Whitepaper:

DEPLOYING POWERBUILDER APPS TO THE CLOUD

Last Updated: December 4 2020

INSTALLABLE CLOUD APPS

The new generation of PowerServer (PowerServer 2021) enables you to automatically deploy PowerBuilder projects as “installable cloud apps”. The apps are installable on desktops or pads. Cloud meaning the database and the database-related operations residing in the cloud (rather than on-premise) with automatic deployment of the client to users over the Internet. In other words, you can think of “installable cloud apps” as the modern desktop app that requires Internet connection to install and run.

POWERSERVER 2021 CHANGES

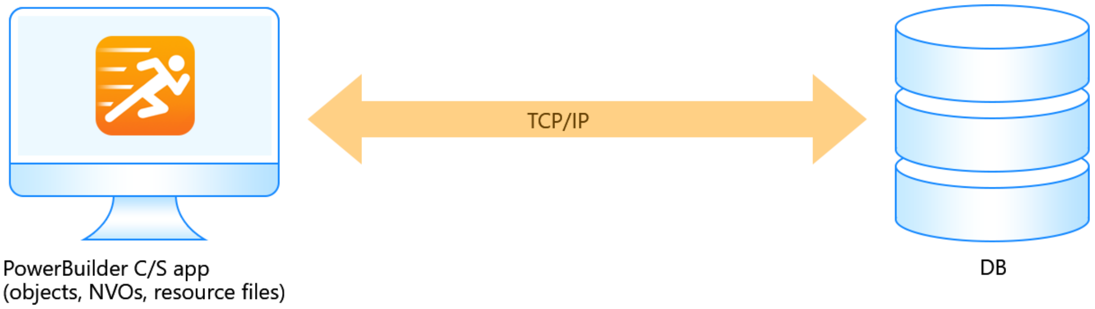
PowerServer 2021 replaces the Web browser implementation of the older generation PowerServer with an installable cloud app implementation. This replacement of the previous implementation is much more powerful, offering the following key advantages.

- Essentially 100% of PowerBuilder features are supported. This is now possible because PowerServer 2021 no longer translates to JavaScript (e.g. Web app) and instead natively uses PowerScript (e.g. desktop app).
- Installable cloud apps are Internet-installable applications. An installable cloud app will be initially installed by accessing the app URL in a web browser. After that, the app has no dependency on a web browser (type, version, or settings) but will run and update itself as needed over the Internet.
- Communication between the desktop and the server is through standard REST/JSON interface (instead of previously the proprietary binary format). Since this is standard REST/JSON, it can be secured using whatever approach you like, for example, OAuth 2.0 for user authentication and AES encryption to protect the JSON data.
- .NET Core is now supported. This changes the landscape for .NET in 3 major ways: 1) it supports cross-OS (e.g. Linux server), 2) the server-side performance is faster, and 3) Microsoft redesigned from the ground up to be cloud-deployable (including Docker compatibility and side-by-side versioning).
- The product license activation mechanism has been revised for cloud compatibility, and even supports serverless options such as Amazon Lambda and Azure Functions.

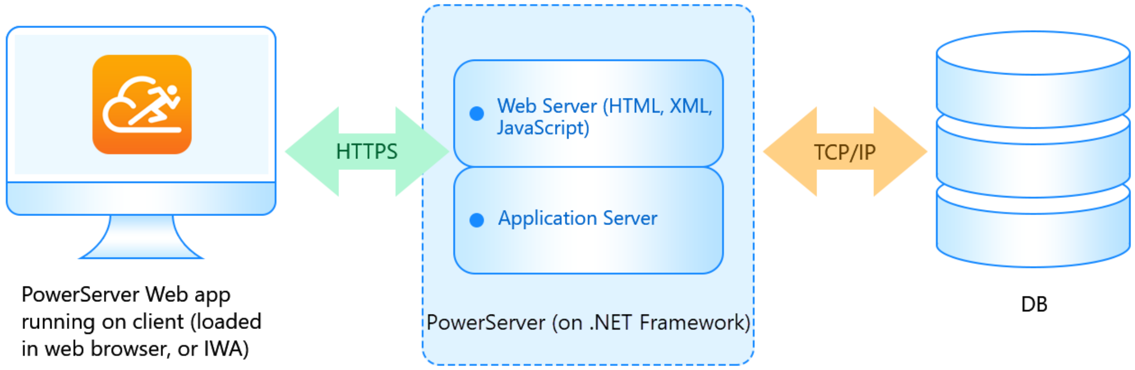
The key tradeoff of this change to the installable cloud app implementation is that PowerServer 2021 can no longer support iOS and Android devices. This is because PowerScript is only supported by the Windows OS.

POWERSERVER ARCHITECTURE COMPARISONS

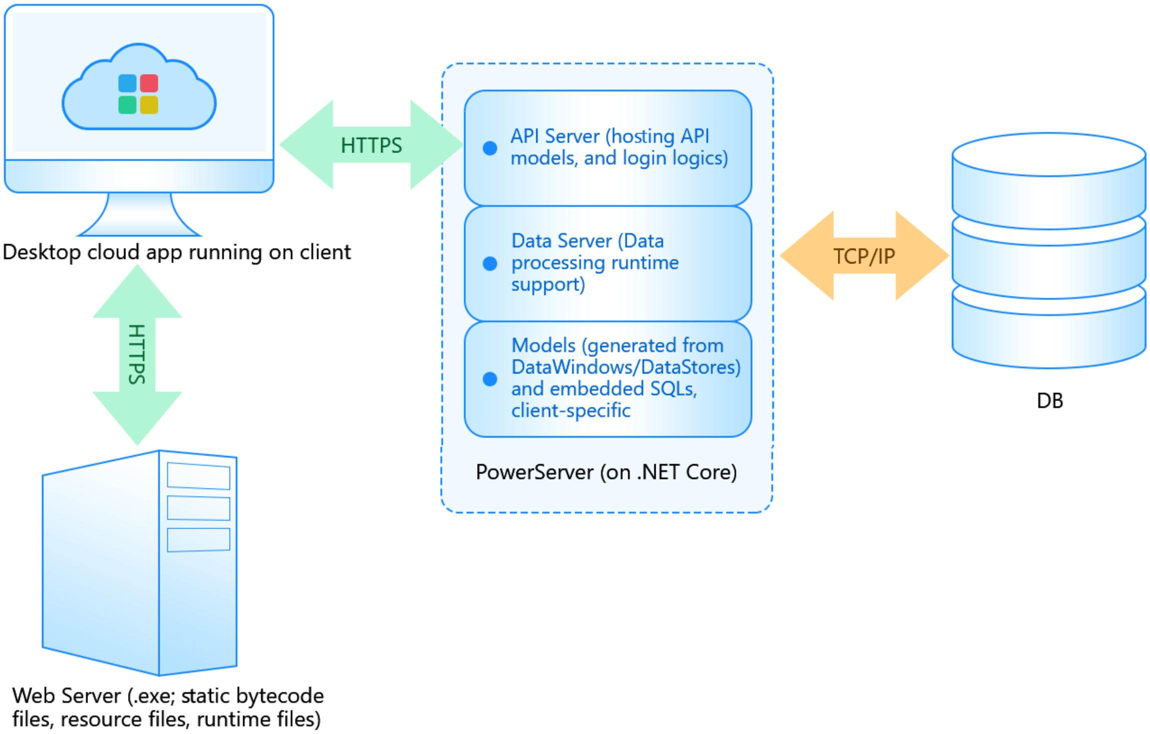
PowerBuilder Client/Server Architecture:



Previous PowerServer Architecture:


HTTPS
Web Server (HTML, XML, JavaScript)
Application Server
PowerServer (on .NET Framework)
TCP/IP
PowerServer Web app running on client (loaded in web browser, or IWA)
DB

PowerServer 2021 Architecture:


Desktop cloud app running on client
HTTPS
HTTPS
Web Server (.exe; static bytecode files, resource files, runtime files)
API Server (hosting API models, and login logics)
Data Server (Data processing runtime support)
Models (generated from DataWindows/DataStores) and embedded SQLs, client-specific
PowerServer (on .NET Core)
TCP/IP
DB

How PowerServer 2021 works?

- PowerServer 2021 adopts a REST API approach. Specifically, the desktop application code is powered by REST APIs that interface with the data sources, and such REST APIs can be hosted in a public or private cloud.
- It is possible to configure load balancing between multiple PowerServer instances. Every PowerServer instance will host both the REST API (referred to as API Server) and the data processing (referred to as Data Server).
- In case of multiple PowerServer instances, sticky sessions will be used to preserve transaction integrity. Specifically, all the requests in one transaction will to be handled by the same server instance.

INSTALLABLE CLOUD APP DEPLOYMENT

Application compilation

Traditionally, when you compile a PowerBuilder application using p-code, PowerBuilder builds PowerBuilder dynamic libraries (PBD files). The PBD files contain everything, such as visual objects, DataWindows/DataStores, PowerScript code, and embedded SQL. All the PBD files must be installed to the desktop and work together to get the application running.

The compilation of installable cloud apps is different from the traditional p-code approach:

- The data access now executes on the server instead of the desktop:

  - All DataWindows/DataStores are automatically converted to .NET models, and then automatically exposed via REST/JSON APIs.
  - All embedded SQLs will be deployed to the server side, and then automatically exposed via REST/JSON APIs.
- All PowerScript and all other objects (besides DataWindows/DataStores) continue to execute on the desktop. In other words, the application business logic written in PowerScript is not automatically partitioned by PowerServer.
- Although it is unnecessary to partition the business logic for installable cloud apps, if you desire to reuse the application business logic elsewhere, or optimize the overall application performance, the PowerScript Migrator product can highly-automate the conversion of such business logic to REST/JSON APIs.
- All PBD files are broken down very granularly into each individual object/definition file. For example, each SRW, SRD, SRU, etc. file would have its individual corresponding p-code file (that have new file extensions, such as .dwo, .apl, .fun, .win, .udo) instead of a monolithic PBD files. Then each version of every p-code file has a unique identifier so that when a user runs the installable cloud app the p-code files can be incrementally downloaded and updated, depending on the window and functionality the user is accessing.

So you will see that with the new compilation approach, it is enhancing PowerBuilder compilation to not only support a client/server model but to also support a cloud model. Simply put, data access is automatically moved from the desktop to the cloud, and the application installation is redesigned to work efficiently over the Internet, providing the key changes necessary to deploy existing client/server projects to the Cloud.

Application deployment

Deployment to the Web server

After the app compilation, the following two types of files will be deployed to the Web server:

- P-code files that are granular in size (not the monolithic PBD format).
- The application .exe file and supporting files such as resource files, OCX files and desktop runtime files (e.g. PBVM). Such files are the same as in client/server PowerBuilder applications. There will be dedicated configuration settings for you to specify these supporting files that will be packaged and deployed with the application to the desktop.

Although a Web server is used, technically it is acting as a file server (i.e. no code executes on the Web server).

Deployment to PowerServer

The following two types of files are converted from DataWindows/DataStores/embedded SQLs, deployed to PowerServer, compiled together with necessary server DLLs, and published as .NET Core applications (REST APIs).

- .NET models;
- SQLs.

Client deployment

Installable cloud apps are Internet-installable applications. An installable cloud app will be initially installed by accessing the app URL in a web browser. After that, the app has no dependency on a web browser (type, version, or settings) but will run and update itself as needed over the Internet.

KEY MANUAL TASKS

PowerServer is a highly-automated cloud deployment solution that requires very little code changes. However, there are a few key tasks that is not automatically handled, requiring manual programming effort:

- User authentication. It is possible for installable cloud apps to apply OAuth2.0 authentication. Some settings will be provided after the application deployment for configuring the user authentication strategy for the application.
- Connection to the database. The SQLCA settings will be configured for the PowerServer as data source settings and dynamic database connection settings, which establishes the connection between the PowerServer and the database.