AIS: A Fast, Disk Space Efficient “Adaptable Installation System” Supporting Multitudes of Diverse Software Configurations

Sergei Mikhailov and Jonathan Stanton – George Washington University

ABSTRACT

Efficiently installing and configuring large sets of computer systems is an important concern for system and cluster administrators. Current solutions usually follow one of the two approaches: an image-based install or a metadata-based custom install. Both approaches limit the opportunities for optimizing the installation time by coupling the system specification with the installation technique and ignoring the relationships between configurations over time (as they evolve with patches and new packages).

The Adaptable Installation System (AIS) is a new model and implementation that attempts to address these shortcomings by taking a hybrid approach to client system installation. As in the metadata-based approach, it uses descriptors to express what the final system should look like in terms of composition and configuration. At the same time, it uses imaging for part of the client re-installation to achieve speed. In this paper we present the design and implementation of AIS along with details on the algorithm that builds images and performance results of running the prototype system on a set of RedHat based machines.

Introduction and Motivation

Efficiently installing and configuring large sets of computer systems is an important concern for system and cluster administrators. Numerous programs that facilitate automated and unattended installations have been created. Generally they follow one of the two approaches: an image-based install or a metadata-based custom install. Both techniques are widely used and effective in certain scenarios. However, they both have two main disadvantages:

1. They couple the specification of the system (golden-server, Kickstart or other configuration file) with the installation technique (over-network disk-image writing, package or OS install tool). This limits opportunities for algorithmic optimization and prevents using the best specification with the best technique in all cases.
2. They both assume a static model of system installation, meaning they assume installation is a singular, fairly heavy-weight event. Although some adaptation to more dynamic environments

Installation Factors				
	Speed of Installation	Number of Configurations	Available Server Storage	Optimal Installation Approach
1	Not a factor	Small	Not a factor	Custom, Image
2	Not a factor	Small	Factor	Custom
3	Not a factor	Large	Not a factor	Custom
4	Not a factor	Large	Factor	Custom
5	Factor	Small	Not a factor	Image
6	Factor	Small	Factor	AIS
7	Factor	Large	Not a factor	AIS
8	Factor	Large	Factor	AIS

Table 1: Factors that affect installation scenarios.

such as Clusters-on-Demand [18] is possible, they do not capture the relationship between system configurations over time.

Table 1 compares the applicability of the two approaches with respect to three typical considerations in large installation/cluster management. The right-most column indicates the preferred installation approach given the values of the three considerations that are shown in the columns to the left. When the speed of client re-installation is not critical, custom install is a better option because of its ability to scale to a large number of system configurations and modest storage requirements. A new system configuration can be supported by creating a corresponding metadata descriptor, i.e., a Kickstart file. Imaging remains a preferred approach when the speed of client re-installation is critical. However, as the number of configurations increase dramatically, it becomes time consuming to manage and requires large amounts of storage on the server.

Both imaging and custom installation approaches fall short of simultaneously supporting fast client re-installations, large number of system configurations and moderate storage usage. The Adaptable Installation System (AIS) is a new model and implementation that attempts to address these shortcomings by taking a hybrid approach to client system installation. As in the metadata-based approach, it uses descriptors to express what the final system should look like in terms of composition and configuration. At the same time, it uses imaging for part of the client re-installation to achieve speed.

The need to support a large number of system configurations while simultaneously allowing fast client re-installations is motivated by the development of ‘‘utility’’ computing and ‘‘clustering on demand’’ services such as the Oceano [4, 10] and Cluster-On-Demand [18] projects. These projects’ business model is that an organization manages computer clusters on behalf of its clients (whether internal or external) and guarantees an agreed upon level of performance. It maintains a large pool of client machines and dynamically reallocates them from one virtual cluster to another as necessary. A second motivation is the continued need to patch or upgrade, and occasionally

System Configuration
A description of the composition of computer system software in terms of the packages installed on this system. In the case of an RPM-based distribution of Linux, a list of all RPMs installed on this system.

System Configuration Descriptor (SCD)
An XML file that describes system configuration at a class level. Configuration is described in terms of Installation Base and additional packages that make up the system.

Installation Base
A minimal, working system configuration. Typically consists of a kernel, C libraries, and a small set of common Unix programs.

Host Configuration
System Configuration, plus any modifications to operating system configuration files needed to achieve the desired host state. Host configuration can be achieved manually, by copying host-specific files from the AIS server, or by running tools such as Cfengine [6].

Host Configuration Descriptor (HCD)
An XML file that specifies host configuration details. It references the SCD on which this host configuration is based.

AIS Server
The machine executing the AIS server code that calculates the contents of each image and builds the cached image files. This machine also hosts the image cache files and package files required for client installation.

Table 2: Definitions of system components.

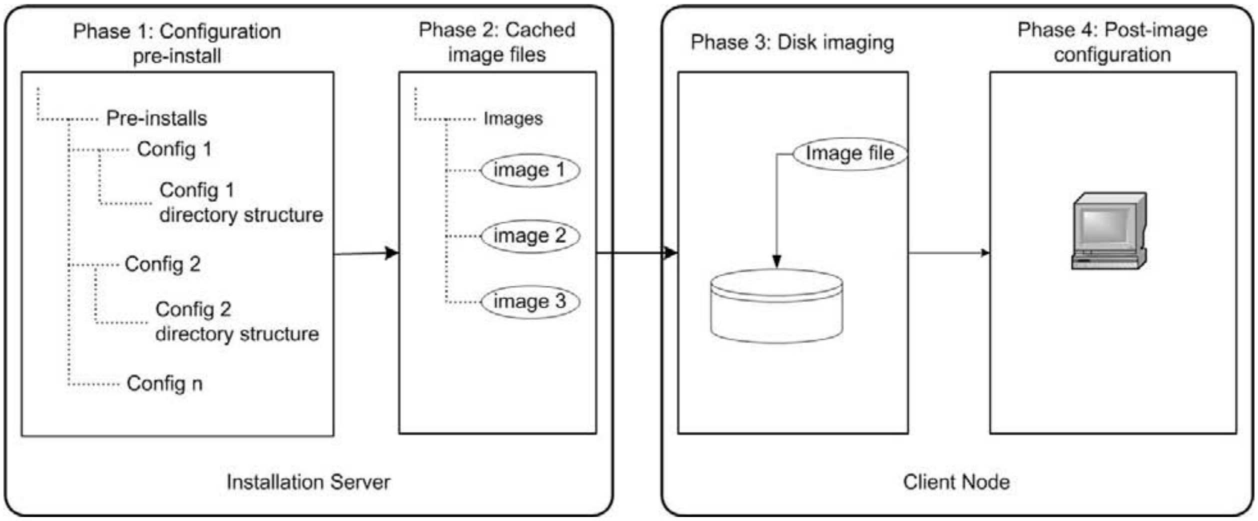

Phase 1: Configuration pre-install
Pre-installs
Config 1
Config 1 directory structure
Config 2
Config 2 directory structure
Config n
Phase 2: Cached image files
Images
image 1
image 2
image 3
Installation Server
Phase 3: Disk imaging
Image file
Phase 4: Post-image configuration
Client Node

Figure 1: Stages in installation process.

downgrade, production servers. This cycle causes a continually increasing set of configurations that differ only by a few package versions. To maintain them as online images would require continually growing storage and management complexity.

```
<ImageConfiguration>
  <InstallBase>fc1</InstallBase>
  <OsBaseAlterations/>
  <Installations>
    <package type="rpm">mozilla-1.4.1-17</package>
    <package type="rpm">other package</package>
        . . .  . . .
  </Installations>
</ImageConfiguration>
```

Figure 2: Sample system configuration descriptor.

```
<HostConfiguration>
  <ScdId>Configuration 1</ScdId>
  <ConfigFiles>
    <file perms="644">/etc/hosts</file>
    <file perms="644">/other/file</file>
    . . .  . . .
  </ConfigFiles>
</HostConfiguration>
```

Figure 3: Sample host configuration file.

AIS Operation

To better understand how AIS operates, it is helpful to know the phases involved in client installation. They are shown in Figure 1. The terms that are used in the section are defined in Table 2.

As seen is Figure 1, AIS operates on two fronts: the installation server and the client machine being installed. On the AIS Server, AIS maintains a repository of System and Host Configuration Descriptors and a repository of installation packages. With this, AIS can reconstruct on disk any system configuration using its descriptors. This is shown as Phase 1 of the overall process in Figure 1. Note that AIS does not store or recover application data files except for host configuration contained in the HCD. The data files can be maintained on network file servers or separate data disks. One such method is described in [17]. Furthermore, AIS creates and maintains a repository of images (Phase 2), which it uses as the first step in reinstalling a client machine. This step is depicted as Phase 3 of the overall process.

The key to AIS operation is that it does not necessarily store images for all system configurations. Furthermore, an image does not have to correspond to any specific configuration. Instead, the content of the image cache is determined by an algorithm and the same image can be used for initiating a re-installation of more than one system configuration. The algorithm aims to minimize the client installation time. In the decision making process it considers all system configurations that might need to be installed and the disk space available for storing image files. Since the image does not necessarily correspond to an exact system configuration, AIS performs the necessary post imaging steps to arrive at the exact system configuration. This step is depicted as Phase 4 in Figure 1.

On the client, AIS is a script that runs as soon as the machine boots. The script uses a third party imaging tool, Frisbee, to retrieve the image from the AIS Server and write it to disk. After this, the script performs additional package installations, as necessary, to achieve the desired system configuration. Finally it performs any host specific configuration.

The following sections provide additional details about AIS operations on the Installation Server and the client machines.

AIS Operations on the AIS Server

Overview

The content of the image cache depends on all the System and Host Configuration Descriptors AIS is managing. The objective is to maintain such a combination of images that it minimizes the time it takes to install the next client machine, while meeting the disk space constraint. In the context of a large number of system configurations it is not feasible to store an image for every configuration. Thus, the images that are maintained do not necessarily correspond to a particular system configuration. Instead, an image can be

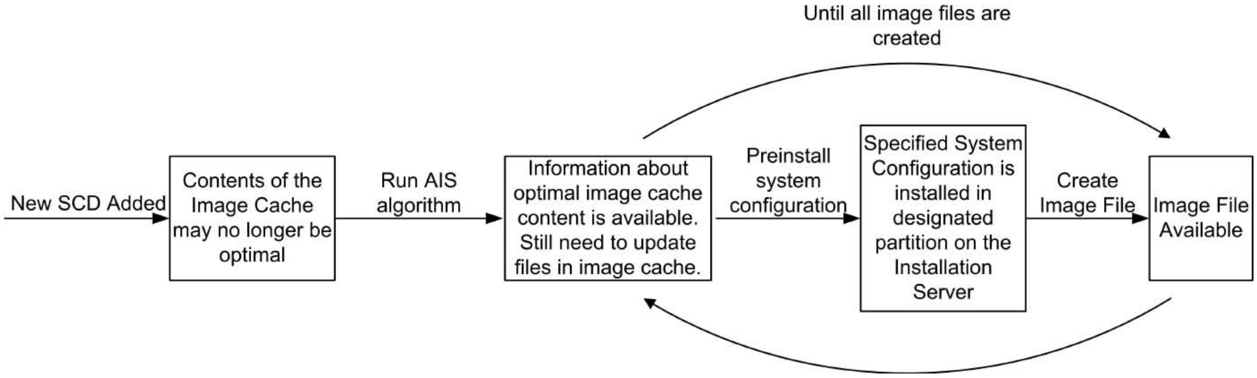

Until all image files are created
New SCD Added
Contents of the Image Cache may no longer be optimal
Run AIS algorithm
Information about optimal image cache content is available. Still need to update files in image cache.
Preinstall system configuration
Specified System Configuration is installed in designated partition on the Installation Server
Create Image File
Image File Available

Figure 4: Adding a new system configuration descriptor requires image cache to be updated.

a mixture of common components from various configurations. This explains why in the post-image configuration phase AIS may need to install additional packages – to arrive at the desired system configuration from a generalized image configuration.

System and Host Configuration Descriptors

For AIS to manage a system configuration, a corresponding System Configuration Descriptor (SCD) must be created. A sample SCD is shown in Figure 2. A SCD specifies configuration of a system only in terms of the installed packages. Each SCD references an InstallBase, which is a minimal working system. The packages that make up the Installation Base plus the packages listed in the Installations section of an SCD constitute all the packages for a given system configuration. A system configuration specified by a SCD represents a class of systems. Any host specific configuration is contained in a Host Configuration Descriptor, HCD. A sample HCD is shown in Figure 3.

Introducing New System Configuration

Figure 4 shows the steps that are taken by AIS when a new System Configuration Descriptor is introduced. The cache content may no longer be optimal since when it was originally calculated the just-added SCD was not considered. This necessitates AIS to rerun the algorithm to determine the new content of the image cache. After the new content of the cache is determined, AIS needs to create the images. This is done by first installing the system configuration in a designated partition on the AIS Server and then running an image creation program. The last two steps are repeated for every image file.

The process of refreshing the image cache takes a considerable amount of time. After AIS determines how many configurations should be in the cache and what should be their content, each of those systems needs to be installed on a dedicated hard disk on the AIS Server in order to create an image. As such, refreshing of the image cache is meant to be an off-line operation scheduled during times when it can complete before starting to server client re-installation requests.

AIS Operations on the Client Machine

Image and Host Customization

On the client, AIS is a script that runs after the client node is booted. The steps involved in the client installation are shown in Figure 5.

The script contacts the AIS Server via HTTP with the client's MAC address. This allows AIS to uniquely identify the client and determine which system configuration should be installed and which image files should be used. The Frisbee server is started that serves the image file and AIS sends the client the command that it should execute to retrieve the image. After the image is written to disk, the client contacts the AIS Server to retrieve the list of any addition packages that might need to be installed to achieve the desired system configuration. After any additional packages are installed, the client script performs host specific configuration by copying OS configuration files from the AIS Server.

Algorithm Details

We will present a specific algorithm for constructing a set of *good* system images to cache. Many other algorithms also exist and we are continuing to investigate them. Each potential algorithm has a different set of tradeoffs. The one we present here is a fairly simple merge-based algorithm that maintains the invariant that all proposed images and targets are complete sets of packages with no missing dependencies. This algorithm produces results that are noticeably better then pure imaging or pure metadata based approaches, however, they are theoretically non-optimal (in the sense of minimizing the time to install any potential requested configuration).

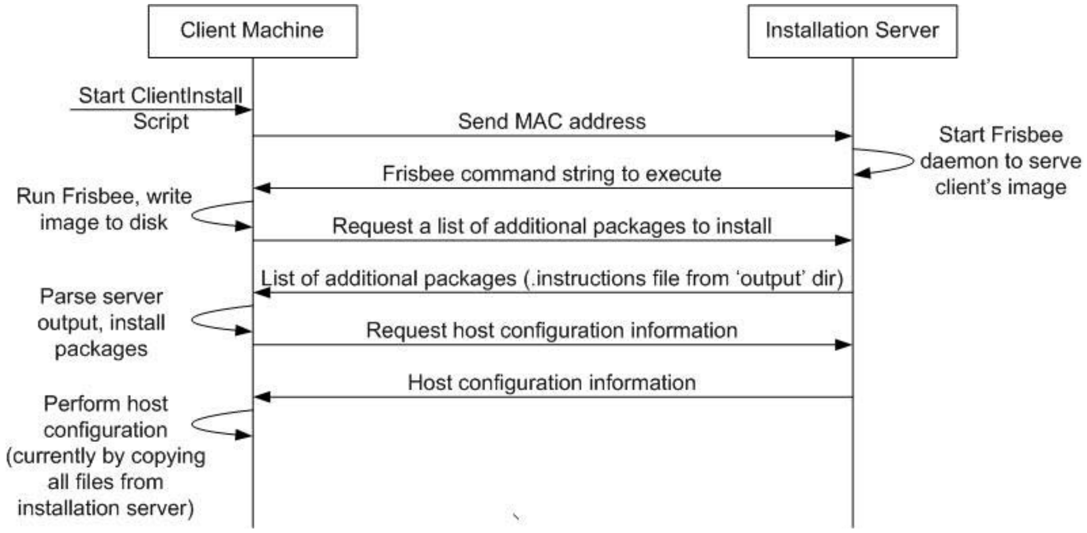

Client Machine
Installation Server
Start ClientInstall Script
Send MAC address
Start Frisbee daemon to serve client's image
Frisbee command string to execute
Run Frisbee, write image to disk
Request a list of additional packages to install
List of additional packages (.instructions file from 'output' dir)
Parse server output, install packages
Request host configuration information
Host configuration information
Perform host configuration (currently by copying all files from installation server)

Figure 5: Client installation process.

The results of experiments conducted using the merge-based algorithm are further discussed later.

Partitioning Available Space Among Installation Bases

The algorithm operates on two levels. First, it determines how much disk space, out of all available for image caching, to allocate to each installation base. Since images are not compatible across installation bases, it is best that at least one image per installation base is available. The amount of space allocated to each installation base is proportional to the installation size of all systems configurations that rely on this base. For example, if the installation size of systems that rely on a given base is thirty percent of the total installation size of all systems AIS manages, then thirty percent of available disk space will be allocated for caching images of this base.

Determining the Composition of Cached Amalgam Images

Merge-based Algorithm

The second step of the algorithm determines the makeup and number of images to create within the space allocated for each installation base. Ideally, there would be enough space to store an image for every system configuration. However, in the context of a large number of configurations, this may not be feasible.

This step continuously merges the two selected configurations into one, until the remaining configurations fit in the allotted space. The two selected configurations are those that share a set of packages which also has the largest installation size. The resultant configuration after the merge is that common subset of packages.

Once this decision step completes, it produces three pieces of output:

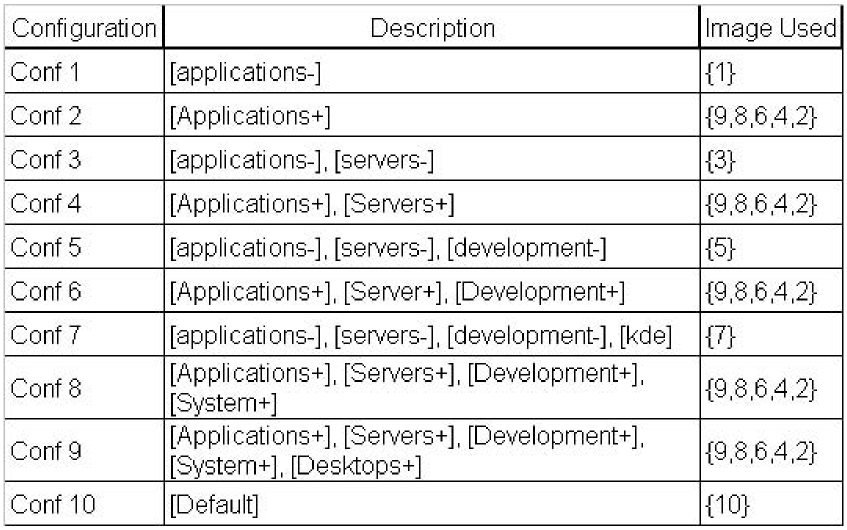
1. A mapping of each system configuration to the image file that will be used in the imaging phase of client installation;
2. An SCD that describes the content of each of the images; and
3. A file that lists any additional packages that need to be installed for each system configuration.

If a particular system configuration was merged in, then there will be no image that represents it exactly. In this case the file will list the missing packages. For those systems that were not merged-in, the file will not list any packages.

Experimental Results

AIS is currently implemented in Python and works with RPM based Linux distributions. The existing Frisbee tools are used to generate compressed disk images and reliably multicast them to the client machines. The image times reported use Frisbee directly. Tables 4, 5, and 6 demonstrate the results of running AIS against ten system configurations. While this is not the large number of configurations AIS is intended to manage, the numbers provide strong evidence of the advantage AIS provides over other approaches. These tests were conducted on a set of X86 PC’s with Pentium3 processors and a 100 Mbps Ethernet network. Although not the newest generation of PC’s, we believe these are representative of actually deployed hardware.

The configurations used in this example are all based on the Fedora Core 1 Linux distribution. They vary in size and composition. The second column of Table 3 describes the content of each configuration in terms of how to achieve it using the redhat-config-packages program. redhat-config-packages, per default comps.xml file in RedHat/Fedora, divides all packages in five global groups: Desktops, Applications, Servers, Development and System. Each package group is further subdivided into subgroups. Each packages can be either mandatory, default or an optional member of a given subgroup. In Table 3 the package group name is



Configuration	Description	Image Used
Conf 1	[applications-]	{1}
Conf 2	[Applications+]	{9,8,6,4,2}
Conf 3	[applications-], [servers-]	{3}
Conf 4	[Applications+], [Servers+]	{9,8,6,4,2}
Conf 5	[applications-], [servers-], [development-]	{5}
Conf 6	[Applications+], [Server+], [Development+]	{9,8,6,4,2}
Conf 7	[applications-], [servers-], [development-], [kde]	{7}
Conf 8	[Applications+], [Servers+], [Development+], [System+]	{9,8,6,4,2}
Conf 9	[Applications+], [Servers+], [Development+], [System+], [Desktops+]	{9,8,6,4,2}
Conf 10	[Default]	{10}

Table 3: Configuration to image file mapping.

capitalized and followed by a plus sign only if every package of every subgroup is installed. The package group name is not capitalized and is followed by a minus sign if only mandatory and default packages of every subgroup are installed. Thus, Conf 5, for example, consists of only mandatory and default packages belonging to the subgroups in Applications, Servers, and Development package groups.

Table 3 shows that after this particular run of AIS, due to the disc constraints provided, the image cache consists of only six image files, as indicated by six unique entries in the rightmost column. For every system configuration in the leftmost column of Table 3 an image file that will be used during client reinstallation is shown in the corresponding row of the rightmost column. For configurations 9, 8, 6, 4, and 2 there is no image in the cache that represents their exact configurations. The amalgam image, depicted as 9, 8, 6, 4, 2, will be used in the imaging phase of client installation for configurations 9, 8, 6, 4, and 2. For other system configurations, there is an image file that represents that configuration.

Table 4 shows the time it took to install each of the ten configurations using the three approaches. RedHat Kickstart was used for custom installation and Frisbee was used to derive the timing results for imaging approach. For those configurations that had a corresponding image in the cache under AIS, the time is nearly identical to Imaging. The one second delay is roughly how long it took to complete the host configuration, as can be seen in Table 6. Those configurations that used the amalgam image, took slightly longer than Imaging because of the additional packages that needed to be installed and host configuration. However, the Imaging approach required almost twice the disk space for storing image files as AIS, as seen in Table 5.

To further demonstrate interesting implications of AIS approach in maintaining a system configurations infrastructure, we compare the behavior of Imaging and AIS under evolving configurations in Figure 6. This can occur when the administrator wants to keep all versions of the same configuration as it evolves (patches applied, new packages installed) throughout its lifetime. In typical imaging approach, an image for every version of the configuration must be maintained, consuming much more storage than the size of the changes.

In Figure 6 AIS maintains only one image, with which it can achieve any version of configuration. As new versions are introduced, they may get incrementally longer to install, because they deviate more from the imaged configuration, as shown in Scenario 1. Alternatively, AIS can update the image so that installation of most recent versions takes the least time, as shown in Scenario 2.

Related Work

A number of tools allow for automated and unattended installations. Tools that allow metadata-based installations, among others, include RedHat Kickstart [15], FAI [7, 8], and LUI [13]. They can support large number of configurations, but take a relatively long time to install and couple the system specification and

	Conf 1	Conf 2	Conf 3	Conf 4	Conf 5	Conf 6	Conf 7	Conf 8	Conf 9	Conf 10
Kickstart	0:37:03	1:06:14	0:52:52	1:13:34	1:03:32	1:14:05	0:40:05	0:54:56	1:14:43	0:20:05
Imaging	0:03:22	0:05:13	0:04:46	0:05:19	0:04:25	0:05:20	0:04:44	0:05:18	0:05:23	0:02:36
AIS	0:03:23	0:05:14	0:04:47	0:07:49	0:04:26	0:08:44	0:04:45	0:09:25	0:09:51	0:02:37

Table 4: Timing of custom install, pure imaging and AIS approaches.

	Installation Pkg. Repository	Image Repository
Kickstart	1.8 GB	---
Imaging	0 or 1.8 GB	9.7 GB
AIS	1.8 GB	5.1 GB

Table 5: Storage requirements for three approaches.

	Conf 1	Conf 2	Conf 3	Conf 4	Conf 5	Conf 6	Conf 7	Conf 8	Conf 9	Conf 10
Imaging	0:03:22	0:05:13	0:04:46	0:05:14	0:04:25	0:05:21	0:04:44	0:05:21	0:05:22	0:02:36
Additional Installations	0:00:00	0:00:00	0:00:00	0:02:34	0:00:00	0:03:22	0:00:00	0:04:03	0:04:28	0:00:00
Host Configuration	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01
Total	0:03:23	0:05:14	0:04:47	0:07:49	0:04:26	0:08:44	0:04:45	0:09:25	0:09:51	0:02:37

Table 6: Detailed timing for AIS.

the installation technique. SystemImager [7, 9] and Frisbee [3] are tools that follow the imaging approach. As such, they install quickly, but lack flexibility in configuration and require disk space that increases with the number of system configuration variations.

A number of more sophisticated host configuration management tools exist which attempt to solve the larger problem of creating and maintaining software configurations across a large number of servers and desktops. These tools are discussed in [2, 1, 14, 5, 16, 12, 11] and are complementary to the AIS approach, as AIS only attempts to solve the problem of fast, efficient installation and could easily be matched with a SCM system for ongoing management. Currently the host configuration phase of the client installation process is achieved by copying over the necessary configuration files from the Installation Server. While this is sufficient for a prototype implementation, the overall value of the system would increase if one of the SCM systems was incorporated as one of the phases of the overall AIS process.

Availability

The AIS system as well as additional documentation is available for download under an open-source license at the web site http://www.ensl.cs.gwu.edu/projects/ais/ .

Future Work

AIS is a prototype implementation of a hybrid approach to installing software on a large number of client machines. The current implementation is limited to RPM. However, conceptually the ideas can carry over to other package management systems, as well as to using multiple package management systems on the same machine, if necessary. What is important is the

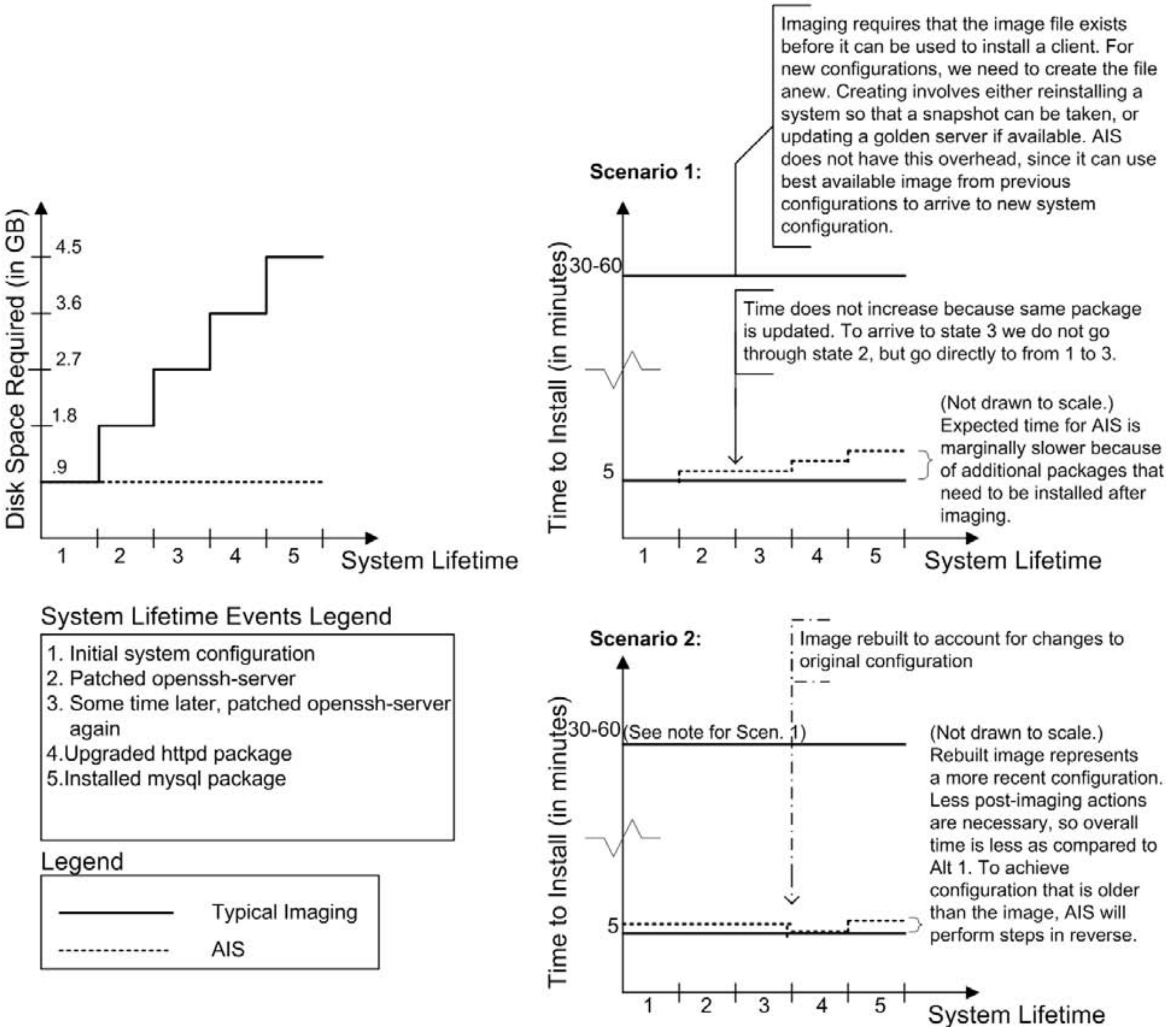


Figure 6: Comparison of disk and installation speed for Imaging and AIS solutions when the system configuration evolves.

ability to determine the makeup of the system's configuration in terms of installed packages, be it from one or more package management systems' databases.

Ability to query and list the content of the system is a significant advantage provided by package management systems; something that is not available by default in systems where a lot of software components are installed from source, "tar.gz" files or similar mechanisms. Furthermore, source installations do not provide dependency information and thus are not naturally suited for AIS-type systems, which construct system images from metadata descriptors.

Conclusion

AIS combines the features of custom install tools and imaging tools and provides a beneficial balance of ease of maintenance, scalability to large number of system configurations, and speed of client re-installation. This paper demonstrates how a hybrid, caching installation system can achieve both fast installation and low disk space use. These ideas can apply to imaging and installation tools other than Frisbee and RedHat Kickstart.

About the Authors

Sergei Mikhailov recently earned a Master's degree in Computer Science from the George Washington University. He currently works for a division of the University's Information Systems and Services department where he splits his time between system administration and web applications development. Contact him electronically at sergei.mikhailov@alumni.gwu.edu .

Dr. Jonathan Stanton received his M.S.E. and Ph.D. degrees in Computer Science from The Johns Hopkins University in 1998 and 2002. He is currently an Assistant Professor in the Computer Science department of the George Washington University. He also co-founded Spread Concepts LLC which provides software tools and designs for high performance, highly available distributed systems. His research interests include distributed systems, secure distributed messaging, network protocols, and middleware support for clustered systems. He is a member of the ACM and the IEEE Computer Society. Reach him electronically at jstanton@gwu.edu .

References

[1] Anderson, Paul, Patrick Goldsack, and Jim Paterson, "SmartFrog meets LCFG: Autonomous Reconfiguration with Central Policy Control," *Proceedings of LISA XVII*, pp. 213-222, 2003.

[2] Anderson, Paul, "Towards a High-level Machine Configuration System," *Proceedings of LISA VIII*, pp. 19-26, 1994.

[3] Hibler, Mark, Leigh Stoller, Jay Lepreau, Robert Ricci, Chad Barb, "Fast, Scalable Disk Imaging with Frisbee," *USENIX Annual Technical Conference Proceedings*, pp. 283-296, 2003.

[4] Appleby, K., "Oceano – SLA Based Management of a Computing Utility," *Proceedings of the Seventh IFIP/IEEE International Symposium on Integrated Network Management*, 2001.

[5] Anderson, Paul and Alistair Scobie, "Large Scale Linux Configuration with LCFG," *Proceedings of the Atlanta Linux Showcase*, pages 363-372, http://www.lcfg.org/doc/ALS2000.pdf , 2000.

[6] Burgess, Mark, "Cfengine: A Site Configuration Engine," *USENIX Computing Systems*, Vol. 8, Num. 3, http://www.cfengine.org/ , 1995.

[7] *Enterprise Infrastructure Workshop Notes*, LISA 03 Workshop, http://www.infrastructures.org/workshop/ , 2003.

[8] *FAI: Fully Automatic Installation for Debian GNU/Linux*, http://www.informatik.uni-koeln.de/fai/ .

[9] Finley, Brian, "VA SystemImager," *USENIX Annual Linux Showcase and Conference Proceedings*, pp. 181-186, 2000.

[10] IBM, *The Oceano Project*, http://www.research.ibm.com/oceanoproject/ .

[11] *Isconf: The Infrastructure Configuration Engine*, http://www.isconf.org/ .

[12] Kanies, Luke, "ISconf: Theory, Practice and Beyond," *Proceedings of LISA XVII*, pages 115-123, 2003.

[13] *Lui Project*, http://www-124.ibm.com/developerworks/projects/lui/ .

[14] Oetiker, Tobias, "TemplateTree II: The Post-Installation Setup Tool," *Proceedings of LISA XV*, pages 170-186, 2001.

[15] RedHat, *Kickstart installations*, http://www.redhat.com/docs/manuals/linux/RHL-9-Manual/custom-guide/ch-kickstart2.html .

[16] Roth, Mark, "Preventing Wheel Reinvention: The psgconf System Configuration Framework," *Proceedings of LISA XV*, pages 205-211, 2003.

[17] Sapuntzakis, Constantine, David Brumley, Ramesh Chandra, Nickolai Zeldovich, Jim Chow, Monica S. Lam, and Mendel Rosenblum, "Virtual Appliances for Deploying and Maintaining Software," *Proceedings of LISA XVII*, pages 181-194, 2003.

[18] Duke University, *COD: Cluster-on-demand*, http://issg.cs.duke.edu/cod/ .