contents of the system tables in the "master" database. The "`dbcc checkdb(master)`" command checks the integrity of all tables in the master database. The "`Password:`", "`1>`" and "`2>`" items are interactive prompts from the "`isql`" utility.

Her database consistency check and her review of the Sybase error log indicated that nothing was amiss. She asked Manuel if anybody had been working on the database that weekend. Alice and Bob angrily denied working over the weekend. Manuel was able to calm them down, but he was becoming nervous himself.

At 10:00am Saul arrived at Site X and was immediately asked to check the UNIX server for problems because the Sybase server was unable to find the application database. He examined the UNIX logs and determined that nothing was amiss. He asked what was changed recently. Alice and Bob were becoming highly defensive since they felt that Manuel was trying to blame them for the outage. Debra and Saul calmed them down while Cindy kept quiet and took more notes.

The team quickly determined that the simplest recovery technique would be to load the on-disk database dump of the application. Debra objected because a missing database indicated a serious problem. If the original cause was not corrected, then they risked another outage. They agreed to verify that the database had been dumped to disk before they rebooted the machine to see if any disks failed when the machine was powered on.

Debra examined the dump files for the application database. She saw that they were newer than they should have been and that they were smaller than they should have been. She notified Saul and Manuel. They agreed that something was seriously amiss and speculated that there was a problem with the Sun hardware. Saul disagreed, but nobody offered an alternate theory.

Countermeasures Assessment of Effectiveness

Site X countermeasures were inadequate for an insider attack. The damage was mitigated by their initial decision to keep the entire network isolated and by their ability to respond to system outages.

The quality of their documentation, installation media, and change control processes enabled them to quickly reconstruct their application. Their outage response was crippled by the lack of current database dumps.

The Site X team did not seriously consider that they had been attacked. They assumed that their problem was strictly technical and that they simply had a serious database outage. They realized they had been sabotaged only after a full day of activity.

Containment Phase

The containment phase occurs after the incident has been investigated. Once the nature and extent of an incident is known, it can be contained so that it does not cause further damage.

Containment Measures

Site X attempted to contain the damage by taking their application offline and posting a maintenance message. Then they isolated the suspect UNIX server by shutting it down for a backup. The focus of their containment efforts was to minimize the loss caused by the outage by restoring the systems as quickly as possible.

Containment Narrative

Manuel asked Saul to lead the recovery effort and then went to discuss the outage with the Organization Y contract contact. Saul directed Alice to put up the "System Undergoing Maintenance" page on the web server and then gathered the team in a conference room. They began planning their recovery options.

They decided to recover the application on the web server machine. This recovery scenario was described as a possible option in the disaster recovery plan, but it had never been attempted.

At 10:30am Debra shut down the Sybase server. Saul brought the UNIX server down to single-user mode and started backing up the server.

At 11:00am, Manuel, Debra, Saul, Alice, and Bob reviewed their situation. Debra, Alice, and Bob used the server inventory book to create a new database server, create a new application database, and modify the application to point to the new data source on the UNIX web server. Cindy updated the NT4 server farm to use the new data source as well.

At 3:00pm Cindy suggested they load a previous backup tape as a test. Since they could not locate the previous week's dump tape, they used an old quarterly dump tape from six months before that they had recently been put in the discard bin. Debra loaded the database.

```
$ tar xf /dev/rmt/0
$ isql -Usa -Sts1252
```

The "`x`" parameter directs the UNIX tar command to extract an archive. The pair of parameters, "`f /dev/rmt/0`", indicate that the archive is located on the tape device.

```
1> load database sitexappl from '/tmp/sitexappl.dmp'
2> go
```

The "`load database`" command instructs Sybase Backup Server to load the "`sitexappl`" database from the database dump file named "`sitexappl.dmp`" located in the "`/tmp`" directory.

Alice and Bob checked the application and determined that everything worked as it should. When Manuel was informed, he inquired about the impact of using 180 day old data. When he understood the limitations, he consulted with the Organization Y contract contact. They agreed to make the Site X application available with the old data and advise the users. The Organization Y people were very concerned about this problem, but agreed that old data was better than no data at all.

As Cindy updated her recovery log, she asked if anybody had heard from Dale. He should have been at work long ago. Bob tried calling his home, but only got an answering machine.

Jump Kit Components

The Site X staff did not have a specially designated 'jump kit'. Instead they relied upon their storage closet full of replacement parts, tools, installation disks, office supplies, and blank media.

Site X had a quarterly dump that was six months old. They had a functional web server. They had a database server they did not trust. They had a room full of processing nodes ready to do work. Their supply closet had a complete set of Sun disks. They had the operating system distribution media for the Sun machines. They had the installation compact disc for Sybase. They had disk images for the processing nodes. They had logs indicating all changes to all systems since they were created. They had a stack of blank compact discs and four unused DLT tapes.

Chain of Custody

Site X did not realize they were the victims of a crime. Their notes and backups were intended to help them recover from a hardware or software failure. They started securing their media after a discussion with the Organization Y security team. However, if they had not kept enough spare parts on hand, they would have simply ignored the security team's recommendation and just attempted to recover.

Detailed Backup of a Victim System

At 10:30am Saul brought the UNIX server down to single-user mode and backed up the server using the following syntax.

```
# ufsdump 0f /dev/rmt/0 /dev/rdsk/c0t0d0s0 /dev/rdsk/c0t0d0s5
/dev/rdsk/c0t0d0s6
```

The "`ufsdump`" utility backs up UNIX filesystems. The "`0`" parameter specifies a full dump of all files. The "`f /dev/rmt/0`" parameter pair indicates the file name of the DLT tape device. The ufsdump command backs up all specified files, filesystems, or devices containing a filesystem. In this case Saul directed it to back up the multiple filesystem by specifying the Solaris device where each filesystem was stored.

Eradication Phase

The UNIX dump completed at 4:30pm. Saul loaded the tape on the UNIX web server. He extracted the database dumps to a subdirectory. Debra then attempted to load the backup file into a test database she had created for that purpose.

```
1> load database sitexappl_test
2> from '/tmp/dbsrv/tmp/sitexappl.dmp'
3> go
```

This Sybase command loads a database named "`sitexappl_test`" from a backup file named "`sitexappl.dmp`" located in the "`/tmp/dbsrv/tmp`" directory.

She found that the application database was empty except for the system tables that every database contains. She looked at the dump header and saw that it was a dump of the "model" database, which is the template used to create all databases in Sybase.

```
1> load database sitexappl_test from
2> '/tmp/dbsrv/tmp/sitexappl.dmp' with headeronly
3> go
```

This load database command instructs the Sybase Backup Server to load the "`sitexappl_test`" database from a dump file named "`/tmp/dbsrv/tmp/sitexappl.dmp`". The "`with headeronly`" clause instructs Backup Server to only report information about the dump file instead of actually loading it into the specified database. Even though the database is not used, it is nonetheless required by the load database command syntax.

At 5:00pm Saul extracted the dump scripts, dump logs, and Sybase account email and reviewed them with Debra. The scripts had not changed in six months. The log files and email indicated successful completion of the dump.

At Debra's suggestion, Saul extracted the Sybase error logs for the database server and the backup server. On examination, they found that the "model" database had been backed up to the filename used by the application database dump. The time was Friday night at 7:00 PM. It was Dale's turn to stay late collecting backup tapes for offsite storage.

Saul and Debra contacted Manuel and informed him that the outage may not have been accidental. As they were discussing the possibility of deliberate data destruction, Alice informed them that the Friday night secure storage container had just been delivered by the offsite storage vendor. Debra checked the database dumps on the tape. They matched the database dump files on disk -- including the very small dump file for the application database.

```
$ tar tvf /dev/rmt/0
```

The "`t`" parameter directs the UNIX tar command to list a table of contents for an archive. The "`v`" parameter directs the tar command to use verbose format, which includes file size. The "`f`" parameter and the associated "`/dev/rmt/0`" argument indicate that the archive is located on the tape device.

At 6:00pm Manuel asked Cindy what systems Dale would be able to access. Cindy produced the password sheet that listed the Administrator passwords for all of the NT4 servers in the server network. Manuel suggested that they examine each NT processing node to make sure there were no surprises left behind. Manuel consulted with the Organization Y contact. Organization Y notified law enforcement and internal security. Manuel called the building engineer to have the combination changed on the computer room lock.

Cindy noticed that processing node 1 had very little free disk space. When she checked the Recycle Bin, she saw a large file and a program called ethereal. Saul asked her to record the data on to a compact disc. Then she shut down the machine and disconnected it from the network.

Once it was recorded, Saul and Bob examined the file. It was a log of network packets. Bob used a copy of ethereal on his workstation to analyze the log contents. The log was from a previous incident where Debra had been called by Dale to investigate a problem that an NT4 server was having with the database. Debra remembered how Dale had insisted that she prove she could connect to Sybase from the ailing machine. She had logged in as "`sa`" and demonstrated access to the application database. Shortly after that, Dale reported that the problem was corrected. Using Debra's recollection as a guide, Bob filtered the

ethereal log and found the Tabular Data Stream login packet. Debra then verified that the password shown in the log was the "sa" password.

At 7:00pm Saul gathered the team in a conference room. They reviewed Cindy's notes and constructed a timeline of events. Their reconstruction had Dale stealing the dump tape and overwriting the Sybase application backup on Friday night. He had ample time to do additional damage. Manuel decided that they would need to rebuild all of the Site X NT4 servers. Everybody agreed to change all passwords.

Recovery Phase

Saul and Alice ordered the secure storage container from the previous quarter. When the container arrived, the quarterly database dump tape was given to Debra. Alice prepared to redeploy the code releases since the quarterly dump.

On advice from Organization Y security, Saul removed all of the hard drives from the database server machine. The disks were bagged, tagged, and logged for future investigation. After installing replacement disks, Saul and Debra reconstructed the system from installation media. Cindy removed the suspicious NT4 machine from the network. The system was tagged and logged for future investigation. She then constructed a replacement machine using installation media.

At 7:30pm Debra realized that Dale could have compromised the master database that controls the Sybase database server. She casually mentioned to Saul that it was a good thing they were going to be forced to create the server from a fresh install. They speculated that Dale might have added or removed login ids from the database.

At 8:00pm Cindy, with help from Alice and Bob, shut down all of the NT4 processing nodes. They then installed fresh copies of NT4sp6 and the application engine. Debra loaded the Site X application database and supervised deployment of the intervening code releases, which included several database changes. Bob and Cindy restored the original data source configuration for the web application. Manuel executed all of the critical reports from the application and then notified the contract contact that they had been able to recover to the previous quarter's backup. The Organization Y contract contact did not consider this a satisfactory resolution and reminded Manuel of the penalties associated with underperformance.

The next day, the Site X team began reprocessing the data from the previous months. They estimated that within three weeks, the application would be fully functional with current data. Site X faced a penalty from Organization Y in excess of $3,000,000. The Site X owner was in a near panic and kept calling Manuel

with ideas for recovering the application faster. They concluded that the most cost effective approach would be to have the salaried staff members work 24 hours a day. At that pace, the penalty would only cost them $1,000,000.

On Wednesday morning, Manuel got a call from Dale's cell phone inquiring about a possible settlement. He offered them three options. They could pay him $250,000 for the old tape. They could pay him $450,000 for the new tape. They could do nothing and he would sell the tapes to another entity. The owner directed Manuel to give Dale whatever was necessary to get the most recent tape. The final agreement included a promise not to prosecute, a letter of recommendation, and $550,000 in exchange for the two tapes.

Lessons Learned Phase

Manuel assigned the task of writing a follow-up report to Cindy. Her findings are summarized below.

Site X relied upon static security policies instead of an active security posture. They had security policies and procedures that were developed without the input of security professionals. Security awareness training was strictly limited to the policies initiated at the time of the contract with Organization Y. The staff was not trained in security principles. Site X carefully tracked the number of times an isolated computer was turned on because it was connected to the internet. They ignored the sheets of paper with all of the administrator passwords for the processing nodes. They did not consider the absurdity of maintaining a different password for each processing node, recording all of them on the same page, and then distributing copies of the list.

Site X allowed budget concerns to undermine security practices. By minimizing the number of backup tapes in the rotation, they magnified the potential for catastrophic data loss. Every week, both of the weekly dump tapes would be in the same room with the database server. As Dale had demonstrated, a single incident could destroy all of them.

The key Site X asset is the database. It was not given adequate protection due to budget concerns and the expense of having a database administrator improve the backup and recovery plan. Site X used unencrypted logins to the database server because it was the default configuration. This weakness resulted in the loss of the application database. Even a small amount of effort would have prevented the password from being readable.

Site X used manual monitoring to ensure the integrity and stability of their systems. There are many tools for infrastructure and application monitoring, many of them free, that could have detected the problems much sooner. Among the detectable events were the disk space shortage on the processing node used

for the sniffer, the batch job failure on the holiday, and the missing application database. A simple report would have identified the account Dale added.

Manuel held a follow up meeting to review Cindy's report. The Site X staff members were upset by the findings and demanded that Manuel allow them to make improvements. They decided to start an action plan before Manuel went to Organization Y with their findings.

In the meeting they agreed to start learning more about security. Several security certifications were discussed and Manuel agreed to pursue a policy change to allow the staff to be reimbursed for their testing expenses. Manuel also agreed to immediately buy more tapes and increase the number of off site secure storage containers.

Debra argued for improvements to the database scripts to improve security. She recommended that they switch to encrypted logins and consider using encrypted connections via SSL. She also recommended that they consider encrypting their dumps to protect the information if the tapes ever get lost or stolen. Manuel agreed to the encrypted login feature because it was easily understood. He asked Debra to prepare a report on the other options so he could understand them better before making a decision.

Saul and Bob argued for automated monitoring. Manuel flatly refused. Instead, he asked Bob and Alice to develop a minimal set of tools to help them identify and report error conditions. He said they would worry about automation later.

When Manuel discussed Cindy's report with Organization Y, he put a great deal of emphasis on the improvements the action plan the team had worked out. The Organization Y contract contact agreed that this was a good starting point and that the future budgets would include annual audits by the Organization Y security team.

Exploit References

The following references describe aspects of the exploit illustrated in this paper.

http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0999
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0327
http://sybooks.sybase.com
http://www.ethereal.com/docs/user-guide/
http://www.freetds.org/tds.html
http://www.secretservice.gov/ntac/its_report_040820.pdf
http://www.sybase.com/content/1030018/ASE_1252_Security_wp.pdf

Additional Information

Many customers resist upgrading to the new authentication techniques available in Sybase ASE 12.5.2. One typical reason to maintain the legacy configuration is the portfolio of applications and earlier Sybase servers that need to be supported. In such an environment it is important to mitigate the exposure.

There are multiple ways to prevent clear text passwords from being sent across the network. The simplest technique is to encrypt the password on the client side before sending it. The "`isql`" utility has a "`-X`" parameter that causes "`isql`" to encrypt the password before sending it to the Sybase Server. Other commonly used Sybase utilities such as "`bcp`" and "`defncopy`" have the "`-X`" parameter as well. The "`bcp`" utility is used to bulk copy data into the database server or out from the database server. The "`defncopy`" utility is used to extract stored procedure source code from the Sybase server. The "`optdiag`" utility, used for reporting and editing Sybase optimizer statistics, does not provide the "`-X`" parameter and thus requires more effort to protect it from eavesdropping. To prevent clear text password sniffing in that case it is necessary to use an encrypted connection. Sybase provides this capability with Kerberos or with SSL. They are described in a White Paper available at the Sybase web site.[22] The white paper also describes advanced authentication mechanisms available in Sybase ASE 12.5.2.

Another common error, not shown in the body of the paper, is to put the password on the command line. This exposes the password to any user on the UNIX server that can execute a "`ps`" command.

There are a variety of methods for hiding the password for Sybase commands. One simple technique that can be used in a script is shown below:

[22] http://www.sybase.com/content/1030018/ASE_1252_Security_wp.pdf

```
isql -U sa -S ts1252 -X <<- EOF
sybase12
      dump database sitexappl to '/tmp/sitexappl.dmp'
      go
EOF
```

The example above uses a UNIX shell "Here Document", the "`<<- EOF`" construct, to illustrate putting the password in the input stream of the isql utility. More information can be found in the Sybase Frequently Asked Questions.[23]

A similar technique can be used with the bulk copy utility:

```
echo sybase12 | bcp sitexappl..dbt3 out dbt3.txt -c -U sa -S
ts1252 -X
```

In this case, the echo sends the password to the input of the bulk copy utility, which will read the password as if the user had typed it. The rest of the "`bcp`" parameters indicate the database and table, the direction of data flow, a filename, the character mode flag "`-c`", the username, the server name, and the client-side encryption flag.

The weakness of the two techniques above is that they still include the password with the command, which is a serious mistake when used with scripts. Even a simple read-only password file is an improvement over embedded passwords in scripts. The next step is to provide a tool to get the password from a protected store and send it to the Sybase utility. Such a tool can be very complex, but to illustrate the point, the UNIX "`cat`" command will suffice.

```
isql -U sa -S ts1252 -X <<- EOF
`cat /etc/sybasepassword.txt`
      dump database sitexappl to '/tmp/sitexappl.dmp'
      go
EOF
```

As before, this example uses a UNIX "Here Document". The difference this time is that the password is replaced by a command substitution, the "`` `cat /etc/sybasepassword.txt` ``" construct which replaces the command between the backticks with the output of the command. The UNIX "`cat`" command provides the contents of a file. The "`/etc/sybasepassword.txt`" file contains the Sybase password and is only readable by the user executing the process. Typically the process owner is a restricted account such as the "`sybase`" user or an application account such as "`sitexappl`".

[23] http://www.faqs.org/faqs/databases/sybase-faq/part11

```
cat /etc/sybasepassword.txt | bcp sitexappl..dbt3 out dbt3.txt -c
-U sa -S ts1252 -X
```

The example above illustrates the same technique with the Sybase bulk copy utility.

References

Bruns, Brian; Lowden, James; Wheeler, Brian; Schaal, Mark; Ziglio, Frediano. "TDS Protocol Documentation." FreeTDS User Guide: A Guide to Installing, Configuring, and Running FreeTDS. Version 1.79. December 2003. URL: http://www.freetds.org/tds.html (16 May 2004).

Cisco Systems, Inc. "Internet Protocols." Internetworking Technology Handbook. February 2002. URL: http://www.cisco.com/univercd/cc/td/doc/cisntwk/ito_doc/ip.htm (30 May 2004).

CVE Editorial Board. "CAN-2003-0327 (under review)." Common Vulnerabilities and Exposures. CVE Version 20030901. May 2003. URL: http://cve.mitre.org/cgi-bin/cvename.cgi?name= CAN-2003-0327 (26 September 2004).

CVE Editorial Board. "CVE-1999-099." Common Vulnerabilities and Exposures. CVE Version 20030901. January 2000. URL: http://cve.mitre.org/cgi-bin/cvename.cgi?name= CVE-1999-0999 (26 September 2004).

Garbus, Jeffrey; Chang, Alvin; Garbus, Penny; Tyrrell, Gary. The Official Administrator's Guide to Sybase ASE 12.5. Plano: Wordware Publishing, Inc., 2002.

Owen, David; Sanchez, Pablo; et al. "Sybase FAQ: 11/19 - Issues, dbccs, isql, bcp." Sybase Frequently Asked Questions. March 2003. URL: http://www.faqs.org/faqs/databases/sybase-faq/part11 (26 September 2004).

Randazzo, Marisa Reddy; Keeney, Michelle; Kowalski, Eileen; Cappelli, Dawn; Moore, Andrew. "Illicit Cyber Activity in the Banking and Finance Sector." Insider Threat Study: Illicit Cyber Activity in the Banking and Finance Sector. August 2004. URL: http://www.secretservice.gov/ntac/its_report_040820.pdf (26 September 2004).

Rapid7, Inc. "Sybase ASE 12.5 Remote Password Array Denial of Service." Rapid7 Advisory. Advisory Number R7-0016. November 2003. URL: http://www.rapid7.com/advisories/R7-0016.html (26 September 2004).

Sharpe, Richard; Warnicke, Ed; Lamping, Ulf. "Ethereal User's Guide." Ethereal User's Guide: V2.00 for Ethereal 0.10.5. 2004. URL: http://www.ethereal.com/docs/user-guide (3 October 2004).

Sybase, Inc. "About Sybase." November 2003. URL: http://www.sybase.com/about_sybase (27 September 2004).

Sybase, Inc. "Adaptive Server Enterprise Addresses Application Security Challenges." 2004. URL: http://www.sybase.com/content/1030018/ASE_Security_wp.pdf (21 August 2004).

Sybase, Inc. "Installation Guide Software Developer's Kit and Open Server 12.5.1 for Microsoft Windows." Software Developer's Kit and Open Server(TM) 12.5.1 MICROSOFT WINDOWS. Document ID: DC36841-01-1251-01. November 2003. URL: http://sybooks.sybase.com/onlinebooks/group-cn/cnp1251e/ocsinst (23 May 2004).

Sybase, Inc. "Installation Guide Software Developer's Kit and Open Server 12.5.1 for UNIX." Software Developer's Kit and Open Server(TM) 12.5.1 UNIX. Document ID: DC34789-01-1251-01. November 2003. URL: http://sybooks.sybase.com/onlinebooks/group-cn/cnp1251e/ocsinunx (23 May 2004).

Sybase, Inc. "Open Client/Server 10.0.3 Supplement for Windows 95." Open Client/Server Release 10.0.3. Document ID: 32671-01-1003-01. September 1995. URL: http://sybooks.sybase.com/onlinebooks/group-cnarc/cnp1003e/w95_supp (23 May 2004).

Sybase, Inc. "Utility Guide." Adaptive Server Enterprise 12.5. Document ID: 30191-01-1250-01. May 2001. URL: http://sybooks.sybase.com/onlinebooks/group-as/asp1250e/util (16 May 2004).

Sybase, Inc. "Utility Programs for UNIX Platforms." Sybase Adaptive Server Enterprise Version 12.0 UNIX Platforms. Document ID: 36124-01-1200-01. October 1999. URL: http://sybooks.sybase.com/onlinebooks/group-as/asp1200e/uxutil (16 May 2004).

Sybase, Inc. "What's New in Adaptive Server Enterprise?" Adaptive Server Enterprise 12.5.1 (Core Documentation Set). Document ID: DC37429-01-1251. November 2003. URL: http://sybooks.sybase.com/onlinebooks/group-as/asg1250e/whatsnew (16 May 2004).

Wikimedia Foundation. "Client-server." Wikipedia, The Free Encyclopedia. October 2004. URL: http://en.wikipedia.org/wiki/Client/server (3 October 2004).

Viega, John; Messier, Matt. "Security is Harder than You Think." Queue. Volume 2, Number 5 (2004): 60-65.

Last Updated: July 25th, 2017

Upcoming Training

Security Awareness Summit & Training 2017	Nashville, TN	Jul 31, 2017 - Aug 09, 2017	Live Event
SANS San Antonio 2017	San Antonio, TX	Aug 06, 2017 - Aug 11, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Salt Lake City 2017	Salt Lake City, UT	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Chicago 2017	Chicago, IL	Aug 21, 2017 - Aug 26, 2017	Live Event
Community SANS Memphis SEC504	Memphis, TN	Aug 21, 2017 - Aug 26, 2017	Community SANS
Mentor Session AW - SEC504	Milwaukee, WI	Aug 23, 2017 - Sep 29, 2017	Mentor
Mentor Session AW - SEC504	New York, NY	Aug 24, 2017 - Sep 08, 2017	Mentor
Mentor Session - SEC504	Denver, CO	Aug 29, 2017 - Oct 10, 2017	Mentor
SANS Tampa - Clearwater 2017	Clearwater, FL	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS San Francisco Fall 2017	San Francisco, CA	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS vLive - SEC504: Hacker Tools, Techniques, Exploits and Incident Handling	SEC504 - 201709,	Sep 05, 2017 - Oct 12, 2017	vLive
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
Mentor AW - SEC504	Santa Clara, CA	Sep 11, 2017 - Sep 22, 2017	Mentor
SANS Dublin 2017	Dublin, Ireland	Sep 11, 2017 - Sep 16, 2017	Live Event
Mentor Session - SEC504	Arlington, VA	Sep 20, 2017 - Nov 01, 2017	Mentor
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
Rocky Mountain Fall 2017	Denver, CO	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS SEC504 at Cyber Security Week 2017	The Hague, Netherlands	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Columbia SEC504	Columbia, MD	Sep 25, 2017 - Sep 30, 2017	Community SANS
Mentor Session - SEC504	Boston, MA	Sep 26, 2017 - Nov 07, 2017	Mentor
Mentor Session AW - SEC504	Houston, TX	Oct 02, 2017 - Dec 11, 2017	Mentor
SANS DFIR Prague 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
Mentor Session - SEC504	Columbia, SC	Oct 03, 2017 - Nov 14, 2017	Mentor
SANS Phoenix-Mesa 2017	Mesa, AZ	Oct 09, 2017 - Oct 14, 2017	Live Event
Community SANS Chicago SEC504	Chicago, IL	Oct 09, 2017 - Oct 14, 2017	Community SANS
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event