



Introduction

The complexity of a typical computing site has increased immensely over the past 10 years, not only in scale but also in the range of services and the intricacy of its interconnections. This has led to systems which can no longer be configured reliably using a purely manual process. For example:

- If a Web server fails, what needs to be done to enable a replacement? This is likely to require a complex chain of reconfigurations, involving firewalls, DNS servers, database servers, and backups. This example is discussed in more detail in section 1.4.
- Can we be sure that the appropriate configuration files on every machine are always set in such a way as to implement the desired security policy? This policy does not just involve individual machines, but also the trust relationships between them.
- Can we be sure that different people, responsible for different aspects of a site, will not change configurations in conflicting ways?
- Can we be sure that a complex service is not configured in such a way that there is a “single point of failure” which has not been identified?

In addition to this increase in complexity, new requirements are presenting new challenges:

- Higher expectations of service reliability call for systems that are able to reconfigure, without manual intervention, to cope with failures of individual machines or components (*autonomics*).
- More complex relationships between machines on an individual site, and between remote sites, imply a more collaborative approach to the development of overall configurations. These are no longer under the complete control of a single individual or small group. This leads to complex problems of security and conflict resolution (*federation*).

Without automated solutions to problems such as those listed above, large computing sites will become increasingly unreliable and unmanageable. Solving these problems is the domain of system configuration.

The State of the Art

An awareness of the need for automated system configuration has been gradually growing within the system administration community itself, and this has spawned a great deal of discussion and a plethora of “homegrown” tools. Many of these can be found in

2 / Introduction

the proceedings of the LISA conferences [9]. However, the vast majority of these tools address only the lowest levels of the problem, and system configuration continues to be a major source of failures (see [60]) and to consume a disproportionate amount of highly skilled manual effort.

Senior system administrators and consultants looking for technologies and approaches to site configuration are faced with a difficult task; there is no clear process for even identifying and evaluating a potential selection of tools, and there will rarely be a single obvious candidate. Most sites end up with a number of imported tools and a large amount of locally developed “glue.” Some sites will decide to develop significant tools of their own, but it is easy to underestimate the resources that this requires, in terms both of skills and experience and of the sheer effort of maintaining critical code which has close dependencies on many rapidly evolving applications: LCFG (see section 5.2), for example, has probably consumed about 10 person-years of direct development effort, supported by twice that effort in configuration-related research. Significant ongoing effort is required to track new operating system and application releases.

So far, tool development appears to have failed to attract a coordinated implementation effort from the open source community. It is also hampered by a lack of appropriate standards, which prevents interoperability and effective tool-sharing. Vendor-supplied tools are frequently used for specific tasks, but they, too, fail to address the overall problem adequately. There is no single, obvious reason for this lack of progress; certainly, the issue is a large one whose solution is well beyond the resources of a system administrator developing code part-time. However, the subject also includes many hard problems whose solution probably involves attracting more interest from computer scientists and theoreticians.

Tools designed to address the real problems of system configuration also face an acceptance challenge from the system administration community; such tools are likely to hide the low-level details of the familiar configuration files and to work with higher-level concepts. This represents a paradigm shift for the average system administrator, comparable to the assembler code programmer who must learn to write distributed applications in Java without worrying about which registers hold which variables! More predictable and trustworthy tools, with clear semantics, are necessary to earn this acceptance.

About This Booklet

This is not a “cookbook”; it does not include detailed descriptions of using specific tools to solve configuration problems, and there is no attempt to provide a comprehensive tool survey, since tools can change rapidly (for some recent evaluations see, e.g., [29], [21]). But neither is this booklet about the developing theory of system configuration (although such theories are summarized in chapter 6). Rather, the booklet is aimed squarely at the working system administrator; it aims to explain clearly the vari-

ous facets of the system configuration problem and to describe how these relate to current tools and future research.

Understanding underlying configuration principles will help system administrators to use “best practice” in applying current configuration tools and procedures; these are often very flexible, and it is all too easy to negate the advantages presented by a perfectly good tool. Descriptions of the various facets of the configuration problem should also help provide system administrators with criteria to evaluate potential tools, since most sysadmins will be able to relate these to concrete examples at their own sites.

In the longer term, a better understanding of the fundamental principles should encourage theorists to address underlying configuration problems and tool developers to incorporate this theory into their products. Ultimately, this will yield new tools and enable system administrators to have more confidence in the ability of these tools to automatically manage the configuration of their sites.

Finally, it should be noted that there is not yet consensus on all aspects of the subject, and some topics may reflect the author’s personal bias. For instance, the LCFG tool is often used in the examples that follow, because it is broad enough to illustrate many of the important issues in a consistent way. However, it is hoped that the presentation is sufficiently logical and objective to allow readers to draw their own conclusions.

The LISA configuration workshops and the lssconf mailing list¹ provide some forums for the topics discussed in this book.

A Word About Operating Systems

This booklet deals mostly with principles that apply equally to any operating system. However, the terminology and most examples are taken from UNIX, and it is expected that this will be the natural background of most readers. UNIX probably has the widest range of configuration tools and practices; at the bottom end, raw UNIX systems provide no clear standard and no comprehensive support for system configuration—many sites without the necessary in-house skills still have very primitive configuration management. At the opposite extreme, the most complex sites and the most sophisticated configuration tools are probably UNIX-based.

Operating systems such as Microsoft Windows have all the same problems and requirements for configuration management, but there is less variety in the tools and approaches; vendor-supplied tools obviously tend to predominate, and these are certainly comparable to the middle ground of the UNIX range. The standard Microsoft tools, for example, are mentioned where appropriate.

The Structure of the Book

Chapter 1 is perhaps the most important chapter of this book; it defines the system configuration problem and describes its various aspects in some detail, without consid-

1. <http://homepages.inf.ed.ac.uk/group/lssconf/>.

4 / Introduction

ering specific solutions. This section is recommended as a prerequisite for understanding the material in the following sections.

Chapter 2 outlines the range of approaches to automating configuration solutions, starting with a largely manual approach and moving towards fully autonomic site management.

Chapter 3 describes some of the ways in which various tools approach different aspects of the configuration problem.

Chapter 4 covers tools and techniques which are specifically used for distributing files and managing software packages. Although this is only one (small) aspect of the overall configuration problem, it is very common to see such tools extended and used to provide primitive configuration management.

Chapter 5 describes a number of tools in more detail. These are chosen to demonstrate the range of different approaches rather than to provide a basis for selecting or using a particular tool.

Chapter 6 briefly describes some of the current issues in configuration theory and research. Less theoretically inclined readers may want to skip this section.

Most chapters end with a list of key points which summarize the important issues. The collection of these key points should provide a good overview of the subject.

Terms in italics are explained in the Glossary.