Realtime
publishers

# New Best Practices in Virtual and Cloud Management: Performance, Capacity, Compliance, and Workload Automation

Greg Shields

sponsored by
vmware®

## Copyright Statement

© 2012 Realtime Publishers. All rights reserved. This site contains materials that have been created, developed, or commissioned by, and published with the permission of, Realtime Publishers (the "Materials") and this site and any such Materials are protected by international copyright and trademark laws.

THE MATERIALS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of Realtime Publishers its web site sponsors. In no event shall Realtime Publishers or its web site sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

The Materials (including but not limited to the text, images, audio, and/or video) may not be copied, reproduced, republished, uploaded, posted, transmitted, or distributed in any way, in whole or in part, except that one copy may be downloaded for your personal, non-commercial use on a single computer. In connection with such use, you may not modify or obscure any copyright or other proprietary notice.

The Materials may contain trademarks, services marks and logos that are the property of third parties. You are not permitted to use these trademarks, services marks or logos without prior written consent of such third parties.

Realtime Publishers and the Realtime Publishers logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners.

If you have any questions about these terms, or if you would like information about licensing materials from Realtime Publishers, please contact us via e-mail at info@realtimepublishers.com.

# Chapter 3: New Best Practices in Configuration & Compliance Management

*Computers may be deterministic, but the people who use them aren't.*

I love that quote, and not just because I'm the one who dreamed it up. If you've ever wondered how a system driven by 1s and 0s can be so irritatingly unpredictable, just take a look at its users. The machine merely does what's asked of it; it's the users whose activities are usually the source of chaotic behavior.

Users in this case needn't necessarily be limited to just regular, non-administrative users. We in IT sometimes forget that we're users too. And, in fact, our activities are often the most impactful on a system.

There's a reason why our activities tend to be more problematic: While regular users interact via published and highly-controlled application interfaces, we administrators have carte blanche. We can perform any action we want, many of which involve heavy resource consumers. As a result, our administrative actions tend to be more difficult to profile.

This scenario impacts our virtual environments all the time. You know the story: One day you hear whispers that an IT service isn't behaving well. A cursory check of its resources shows nothing of interest. It isn't until much later that you learn some *other* administrator has been executing a dozen *other* tasks on the same host. Externalities in a shared virtual environment often cause more impact than the systems themselves.

That said, we must never forget that computers are deterministic, so there's an argument that unpredictable behaviors are merely *behaviors you're just not monitoring*. As the previous chapter argues, the seemingly chaotic virtual environment is really just one with a lot of variables. Sometimes, as Figure 3.1 suggests, those variables are the actions being performed by that system's various caretakers.

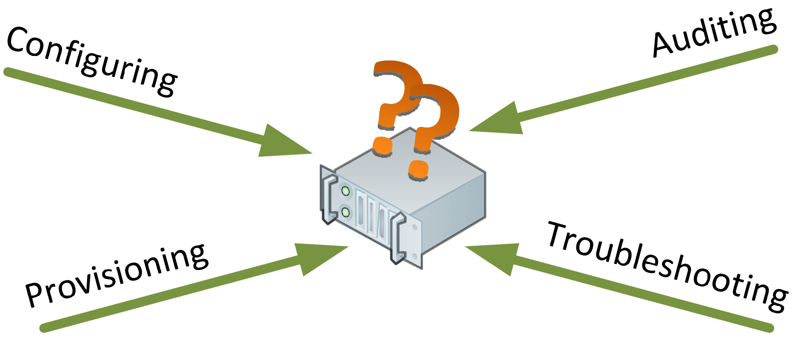


**Figure 3.1: Independent actions beget unpredictability.**

Think for a minute about the actions that can happen—often simultaneously—in a system of shared resources. One administrator makes a configuration change. Another provisions a new virtual machine or application. At the same time, a third troubleshoots some issue while a fourth is auditing the system for exactly those configurations being changed.

*Chaos, indeed.*

## The Black Box Approach to Configuration Management

Bringing order to the chaos requires implementing yet another new best practice: configuration management. Not just any configuration management; rather, a very special kind that fits the unique characteristics of a shared virtual environment.

That "special kind" of configuration management ties directly into the concepts and functionalities discussed to this point. Chapters 1 and 2 discuss at length the need for assistive tools in managing performance and capacity. You should, at this point, recognize that there are limits to how many variables your unaided brain can monitor and correlate. Chapter 2's conclusion further foreshadows the notion that the performance, capacity, and configuration management activities are more tied together than you'd think. When people enact change without coordination, the result is unexpected behavior.

So, what's the solution? As you might expect, the right solution follows the same black box approach discussed earlier. At this point, your black box is already collecting data and generating recommendations. All it needs are a few extra features that correlate *what's being changed* with *what's being monitored*.

Figure 3.2 gives you an idea of these new features. By continuously monitoring your environment's inventory, you gather a reference catalog of each configuration item. Centralizing execution ensures that every change is always logged to a central location. Integrating provisioning tasks adds intelligence about why and where resources are being consumed. These three facilitate the fourth new feature, automating remediation, which enables rolling back an environment to a previous state should the need arise.

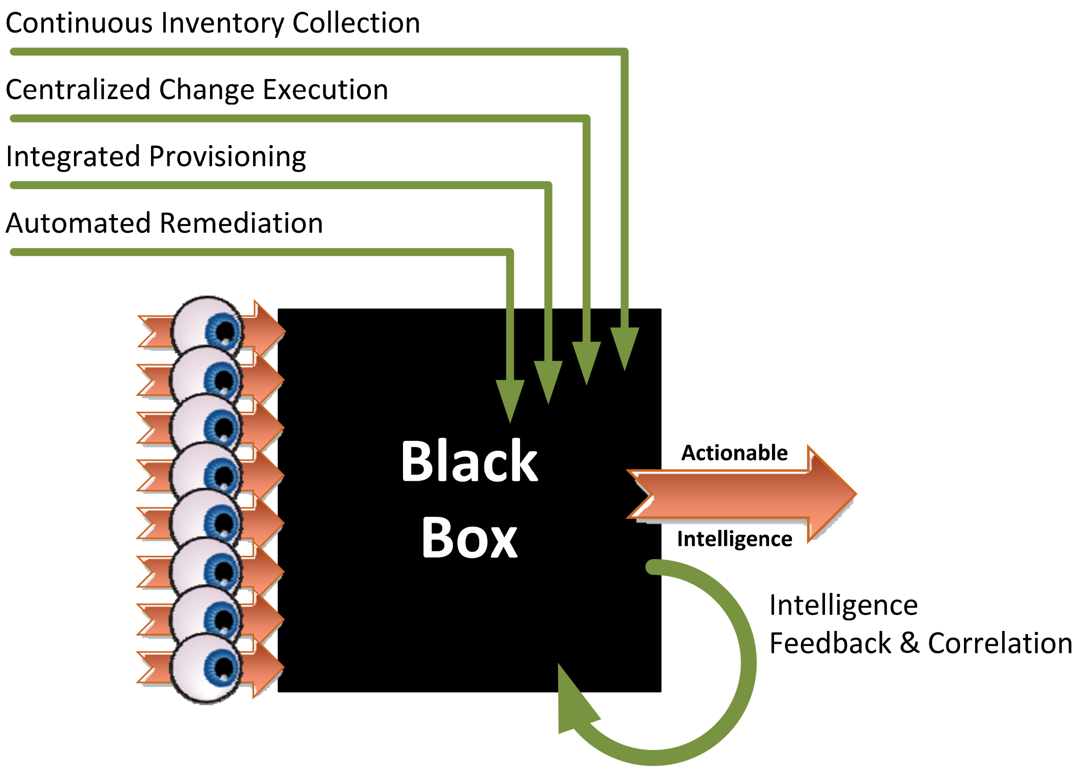

Continuous Inventory Collection
Centralized Change Execution
Integrated Provisioning
Automated Remediation
Black Box
Actionable
Intelligence
Intelligence Feedback & Correlation

**Figure 3.2: Integrating configuration management into the black box.**

As you already know from previous chapters, the primary goal of the black box is to distill the fire hose of metrics data that is constantly being captured. That distillation converts raw metrics into suggested actions that define what you might do next. Although previous efforts focused on the performance and capacity suggestions, that data can also drive a kind of feedback loop that takes into account each configuration change:

- You make a change
- The change alters the environment's behavior
- A further change is suggested
- Repeat

The benefit of this feedback is in being able to *correlate the environment's behaviors with the execution of a change*. Doing so, in a way, makes your users as deterministic as the systems they manage.

## Enacting Change, on the Virtual Machine and in the Virtual Machine

You must first recognize that changes can happen on any virtual environment element. Virtual machines are assigned varying resource levels. Virtual (and physical) network switches are reconfigured to meet updated requirements. Additional storage is routinely provisioned. And, most notably, configurations *inside each virtual machine* are modified to solve problems, tweak performance, or enhance security.

Various change management solutions have existed for years that facilitate the process. Historically, these solutions have tended to focus on what's inside the operating system (OS). They haven't really cared whether the computer is physical or virtual. That disinterest works when change management alone is your goal; *it doesn't work when your needs include a change's impact on shared resources*.

The historical solutions are also limited by their visibility. They simply aren't designed to interact with a virtual environment's hypervisor or host cluster resources. As a consequence, the new best practice integrates change management activities into the virtual platform's management tools.

Figure 3.3 shows one approach to this design. In it, the same virtual platform tools that facilitate performance and capacity management are used to enact change on hypervisor objects as well as inside the virtual machine. Agents installed into each guest machine facilitate the installations, updates, or other configuration changes that source from the virtual platform manager. These agents are necessary because they are installed into the virtual machine's OS. This installation allows them to execute actions in the correct administrative context.

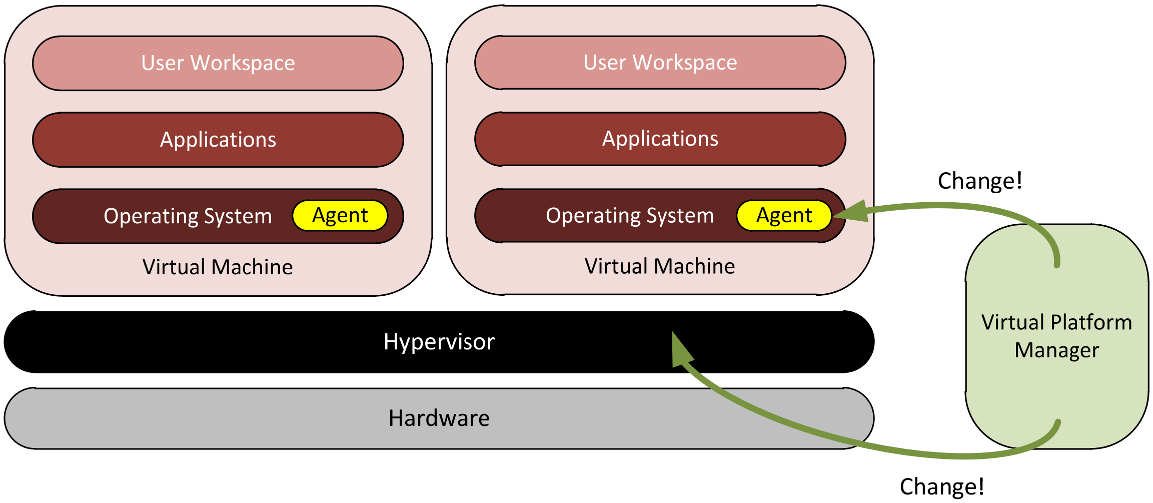


**Figure 3.3: Virtual platform agents…of change.**

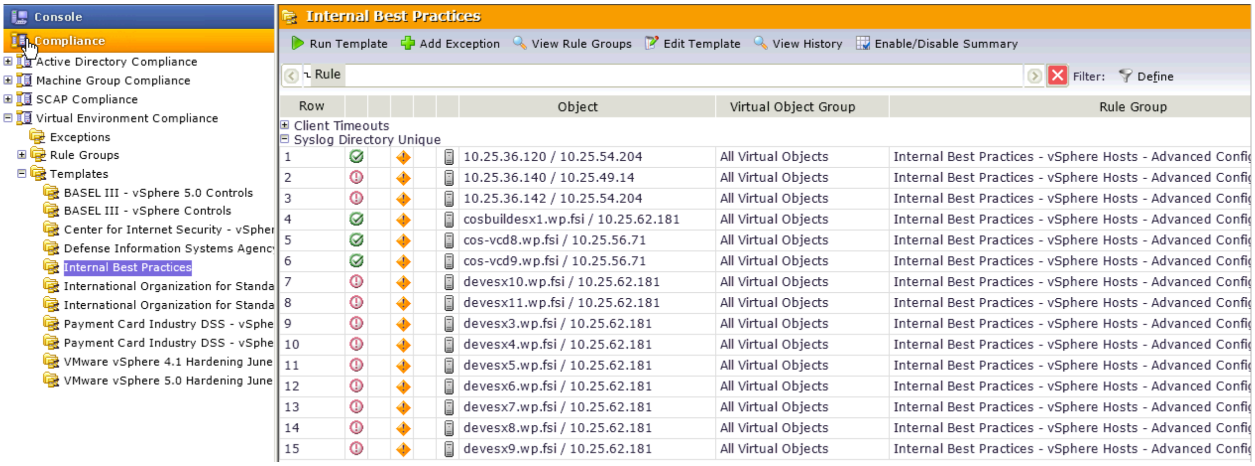
**Not Necessarily in the Box**

This added functionality isn't necessarily going to be part of the core tools a virtual platform offers. Those core tools facilitate performing actions on virtual environment objects; however, they tend not to add all the extra features this chapter is referring to. These features usually cost extra.

## Wielding the Vast Power of Undo

Unifying configuration management with the additional best practices suggested in this guide isn't the only benefit of this approach to configuration management. When your virtual environment enjoys continuous inventory collection and centralized change execution, you gain a very key additional benefit: *the universal undo*.

Recognize that this isn't just any old undo. Rather, it is one that's backed by all the feature sets intrinsic to virtualization. The right solution will have the ability to back out changes to hypervisor objects, returning them to previous configurations. Figure 3.4 shows a report on a change, detailing what happened, where it happened, and when it was detected. With the right solution, undoing these changes can happen via a combination of real-time detection, identification, and remediation of the offending configuration. This series of events ensures that an errant change can quickly be removed with minimal impact.



- ⊞ Active Directory Compliance
- ⊞ Machine Group Compliance
- ⊞ SCAP Compliance
- ⊟ Virtual Environment Compliance
  - Exceptions
  - ⊞ Rule Groups
  - ⊟ Templates
    - BASEL III - vSphere 5.0 Controls
    - BASEL III - vSphere Controls
    - Center for Internet Security - vSpher
    - Defense Information Systems Agenc
    - Internal Best Practices
    - International Organization for Standa
    - International Organization for Standa
    - Payment Card Industry DSS - vSphe
    - Payment Card Industry DSS - vSphe
    - VMware vSphere 4.1 Hardening June
    - VMware vSphere 5.0 Hardening June

⊞ Client Timeouts
⊟ Syslog Directory Unique

| Row | | | | | | Object | Virtual Object Group | Rule Group |
|---|---|---|---|---|---|---|---|---|
| 1 | ✓ | | ◆ | | ▯ | 10.25.36.120 / 10.25.54.204 | All Virtual Objects | Internal Best Practices - vSphere Hosts - Advanced Confi |
| 2 | ! | | ◆ | | ▯ | 10.25.36.140 / 10.25.49.14 | All Virtual Objects | Internal Best Practices - vSphere Hosts - Advanced Confi |
| 3 | ! | | ◆ | | ▯ | 10.25.36.142 / 10.25.54.204 | All Virtual Objects | Internal Best Practices - vSphere Hosts - Advanced Confi |
| 4 | ✓ | | ◆ | | ▯ | cosbuildesx1.wp.fsi / 10.25.62.181 | All Virtual Objects | Internal Best Practices - vSphere Hosts - Advanced Confi |
| 5 | ✓ | | ◆ | | ▯ | cos-vcd8.wp.fsi / 10.25.56.71 | All Virtual Objects | Internal Best Practices - vSphere Hosts - Advanced Confi |
| 6 | ✓ | | ◆ | | ▯ | cos-vcd9.wp.fsi / 10.25.56.71 | All Virtual Objects | Internal Best Practices - vSphere Hosts - Advanced Confi |
| 7 | ! | | ◆ | | ▯ | devesx10.wp.fsi / 10.25.62.181 | All Virtual Objects | Internal Best Practices - vSphere Hosts - Advanced Confi |
| 8 | ! | | ◆ | | ▯ | devesx11.wp.fsi / 10.25.62.181 | All Virtual Objects | Internal Best Practices - vSphere Hosts - Advanced Confi |
| 9 | ! | | ◆ | | ▯ | devesx3.wp.fsi / 10.25.62.181 | All Virtual Objects | Internal Best Practices - vSphere Hosts - Advanced Confi |
| 10 | ! | | ◆ | | ▯ | devesx4.wp.fsi / 10.25.62.181 | All Virtual Objects | Internal Best Practices - vSphere Hosts - Advanced Confi |
| 11 | ! | | ◆ | | ▯ | devesx5.wp.fsi / 10.25.62.181 | All Virtual Objects | Internal Best Practices - vSphere Hosts - Advanced Confi |
| 12 | ! | | ◆ | | ▯ | devesx6.wp.fsi / 10.25.62.181 | All Virtual Objects | Internal Best Practices - vSphere Hosts - Advanced Confi |
| 13 | ! | | ◆ | | ▯ | devesx7.wp.fsi / 10.25.62.181 | All Virtual Objects | Internal Best Practices - vSphere Hosts - Advanced Confi |
| 14 | ! | | ◆ | | ▯ | devesx8.wp.fsi / 10.25.62.181 | All Virtual Objects | Internal Best Practices - vSphere Hosts - Advanced Confi |
| 15 | ! | | ◆ | | ▯ | devesx9.wp.fsi / 10.25.62.181 | All Virtual Objects | Internal Best Practices - vSphere Hosts - Advanced Confi |

**Figure 3.4: Who did what, when?**

Certain changes can also be rolled back with the assistance of virtual machine snapshots. These snapshots are built into every major virtual platform today, and provide a way to return a virtual machine back to the snapshot's point in time. With them, you should easily see the possibilities in orchestrating universal undo:

- You instruct the solution to execute a change
- The solution snapshots the virtual machine prior to execution
- That change later needs to be removed
- The solution reverts the virtual machine back to the snapshot

The experienced virtual administrator should at this point recognize that this functionality can be a double-edged sword. Snapshots are absolutely useful for reverting virtual machines, but keeping too many snapshots lying around can be a bad thing. Every linked snapshot impacts performance as well as consumes storage space. Snapshots have also been notorious for creating problems when they're allowed to exist for too long.

For these reasons, you should expect additional logic built into the solution can later consolidate snapshots after a predetermined period. Doing so ensures that snapshots don't remain resident for extended periods, essentially offering the best of both worlds. The right solution balances both requirements, delivering a reduction in change-related risk while limiting the risks inherent to the snapshots themselves.

**Useful for Anticipated; Really Useful for Unanticipated**

I can't overstate the benefit in offering global undo functionality. Not only *anticipated* changes can be rolled back when they're found to be ineffective or problematic but also *unanticipated* changes. Remember that the defining point of this solution is to centralize the dissemination of changes. When that solution also supports rollback, administrators gain the ability to click once and back out any change they find.

## Configuration Management Feeds Compliance Management

You can't talk about configuration management these days without a nod to maintaining compliance. Compliance management in this discussion refers not only to external guidelines handed down from regulation and/or security policies but also internal guidelines that ensure systems are configured to meet best practices. As you can probably guess, the activities in compliance management fit perfectly within our black box approach:

- Services feeding data into your black box are constantly capturing inventory information
- That inventory information contains auditable configuration items on hypervisor objects as well as inside virtual machines
- Centralizing provisioning and the execution of changes through the same system ensure configuration updates are similarly captured
- Feedback keeps everyone honest

Masking a few of these out for a minute (Figure 3.5) allows us to focus on two capabilities that are central to compliance management's new best practices. One of those, compliance templates, is new to our black box diagram. Compliance templates are exactly as they seem: lists of configurations that define specific settings required to meet an established baseline.

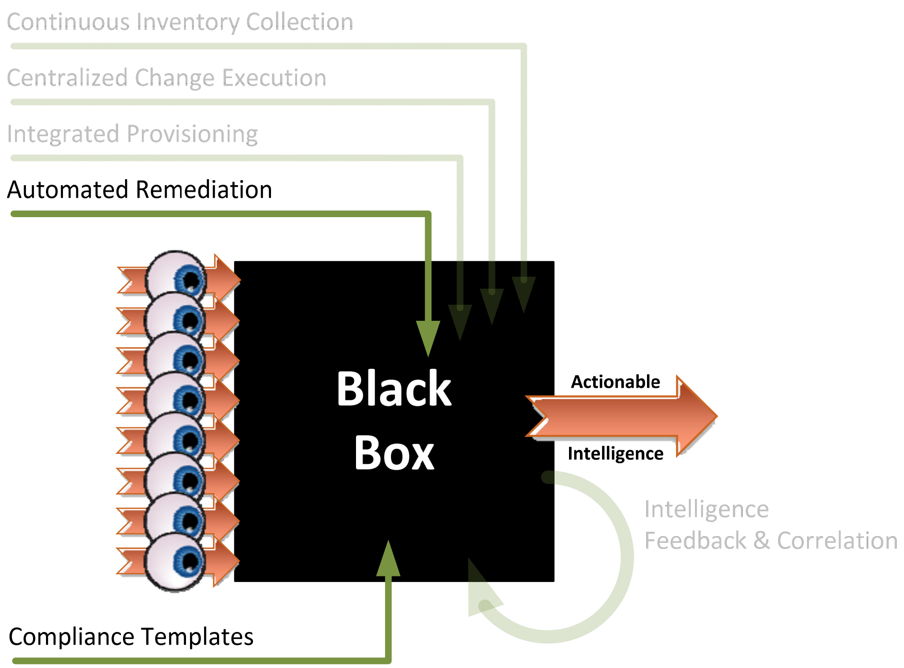


**Figure 3.5: Templates and remediation.**

These baselines can be anything. One baseline might identify settings that define a corporate security policy. Another might outline requirements handed down by an external regulatory agency. A third may include settings that follow performance best practices.

*Your limit is only your imagination.*

### What Makes a Template?

Notwithstanding what its baseline attempts to accomplish, a prototypical template generally comprises three elements:

- The configuration item
- The compliance rule
- The validation code used to verify the setting

That third element, the validation code, is where compliance templates offer their greatest utility. It is at the same time the least likely to be directly managed by a virtual environment's administrators (but more on that in a moment).

The job of the validation code is to automatically validate compliance settings on a monitored object. Recall from Chapter 1 that virtually every data center-class piece of hardware exposes collectable metrics (see Figure 3.6). So too does every hypervisor object as well as most configuration items within an OS and its installed applications. Now augmented with its configuration management functionality, our black box now possesses the ability to execute code against those objects.

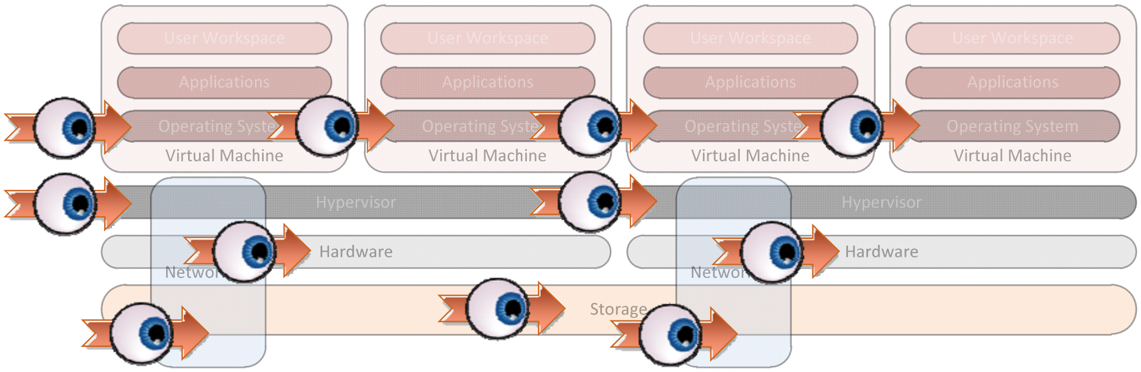
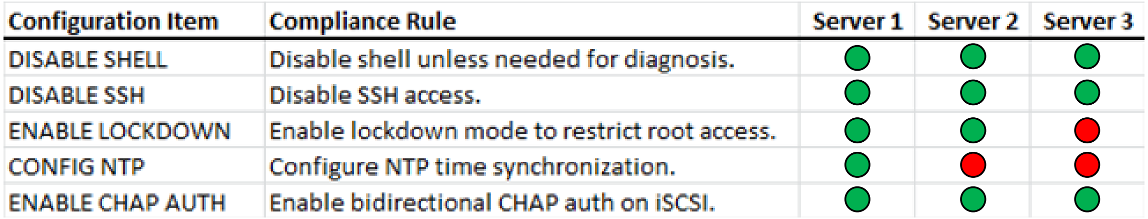


**Figure 3.6: Monitors, (still) everywhere.**

*So, why not use those features to automate each object's validation?*

You can do so with the right solution. In fact, that automation enables the compliance management activity to occur constantly and generally without administrator intervention. Such a solution can evaluate compliance templates with each inventory collection pass, and then again when changes are either invoked through the system or identified as having occurred.

What results is an easy-to-read report that outlines the settings on each object that aren't meeting the baseline. Figure 3.7 illustrates a representation. Reports like these ease the job of auditing for auditors and security officers. These reports also ease troubleshooting, giving administrators a heads-up warning when an errant change has taken an object out of compliance. Similar to Chapter 2's capacity management stoplight charts, these reports enable you to drill down from these high-level monitors to expose additional detail.



| Configuration Item | Compliance Rule | Server 1 | Server 2 | Server 3 |
|---|---|---|---|---|
| DISABLE SHELL | Disable shell unless needed for diagnosis. | 🟢 | 🟢 | 🟢 |
| DISABLE SSH | Disable SSH access. | 🟢 | 🟢 | 🟢 |
| ENABLE LOCKDOWN | Enable lockdown mode to restrict root access. | 🟢 | 🟢 | 🔴 |
| CONFIG NTP | Configure NTP time synchronization. | 🟢 | 🔴 | 🔴 |
| ENABLE CHAP AUTH | Enable bidirectional CHAP auth on iSCSI. | 🟢 | 🟢 | 🟢 |

**Figure 3.7: Automatically validating compliance.**

Centralizing change execution needn't necessarily stop with running validation scripts. The contents of these reports can further drive another capability: *automated remediation*. That remediation process leverages aspects of the template's validation code—often in cooperation with administrator input or established policies—to automatically correct any inappropriate settings the moment they're deemed out of compliance. The activity then simplifies to four principle steps: inventory, validate, alert, and remediate.

### Overcoming the Achilles Heel: Who Authors the Templates?

All of these templates and automations offer incredible benefit until you realize a fairly sizeable Achilles Heel: *They can be cumbersome to construct.* Their challenge is two-fold. The first part is in translating a regulation's requirements into specific settings that require monitoring. Try this one, as an exercise:

> PCI-DSS Requirement 6.5.9 requires the implementation of controls that inhibit Cross-Site Request Forgery (CSRF) attacks. It specifies a testing procedure that states, "Do not reply on authorization credentials and tokens automatically submitted by browsers."

Which of your virtual environment's objects need monitoring to meet this requirement? What interfaces expose the necessary information? Exactly what code needs to be written to verify their configuration and remediate it when it's found to be incorrect? The answers to all of these questions are complex, and likely too complex for many administrators to answer comfortably.

The problem's second part is arguably more insidious because it pits your opinion against the opinion of the person performing the audit: Assuming you determine an answer to the first part, *how do you know yours is the correct answer?*

All by yourself, you don't. Or, more specifically, you don't without testing your assertion against those of your auditors. Unlike the other activities discussed in this guide, compliance management is above all a test of interpretation. Most compliance mandates (internal or external) are purposely written to be open to interpretation. Their purposeful vagueness can create big challenges when opinions on implementation disagree.

### Federating Interpretation

One solution that is swiftly becoming the new best practice involves mutually trusting the efforts of others. The concept works much like identity federation, where the authorization to use a service requires trusting the successful authentication from an outside authority.

In the case of compliance management, the authority is an external agency while the service is the verification of compliance. These external agencies have gone through the effort to research and publish their interpretations. These are security best practices like those developed by the Defense Information Systems Agency (DISA), the National Institute of Standards and Technology (NIST), and the Center for Internet Security (CIS). They can be vendor best practices such as VMware's and Microsoft's Hardening Guidelines. They can also be industry and regulatory mandates such as the Sarbanes-Oxley (SOX) Act, Payment Card Industry (PCI) standards, Health Insurance Portability and Accountability Act (HIPAA), and Federal Information Security Management Act (FISMA).

The compliance template in this case is the literal interpretation of the regulation's mandates. When that template is made available by the regulatory agency that owns the mandate, you can reasonably assume that meeting their template's guidelines goes far towards meeting their regulation's guidelines as well.

> **Note**
>
> An important disclaimer: Even with a set of mutually-trusted templates, this activity is still subjective. Your templates might not cover every aspect of your entire solution. They might only relate to those the virtual platform touches. That said, using templates that your auditors already trust goes far towards helping them fulfill their due diligence.

The last point to be made respects the notion that even externally-trusted templates won't meet all your baseline needs. A good solution will provide a mechanism for ingesting existing templates as well as constructing your own for any environment-specific requirements that aren't captured elsewhere.

## Activity Consolidation: The New Best Practice

By this point, you've probably recognized the central theme behind this guide's new best practice assertions: activity consolidation. By following the black box approach and layering each activity's features on top of the other, a virtual environment gains a single pane of glass for managing every behavior. With a well-constructed metrics engine, performance and capacity can be managed via the same platform that handles configurations and compliance.

The next and final chapter concludes this conversation by adding one more major activity to the model: *workload automation*. In Chapter 4, you'll learn how the bundling of configurations into runbooks and policies goes far towards automating large-scale and complex activities. As before, all these best practices fit together to create a cohesive management experience.