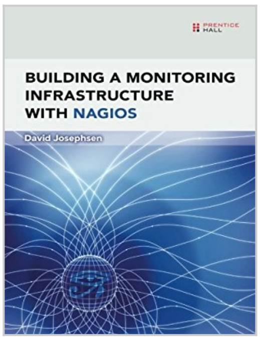

Building a Monitoring Infrastructure with Nagios

By David Josephsen

Building a Monitoring Infrastructure with Nagios By David Josephsen

Build real-world, end-to-end network monitoring solutions with Nagios

This is the definitive guide to building low-cost, enterprise-strength monitoring infrastructures with Nagios, the world’s leading open source monitoring tool. Network monitoring specialist David Josephsen goes far beyond the basics, demonstrating how to use third-party tools and plug-ins to solve the specific problems in your unique environment. Josephsen introduces Nagios “from the ground up,” showing how to plan for success and leverage today’s most valuable monitoring best practices. Then, using practical examples, real directives, and working code, Josephsen presents detailed monitoring solutions for Windows, Unix, Linux, network equipment, and other platforms and devices. You’ll find thorough discussions of advanced topics, including the use of data visualization to solve complex monitoring problems. This is also the first Nagios book with comprehensive coverage of using Nagios Event Broker to transform and extend Nagios.

- Understand how Nagios works, in depth: the host and service paradigm, plug-ins, scheduling, and notification
- Configure Nagios successfully: config files, templates, timeperiods, contacts, hosts, services, escalations, dependencies, and more
- Streamline deployment with scripting templates, automated discovery, and Nagios GUI tools
- Use plug-ins and tools to systematically monitor the devices and platforms you need to monitor, the way you need to monitor them
- Establish front-ends, visual dashboards, and management interfaces with MRTG and RRDTool
- Build new C-based Nagios Event Broker (NEB) modules, one step at a time
- Contains easy-to-understand code listings in Unix shell, C, and Perl

If you’re responsible for systems monitoring infrastructure in any organization, large or small, this book will help you achieve the results you want–right from the start.

David Josephsen is Senior Systems Engineer at DBG, Inc., where he maintains a collection of geographically dispersed server farms. He has more than a decade of hands-on experience with Unix systems, routers, firewalls, and load balancers in support of complex, high-volume networks. Josephsen's certifications include CISSP, CCNA, CCDA, and MCSE. His co-authored work on Bayesian spam filtering earned a Best Paper award at USENIX LISA 2004. He has been published in both ;login and Sysadmin magazines on topics relating to security, systems monitoring, and spam mitigation.

Introduction
CHAPTER 1 Best Practices
CHAPTER 2 Theory of Operations
CHAPTER 3 Installing Nagios
CHAPTER 4 Configuring Nagios
CHAPTER 5 Bootstrapping the Configs
CHAPTER 6 Watching
CHAPTER 7 Visualization
CHAPTER 8 Nagios Event Broker Interface
APPENDIX A Configure Options
APPENDIX B nagios.cfg and cgi.cfg
APPENDIX C Command-Line Options
Index

Building a Monitoring Infrastructure with Nagios

By David Josephsen

Building a Monitoring Infrastructure with Nagios By David Josephsen

Build real-world, end-to-end network monitoring solutions with Nagios

This is the definitive guide to building low-cost, enterprise-strength monitoring infrastructures with Nagios, the world's leading open source monitoring tool. Network monitoring specialist David Josephsen goes far beyond the basics, demonstrating how to use third-party tools and plug-ins to solve the specific problems in your unique environment. Josephsen introduces Nagios "from the ground up," showing how to plan for success and leverage today's most valuable monitoring best practices. Then, using practical examples, real directives, and working code, Josephsen presents detailed monitoring solutions for Windows, Unix, Linux, network equipment, and other platforms and devices. You'll find thorough discussions of advanced topics, including the use of data visualization to solve complex monitoring problems. This is also the first Nagios book with comprehensive coverage of using Nagios Event Broker to transform and extend Nagios.

- Understand how Nagios works, in depth: the host and service paradigm, plug-ins, scheduling, and notification
- Configure Nagios successfully: config files, templates, timeperiods, contacts, hosts, services, escalations, dependencies, and more
- Streamline deployment with scripting templates, automated discovery, and Nagios GUI tools
- Use plug-ins and tools to systematically monitor the devices and platforms you need to monitor, the way you need to monitor them
- Establish front-ends, visual dashboards, and management interfaces with MRTG and RRDTool
- Build new C-based Nagios Event Broker (NEB) modules, one step at a time
- Contains easy-to-understand code listings in Unix shell, C, and Perl

If you're responsible for systems monitoring infrastructure in any organization, large or small, this book will help you achieve the results you want–right from the start.

David Josephsen is Senior Systems Engineer at DBG, Inc., where he maintains a collection of geographically dispersed server farms. He has more than a decade of hands-on experience with Unix systems, routers, firewalls, and load balancers in support of complex, high-volume networks. Josephsen's certifications include CISSP, CCNA, CCDA, and MCSE. His co-authored work on Bayesian spam filtering earned a Best Paper award at USENIX LISA 2004. He has been published in both ;login and Sysadmin magazines on topics relating to security, systems monitoring, and spam mitigation.

Introduction
CHAPTER 1 Best Practices
CHAPTER 2 Theory of Operations
CHAPTER 3 Installing Nagios
CHAPTER 4 Configuring Nagios
CHAPTER 5 Bootstrapping the Configs
CHAPTER 6 Watching
CHAPTER 7 Visualization

CHAPTER 8 Nagios Event Broker Interface
APPENDIX A Configure Options
APPENDIX B nagios.cfg and cgi.cfg
APPENDIX C Command-Line Options
Index

Building a Monitoring Infrastructure with Nagios By David Josephsen Bibliography

- Sales Rank: #760170 in Books
- Published on: 2007-03-02
- Released on: 2007-02-20
- Original language: English
- Number of items: 1
- Dimensions: 9.10" h x .70" w x 6.90" l, 1.11 pounds
- Binding: Paperback
- 264 pages

Introduction

This is a book about untrustworthy machines. Machines in fact, which are every bit, as untrustworthy as they are critical to our well-being. But then I don't need to bore you with laundry lists of how prevalent computer systems have become, or horror stories about what can happen when they fail. If you picked up this book, then I'm sure you're well aware of the problems; layer upon layer of interdependent libraries hiding bugs in their abstraction, script kiddies, viruses, DDOS attacks, hardware failure, end-user error, back-hoe's, hurricanes, and on and on. It doesn't matter whether the root-cause is malicious, or accidental, your systems will fail, and when they do, only two things will save you from the downtime; redundancy, and monitoring systems.

Do it right the first time

In concept, monitoring systems are simple, an extra system, or collection of systems whose job it is to watch the other systems for problems. For example the monitoring system could periodically connect to a web server, to make sure it responds, and if not, send notifications to the administrators. And while it all sounds quite straightforward, monitoring systems have grown into expensive, complex pieces of software. Many now have agents larger than 500Mb, include proprietary scripting languages, and sport price tags above $60,000.

When implemented correctly, a monitoring system can be your best friend. It can notify admins of glitches before they become crises, help architects tease out patterns corresponding to chronic interoperability issues, and give engineers detailed capacity planning info. A good monitoring system will help the security guys correlate interesting events, show the network operations center personnel where the bandwidth bottlenecks are, and provide management much needed high level visibility into the critical systems they bet their business on. A good monitoring system can help you uphold your service level agreement (SLA), and even take steps to solve problems without waking anyone up at all. Good monitoring systems save money, bring stability to complex environments, and make everyone happy.

When done poorly however, the very same system can wreak havoc. Bad monitoring systems cry wolf at all hours of the night so often that nobody pays attention anymore, they install backdoors into your otherwise secure infrastructure, leech time and resources away from other projects, and congest network links with megabyte upon megabyte of health checks. Bad monitoring systems can really suck.

Unfortunately, getting it right the first time isn't as easy as you might think, and in my experience, a bad monitoring system doesn't usually survive long enough to get fixed. Bad monitoring systems are just too much of a burden on everyone involved, including the systems being monitored. In this context, it's easy to see is why large corporations, and governments employ full-time monitoring specialists, and purchase software with six-figure price tags. They know how important it is to get it right the first time.

Small to medium sized businesses and universities can have environments as complex or even more complex then large companies, but they obviously don't have the luxury of high-priced tools, and specialized expertise. Getting a well-built monitoring infrastructure in these environments, with their geographically dispersed campuses and satellite offices can be a challenge. But having spent the better part of the last 7 years building and maintaining monitoring systems, I'm here to tell you that not only is it possible to get it done right the first time, but you can do it for free, with a bit of elbow grease, some open source tools, and a pinch of imagination.

Why Nagios?

Nagios is in my opinion the best system and network monitoring tool available, open source or otherwise. Its modularity and straightforward approach to monitoring makes it easy to work with and highly scalable. Further, Nagios' open source license makes it freely available and easy to extend to meet your specific needs. Instead of trying to do everything for you, Nagios excels at interoperability with other open source tools, which makes it very flexible. If you're looking for a monolithic piece of software with checkboxes that solve all your problems, this probably isn't the book for you, but before you stop reading, give me another paragraph or two to convince you that the checkboxes aren't really what you're looking for.

The commercial offerings get it wrong mainly because their approach to the problem assumes that everyone wants the same solution. To a certain extent, this is true. Everyone has a large glob of computers and network equipment, and wants to be notified if some subset of it fails. So if you want to sell monitoring software, the obvious way to go about it is to create a piece of software that knows how to monitor every conceivable piece of computer software and networking gear in existence. The more gadgets your system can monitor, the more people you can sell it to. To someone who wants to sell monitoring software, it's easy to believe that monitoring systems are turnkey solutions, and whoever's software can monitor the largest number of gadgets wins.

The commercial packages I've worked with all seem to follow this logic. Not unlike the borg, methodically locating new computer gizmos and adding the requisite monitoring code to their solution, or worse, acquiring other companies who already know how to monitor lots of computer gadgetry, and bolting that companies code on to their own. They quickly become obsessed with features, creating enormous spreadsheets of supported gizmos. Their software engineers exist so that the pre-sales engineers can come to your office and say to your managers through seemingly layers of white gleaming teeth; "Yes our software can monitor that".

The problem is, monitoring systems are not turnkey solutions. They require a large amount of customization before they really start solving problems, and herein lay the difference between people selling monitoring software and those designing and implementing monitoring systems. When you're trying to build a monitoring system, a piece of software that can monitor every gadget in the world by clicking a checkbox is not as useful to you as one that makes it easy to monitor what you need, in exactly the manner that you want. By focusing on *what* to monitor, the proprietary solutions neglect the '*how'*, which limits the context in which they may be used.

Take 'ping' for example. Every monitoring system I've ever dealt with uses ICMP Echo requests, otherwise known as 'pings' to check host availability in one way or another. But if you want to control *how* a proprietary monitoring system uses ping, architectural limitations become quickly apparent. Lets say I want to specify the number of ICMP packets to send or want to be able to send notifications based on the round trip time of the packet in microseconds instead of simple pass/fail. More complex environments may necessitate that I use IPv6 pings, or that I portknock1 before I ping. The problem with the monolithic, feature-full approach is that these changes represent changes to the core application logic, and are therefore non-trivial to implement.

In the commercial monitoring applications I've worked with, if these ping examples could be performed at all they would require re-implementing the ping logic in the monitoring system's proprietary scripting language. In other words, you would have to toss out the built-in ping functionality altogether. Perhaps, being able to control the specifics of ping checks is of questionable value to you, but if you don't really have any control over something as basic as ping, what are the odds, that you'll have finite enough control over the most important checks in your environment? They've made the assumption that they know *how* you want to ping things, and from then on it was game over; they never thought about it again. And why would they? The ping feature is already in the spreadsheet after all.

When it comes to gizmos, Nagios' focus is on modularity. Single purpose monitoring applets called 'plugins' provide support for specific devices and services. Rather than participating in the feature arms race, hardware support is community driven. As community members have a need to monitor new devices or services, new plugins are written, and usually a good bit more quickly than the commercial apps add the same support. In practice Nagios will always support everything you need it to, and without ever needing to upgrade Nagios itself. Nagios also provides the best of both worlds when it comes to support, with several commercial options, as well as a thriving and helpful community that provides free support through various forums and mailing lists.

Choosing Nagios as your monitoring platform means that your monitoring effort will be limited by your own imagination, technical prowess, and political savvy. Nagios can go anywhere you want it to, and the trip there is usually pretty simple. And while Nagios can do everything the commercial apps can and more, and without the bulky, insecure agent install, it usually doesn't compare favorably to commercial monitoring systems simply because when spreadsheets are parsed, Nagios doesn't have as many checks. In fact if they're counting correctly, Nagios has no checks at all, because technically it doesn't know *how* to monitor anything; it prefers that you tell it how. 'How' in fact, is exactly the variable that the aforementioned checkbox cannot encompass. Checkboxes cannot ask 'how', and therefore you don't want them.

What's in this book?

While Nagios is the biggest piece of the puzzle, it's only one of the myriad of tools that make up a world-class open source monitoring system. With several books, superb online documentation, and lively and informative mailing lists, it's also the best-documented piece of the puzzle. So my intention in writing this book is to pick up where the documentation leaves off. This is not a book about Nagios, as much as it is a book about the construction of monitoring systems using Nagios, and there is much more to building monitoring systems than configuring a monitoring tool.

I'll cover the usual configuration boilerplate, but configuring and installing Nagios is not my primary focus. Instead, in order to help you build great monitoring systems I need to introduce you to the protocols, and tools that enhance Nagios' functionality and simplify its configuration. I need to give you an in-depth understanding of the inner workings of Nagios itself so you can extend it to do whatever you might need. I need to spend some time in this book exploring possibilities, because Nagios is only really limited by what you feel it can do. And finally, I need write about things only loosely related to Nagios, like best practices, SNMP, visualizing time-series data, and various Microsoft scripting technologies like WMI and WSH.

Perhaps most importantly, I need to document Nagios itself in a different way than it normally is. By introducing it in terms of a task efficient scheduling and notification engine, I can keep things simple while at the same time talking about the internals up front. Rather than relegating important information to the seldom-read 'advanced' section I'll empower you early by covering topics like plugin customization and scheduling as core concepts.

Although the chapters more or less stand on their own, and I've tried to make the book as reference friendly as possible, I think it reads better as a progression from start to end. I encourage you to read from cover to cover, skipping over anything you are already familiar with. The text is not large, but I think you'll find it dense with information and even the most seasoned monitoring veterans should find more than a few nuggets of wisdom.

The chapters tend to build on each other, and casually introduce Nagios specific details in the context of more general monitoring concepts. Because there are many important decisions that need to be made before any software is installed, I begin with best practices, in Chapter 1. This should get you thinking in terms of what needs to take place for your monitoring initiative to be successful such as how to go about implementing, whom to involve, and what pitfalls to avoid.

Chapter 2 builds on Chapter 1's general design guidance by providing a theoretical overview of Nagios from the ground up. Rather than inundating you with configuration minutiae, Chapter 2 will give you a detailed understanding of how Nagios works without being overly specific about configuration directives. This knowledge will go a long way toward making configuration more transparent later.

Before we can configure Nagios to monitor our environment, we need to install it. Chapter 3 should help you install Nagios, either from source or via a package-manager.

Chapter 4 is the dreaded configuration chapter. Configuring Nagios for the first time is not something most people consider to be fun, but I hope I've kept it as painless as possible by taking a bottom up approach, only documenting the most used and required directives, providing up front examples, and specifying exactly what objects refer to what other objects and how.

Most people who try Nagios become very attached to it2, and are loathe to return to using anything else. But if there is a universal complaint, it is certainly configuration. Chapter 5 takes a bit of a digression in order to document some of the tools available to make configuration easier to stomach. These include automated discovery tools as well as graphical user interfaces.

In Chapter 6 we are finally ready to get into the nitty-gritty of watching systems, including specific examples with Nagios plugin configuration syntax solving real world problems. I begin with a section on watching Microsoft Windows boxes, followed by a section on Unix, and finally the 'other stuff' section, which encompasses networking gear, and environmental sensors.

Chapter 7, covers one of my favorite topics, data visualization. Good data visualization solves problems that couldn't be solved otherwise, and I'm excited about the options that exist now as well as what's on the horizon. With fantastic visualization tools like RRDTool and no fewer than 12 different glue layers to choose from, graphing time series data from Nagios is getting easier every day, but this chapter doesn't stop at mere line-graphs.

And finally, now that you know the rules, it's time to teach you how to break them. At the time of this writing Chapter 8 is the only documentation I'm aware of that covers the new Nagios Event Broker interface. The event broker is the most powerful Nagios interface available. Mastering it rewards you with nothing less than the ability to rewrite Chapter 2 for yourself by fundamentally changing any aspect of how Nagios operates or extending it to meet any need you might have. I describe how the event broker works, and walk you through building a NEB module.

Who should read this book?

If you are a systems administrator with a closet full of Unix systems, Windows systems, and assorted network gadgetry, and need a world class monitoring system on the cheap, this book is for you. Contrary to what you might expect, building monitoring systems is not a trivial undertaking. Constructing the system that potentially interacts with every TCP-based device in your environment requires quite a bit of knowledge on your part. But don't let that give you pause, systems monitoring has taught me more than anything else I've done in my career, and in my experience no matter what your level of knowledge, working with monitoring systems has a tendency to constantly challenge your assumptions, deepen your understanding, and keep you right on the edge of what you know.

To get the most out of this book, you should have a pretty good handle on the text-based internet protocols you use regularly such as SMTP, and HTTP. Although it interacts with Windows servers very well, Nagios is meant to run on Linux, which makes the text pretty Linux heavy, so a passing familiarity with Linux, or Unix-like systems is helpful. Although not strictly required, you should also have some programming skills. The book has a fair number of code listings, but I've tried to keep them as straightforward and easy-to-follow as possible. With the exception of Chatper 8, which is exclusively C, the code listings are written in either Unix shell, or Perl.

Perhaps the only strict requirement is that you approach the subject matter with a healthy dose of open curiosity. If something seems unclear, don't be discouraged, check out the online documentation, ask on the lists, or even shoot me a mail, I'd be glad to help if I can.

Have fun!

--dave.

davesbook@skeptech.org

---

1 http://www.portknocking.org/

2 Dare I say love it?

Read Building a Monitoring Infrastructure with Nagios By David Josephsen for online ebook

Building a Monitoring Infrastructure with Nagios By David Josephsen Free PDF d0wnl0ad, audio books, books to read, good books to read, cheap books, good books, online books, books online, book reviews epub, read books online, books to read online, online library, greatbooks to read, PDF best books to read, top books to read Building a Monitoring Infrastructure with Nagios By David Josephsen books to read online.

Online Building a Monitoring Infrastructure with Nagios By David Josephsen ebook PDF download

Building a Monitoring Infrastructure with Nagios By David Josephsen Doc

Building a Monitoring Infrastructure with Nagios By David Josephsen Mobipocket

Building a Monitoring Infrastructure with Nagios By David Josephsen EPub

4B23E8QHG5T: Building a Monitoring Infrastructure with Nagios By David Josephsen