Adobe® LiveCycle® ES2 Infrastructure Selection

Analyzing architectures and choosing an infrastructure topology

Introduction

This technical guide provides insight and recommendations around the infrastructure for deployment of Adobe LiveCycle ES2. It also provides assistance in the preparation of an environment prior to installation. Details provided are specifically targeted at deployments of ES2 or later. The Appendix provided contains valuable documentation references that should be downloaded and reviewed prior to deployment.

Three main topics are covered:

• Deployment architectures of Adobe LiveCycle ES2 - An overview of LiveCycle ES2 from a deployment professional’s perspective
• Choosing an infrastructure and deployment topology - Infrastructure and deployment options
• Considerations for deployment - Outlines of practical aspects of a deployment

This technical guide focuses on the Foundation-based LiveCycle ES2 modules because of their complex dependencies and requirements. It covers all the modules except Adobe LiveCycle Data Services ES2, since it has greater infrastructure support.

Deployment architecture of Adobe LiveCycle ES2

It is helpful to understand the high-level deployment architecture of LiveCycle that is tailored for people planning a deployment. This model will help you better understand the dependencies, communications paths, and prerequisites that will have to be present (see Figure 1).

LiveCycle software is based on J2EE. It requires features that are only present in a complete J2EE implementation such as Enterprise JavaBeans (EJB). Prior versions of LiveCycle required JMS, ES2 no longer requires JMS. LiveCycle’s use of J2EE is sophisticated enough that it is not portable between different versions and vendor implementations of J2EE. This is not because vendor-specific extensions are used per se, but because minor idiosyncrasies can be significant. LiveCycle ES2 platform support requires specific versions of vendor-supported J2EE application servers.

One aspect of being J2EE-based is the need for a relational database. The ES2 table space contains data that ES2 manages for both product and end-user data persistence. The database can grow substantially, especially when Process Management is used to manage long-lived processes. It is important to monitor the database size and plan accordingly. Depending on the use of ES2, applications may not accumulate data and may not be particularly sensitive to database performance. This may be the case if long-lived processes are not being used or if integration with LDAP directories is not being performed. If long-lived processes are being used, database performance becomes important.
ES2 can generate and manage large amounts of data. Much of which is in the form of documents such as PDF or XML. **These documents are placed into a directory known as the Global Document Storage (GDS). This directory is a critical part of the deployment.** It is often overlooked but is just as critical as the database to the deployment. The GDS directory is a single instance in a cluster. It contains ES2 services (DSC files), process document variables, and application resources such as templates, fragments, schemas, and images. The GDS directory can commonly grow to more than 10GB, so it is important to place it in an adequate file system. Access times will have a significant impact on the overall performance. Directories are also required for fonts and temporary processing.

![Deployment architecture of LiveCycle ES2](image)

**Figure 1: Deployment architecture of LiveCycle ES2**

ES2 modules may use native processes. LiveCycle Foundation always provides support for native process management, but this is only used by a subset of modules (Adobe LiveCycle Forms ES2, LiveCycle Output ES2, and LiveCycle PDF Generator ES2). Native processes in ES2 are extremely well managed and do not undermine the performance or reliability of the overall application. When native processes are used, they are executed in a process instance different from the application server process and their complete life cycle is managed by LiveCycle. Native processes only execute and communicate with the local installation of ES2. Communication between the LiveCycle components running inside the J2EE application server and the native processes is accomplished via **CORBA IIOP**. There are two kinds of native processes in ES2. Pure native components that appear in the process list as executables, and external Java processes that appear as Java processes in the process list.

Optional components that LiveCycle may interoperate with will need to be considered as part of a deployment plan.

**Authentication to ES2 is not managed by J2EE.** ES2 manages its own authentication and access control lists that are richer and more business application centric than the standard J2EE implementation. By default, this is a local security repository known in ES2 terms as a “local domain”. In most enterprise deployments ES2 will be integrated with a corporate LDAP directory known in ES2 terms as an “enterprise domain”. **Authentication is a LiveCycle configuration and not a configuration of the Java Authentication and Authorization Service (JAAS) application server.** It is the LiveCycle administrator who implements security.
ES2 can integrate with existing enterprise content management systems or can be deployed with its own embedded content management system, Adobe LiveCycle Content Services ES2. By default, ES2 has an internal repository for LiveCycle application resources. This is an inherent capability. Enterprise content management (ECM) integration adds a richer feature set than the internal repository.

Use of a storage area network (SAN) is recommended in cluster configurations. Because ES2 has critical failure and performance constraints related to the GDS directory, it is sometimes important to consider placing the directory on a high-performance, reliable network system such as a SAN.

ES2 can be configured to work with e-mail. This capability is mainly used with LiveCycle Process Management ES2 software.

In deployments of Adobe LiveCycle Digital Signatures ES2 software, there may be the need to have added safeguards around the access to corporate private keys. One way to do this is to use a hardware security module (HSM). An HSM provides a secure key store and efficient signing implementation. In some ways, it can be considered a secure black box that removes the need for software to access the PKI private keys and signing algorithms. Digital signing occurs with a locally managed private key credential stored in the ES2 database, whereas when signing with an HSM, the private key is used only within the HSM hardware for added security. Provisions for certificate revocation list (CRL) or online certificate status protocol (OCSP) might also have to be made to allow ES2 to reach out through a corporate firewall to perform certificate revocation checking.

Choosing an infrastructure and deployment topology

There are two primary variables when choosing the most appropriate way to realize an ES2 installation. Selecting the third-party infrastructure, such as a J2EE application server, database, and operating system is one approach, followed by selection of the right topology for the deployment. Whether it’s a single server, one of the cluster options, or server load balancing. When you consider your organization’s requirements that list of options will become shorter. Each has certain considerations that need to be highlighted.

Selection of a J2EE application server is dependent on what skills and standards exist within an organization. ES2 supports IBM WebSphere, BEA WebLogic, and the JBoss application server. Technically, all of these application servers will be able to deliver the requirements of a LiveCycle deployment, small or large scale. The recommendations in this document provide guidance based on the level of effort and complexity that may be involved.

Selection of an operating system and database is similarly open. ES2 supports:

- IBM AIX® 5.3 and 6.1 on Power-compatible systems,
- Microsoft® Windows Server® 2003 and 2008 on Intel-compatible systems,
- Red Hat® and SUSE™ Linux® on Intel-compatible systems, and
- Sun® Solaris® on Sparc-compatible systems.

In the case of databases, there is support for:

- IBM DB2,
- Microsoft SQL Server,
- MySQL, and
- Oracle.

Note that not all combinations of the third-party infrastructure are supported. The support matrix is sparse, and you have to review the documentation listed in the appendix to ensure that the combination is listed as supported.

The hardware architecture is limited to the most common options. Windows and Linux are supported on Intel and AMD x86 architectures with the option of 64-bit extensions. Solaris is supported on SPARC architectures. AIX is supported on IBM Power systems.
Before going into the high-level topology decision flow, it is important to understand what the topology options are. ES2 has the following topology options:

- A turnkey topology is a very popular single-server deployment. It is appropriate for simple processes, evaluations, development, and user acceptance test (UAT) servers. This is the most common ES2 deployment option. It is offered on Windows architectures and comes pre-configured with JBoss and MySQL, and may also support a pre-existing MS-SQL Server or Oracle database. It can be implemented in a relatively short amount of time.

- A single-server topology deploys ES2 into a single instance of the application server. With a single server topology you must install and configure the application server and database before beginning the deployment. This topology supports a wide variety of application servers, operating systems, and databases. LiveCycle Configuration Manager (LCM) can be used to automatically configure WebSphere and Weblogic or a pre-configured JBoss tree.

- A cluster topology, including vertical, horizontal, and mixed clustering of the application server, is the most advanced deployment option and delivers the richest capabilities. Similar to a single-server deployment, you must install and configure the application server and database before beginning the installation of ES2. LiveCycle Configuration Manager can be used to automatically configure WebSphere and Weblogic or a pre-configured JBoss tree.

- A server load-balancing (SLB) topology is a hybrid implementation, mixing the simplicity of turnkey or single-server with some of the richness of a cluster topology. Since SLB is not a true cluster topology, it is simpler to deploy. This option uses independent servers in conjunction with an IP load balancer to gain throughput and availability. You may use all ES2 modules, except LiveCycle Process Management ES2 and LiveCycle Rights Management ES2, with the SLB topology. A downside of this topology is the fragmentation of management. There is no longer a centralized management capability; all management needs to be repeated on each participating server.

Note that LiveCycle Process Management ES2 may be used with the SLB topology if long-lived processes are avoided. Using long-lived processes in LiveCycle Process Management ES2 is normal though, and leveraging LiveCycle Process Management ES2 may exclude the SLB topology from consideration.

The decision flow is graphically represented in Figure 2. The tree accounts for the experience seen deploying LiveCycle over several years. If possible, the most rapid avenue to a fully working deployment is to use a turnkey topology. There are some implicit limitations, such as Windows only support and single-server deployment. However, these limitations must be balanced with SLB’s greater ease of use.

If the turnkey topology’s limitations are not acceptable, then the next option may be to perform a single-server deployment. Single-server deployment allows the use of any supported platform combination.

Multi-server environments introduce significantly greater complexity and can have longer deployment times. A cluster topology is recommended. This is a very advanced configuration and requires dedicated J2EE skills. It is the most common multi-server configuration in use with LiveCycle.

It is possible to use an SLB topology for multi-server environments if ES2 is being used as a purely stateless service (black box). Between requests, ES2 is not maintaining any request data.
At this point, it is useful to work through the decision tree and work out the acceptable topologies for your deployment.

![Topology decision flowchart](image)

After reviewing which topologies may potentially work, this guide will review in greater detail each topology introducing specific application server traits. It is not pertinent to discuss operating systems or databases, as they do not substantially alter the topology discussion. They do impact performance and administration but those topics are not within the scope of this document.

**Turnkey**

The turnkey option can be used in two topologies. Either as the turnkey, or as a deployment option for the servers that participate in SLB. The turnkey option is popular because of its ease of use. It is:

- Ready to use - Turnkey contains an application server, a database, and an environmental setup so ES2 is ready to use once you exit the install and configuration programs
- Ease of deployment - Aside from its shrink-wrapped nature, turnkey also has an "express" configuration mode that further simplifies the deployment task by choosing common defaults for many configurations. In some circumstances, very few Java skills are required to complete a deployment
- The turnkey can now be used in 64-bit deployments and with existing MS SQL or Oracle servers
- Common - Because this is the most common deployment option, inside Adobe, customers, and developers, it is a very mature topology

Turnkey is limited to the following platform combinations:

- JBoss 4.2.1
- Windows Server 2003 SP2/R2, 2008 R1
- Embedded MySQL 5.1.30 or supported Oracle and MS SQL Server databases
- 32 or 64-bit systems
The turnkey installation with the MySQL option will set up Windows services to launch the database and server components automatically. The turnkey with Oracle or MS SQL Server options will set up Windows services to launch the server components while the database services remain unchanged.

Turnkeys with the MySQL option are appropriate when you wish to treat the embedded application server and database as an atomic unit. Typically in these cases, the infrastructure is considered self-contained, and the database and application server are not individually administered. Also, the data in the database would not normally be considered directly managed by database administrators.

If your organization has incompatible requirements about how business data is managed, then the Oracle or MS-SQL Server options may be feasible. The distinction being that if the ES2 implementation is directly managing audited business processes then the business may require that data be stored in a regulation approved data store. If MySQL is not a regulation-approved database in your organization, then the default turnkey with MySQL is not a good option because it must store the ES2 process data in its MySQL database.

Although the turnkey topology is relatively ease of use and is discrete in nature, it still requires the use of fundamental IT best practices such as failure recovery procedures, monitoring system resource availability, and the implementation of formal, documented administration procedures. There are cases in which these best practices can be relaxed, for instance when the server is easy to rebuild without data loss, is a development server, or is only used for evaluation.

**Single server**

Single-server topologies allow the use of a varied array of infrastructures. You may be required to use this topology because of your organization’s specific standards or available infrastructure. Although there are a significant number of supporting tools to configure and deploy into this kind of environment, you will still need to have a good level of application server proficiency.

The exact procedure for setup and configuration for the following configurations is documented in the “Preparing to Install ES2” and other installation documents. See the appendix for references.

**WebSphere**

A single-server deployment of ES2 on WebSphere can use either the Base or Network Deployment (ND) editions of WebSphere. The Express and Community editions are not supported.

**WebLogic**

A single-server deployment of ES2 on WebLogic can use either the Standard or Enterprise editions of WebLogic Server.

Adobe ES2 requires a managed server when using WebLogic.

**JBoss**

A single-server deployment of ES2 on JBoss supports JBoss 4.2.0 and 4.2.1 (EAP 4.2 and 4.3). There are two methods of deployment. The recommended option is to use the Adobe Pre-configured JBoss tree. This is the standard download from JBoss with basic configurations set for ES2. This provides a good starting point. You can also go directly to JBoss, download the same versions, and apply all the configurations manually as documented. The Adobe pre-configured JBoss tree is used in the turnkey deployment.

The JBoss configuration for ES2 is mostly a manual task in a single-server deployment unlike the other application servers. Deployment tasks include editing XML configuration files, copying files, and can be detail-oriented work.

**Cluster**

Clustering is a very advanced deployment option. It is the most common multi-server topology in real-world use. Adobe recommends a cluster in cases in which scalability, high availability, and load distribution are required. Application server-specific skills in clustering are required.
In an ES2 cluster, all cluster members must be identical. They must be the same version of the application server, have the same version of ES2, and have the same modules deployed. There is no way to separate the ES2 EAR files into web-tier and enterprise-tier components. Remote clients can integrate with the ES2 cluster using the ES2 client libraries or web services.

There are multiple variants of clustering, even with the same type of application server. There are also multiple options for how to implement each layer in the cluster's infrastructure, including hardware, database, and file systems. LiveCycle does not support them all. Be sure to review the clustering guides to understand exactly how the cluster must be setup.

**WebSphere**

A cluster deployment of ES2 on WebSphere must use the Network Deployment (ND) edition of WebSphere. Adobe requires a specific WebSphere cluster topology. This is not optional. Adobe has worked with IBM to ensure the most appropriate possible WebSphere topology is used.

The exact procedure for setup and configuration is documented in the "Preparing to Install ES2 (clustering)" and "Configuring Adobe ES2 Application Server Clusters Using WebSphere" documents.
WebLogic

A cluster deployment of ES2 on WebLogic requires the Enterprise edition of WebLogic Server. Adobe recommends the default WebLogic cluster configuration. This provides the necessary characteristics. The exact procedure for setup and configuration is documented in the "Preparing to Install ES2 (clustering)" and "Configuring ES2 Application Server Clusters Using WebLogic" papers.

Figure 4: WebLogic topology
JBoss

A cluster deployment of ES2 on JBoss supports JBoss 4.2.0 and 4.2.1 (EAP 4.2 and 4.3). In a new cluster 4.2.1 or EAP 4.3 are recommended. There are two methods of deployment. The recommended option is to use the Adobe pre-configured JBoss.

All JGroups must run with UDP support.

The exact procedure for setup and configuration is documented in the "Preparing to Install ES2" and "Configuring ES2 Application Server Clusters Using JBoss" documents.

Considerations for a cluster

In a cluster, there will be a single shared database for all the members running ES2. ES2 supports DB2, Oracle, and Microsoft SQL Server for clustered configurations. MySQL is not supported in a cluster deployment.

An ES2 cluster must contain one or more shared directories across the cluster. There is always the single shared GDS directory. This is a critical component of the overall deployment and needs to be high performance and extremely reliable. It cannot experience any data loss. Optionally, there are directories for fonts and resources, such as templates, images, and PDF documents. These optional directories can either be local to each member of the cluster or on a cluster-wide shared directory. In either case, they need to appear on an identical path on each cluster member.

Timing services in a cluster can be critical, especially when using virtualization products like VMware ESX. All cluster members need to have synchronized time to help ensure the Security Assertions Markup Language (SAML) tokens work correctly. Without this intermediary, LiveCycle communication will fail.

Server load balancing

The SLB topology uses common web server farm techniques to distribute requests to ES2 instances that are atomic and non-clustered. The two most common ways of doing this are using a dedicated IP load-balancing network appliance such as F5's BIG-IP Local Traffic Manager (LTM) or by using a clustered web farm to manage state and have the ES2 atomic instances provide services to the web tier.
The key to this topology is that each ES2 instance has no dependency on any other instance and that the
requests to them are absolutely stateless. Most products except LiveCycle Process Management ES2 are
compatible with SLB. In theory, LiveCycle Process Management ES2 would be compatible with SLB if long-
lived processes were avoided; however, in practice, long-lived processes are a common type of transaction.
Similarly, LiveCycle Rights Management ES2 is not a good candidate for SLB deployment as it maintains state.

Server load balancing has a number of advantages, although Adobe does recommend clustering for its ability
to support all of the ES2 features. SLB advantages include reducing the level of J2EE skills required, making it
more appropriate for .NET-centric customers; reducing failures due to inherent complexity; and reducing the
infrastructure cost by allowing a less expensive software infrastructure (WebSphere Base vs. ND; MySQL vs.
Oracle).

Considerations for an SLB topology
In the case of an SLB, each ES2 instance is self-contained. There is an ES2 database per instance and a GDS
directory per instance. Generally, user-based authentication and access control is not realistic. The topology
should use generic system accounts that are role-centric, such as a “form render user.”

Deployment of applications built with ES2 is different in an SLB than in any other topology. In the other
topologies, there is a single, logical ES2 system. When an application is created or deployed, it is available on
all members of the topology. In an SLB topology, each instance is isolated. For an ES2 application to be
available on all instances, the deployment of the application needs to be carried out on each instance. So when
a LiveCycle archive (LCA) needs to be deployed, it must be deployed multiple times, once to each instance.

ES2 platform support
A summary of supported platforms appears below. For detailed requirements, review the “Preparing to Install
ES2” guide.

32-bit application server support
Below are the platform combinations that are supported on 32-bit application servers with a required 32-bit
operating system.

<table>
<thead>
<tr>
<th>Platform combinations on 32-bit application servers</th>
<th>Windows Server 2003</th>
</tr>
</thead>
<tbody>
<tr>
<td>JBoss 4.2.1 (EAP 4.3)</td>
<td>MySQL</td>
</tr>
<tr>
<td>WebLogic 10gR3</td>
<td>MySQL</td>
</tr>
<tr>
<td>Hardware architectures supported</td>
<td>Intel x86 compatible</td>
</tr>
</tbody>
</table>

Table 1: Platform combinations on 32-bit application servers
64-bit application server support

Table 2 below outlines the platform combinations that require a 64-bit application server and therefore require a 64-bit operating system.

<table>
<thead>
<tr>
<th>Platform combinations on 64-bit application servers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Platform combinations on 64-bit application servers</strong></td>
</tr>
<tr>
<td><strong>Windows Server 2003/2008</strong></td>
</tr>
<tr>
<td>JBoss 4.2.1, 4.2.0 (EAP 4.3, 4.2)</td>
</tr>
<tr>
<td>WebSphere 6.1, 7.0</td>
</tr>
<tr>
<td>WebLogic 10gR3</td>
</tr>
<tr>
<td>Hardware architectures supported</td>
</tr>
</tbody>
</table>

Table 2: Platform combinations on 64-bit application servers

Considerations for deployment

Depending on the type of deployment topology selected, a large number of individuals may be needed to commission an ES2 system. If the topology is a turnkey or single-server deployment, a single IT specialist may be used. A cluster with two or more members typically will require planning by several IT individuals.

Overview of LiveCycle installation programs

There are four general LiveCycle programs or processes involved in the commissioning of an ES2 system.

1. ES2 server installer
2. ES2 Configuration Manager
3. Service container (DSC deployment)
4. LiveCycle Administration UI (web based)

The ES2 server installer primarily manages the file delivery of the LiveCycle server. After an installation is complete the installer walks through a sequence of tasks to create a directory tree that is ready for LCM to configure. The directory tree that the installer provides is a valuable data set and should be carefully maintained and archived for future use during system maintenance (applying service packs, patches and upgrades).

Installer modes include:

- Graphical mode, the default mode and requires a windowing interface.
- Command-Line Interface (CLI) mode, intended to be used by advanced users of LiveCycle or in server environments that do not support the use of a graphical user interface (GUI). The CLI runs in console mode with one interactive session for all install operations. This mode does not use any windowing capabilities. It is included for cases in which the target server for installation does not have a windowing system or in which the windowing system is not usable due to lack of bandwidth or responsiveness.
LiveCycle Configuration Manager is a wizard-like tool used to configure, deploy, and validate ES2 components on the application server. You can optionally use LiveCycle Configuration Manager to configure the application server and deploy the product EAR files to the application server. The LiveCycle Configuration Manager is designed to configure minimum requirements for a basic runtime and leaves configuration to the LiveCycle Administration UI (a web application for LiveCycle administration). LiveCycle Configuration Manager supports the following modes for critical enterprise use cases:

- **Graphical mode**, the default and most common mode. LiveCycle Configuration Manager is run on a system that uses the same operating system on which the target runtime servers are based and manipulates the target application server and LiveCycle deployment units from that location.

- **CLI**, intended for advanced users of LiveCycle and may be useful in server environments that do not support the Configuration Manager GUI. In addition, the CLI is useful in cases where many similar deployments need to be performed.

- **Graphical mode (targeted)** is being deprecated in favor of CLI mode. LiveCycle Configuration Manager is run on a Windows system, but the target runtime servers are based on UNIX®. LiveCycle Configuration Manager utilizes the windowing capabilities of Windows; UNIX systems do not require a working X Window System™ environment.

The WebSphere or WebLogic application servers must be started so that LiveCycle Configuration Manager can perform configuration tasks on them. You may configure an application server that is installed on a different computer than the one on which you are running LiveCycle Configuration Manager. LiveCycle Configuration Manager must have access to the application server library files, this is typically done with a local application server install (it does not have to be running).

**Considerations for installation**

The process of commissioning a significant enterprise system such as ES2 is not trivial. Due diligence to document the procedures and configurations is critical. Documentation, archives of the installation media, serial numbers (where used), electronic certificates, and post-installation directories are valuable aspects of the installation and will be required for maintenance and future support.

**Future-proofing your deployment**

There are several infrastructure decisions that can have significant implications as to how easy it is to upgrade to new infrastructure and new versions of LiveCycle. Some insights into how Adobe is planning for future upgrades will allow you to make appropriate decisions now.

**Service packs**

Apply the latest service packs to your infrastructure whenever possible. This primarily includes ES2 and the application servers. Often defect resolutions are provided in the service pack or via a patch on top of a current service pack. For application servers, one of the first issue resolution procedures the vendor’s support organization will ask to be tried is to apply the latest service, fix or maintenance pack.

Plan to install ES2 and infrastructure service packs in your quarterly project schedule.
Assigning disk space
When planning the amount of disk space for a deployment, especially in the case of virtualization, take into account the long-term requirements of LiveCycle. The system requirements for the initial ES2 deployment are fairly sizable, especially in the case of temporary space which is can be considerable on UNIX. The size of ES2 service packs and the rollback capability used could make disk requirements grow quickly (see Table 3). Allow for growth over the lifetime of a deployment.

<table>
<thead>
<tr>
<th>ES2 hard-disk requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation</td>
</tr>
<tr>
<td>Primary install requirement</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Service packs</td>
</tr>
<tr>
<td>Total</td>
</tr>
</tbody>
</table>

Table 3: ES2 hard-disk requirements

Database selection
It is not possible to move the ES2 database to a different database type, for example from MySQL to Oracle. It is important to be comfortable with the selection you start with. Moving to a different database product will require a new installation. All previous data (for example, long-lived processes in LiveCycle Process Management ES2) will not migrate forward. The applications you created with ES2, however, can be easily moved to the new system.

LiveCycle cache management
ES2 has a built-in cache capability to synchronize changes inside a single instance and across members in a cluster. In a single instance, this is not generally apparent and does not need special attention. In a cluster, this may need to be taken into account. In a cluster, all the members will be sending and receiving cache events from other members over UDP. The events must be able to reach all members in the cluster, and network performance is important. Network reliability is critical as even short network interruptions will cause problems that may require the cluster to be restarted. By default, UDP multicasting is used for members to announce their presence and locate other members. Once members join, cache events are always sent over UDP. Alternatively, a separate Locator service may be running on the same network as the cluster. The Locator service can have backups and all Locator addresses must be configured into each member of the cluster. Locators communicate over TCP/IP, but event notifications still occur over UDP. Using Locators is the preferred deployment option.

Transaction execution
A request will fully execute on a single member or server. This has two important implications. Each ES2 server must have identical capabilities, and a request will not be processed on more than one member or server. There is low amount of additional network bandwidth required other than the requirement of a request to a member/server and the member/server to database and GDS directory. The failure path is also simplified. Should a synchronous request fail due to a system failure, it does not get restarted on a second node. The failure must be handled by the calling application. An asynchronous request will normally be returned to the pool and rerun on another node.

Communications outside the firewall
Some ES2 modules may require access to resources outside the demilitarized zones and corporate network. Examples of this are the Certificate Revocation Lists (CRL) and Online Certificate Status Protocol (OCSP) checks. This is configured on the server and has various options. LiveCycle can be deployed in such a way as to never require external access.
Environmental considerations

Certain LiveCycle ES modules have particular environmental requirements. Successful deployment requires that these settings are accurately accounted for. Adobe LiveCycle PDF Generator ES2 relies on three types of environmental settings:

- Versions of third-party desktop products (native document conversion). The deployment must have specific versions of Acrobat and Microsoft Office in particular. LiveCycle PDF Generator ES2 automates the use of these products to convert Office documents to PDF. Note that Microsoft Office must be started at least once to complete its installation prior to running LiveCycle.

- Windows user account. See Installation guides as to the special requirements for this user.

- Acrobat post configuration. The installation of Acrobat requires additional configurations to work with LiveCycle PDF Generator ES2. This is done by the ES2 installer or may be done later using provided scripts.

LiveCycle PDF Generator ES2 has different capabilities on different operating systems.

<table>
<thead>
<tr>
<th>Operating system features</th>
<th>Windows</th>
<th>Solaris/Linux</th>
<th>AIX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Office docs</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Open Office</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>HTML pages</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Export PDF</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PS/PRN to PDF</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Image to PDF</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>OCR PDF or images to PDF</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Convert PDF and Assembler</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 4: Operating system features

LiveCycle PDF Generator ES2 requires an additional 32-bit JDK on 64-bit Windows. The 32-bit JDK must have an environment variable set (JAVA_HOME_32). This does not affect the application server’s JDK, which will be 64-bit. The ES2 turnkey deployment option supplies these prerequisites automatically. The reason for this additional requirement is that the automation of desktop applications e.g. Microsoft Office requires loading 32-bit dynamic link libraries (DLLs). LiveCycle does this in a process separate from the application server process for reliability, 32-bit DLLs can only be loaded into 32-bit processes.

LiveCycle PDF Generator ES2 also has scalability characteristics different from ES2. The most common case is when Microsoft Office document conversion is being used (native document conversion), and the conversion process is only able to utilize a single CPU at a time (single threaded) (see Table 5).

<table>
<thead>
<tr>
<th>Multi-threaded vs. single-threaded conversion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conversion type</td>
</tr>
<tr>
<td>Office docs</td>
</tr>
<tr>
<td>Open Office</td>
</tr>
<tr>
<td>HTML pages</td>
</tr>
<tr>
<td>Export PDF</td>
</tr>
<tr>
<td>PS/PRN to PDF</td>
</tr>
<tr>
<td>Image to PDF</td>
</tr>
<tr>
<td>OCR PDF or images to PDF</td>
</tr>
<tr>
<td>Convert PDF and Assembler</td>
</tr>
</tbody>
</table>

Table 5: Multi-threaded vs. single-threaded conversion

Additional CPUs may see lighter loads due to other supporting processes, but no significant value is seen beyond one CPU when running conversions that are single threaded. ES2 in general will scale to use large numbers of CPUs. There have been effective deployments done with 16-CPU systems, though these typically require careful management of other bottlenecks such as RAM and disk I/O.
Due to this characteristic of LiveCycle PDF Generator ES2 native document conversion, increasing throughput and performance above what a single CPU delivers requires horizontal scaling of the topology. For a six-CPU throughput requirement, either a six-member horizontal cluster or a six-server SLB configuration would be used.

**Recommendations for system virtualization**

In most cases, virtualization is transparent to ES2. The vendors that provide the underlying virtualization technology do so in ways that guarantee software functions without modification. There are some exceptions.

Areas where virtualization can break down are with direct hardware access, and moving a running image to a second virtual deployment. In the case of Intel x86-compatible hardware and direct hardware access, use of an HSM with virtualization is not always possible. LiveCycle Digital Signatures ES2 usage with an HSM under virtualization requires some investigation to help ensure compatibility.

ES2 relies on the guarantees of the virtualization providers that compatibility exists with real physical hardware. ES2 is tested on VMware using guest Windows and Linux operating systems as well as Power LPars and Solaris 10 Zone partitioning. In cases where a defect arises in the virtualization scenario, support has to be provided by the infrastructure vendor if it cannot be replicated on bare hardware. It is important for the group deploying with virtualization to have specific expertise in the virtualization technology being used, as it can introduce performance constraints.

**Deployment activities by role**

With turnkey deployments there will likely only be a single administrator. For other deployment options, there will likely be several types of administrators involved.

The following is an outline of the tasks required by the core IT roles to deploy an ES2 system. In many cases, these roles may be performed by the same person.

**Software licensing administrator**

Ensure that you have received all the required material and it has been made available to the installation team. It is important to archive these in a safe and well known location.

1. The documented licenses for the ES2 modules your organization has purchased. Ensure they are for the correct platform and are for the correct mode (evaluation, development, production)
2. Adobe LiveCycle Reader Extensions ES2 credential PFX file and credential password
3. ES2 media (disk set) or ESD download location and credentials
4. ES2 current Service Pack ESD download location
5. Acrobat Pro Extended Serial Numbers

**Server administrator on Windows:**

1. Verify the correct system prerequisites (see appendix)
2. Expand the temporary file system to at least 3 GB of available space for install. It can be reduced after the installation
3. Make available at least 10GB of storage for the ES2 binaries and runtime. For a small number of modules, 5GB may be adequate, but the requirement will grow with patches and service packs

**Server administrator on UNIX:**

1. Verify the correct system prerequisites (see appendix)
2. Expand the /tmp filesystem to at least 3GB of available space for install. It can be reduced after the installation
3. Make available at least 10GB of storage for the ES2 binaries and runtime. For a small number of modules, 5GB may be adequate, but the requirement can grow with patches/service packs.
Application server administrator
1. Install and prepare the application server. If clustered, set up the basic cluster and verify it is operational.
2. Provide the username and password of the user with whose credentials the application server is to run. LiveCycle has to be installed and configured while logged in as this user.

Database administrator (DBA)
1. Prepare a database server with enough available resources to support ES2. Create a database instance and user for ES2 according to the requirements in "Preparing to Install LiveCycle ES".
2. Provide database connection information and credentials to the LiveCycle ES2 administrator.
3. Be available during the ES2 deployment in case the database tables needs to be dropped.
4. When the ES2 deployment is complete, be prepared to do a full LiveCycle database backup and real-time backups if required.

Network administrator
1. Provide permanent IP addresses for all servers running ES2.
2. Open the required ports if ES2 is being installed into a DMZ, for example, HTTP, SOAP, or RMI.
3. If clustering is using the default ES2 cache locator protocol, verify that UDP broadcasts are available on the network between cluster nodes.

LDAP administrator
If the ES2 user management is going to integrate with the corporate directory:
1. Create a read-only user ID in the LDAP directory so that ES2 can sync with the corporate directory LDAP.
2. Provide the following information to the ES2 administrator:
   a. Fully qualified LDAP name for the user ID to access LDAP.
   b. Password for the user ID.
   c. Base DN to start the LDAP search.
   d. Fully qualified name of the OU that contains the users and groups who will be authorized to access ES2.

Storage server administrator
If this is a cluster deployment:
1. Create a high availability network share for use as the GDS directory. Ensure it can be accessed by all cluster members.
2. Work with the database administrator to set up the appropriate backup regime.

LiveCycle administrator
1. Coordinate with all the participants and ensure they are prepared for the day of the installation.
2. Execute the ES2 installation.
3. Verify the functioning of the install.
4. Record the installation procedure, file sets, and configuration.

Client desktop administrator
If this is a remote UNIX install:
1. Install and configure an X client on the Windows desktop that will be used to access the UNIX boxes. This may be Hummingbird Exceed, WRQ Reflection, Cygwin, etc.
Appendix

This appendix contains references to important documents and procedures for installation and deployment.

**Preparing to Install ES2** - ES2 system requirements

Before installing

**Preparing to Install LiveCycle ES**

Describes the steps to complete before installing and configuring modules on a J2EE application server.

Setting up your development environment

**Installing Your Development Environment**

Describes how to install LiveCycle Workbench ES2 and the LiveCycle software development kit (SDK) on an application development computer.

Setting up your runtime environment

**Installing and Deploying ES2 for JBoss Using Turnkey**

Describes how to install, configure, and deploy modules using the turnkey installation.

**Installing and Deploying ES2 for JBoss**

Describes how to install, configure, and deploy modules on a JBoss application server.

**Installing and Deploying ES2 for WebLogic**

Describes how to install, configure, and deploy modules on a WebLogic application server.

**Installing and Deploying ES2 for WebSphere**

Describes how to install, configure, and deploy modules on a WebSphere application server.

**Configuring ES2 Application Server Clusters Using JBoss**

Describes how to configure a JBoss cluster and deploy ES2 to the cluster.

**Configuring ES2 Application Server Clusters Using WebLogic**

Describes how to configure a WebLogic cluster and deploy ES2 to the cluster.

**Configuring ES2 Application Server Clusters Using WebSphere**

Describes how to configure a WebSphere cluster and deploy ES2 to the cluster.

**LiveCycle Data Services ES2 Installation**

Describes how to install and configure LiveCycle Data Services ES2.

Additional recommended reading

**Hardening and Security for ES2**

For more information and additional product details:

http://www.adobe.com/devnet/livecyle/

We welcome your comments. Please send any feedback on this technical guide to LCES-Feedback@adobe.com.

Adobe, the Adobe logo, Acrobat, LiveCycle, and Reader are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. Intel is a registered trademark of Intel Corporation in the United States and other countries. IBM and AIX are trademarks of International Business Machines Corporation in the United States, other countries, or both. Linux is a registered trademark of Linus Torvalds in the U.S. and other countries. Microsoft, Windows, and Windows Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. SUSE is a trademark of Novell, Inc. Red Hat is a trademark or a registered trademark of Red Hat, Inc. in the United States and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc. Sun, Java, and Solaris are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. UNIX is a registered trademark of The Open Group in the U.S. and other countries. X Window System is a trademark of The Open Group. All other trademarks are the property of their respective owners.