

Using DeployR to solve the R integration problem

Microsoft Advanced Analytics Team

April 2016

Introduction

Organizations use analytics to empower decision making, often in real time. That means the ability to easily embed and share the results of R analytics within existing web, desktop, and mobile applications, plus backend systems, is vital. As important as real-time delivery of analytics is to this process, users also need to consider security in terms of identity and data integrity, and scale in terms of workload and throughput. DeployR from Microsoft delivers the power and flexibility of R securely and at scale to all of your decision-making systems.

DeployR is an integration technology for deploying R analytics inside web, desktop, mobile, and dashboard applications, as well as backend systems. DeployR turns your R scripts into analytics web services, so R code can be easily executed by applications running on a secure server.

Using analytics web services, DeployR also solves key integration problems faced by those adopting R-based analytics alongside existing IT infrastructure. These services make it easy for application developers to collaborate with R programmers to integrate R analytics into their applications without any R programming knowledge.

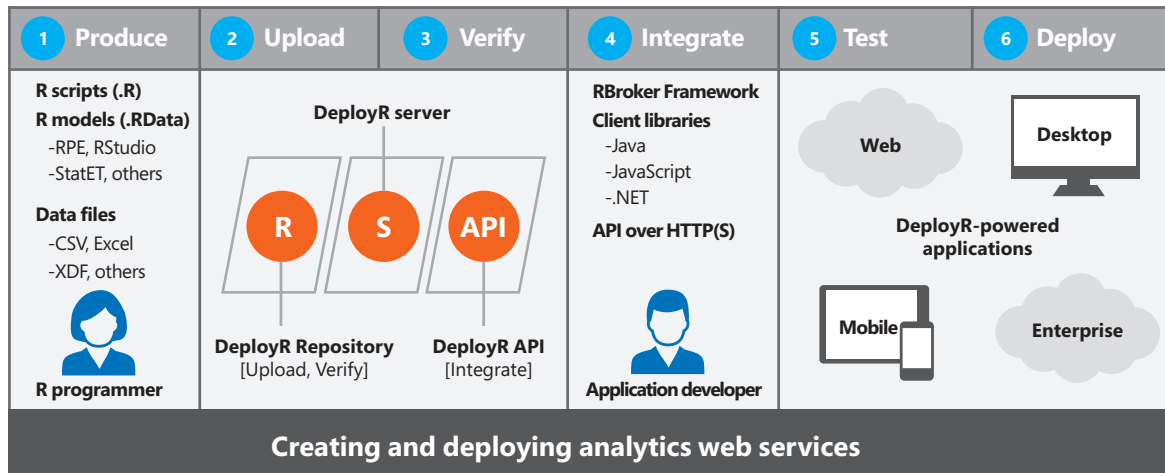
DeployR is available in two editions: DeployR Open and DeployR Enterprise. DeployR Open is a free, open-source solution that is ideal for prototyping, building, and deploying noncritical business applications. DeployR Enterprise scales for business-critical applications and offers support for production-grade workloads, as well as seamless integration with popular enterprise security solutions such as single sign-on (SSO), Lightweight Directory Access Protocol (LDAP), Active Directory, or pluggable authentication modules (PAM).

Analytics web services

In DeployR, we refer to any web service that exposes R analytics capabilities over the network as an analytics web service. While “web services” is commonly used in the context of browser-based web applications, these services—in particular, analytics web services—can just as easily be integrated inside desktop, mobile, and dashboard applications, as well as backend systems. For example, when you upload an R script into the DeployR Repository Manager, the script may then be executed as an analytics web service by any application with appropriate permissions.

Basic DeployR workflow

This diagram captures the basic workflow used by R programmers and application developers when collaborating on the delivery of solutions powered by analytics web services.



Workflow in action

The workflow is simple: An R programmer develops an R script (using standard R tools) and publishes that script to the DeployR server, where it becomes available for execution as an analytics web service. Once published, R scripts can be executed by any authorized application using the DeployR application programming interface (API).

The following real-world scenario demonstrates the key concepts introduced in the preceding workflow diagram.

While this scenario is provided as an example only, it highlights how DeployR acts as an effective bridge between R programmer and application developer workflows. By supporting a clean separation of concerns, DeployR delivers arbitrarily sophisticated R analytics to any network-enabled application, system, or software solution.

The scenario

As a solutions provider at an insurance company, you have been tasked with enhancing an existing customer support system to deliver real-time, high-quality policy quotes to customers who contact your customer support center.

The plan

To deliver this solution, you decide to leverage the power of R-based predictive analytics with DeployR. This relies on a real-time scoring engine architecture to rapidly generate and return accurate scores to your existing customer support system. These scores can then be used to help drive the customer experience to a successful conclusion. At a high level, such a DeployR-powered solution can be realized as follows:

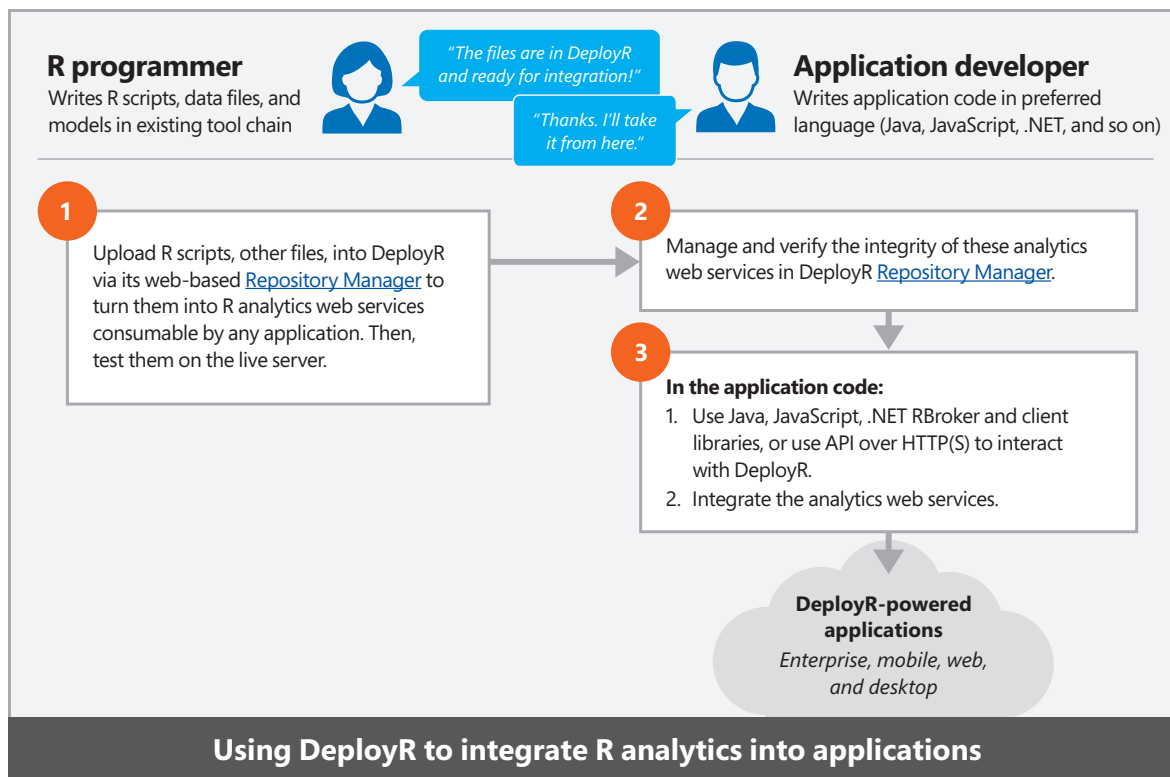
1. **Produce:** R programmers begin by building an appropriate predictive model and scoring function for insurance policy quotes using their existing analytics tools and datasets.
2. **Upload:** Once the model and scoring function are ready, the R programmer uploads these files to the DeployR repository via the Repository Manager. Uploading the scoring function as an R script will automatically turn that script into an analytics web service.
3. **Verify:** As a final step, the R programmer should test and verify the scoring function against a live DeployR server via the Repository Manager.

Handoff from R programmer to application developer

4. **Integrate:** Application developers choose and download their preferred client application integration tool: RBroker Framework, client library, or the raw API.
5. **Test:** With their tool selected, application developers implement the integration between the customer support system and the DeployR-powered analytics web service. This can be an iterative process with testing at each stage of development.
6. **Deploy:** Once application developers are confident their integration is complete, tested, and verified, it is time to deploy your enhanced customer support system to your live production environment.

Roles and responsibilities

This diagram shows how R programmers and application developers collaborate when working with DeployR.



R programmers

Role

R programmers, sometimes referred to as data scientists, typically focus on developing analytics solutions using their existing analytics tool chain—the R Productivity Environment (RPE), RStudio, StatET, and others. To maximize the impact of these R programmers, DeployR is designed to encourage minimal change in this workflow. Therefore, R programmers can remain focused on creating the R code, models, and data files necessary to drive your analytics solutions without having to concern themselves with how these outputs will eventually be used by application developers in their software solutions.

In DeployR, there is just one tool for the R programmer: the Repository Manager. When the R programmer uploads R scripts, models, and data files into the Repository Manager, they automatically turn into analytics web services that can be consumed by any software solution.

The Repository Manager is a web-based tool that serves as a bridge between the R programmer's work and the deployment of that work into the DeployR repository. Once deployed, application developers can create DeployR-powered client applications and integrations. You can review documentation for the Repository Manager at aka.ms/r/deployr/docs/repository-manager.

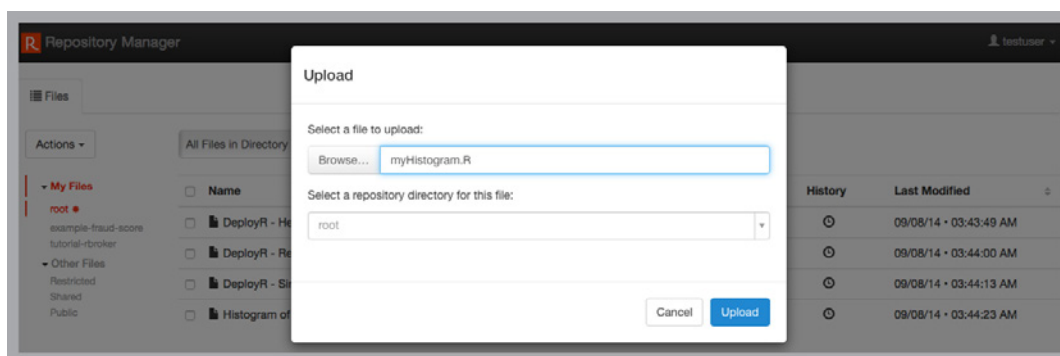
Responsibilities

With DeployR, the recommended steps for an R programmer are both simple and familiar:

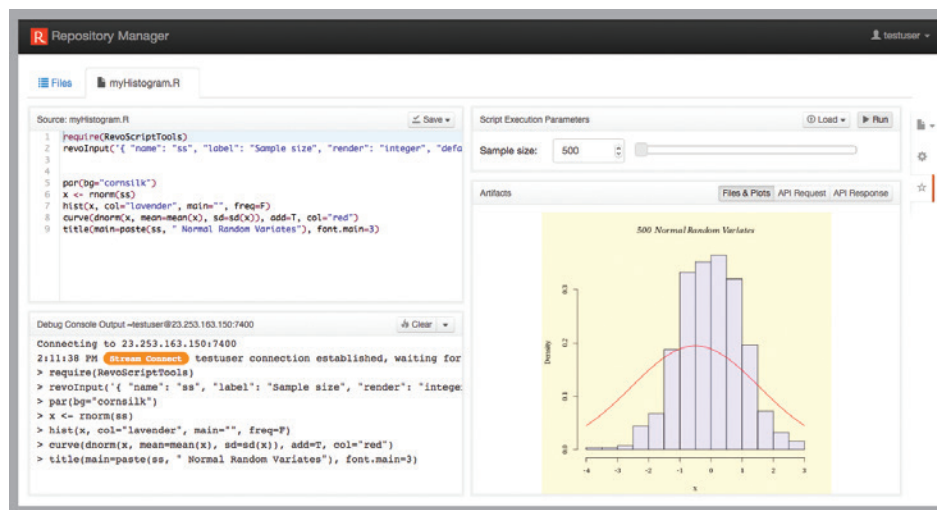
1. Use your preferred R integrated development environment (IDE) to develop and test your R scripts, models, and datasets.
2. Use the functions from the `deployrUtils` R package to make your R code portable. Learn more here: aka.ms/r/deployr/docs/deployrutils. You can install `deployrUtils` locally using your IDE, R console, or terminal window from GitHub: github.com/Microsoft/deployrUtils/releases.

Note: Once you've uploaded your R scripts into the Repository Manager in the next step, you'll be able to test and visualize these inputs in the Test tab before they are integrated into the application.

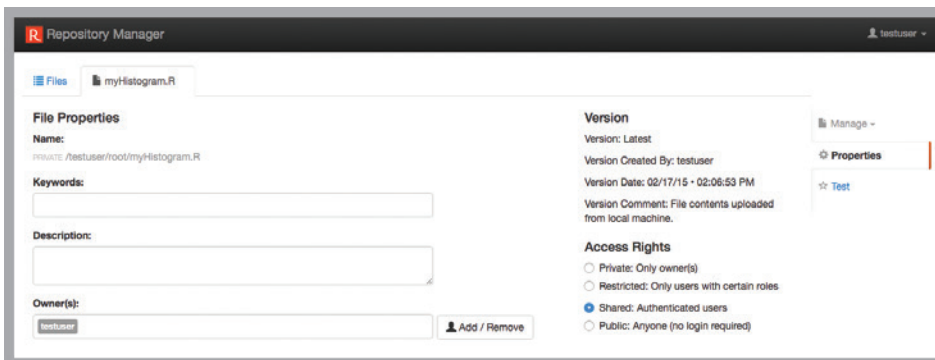
3. Upload these files into the DeployR repository using the Repository Manager.



4. Test your R scripts live on DeployR in the Repository Manager.



5. Once testing is complete, make your files available for use by the application and its developer by setting the appropriate access rights on each file.



The screenshot shows the DeployR Repository Manager interface with the 'File Properties' and 'Version' tabs selected for the file 'myHistogram.R'. The 'File Properties' tab displays the file name, keywords, description, and owner. The 'Version' tab shows the version history, including the latest version created by 'testuser' on 02/17/15. The 'Access Rights' section shows the file is set to 'Shared: Authenticated users'.

6. Inform your application developers that the new R scripts, models, and/or datasets are available in the DeployR repository and ready for integration.

Application developers

Role

Unlike the R programmer who focuses solely on developing R scripts, models, and data files, the application developer does not need to know any R. Instead, this role focuses solely on integrating the output of the R programmer's work (analytics web services) into their applications. The handoff occurs in the Repository Manager. The separation of responsibilities between R programmers and application developers is critical to working effectively in the DeployR environment.

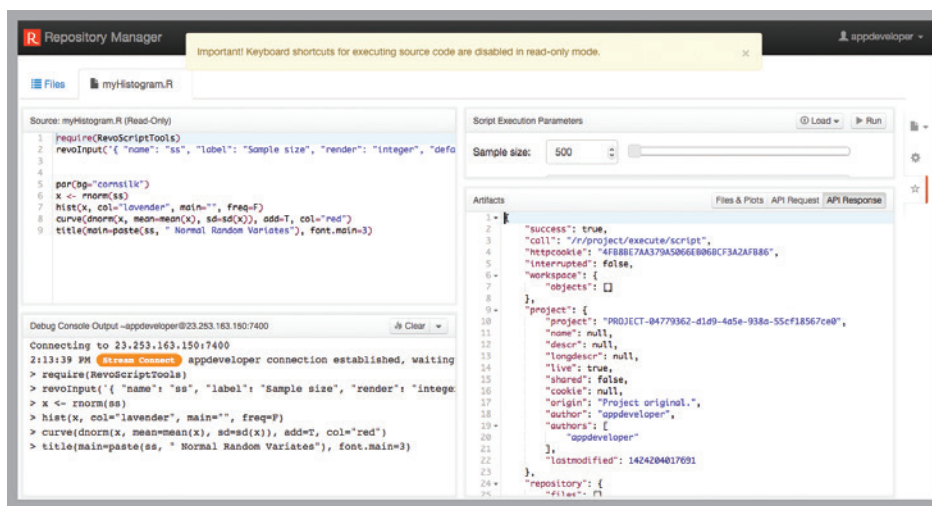
There are several client application integration tools available to application developers in DeployR. The RBroker Framework and client libraries—which are provided in Java, JavaScript, and .NET—greatly simplify the integration for anyone working in those languages. However, to integrate analytics web services using other programming languages, the [API Reference Guide](#) details everything you need to know.

Responsibilities

Regardless of your integration tools—or whether you are building from scratch or enhancing an existing design—the basic steps for leveraging DeployR-powered analytics web services inside any application remain simple and consistent:

1. Consult with the R programmers responsible for developing the analytics outputs to determine your application's analytics dependencies (inputs and outputs).
2. Verify these dependencies in the Repository Manager by testing the R scripts live on DeployR. For each script, you can inspect the API request and API response in the Artifacts pane to learn how your application needs to interact with that script.

Note: Once the dependencies are in the DeployR repository, they become accessible via analytics web services.



3. In order to integrate with application code, select a DeployR client application integration tool. Examples include the RBroker Framework or a client library in Java, JavaScript, or .NET. If working in another language, read the [API Reference Guide](#). To learn more about these tools and download them, visit aka.ms/r/deployr/docs/developer-tools.
4. Build or extend your application using your DeployR-powered analytics web services.

DeployR client application integration libraries and API

The DeployR API exposes a wide range of R analytics services to client application developers. These services are exposed using standards-based JSON/XML and are delivered as web services over HTTP(S). This standards-based approach lets you integrate DeployR services within just about any client application environment.

The following libraries are available to DeployR users.

The RBroker Framework

The RBroker Framework is the simplest way to integrate DeployR-enabled analytics web services inside any Java, JavaScript, or .NET application. Just define an RTask, submit your task to an instance of RBroker, and then retrieve your RTaskResult. It really is that simple.

The RBroker Framework works well in most real-world use cases. However, in a case where the RBroker Framework does not support your requirements, you can work directly with the client library APIs, which expose more complete API coverage for Java, JavaScript, and .NET developers. For example, the RBroker Framework is optimized for transactional, on-demand analytics, where each invocation of an analytics web service is a standalone operation that executes on a stateless R session environment. If your application requires a long-lived stateful R session environment, then we recommend the DeployR client libraries, which offer support for stateful operations on DeployR-managed projects.

Visit aka.ms/r/deployr/docs/developer-tools to download the framework and API documentation. The RBroker Framework tutorial and examples are also available.

The DeployR client libraries

The DeployR client libraries—available in Java, JavaScript, and .NET—offer broader access to the complete DeployR API. The client libraries provide many significant advantages over working directly with the raw API, including simplified service calls, encoding of call parameter data, and automatic handling of response markup on the API.

Visit aka.ms/r/deployr/docs/developer-tools to download the client library and API documentation for your preferred language. The client library tutorial and examples are also available.

The API Reference Guide

This comprehensive reference guide provides the definitive documentation for every DeployR server API service. In it, you can learn all about DeployR users, projects, jobs, and repository services. The guide also covers how R object data encodings are handled for all data entering and leaving the DeployR server.

This guide serves as an excellent resource whenever you are unsure of how best to use the client libraries or whenever you would like to understand the full set of parameter values available for a particular API call.

The API Reference Guide covers everything you need to know to work directly with the raw API, making it particularly helpful to those who are integrating DeployR-powered R analytics using a language other than Java, JavaScript, or .NET.

You can learn more at aka.ms/r/deployr/docs/api-reference-guide.

DeployR security

DeployR supports a highly flexible, enterprise-grade security framework that verifies identity, enforces permissions, and ensures privacy.

Identity is established and verified using many well-known authentication solutions. Basic authentication, using username and password credentials, is available to all by default. The DeployR Enterprise edition extends support for authentication by providing a seamless integration with established enterprise security solutions including SiteMinder Single Sign-On, PAM authentication, LDAP authentication, and Active Directory authentication.

Permissions are enforced at runtime to control access to DeployR-managed resources. The DeployR administrator has a wide range of access control policies that can be tailored to specific deployment needs, including IP filters and cross-origin resource sharing (CORS) policies. Permissions for individual users within DeployR may also be inherited directly from existing configurations within the established enterprise security solutions indicated above. Individual users can also make use of repository-managed file access controls that offer fine-grained access controls on individual files such as R scripts, models, and data files.

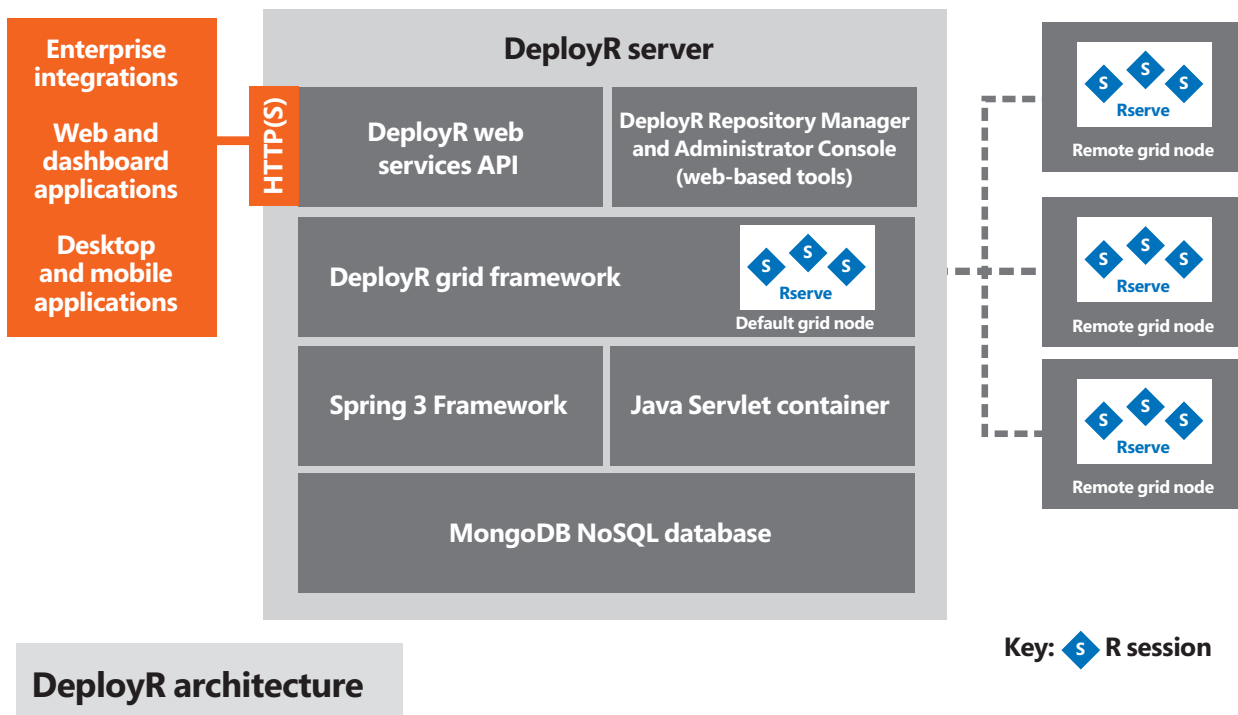
Privacy is ensured through a combination of identity verification, file access controls, database security, and the use of HTTPS to ensure data encryption for all data entering or leaving the server.

You can learn more at aka.ms/r/deployr/docs/security.

DeployR architecture

DeployR is a standalone server product—potentially sitting alongside but never directly connected with other systems—that can be deployed on site and in private, public, or hybrid cloud environments.

Behaving like an on-demand R analytics engine, DeployR exposes a wide range of related analytics services via a web services API.



The fact that DeployR is a standalone product means that any software solution, whether it's a backend enterprise messaging system or a client application running on a mobile phone, can leverage DeployR-powered analytics services.

DeployR Enterprise supports a scalable grid framework, providing load-balancing capabilities across a network of node resources. On the other hand, DeployR Open supports a static grid framework of a single, local grid node with a fixed slot limit. For more on planning and provisioning your grid framework, see the Scale & Throughput documentation at aka.ms/r/deployr/docs/throughput.

Summary

DeployR facilitates the collaboration between R programmers and application developers, allowing each to work with the tools they know best. DeployR can turn any R script into a secure, executable analytics web service. This allows the application programmer to easily integrate results into their applications without having to learn R. These analytics web services run behind a firewall.

DeployR Open is a 100 percent open-source project, and it includes many features previously only available as part of DeployR Enterprise. DeployR Open is deployed as a single node and, as such, it is mainly designed for prototyping, building, and deploying applications when the expected load on the server is low.

If you anticipate a need to scale to multiple server resources to handle increased workload or improved throughput—or if you want to enjoy seamless integration with popular enterprise security solutions such as single sign-on (SSO), Lightweight Directory Access Protocol (LDAP), Active Directory, or pluggable authentication modules (PAM)—consider upgrading to DeployR Enterprise.

For more information on DeployR or to download DeployR Open, visit our website at aka.ms/r/deployr/docs.

About open-source R

R is the most widely used statistical language, with more than two million users worldwide. Top university talent is graduating with R skills, ready to help global enterprises innovate and realize value from big data. Microsoft supports the growing R community with open-source contributions, user group sponsorships, and free RRE licenses for academia.