Networking Fundamentals

Part of the SolarWinds IT Management Educational Series

VOLUME 1

The Basics of Application Monitoring

This paper examines application management and the interaction of applications and networks. As part of the Network Fundamentals series, it is intended for readers who have a basic understanding of networks, network management and applications.

solarwinds

Table of Contents

Document Revised: Oct 1, 2010

The Way We Were

In the 1980s, networks were so rare that the management of applications was naturally left to the application developer/support or, in most cases, to the end user. Tools for application analysis were few and hard to find. The "application expert" could fool with configuration files if they knew what they were doing, search BBS's for clues, upgrade DOS and then resort to a series of prayers and reboots. In the 1990s, networking exploded and applications that used to save data locally could now save it to file servers, and multiple users could access the files from different locations - the floppy disk "sneaker-net" was on the way out. With the introduction of new technologies such as middleware and application servers, networks were expanding their use from file sharing to application sharing. User stations still processed data locally for the most part and shared the results with other users using network storage. Then something changed with network applications.

Developers began to see that the applications could be used as efficient tools to carry out complete business transactions. This meant that an application could mingle technologies like file sharing, database queries, local processing and remote procedures to accomplish what might otherwise be time consuming and labor intensive. In a very short time, application abilities and complexities expanded greatly, but the management of applications was slower to change. Application developers and application support specialists were still responsible for application performance but they usually knew very little about the network that enabled them. Network engineers were expanding network reach quickly, but knew little about the applications the network carried.

Enter the age of finger pointing!

"My code is fine, the network is slow — that's the problem."

"The network has plenty of bandwidth, something must be wrong with your application."

Stop the Madness

As you can see, the above dialogue could be classified as less than productive. Both the network engineer and the application developer had some tools to diagnose what was happening in their area of expertise, but they each had little to no visibility into how the network and applications were working together. The two departments were experiencing the effects of technology silos. Something had to change in how we defined and measured applications.

In the mid to late 1990's that change happened, and management vendors on both sides of the fence began to offer solutions that included application and network monitoring features. These included:

- Hardware probes that sniff traffic and analyze what is on-the-wire
- Synthetic transaction software that inserts and measures artificial transactions to model actual user transactions
- Distributed software agents to measure user transactions
- Flow analyzers to discover what applications were consuming bandwidth, application vendor modules focused on managing their own application
- Network management system (NMS)-based application monitoring, which combined application and management packages

Many of these have been offered as "stop the finger pointing" solutions. Some argue they could be better described as qualifying the finger pointing. Today there are several network management offerings which focus simply on the network, but you'd be hard pressed to find an application management solution that does not also examine portions of the network. For the purpose of this paper, we'll limit the discussion to NMS-based application monitoring. We'll also assume that the network is healthy and is well monitored for bandwidth availability, response time and uptime throughout.

Enough History! Let's Get Started Defining and Monitoring an Application

From the first two sections, we can see that it's not enough to consider an application a service or process running on a machine. Applications do include those resources but also consist of the network components and shared services on which they depend. While there is no set definition of what exactly comprises an application, I believe most applications can be defined by and monitored by the following:

- Specific server processes/daemons and services
- Protocol port(s)
- User experience

Yes — there are a whole bunch of other technologies we could examine to look at applications, but what we're after here is the basics of application management, so that is where we will focus. Let's look at how each of these components can be monitored.

Server Processes/Service and Daemon Monitoring

Windows servers — By far the most common method of monitoring processes and services is Windows Management Instrumentation (WMI). WMI is Microsoft's implementation of a couple of standards: CIM and WBEM. WMI offers monitoring of counters on a Windows system to track machine-level items like memory, processors, etc., and OS information like server work queues, processes, services and connection status. Functionally, WMI is a lot like SNMP in its use of counters, polling and information bases. The best way to get to know WMI is to give it a spin.

Download and try the free SolarWinds WMI Monitor tool. Make sure to spend some time with the WMI browser included in the free tool. It's a good way to see firsthand the types of objects you can manage with WMI and how WMI operates.

Linux or Unix servers — These servers have been around for decades and have mature implementations of SNMP. Like WMI, SNMP on these servers monitors machine-level and OS components. Unlike WMI, SNMP is a completely open standard. You can use a MIB browser and walk through a server's MIB much the way you do in a WMI browser.

As an example, if we want to see that an FTP server is able to operate correctly we should monitor the FTP process on that server. This can be accomplished by setting up a WMI monitor and monitoring the FTP process WMI counter on the FTP server. We could also do this using an SNMP monitor for the FTP process.

Protocol Port Monitoring

Protocols can be thought of as two-way radios: they allow bidirectional information flow between two or more points. The protocol ports in this analogy are the radio frequencies. They offer multiple channels of communication within dedicated spectra. So TCP port 20 is like TCP channel 20. If you were able to tune in and listen to port 20 you'd be hear computers delivering files, as TCP port 20 is reserved for File Transfer Protocol (FTP). If you tuned to port 21 you'd hear computers asking for file transfers that will be sent on port 20. These two ports work together to supply FTP control (port 21) and an FTP file carrier (port 20). So if you just want to see if files are able to be transferred, you only need to setup a monitor for port 20 on the FTP server. On the other hand, if you want to make sure the FTP network service is able to function, you would monitor both ports. There's actually a trick we can do to monitor port 21 directly and 20 indirectly. We'll see that below.

"10.1.1.1 calling 10.1.1.2, please ship me file1.txt on port 20"

"Hi 10.1.1.1, this is 10.1.1.2, I'll meet you on port 20 with file1.txt"

This example is a bit more elaborate than actual protocol port monitoring. In protocol port monitoring what we do is look to make sure the server is able to use the specified port rather than listening to the conversations on that port. Examining the conversations is accomplished by flow analyzers and sniffers. That level of monitoring is overkill for the type of application monitoring we are discussing here.

Now we can monitor ports and determine FTP is working port-wise, but that is only two components of the FTP network service.

There is also an FTP service running on the server OS that receives the requests and serves the files. As we learned in the previous section we could monitor the FTP process using WMI or SNMP. Now that we have combined the FTP service monitoring with FTP port monitoring we are close to claiming we are monitoring the FTP application. But what if the service and ports are operational and someone has deleted the target file to be served? While the components of FTP are operational, the FTP network service cannot complete user requests for the absent file. We should test the service by requesting the file.

User Experience Monitoring

This component of application monitoring involves making an automated request of an application to model a real user request. These are called synthetic transactions because they mimic a user transaction but don't originate from a user. Typically they are initiated from the NMS. Let's add a user experience monitor for FTP to the process and protocol monitors above.

The FTP user monitor in this case will make an FTP request to the FTP server from the NMS. The request can be for a file normally requested as part of a business transaction or a test file. When the NMS makes the request for the file it can record the following:

- Time to complete transfer of the file
- File integrity via checksum

Here's the trick I mentioned: If we monitor TCP port 21, we examine FTP requests. Now by placing a user experience monitor on FTP, we are also testing TCP port 20 indirectly and monitoring the actual ability to serve a file. After all, if TCP port 20 is not functioning the user experience transfer will not happen.

Let's review how we've created a holistic application monitor using each of the component monitors above:

What application do we want to monitor?

- FTP of file1.txt from FTP server001.

What components define this application instance?

- Service monitor — The FTP service on server001
- Protocol port monitor — TCP ports 20 and 21 on server001
- User experience monitor — File1.txt delivery

How will we measure each one of these?

Create an application monitor for FTP which includes:

- Status of the FTP service on server001 (WMI or SNMP service/process monitor)
- Status of TCP ports 20 and 21 on server001 (protocol monitor)
- Delivery of file1.txt (user experience monitor)

Often these component monitors are grouped to make an application template. This can help to save a great deal of time when there are multiple instances of an application to monitor. Instead of having to create a new application monitor for FTP on server002, wouldn't it be nice to apply the template created for server00 instead?

Simple and Complex Applications

For our purposes, I'll define these as:

- Simple application = an application that operates as a process, independently from other applications or processes
- Complex application = an application that requires multiple processes and services and interacts with other applications to complete its functions

This does not imply anything about size or ease of use of the applications, it just refers to application and process/service interaction.

In our example we created an application monitor for a simple application, FTP, which happens to be a network service. Let's take SolarWinds Orion Network Performance Monitor with the Orion NetFlow Traffic Analyzer module as an example of a complex application.

Here are some of the components that we could monitor to make a complete application monitor:

Orion NPM Server Services Monitors — WMI

- SNMP service, SNMP trap service, SolarWinds alerting engine, SolarWinds job engine, SolarWinds NetFlow service, SolarWinds information service, IIS, WWW publishing service.
- Orion NPM server port monitors.
- TCP port 80 — HTTP for web console.

User Experience Monitors

- HTTP monitor of Orion home page. This monitor will log in to the Orion home page automatically and test both authentication and delivery of the http home page.
- SQL monitor. This monitor will automatically query the SQL server and return the results.

That makes 10 individual monitors to monitor the application, and you could probably add a few more. These monitors cover all three of the component areas we defined in Section 3 — service monitors, port monitors and user experience monitors. Here is where application templates are crucial. With 10 or more components being used to monitor an application, there is plenty of opportunity for mistakes if you had to create the application monitor from scratch each time. If you apply a verified, saved application monitor template, the risk of applying monitors with mistakes is eliminated. Another option would be to create an application monitor template for the SQL server and one for the DNS server that NetFlow uses to resolve endpoint IP addresses to domains.

Summary

We have seen the following points in the above sections:

- An application is not just a service or process running on a server.
- Applications depend on multiple resources to operate correctly, including:
  - Services/processes — both the actual application services and associated services which support the application
  - Server resources (RAM, CPU, hard drives, etc.)
  - Protocol ports
  - Secondary applications such as SQL in section 4
- Applications can be effectively monitored by:
  - examining the application components and applying the proper monitors, including:
  - WMI and SNMP component monitors for processes/services
  - Port component monitors for protocol ports
  - User experience monitors to demonstrate the application functionality.

I hope this has been a helpful overview. If you have any questions or comments, participate in our online community Thwack, made of over 58,000 users of SolarWinds products.

Related SolarWinds Products

Orion Network Performance Monitor

Orion Network Performance Monitor Highlights:

- Monitors and analyzes real-time, in-depth, network performance statistics for routers, switches, wireless access points, servers, and any other SNMP-enabled devices
- Simplifies network issue investigation with drill down maps and Top 10 views of your global network
- Gets you up and running in less than an hour with do-it-yourself deployment
- Scales to accommodate growth and management needs
- Enables advanced alerting for correlated events, sustained conditions, and complex combinations of device states
- Monitors the energy consumption of Cisco® EnergyWise-enabled network devices and displays policies that regulate energy consumption

Get more information on Orion Network Performance Monitor here.

Evaluate Orion NPM in a live demo environment here.

Download a fully functioning, 30-day evaluation of Orion NPM here.

Orion Application Performance Monitor

Orion Application Performance Monitor Highlights:

- Monitor virtually any application with the Community Content Sync
- Monitor your applications in minutes with the Application Discovery Engine
- Manage global template settings across your organization with Dynamic Templates
- Simulate and measure Quality of Experience (QoE) with User Experience Monitors
- Migrate from open-source solutions such as Nagios® with the Open Source Script Processor
- Remotely monitor any WMI performance counters to proactively troubleshoot application issues before impacting users.
- Manage advanced alerts for correlated events, sustained conditions, and complex combinations of device states
- Leverage network service monitors to gauge DNS, IMAP4, POP3, and MAPI performance
- Investigate network fault, performance and applications issues from a single intuitive web console

Get more information on Orion Application Performance Monitor here.

Evaluate Orion APM in a live demo environment here.

Download a fully functioning, 30-day evaluation of Orion APM here.

SolarWinds WMI Monitor Free Tool

SolarWinds WMI Monitor Free Tool Highlights:

- Monitor real-time performance metrics on any Windows server or application
- Leverage a large selection of pre-built and community generated application templates
- Modify or design your own application templates with the built-in WMI browser
- Share your application templates with the world – one click posts your template to thwack, the SolarWinds community site
- Use the application templates from WMI Monitor with Orion Application Performance Monitor if you need to monitor multiple applications or require more powerful monitoring

Get more information on the SolarWinds WMI Tool here.

Download the SolarWinds WMI Monitor Free Tool here.

About SolarWinds

SolarWinds is rewriting the rules for how companies manage their networks. Guided by a global community of network engineers, SolarWinds develops simple and powerful software for managing networks, small or large. Our company culture is defined by passion for innovation and a philosophy that network management can be simplified for every environment.

SolarWinds products are used by more than one million network engineers to manage IT environments ranging from ten to tens of thousands of network devices. Comprised of fault and performance management products, configuration and compliance products, and tools for engineers, the SolarWinds product family is trusted by organizations around the globe to design, build, maintain, and troubleshoot complex network environments.

SolarWinds is headquartered in Austin, Texas, with sales and product development offices around the world. Join our online community of experts at thwack.com!

About the Author

Andy McBride is a Technical Specialist for SolarWinds focusing on making knowledge of networking and network management accessible to customers and prospects of all levels. The "Networking Fundamentals" series is specifically written for an audience with limited prior exposure to these technologies. Andy's technical background includes seven years at International Network Services (INS) as a Network Engineer and Managing Consultant, three years as a Novell Certified Instructor and five years as a Network Performance Products Manager with BT-Infonet. Prior to entering technology, Andy worked in aerospace on projects such as the SR-71, F-117, F-22, L-1011, F-18 and the space shuttle main engine. Andy has a degree in Chemistry but was wise enough to never work in that field. Andy invites you to follow him on Twitter, @McBrideA, and can be contacted on Thwack, McBrideA.