logic fruit Technologies

# AES-256 Encryption Application

## *User Guide*

**UG101 (v1) August 12, 2021**


logic fruit
Technologies

# Revision History

The following table shows the revision history for this document.

| Section | Revision Summary |
|---|---|
| **12/08/2021 Version 1.0.1** | |
| | |
| | |
| | |
| | |

logic fruit Technologies

# Contents

# Introduction

# Chapter 1

## Application Description

Our AES-256 application processes plain data blocks, generates cipher data blocks which is made up of seemingly random characters using cipher keys of 256 bits (32 bytes). AES uses symmetric key encryption, which involves the use of only one secret key to cipher and deciphers the information. To safeguard data against unauthorized access our application supports a user-generated key (custom key) and a default key for encryption and decryption.

The AES-256 application performs FPGA accelerated AES-256 encryption on the data provided by the user in the form of an input file. The data is read by the application and pushed to the Alveo U200 card which performs the encryption. After the encryption, the user will get an output file with the encrypted data in it.

The application currently is supported to run on-premises Alveo U200 card.

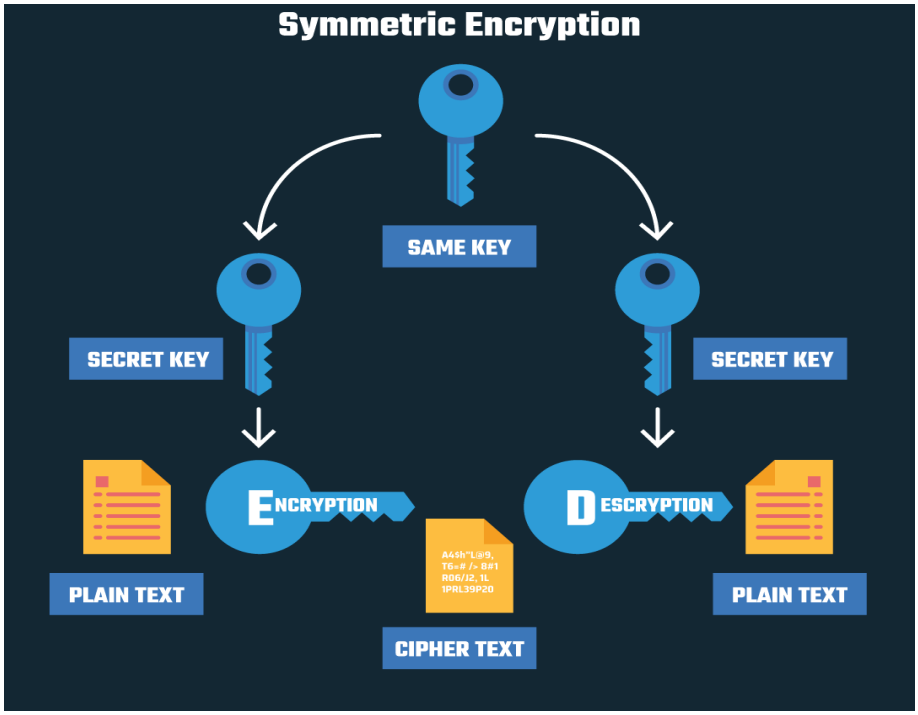



*Figure 1:* **AES-256 Symmetric Encryption**

# Features of AES-256 application

1) **256-bit Encryption Key:** The application uses a key of length 256 bits. Hence it makes the AES algorithm more robust against hacking.
2) **Default and Custom Keys:** Application supports both a default key and a custom key for encryption and decryption.
3) The default key is hidden inside the code and is used when user does not want to use or generate a custom key.
4) Custom key is a predefined 256 bit key provided by the user for encryption. Same key is required for the decryption.
5) **Key Validation:** Application validates the custom key before performing any encryption-decryption.
6) **Key Generation:** Application supports 256 bit custom key generation as well. If the user provides an empty file or a file that doesn't exists, then the application will generate a random 256 bits key, and write it into that file.

# Digital Rights Management

The AES 256 usage is protected and monitored through digital rights management (DRM) provided by Accelize. The DRM IP is part of the encoder binary running on the FPGA. A separate DRM application is provided with the encoder release package.

Use the following steps to subscribe and run DRM:

1. Install DRM from https://tech.accelize.com/documentation/stable/ drm_library_installation.html.
2. Create an account in the DRM portal: Xilinx App Store Portal.
3. Generate and save an access key (cred.json) file from the portal: https://portal.accelize.com/front/customer/apicredential
4. Subscribe to the AES  in the DRM portal.

For more information on the Accelize DRM IP, contact the Xilinx support team or the Accelize team.

Chapter 2

# System Requirement

The system requirements for running AES-256 app on Alveo™ U200 Data Center accelerator cards are listed below:

| Components | Output ( host machine ) |
|---|---|
| OS Version | Operating System: Ubuntu 18.04.5 LTS |
| Kernel Version | Kernel: Linux 5.4.0-73-generic |
| Vitis Version | Vitis v2020.2 (64-bit) |
| Driver Used | Xclmgmt , xocl |
| Specification about the card | Card type: u200<br>Flash type: SPI<br>Flashable partition running on FPGA:<br>xilinx_u200_xdma_201830_2,[ID=0x5d1211e8],[SC=4.2.0]<br>Flashable partitions installed in system:<br>xilinx_u200_xdma_201830_2,[ID=0x5d1211e8],[SC=4.2.0] |
| BSP Version | [0000:01:00.0] |
| Docker Version | 20.10.6 |

Chapter 3

# Application Running

## *Step-1: Obtain an Account Access Key*

An access key is required to authenticate a user and grant them access to the application based on their entitlements. To obtain your account access key, follow these steps:

1. Login to Xilinx App Store
2. Click the button labeled "Manage Account" to view entitlements.
3. Click the "Access Key" link on the left side menu
4. Click the "Create an Access Key" button.
5. Download the resulting file "cred.json" to the home location or recommended to in /tmp folder

## *Step-2: Host Setup*

This step involves installing Xilinx Runt Time (XRT) for Alveo U200. XRT host application is supported on Ubuntu 16.04 /18.04 and CentOS 7.x.

With sudo access, use the following command to download and run the setup script:

1. Clone GitHub Repository for Xilinx Base Runtime

   ***git clone https://github.com/Xilinx/Xilinx_Base_Runtime.git***

2. Go to the Xilinx Base Runtime

   ***cd Xilinx_base_Runtime***

3. Run the Host Setup Script

   ***./host_setup.sh -v 2020.2***

**Note:** *Please wait for the installation to complete. During this time, you may need to press [Y] to continue the host setup.*

4. If you choose to flash the FPGA, you will need to cold reboot the local machine after the installation is completed to load the new image on the FPGA. The script for host setup can be used to set up other versions of XRT and shell.
   Please check this link for more details: https://github.com/Xilinx/Xilinx_Base_Runtime

## *Step-3: Install Docker*

With sudo access, use the following command to run the utility script to install docker.

1. Go to Xilinx_Base_Runtime utilities directory

   ***cd Xilinx_Base_Runtime/utilities***

2. Run the Docker installation script

   ***./docker_install.sh***

3. To verify that the Docker has been properly installed you can run

   ***docker run hello-world***

4. If docker is already installed, enable it:

   **systemctl restart docker**

   **systemctl enable docker**

## *Step-4: Run Application*

Enter the following commands in a terminal window to run the application:

1. Setup Environment Variables by script from Xilinx_Base_Runtime

   ***source Xilinx_Base_Runtime/utilities/xilinx_docker_setup.sh***

2. Pull the Docker Image from Docker Hub

   ***docker pull hubxilinx/logicfruit_aes256_u200:latest***

3. Run this single command to run the Docker Image and the AES-256 app

   ***docker run --rm ***

   ***-v $(pwd)/cred.json:/cred.json ***

   ***-v $(pwd)/input.bin:/input.bin ***

   ***-v $(pwd)/output.bin:/output.bin ***

   ***$XILINX_DOCKER_DEVICES ***

   ***--shm-size=64G ***

***hubxilinx/logicfruit_aes256_u200:latest ***

***./xclbin/rtl_adder_pipes_hdk_4.2.1.0_vitis_2020.2_u200_xdma_201830_2.xclbin***

***input.bin output.bin***

4. Run this single command to run the AES-256 app using custom key:

***docker run --rm ***

***-v $(pwd)/cred.json:/cred.json ***

***-v $(pwd)/input.bin:/input.bin ***

***-v $(pwd)/output.bin:/output.bin ***

***-v $(pwd)/key.bin:/key.bin***

***$XILINX_DOCKER_DEVICES ***

***--shm-size=64G ***

***hubxilinx/logicfruit_aes256_u200:latest ***

***./xclbin/rtl_adder_pipes_hdk_4.2.1.0_vitis_2020.2_u200_xdma_201830_2.xclbin input.bin output.bin key.bin***

# Chapter 4

## Performance Figures

This is a performance evaluation between a standard AES-256 application in C  running on processor and this FPGA accelerated AES-256 app. The performance data is based only on the data encryption time taken on CPU and FPGA.

| Input Data Size | Avg. Encryption Time with FPGA (sec) | Avg. Encryption Time with CPU (sec) | Speed Improvement with FPGA (x Times) |
|---|---|---|---|
| 4KB | 0.00107254425 | 0.00335381 | 2.566565435 |
| 32 KB | 0.00130971 | 0.02946895 | 20.32131159 |
| 80 KB | 0.001677895 | 0.0403952 | 25.88165529 |
| 1MB | 0.0065421175 | 0.363072 | 66.47763511 |
| 10MB | 0.0521673 | 3.31045 | 79.48446632 |
| 100 MB | 0.0638202 | 3.69715 | 71.15443073 |
| 512 MB | 2.329245 | 168.707 | 90.55445005 |
| 1GB | 4.778735 | 532.2402165 | 120.8279349 |
| 10GB | 8.565035 | 1013.129095 | 123.5597472 |

**Test Setup Specs:**

- Processor: Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz, 8 cores
- RAM: 16 GB

# Chapter 5

# Troubleshooting

## *App hanging, crashing, or behaving abnormal*

If the AES application stops in between or works abnormally then, perform the following checklist:

- Check whether the FPGA is connected and in a working state.

```
source /opt/xilinx/xrt/setup.sh xbutil list
```

- Check whether the FPGA is connected and in a working state

  ***sudo lspci -vd 10ee:***

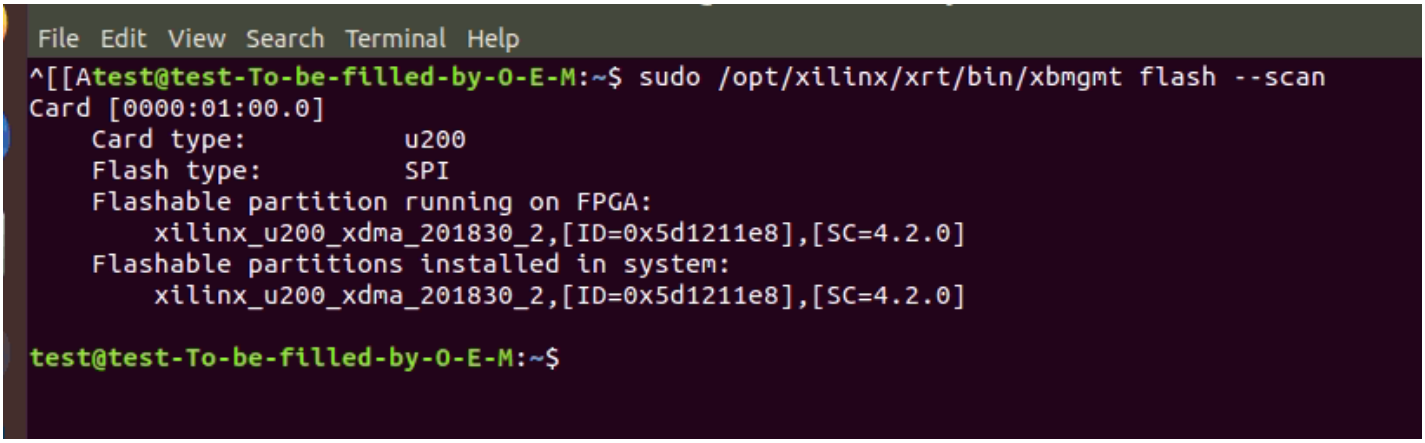
```
test@test-To-be-filled-by-O-E-M:~$ sudo lspci -vd 10ee:
[sudo] password for test:
01:00.0 Processing accelerators: Xilinx Corporation Device 5000
        Subsystem: Xilinx Corporation Device 000e
        Flags: bus master, fast devsel, latency 0
        Memory at f2000000 (64-bit, prefetchable) [size=32M]
        Memory at f4000000 (64-bit, prefetchable) [size=128K]
        Capabilities: [40] Power Management version 3
        Capabilities: [60] MSI-X: Enable+ Count=33 Masked-
        Capabilities: [70] Express Endpoint, MSI 00
        Capabilities: [100] Advanced Error Reporting
        Capabilities: [1c0] #19
        Capabilities: [400] Access Control Services
        Capabilities: [410] #15
        Kernel driver in use: xclmgmt
        Kernel modules: xclmgmt

01:00.1 Processing accelerators: Xilinx Corporation Device 5001
        Subsystem: Xilinx Corporation Device 000e
        Flags: bus master, fast devsel, latency 0, IRQ 16
        Memory at f0000000 (64-bit, prefetchable) [size=32M]
        Memory at f4020000 (64-bit, prefetchable) [size=64K]
        Memory at e0000000 (64-bit, prefetchable) [size=256M]
        Capabilities: [40] Power Management version 3
        Capabilities: [60] MSI-X: Enable+ Count=33 Masked-
        Capabilities: [70] Express Endpoint, MSI 00
        Capabilities: [100] Advanced Error Reporting
        Capabilities: [400] Access Control Services
        Capabilities: [410] #15
        Kernel driver in use: xocl
        Kernel modules: xocl
```

  This should be the required output if not please do reinstall check the Troubleshooting page.

- We can review the shell capabilities with by command as shown below

***sudo /opt/Xilinx/xrt/bin/xbmgmt flash –scan***




- Unload/reload XRT drivers:

  Use modprobe -r to remove the drivers as shown below

  **sudo modprobe -r xocl**

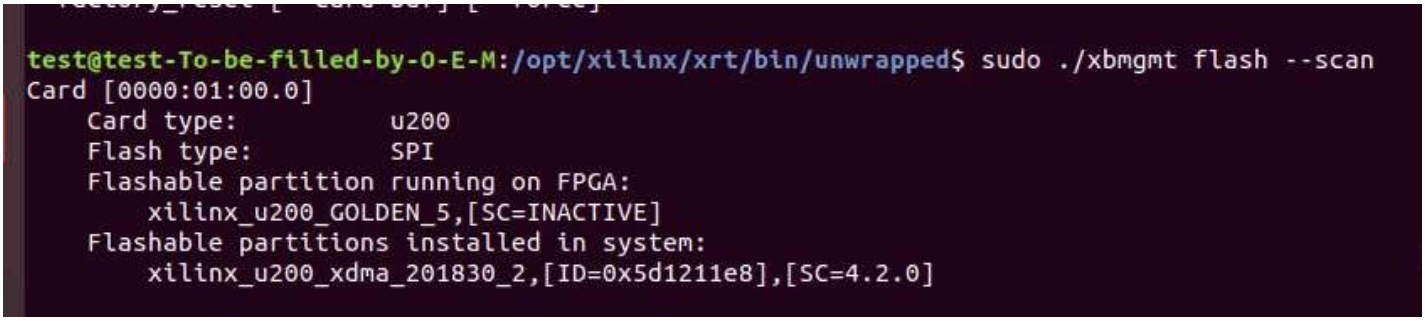
  **sudo modprobe -r xclmgmt**

  Use modprobe to reload the drivers as shown below

  **sudo modprobe xclmgmtsudo modprobe xocl**

  Order matters for both of these commands. xocl depends on xclmgmt.

- Flash the card with a deployment platform:




Once the card is up and running in the system, a deployment platform will need to be flashed onto the card before xbutil validate passes and applications can be run. To flash the card with a deployment platform follow the below steps:

- Run sudo xbmgmt flash --scan

- If Flashable partitions installed in the system: (None) is the output please install the latest packages from the Alveo landing page for your installed card(s)
- Follow the process for Card install to install the platforms on the machine.
- Run sudo xbmgmt flash --update --shell <xilinx_uxx> to flash the platform onto the card. This command should be provided during platform installation, shown below:

| |
|---|
| Partition package installed successfully. |
| Please flash card manually by running below command: |
| sudo /opt/xilinx/xrt/bin/xbmgmt flash --update --shell xilinx_u200_xdma_201830_2 |
| ~]$ sudo xbmgmt flash --update --shell xilinx_u200_xdma_201830_2 |
| Status: shell needs updating |
| Current shell: xilinx_u200_GOLDEN_9 |
| Shell to be flashed: xilinx_u200_xdma_201830_2 |
| Are you sure you wish to proceed? [y/n]: y |
| Updating shell on card[0000:05:00.0] |
| INFO: ***Found 353 ELA Records |
| Enabled bitstream guard. Bitstream will not be loaded until flashing is finished. |
| Preparing flash chip 0 |
| Erasing flash..... |
| Programming flash..... |
| Cleared the bitstream guard. Bitstream now active. |
| Successfully flashed Card[0000:05:00.0] |
| 1 Card(s) flashed successfully. |
| Cold reboot machine to load the new image on card(s). |

- Cold boot the server
- Run sudo xbmgmt flash --scan
- Now platform installed in host and card are the same
- If this is a DFX-2RP platform, go to Programming DFX-2RP shell partitions
- If there is a different number in the SC= line between the FPGA and the system for the platform on the card, update the SC firmware, example below:

| |
|---|
| :~> sudo xbmgmt flash --update |
| Status: SC needs updating |
| Current SC: 5.0.20 |
| SC to be flashed: 5.0.27 |
| Updating SC firmware on card[0000:05:00.0] |
| Stopping user function... |
| INFO: found 4 sections |
| ..... |
| INFO: Loading new firmware on SC |
| Successfully flashed Card[0000:05:00.0] |

# *Reverting the card to factory image:*

The Alveo card can be reverted to the factory image, also known as golden. This requires that XRT release 2019.2 or later is installed on the same system as the Alveo accelerator card. The steps to revert the card using this method are listed below.

1. Open a terminal window.
2. Run the following command, where card_bdf is the BDF of the card to revert to golden.

*$ sudo xbmgmt flash --factory_reset --card <card_bdf>*

3. Enter y to continue. The following message is displayed on completion.

*Shell is reset successfully*

*Cold reboot machine to load new shell on card*

4. Cold boot the system so the card FPGA uses the new image.
5. Confirm the card has been reverted to factory image by running the following command.

*$ sudo xbmgmt flash --scan*

6. An output similar to the following is displayed.

| |
|---|
| Card [0000:65:00.0] |
| Card type: uxx |
| Flash type: SPI |
| Flashable partition running on FPGA: |
| xilinx_uxx_GOLDEN_x,[SC=x.x] |
| Flashable partitions installed **in** system: (**None**) |

In this output, under the Flashable partition running on an FPGA, note GOLDEN in the name. This indicates that the card has successfully been reverted to the factory image.

IMPORTANT! If the GOLDEN_2 image is running on the FPGA, carefully review the design advisory for Alveo data center Accelerator card golden corruption, found in AR 71915. Complete the repair instructions associated with the Xilinx Answer before proceeding.

# Chapter 6

## Important Commands

The important command which the user can use for checking if the FPGA is running fine. Through these commands, we can validate the card, flash the card, check its thermal temperature, and other useful information about the FPGA card.

| Commands | Usage |
|---|---|
| xbutil scan | List the card available on the devices, the XRT information & the system configurations |
| xbutil validate | This command will check the proper functioning of the FPGA on your host machine and even will flash the FPGA card on the host machine |
| xbutil query | This command will show all the FPGA card's important information, such as card temperature, card memory, and power supplied to the card. |
| xbutil reset | To reset the XRT<br>NOTE: this command should only be run when the XRT is not working or the machine is not able to access the device present on the host machine or the application is stuck |

# References

These documents provide supplemental material useful with this guide:

1. Programming the Host Application
2. AES_Documentation
3. Debug_Page
4. Debugging of XRT

# Additional Resources and Legal Notices

## Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see Xilinx Support.

## Documentation Navigator and Design Hubs

Xilinx® Documentation Navigator (DocNav) provides access to Xilinx documents, videos, and support resources, which you can filter and search to find information. To open DocNav:

- From the Vivado® IDE, select **Help → Documentation and Tutorials**.
- On Windows, select **Start → All Programs → Xilinx Design Tools → DocNav**.
- At the Linux command prompt, enter docnav.

Xilinx Design Hubs provide links to documentation organized by design tasks and other topics, which you can use to learn key concepts and address frequently asked questions. To access the Design Hubs:

- In DocNav, click the **Design Hubs View** tab.
- On the Xilinx website, see the Design Hubs page.

***Note*:** For more information on DocNav, see the Documentation Navigator page on the Xilinx website.

## Please Read: Important Legal Notices

Sale which can be viewed at https:// www.xilinx.com/legal.htm#tos; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in such critical applications, please refer to Xilinx's Terms of Sale which can be viewed at https://www.xilinx.com/legal.htm#tos.

**AUTOMOTIVE APPLICATIONS DISCLAIMER**

AUTOMOTIVE PRODUCTS (IDENTIFIED AS "XA" IN THE PART NUMBER) ARE NOT WARRANTED FOR USE IN THE DEPLOYMENT OF AIRBAGS OR FOR USE IN APPLICATIONS THAT AFFECT CONTROL OF A VEHICLE ("SAFETY APPLICATION") UNLESS THERE IS A SAFETY CONCEPT OR REDUNDANCY FEATURE CONSISTENT WITH THE ISO 26262 AUTOMOTIVE SAFETY STANDARD ("SAFETY DESIGN"). CUSTOMERS HALL, BEFORE USING OR DISTRIBUTING ANY SYSTEMS THAT INCORPORATE PRODUCTS, THOROUGHLY TEST SUCH SYSTEMS FOR SAFETY PURPOSES. USE OF PRODUCTS IN A SAFETY APPLICATION WITHOUT A SAFETY DESIGN IS FULLY AT THE RISK OF THE CUSTOMER, SUBJECT ONLY TO APPLICABLE LAWS AND REGULATIONS GOVERNING LIMITATIONS ON PRODUCT LIABILITY.

**Copyright**