# Introduction to Database and Application Worms

## White Paper

# INTRODUCTION

In the recent past, a new set of threats have emerged – worms that propagate through vulnerabilities in databases rather than through more traditional operating system or web server holes. Despite their lack of sophistication, these worms have been somewhat successful because of the poor state of database security. Security in databases has generally been ignored and the threat management of these applications has been non-existent.

Most recently, a new worm emerged called the Spida worm. This is considered the first successful worm to propagate through databases. Fortunately, the Spida worm did not have a destructive payload.  More information on the Spida Worm can be found at:

> http://www.appsecinc.com/resources/alerts/mssql/02-0002.html
> http://www.cert.org/incident_notes/IN-2002-04.html

Last year (2001) a different Microsoft SQL Server worm known as Voyager Alpha Force was found in "the wild". However, Voyager Alpha Force never gained critical mass and fortunately died out without causing any harm.  More information on Voyager Alpha Force can be found at:

> http://www.cert.org/incident_notes/IN-2001-13.html

In terms of its method of breaking into databases, the Spida worm was extremely primitive. The Spida worm attacked Microsoft SQL Server databases that had a blank password for the "sa" login which  is the ultimate administrator account within a Microsoft SQL Server database. Unfortunately, if the administrator of the database does not go through the trouble of setting a password for the sa login on the database, there is little hope that the administrator will ever attempt to seriously secure the database. As you see, the Spida worm only penetrated databases with essentially no security and yet managed to infect a large number of databases.

# WHAT MAKES A WORM SUCCESSFUL?

The damage caused by a worm is dependent on several factors:

1) The number of targets for the worm
2) The success rate of infection
3) The resilience of the worm

A critical factor in the effect of a worm is the quantity of potential targets. Most people assume that databases are always behind firewalls. Unfortunately this is not always the case. Databases are a critical piece of an organizations infrastructure and cannot always be hidden behind a firewall.

The success rate of infection is critical to whether or not the worm is able to spread through to other systems. The Spida worm was effective because a large number of Microsoft SQL Server databases have blank "sa" passwords. Those databases with non-blank passwords were not infected.

The last factor is the resilience of the worm. Is it hard to detect? Does it leave a back door? Are there any bugs that cause the infection to fail on certain databases? A well-developed worm is much harder to fight than a "noisy" and "sloppy" worm that is easy to remove from the system.

# THE FUTURE OF DATABASE WORMS

More sophisticated variants of database worms are expected to become more prevalent over the next 18+ months given the success of the Spida Worm. These worms will clearly be aimed at meeting the three factors listed in the previous section (*What Makes a Worm Successful*).

It is predicted that future worms will not simply propagate through Microsoft SQL Server databases with blank "sa" passwords, but will also attempt dictionary attacks on the "sa" login. Future worms will detect a remote database, and will attempt to connect as the "sa" login using a dictionary of thousands of common passwords. Using this strategy, the infection rate of the worm will be slower, but the success rate of infection will significantly increase leading to a dramatic overall increase in the damage caused by the worm.

In order to gain critical mass, it seems likely that worms will develop as multi-headed worms. These multi-headed worms will propagate not through a single platform but rather through multiple database platforms, such as through a combination of Oracle, Microsoft SQL Server, and MySQL. Infecting any single database platform may not be enough to gain critical mass, but a worm spreading through multiple database platforms would indubitably gain critical mass.

A real concern is that database worms will expand outside the realm of Microsoft SQL Server to the next likeliest avenue of attack - Oracle. With Microsoft SQL Server, providing a minimum level of security is very simple – set a strong password on the sa login. Oracle is a much more complex system. With Oracle, there are 15 default accounts and passwords installed straight out of the box. There is also a "listener" password that is not set by default. An Oracle worm would likely have a much higher rate of success for infecting each system it found.

Another likely advancement we will see in worms is more intelligence in discovering targets. The worms we have seen so far have attacked applications only on default ports. Microsoft SQL Server does not always exist on default ports. It can be configured to listen on a different port, and for a named instance of Microsoft SQL Server, a port redirector is installed on port 1434 and the actual port the database listens on is randomly selected. By querying the redirector, you are able to determine the port Microsoft SQL Server is listening on. The Spida worm was not intelligent enough to detect if a SQL Server was on a different port. By building a worm that intelligently finds Microsoft SQL Server on a non-standard port, the potential targets for a worm increases.

Oracle is very similar. Many people expose Oracle to the Internet, but then change the port number to something other than the default in an attempt to hide the application. By default, Oracle runs on port 1521. Very frequently, many people run Oracle on port 1522 or 1527 to make it less of a target. When someone does this, they develop a false sense of security, and will often compensate by neglecting the real security of Oracle. An intelligent worm may take advantage of this, and propagate through databases on non-standard ports.

# APPLICATION WORMS

Beyond the next 18 months, the next generation of worms will be based on application level attacks. What this means is that worms will look for holes in application code, use these holes to gain control of the system, and propagate by looking for similar holes in other applications. For example, a worm may crawl through the pages of a web site looking for SQL Injection vulnerabilities. Once a vulnerability within the website is found, the worm would inject SQL commands that would infect the database and propagate to other web applications.

There is an additional risk of these application level attacks. They will not only affect the Internet, but could also have a serious impact on networks behind the firewall. If a worm were able to tunnel through a web server and infect the database behind the firewall, it would be able to infect the network behind the firewall. This would result in a drastically increased amount of damage caused by the worm.

# SOLUTION TO ADDRESS THE THREAT

One solution offered by many in the security community is to stop all traffic at the perimeter of the organization, specifically at the firewall. From the security professional's perspective, this may be reasonable. Unfortunately, from a business perspective this is not a reasonable request. These databases are exposed to the Internet for a reason. Business partners and customers need access to these databases, and architecting the current software is not possible in the short-term and often not cost effective in the long-term.

A better solution is to actively lock down these databases and applications. Databases and other critical infrastructure applications have a security system all of its own that is as robust as their operating/network operating system counterparts. With that said, this brings to light a whole other level of security issues and vulnerabilities to be aware of and address:

- ❑ Denial of Services (DoS)
- ❑ Identification/Password Control Issues
- ❑ Misconfigurations
- ❑ System Integrity Issues
- ❑ Access Control Issues

It is therefore important to become more knowledgeable about database security, or to find partners that provide solutions for locking down these applications. A good starting place is with the database vendors themselves. Each of them provide their own security bulletins, and even lock down procedures to safeguard databases from hackers:

## MICROSOFT
http://www.microsoft.com/security/

## ORACLE
http://otn.oracle.com/deploy/security/content.html

## MYSQL
http://www.mysql.com/doc/P/r/Privilege_system.html

## ABOUT APPLICATION SECURITY, INC. (APPSECINC)

AppSecInc is the pioneer in designing, developing, and managing application security solutions for the enterprise.  AppSecInc products proactively secure enterprise applications by discovering, assessing, and protecting the database against rapidly changing security threats.  We give organizations the confidence to extend business with customers, partners and suppliers across networks and the Internet.  Our security experts, combined with our strong support team, deliver the most up-to-date application safeguards to minimize risk and eliminate impact on business.  Please contact us at 1-866-927-7732 to learn more, or visit us on the web at www.appsecinc.com.