

FPGA Attestation for Azure NP VMs

The FPGA Attestation service performs a series of validations on a design checkpoint file (called a “netlist”) generated by the Xilinx toolset and produces a file that contains the validated image (called a “bitstream”) that can be loaded onto the Xilinx U250 FPGA card in an NP series VM.

Pre-Requisites

You will need an Azure subscription and an Azure Storage account. The subscription gives you access to Azure and the storage account is used to hold your netlist and output files of the attestation service.

We provide Powershell and Bash scripts to submit attestation requests. The scripts use the Azure CLI, which can run on Windows and Linux. Powershell can run on Windows, Linux and MacOS.

Azure CLI download (required):

<https://docs.microsoft.com/en-us/cli/azure/install-azure-cli?view=azure-cli-latest>

Powershell for Windows, Linux and MacOS download (only for powershell scripts):

<https://docs.microsoft.com/en-us/powershell/scripting/install/installing-powershell?view=powershell-7>

You will need to have your tenant and subscription ID authorized to submit to the attestation service. Visit <https://aka.ms/AzureFPGAAttestationPreview> to request access.

Building Your Design for Attestation

The preferred Xilinx toolset for building designs is Vitis 2020.2. Netlist files that were created with an earlier version of the toolset and are still compatible with 2020.2 can be used. Make sure you have loaded the correct shell to build against. The currently supported version is xilinx_u250_gen3x16_xdma_2_1_202010_1. The support files can be downloaded from the Xilinx Alveo [lounge](#).

You must include the following argument to Vitis (v++ cmd line) to build an xclbin file that contains a netlist instead of a bitstream.

```
--advanced.param compiler.acceleratorBinaryContent=dcf
```

Logging into Azure

Prior to performing any operations with Azure, you must log into Azure and set the subscription that is authorized to call the service. Use the `az login` and `az account set -s <Sub ID or Name>` commands for this purpose. Further information about this process is documented here:

<https://docs.microsoft.com/en-us/cli/azure/authenticate-azure-cli?view=azure-cli-latest>. Use either the ‘sign in interactively’ or ‘sign in with credentials’ option on the command line.

Creating a Storage Account and Blob Container

Your netlist file must be uploaded to an azure storage blob container for access by the attestation service.

Refer to this page for more information on creating the account and a container and uploading your netlist as a blob to that container: <https://docs.microsoft.com/en-us/azure/storage/blobs/storage-quickstartblobs-cli>.

You can also use the portal for this as well.

Upload Your Netlist File to Azure Blob Storage

There are several ways to copy the file; an example using the `az storage upload` cmdlet is shown below. The az commands run on both Linux and Windows. You can choose any name for the “blob” name but make sure to retain the xclbin extension.

```
az storage blob upload --account-name <storage account to receive
netlist> --container-name <blob container name> --name <blob filename> --file
<local file with netlist>
```

Download the Attestation Scripts

The Validation scripts can be downloaded from the following Azure storage blob container:

<https://fpgaattestation.blob.core.windows.net/validationscripts/validate.zip>

The zip file has two powershell scripts, one to submit and the other to monitor while the third file is a bash script which performs both functions.

Running the Attestation Scripts

To run the scripts, you will need to provide the name of your storage account, the name of the blob container where the netlist file is stored and the name of the netlist file. You will also need to create a Service shared access signature (SAS) that grants read/write access **to your container** (not the netlist). This SAS is used by the attestation service to make a local copy of your netlist file and to write back the resulting output files of the validation process to your container.

An overview of shared access signatures is available [here](#) with specific information about the Service SAS available [here](#). The Service SAS page includes an important caution about protecting the generated SAS. **Please read the caution to understand the need to keep the SAS protected from malicious or unintended use.**

You can generate a SAS for your container using the `az storage container generate-sas` cmdlet. Specify an expiry time in UTC format that is at least a few hours past the time of submission; around 6 hours should be more than adequate.

If you wish to use virtual directories, you must include the directory hierarchy as part of the container argument. For example, if you have a container named “netlists” and have a virtual directory named

“image1” that contains the netlist blob, you would specify “netlists/image1” as the container name. Append any additional directory names to specify a deeper hierarchy.

Powershell

```
$sas=$(az storage container generate-sas --account-name <storage acct name>
--name <blob container name> --https-only --permissions rwc --expiry <e.g.,
2021-01-07T17:00Z> --output tsv)
```

```
.\Validate-FPGAImage.ps1 -StorageAccountName <storage acct name> -Container
<blob container name> -BlobContainerSAS $sas -NetlistName <netlist blob
filename>
```

bash

```
sas=`az storage container generate-sas --account-name <storage acct name>
--name <blob container name> --https-only --permissions rwc --expiry <e.g.,
2021-01-07T17:00Z> --output tsv`
```

```
validate-fpgaimage.sh --storage-account <storage acct name> --container <blob
container name> --netlist-name <netlist blob filename> --blob-container-sas
$sas
```

Checking on the Status of Your Submission

The Attestation service will return the orchestration ID of your submission. The submission scripts automatically start monitoring the submission by polling for completion. The orchestration ID is the primary way for us to review what happened to your submission so please keep that in case you have an issue. As reference points, attestation takes about 30 minutes to complete for a small netlist file (300MB in size); a 1.6GB file took an hour.

You can call the Monitor-Validation.ps1 script at any time to get status and results of attestation, providing the orchestration ID as an argument:

```
.\Monitor-Validation.ps1 -OrchestrationId <Orchestration ID>
```

Alternatively, you can submit HTTP post request to the attestation service endpoint:

https://fpga-attestation.azurewebsites.net/api/ComputeFPGA_HttpGetStatus

The request body should contain your Subscription ID, Tenant ID, and orchestration ID of your attestation request:

```
{
  "OrchestrationId": "<orchestration ID>",
  "ClientSubscriptionId": "<your subscription ID>",
  "ClientTenantId": "<your tenant ID>"
}
```

After Validation has Completed

The service will write its output back to your container. If the validation pass succeeds, your container will have the original netlist file (abc.xclbin), a file with the bitstream (abc.**bit**.xclbin), a file that identifies the private location of your stored bitstream (abc.**azure**.xclbin) and 4 log files: one for the startup process (**abc-log.txt**) and one each for the 3 parallel phases that perform the validation. These are named ***-logPhaseX.txt** where X is a number for the phase. The **azure**.xclbin is used on your VM to signal the uploading of your validated image to the U250.

If validation failed, an **error-*.txt** file is written indicating which step failed. Also check the log files if the error log indicates that attestation failed. When contacting us for support, please be sure to include all these files as part of the support request along with the orchestration ID.

Using the Azure Portal

You can use the Azure Portal to create your container as well as uploading your netlist and downloading the bitstream and log files. Submitting an attestation request and monitoring its progress through the portal is not supported at this time and must be done through scripts as described above.

Reporting Issues

Send feedback to AzureFPGAFeedback@service.microsoft.com