DATA SECURITY IN IoT DEVICES using Key Lifecycle Management

Presented By:
Alisha Joshi
Kaveesha Shah
Laxmi Garde
Pratima Lunkad

Sponsored by:

Internal Guide:
Prof. Saurabh Mengale

Agenda

- Introduction
- Background
- Problem Statement
- High Level Design
- Technologies used
- System modules
- Implementation Details
- Applications
- Future Scope
- Conclusion

IoT – The concept

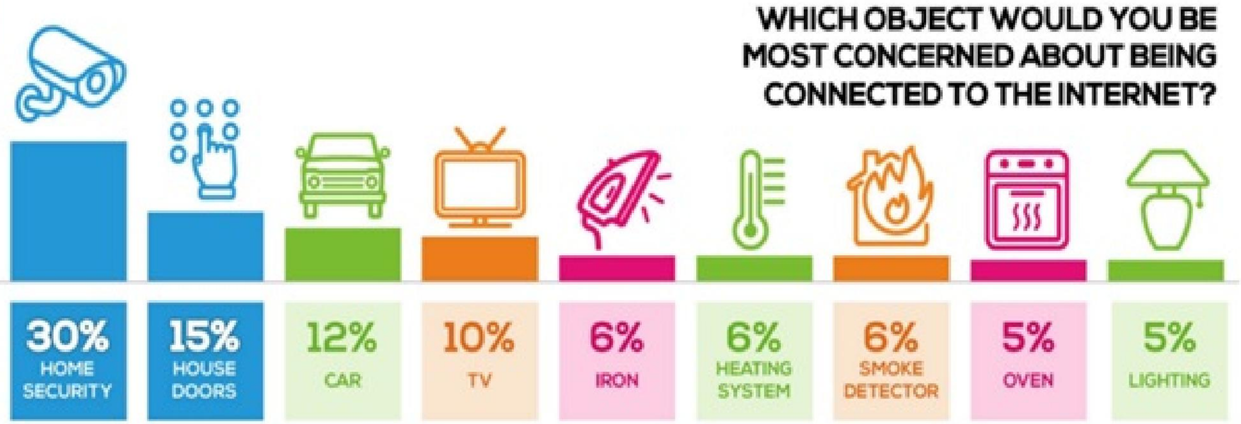
- Internet of Things

  Everything on the planet will become things on internet via n/w and the nodes/sensors in that n/w will be able to transfer data over that n/w.

- If understood and secured, IoT will enhance communications, lifestyle and delivery of services.

Background

With increase in number of IoT devices and their proximity to human life, security and privacy of these devices is of great importance.



Existing IoT security mechanisms and their drawbacks

- Current systems use only SSL as a security measure.

- Privacy concern: personal information collected is stored as plaintext.

- If the device itself is stolen, information is exposed and the security fails before even reaching the SSL level. Hence physical security is another major concern.

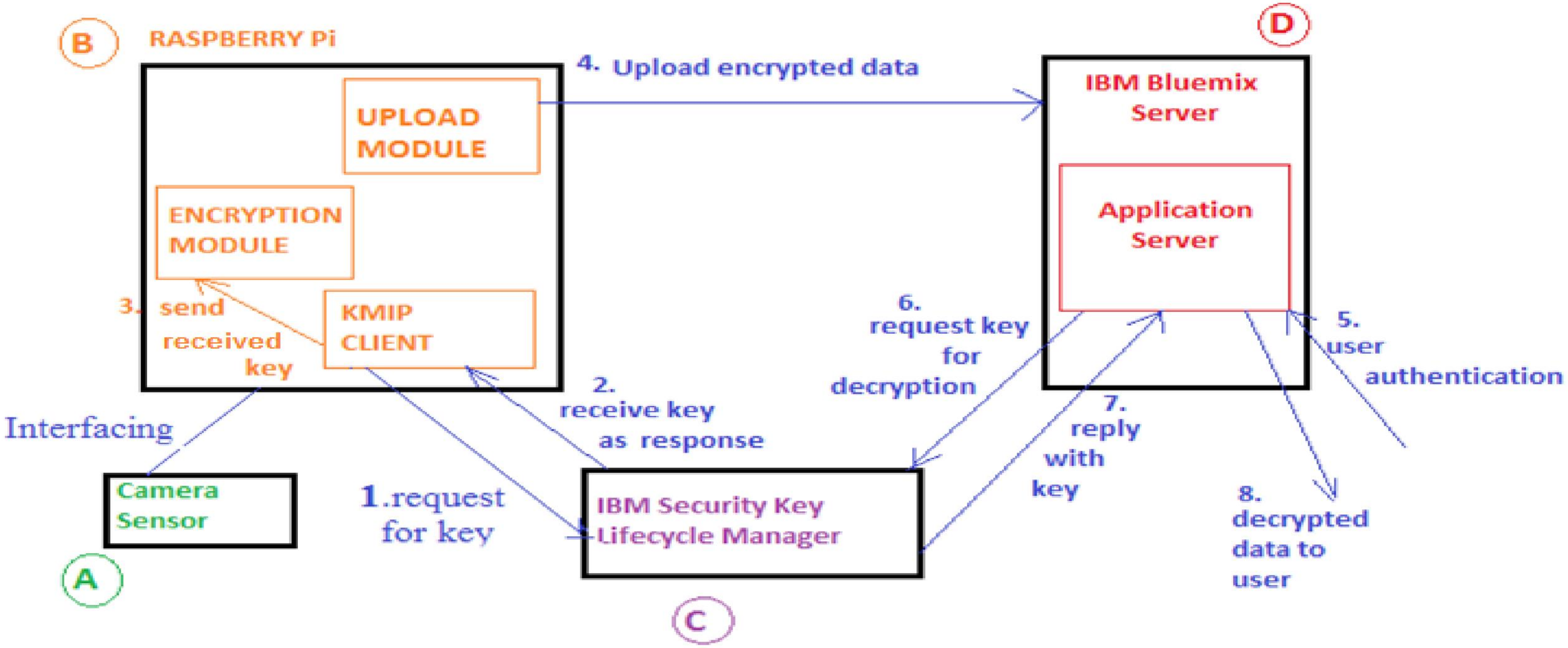
- Encryption of data using key stored on the device itself is a big drawback because if the key is compromised or availed by the hacker, the data is at risk.

Problem Statement

- A system that ensures data security on IoT devices where
  - ***sensor data is encrypted*** before being exposed in the network
  - security of keys and data is reinforced by using key lifecycle management.

- To ensure safety of camera sensor data by encrypting it using the keys which are managed by the IBM Security Key Lifecycle Manager and uploading the encrypted data on the Bluemix cloud service to ensure safe delivery of data only to the authenticated user.

High Level Design



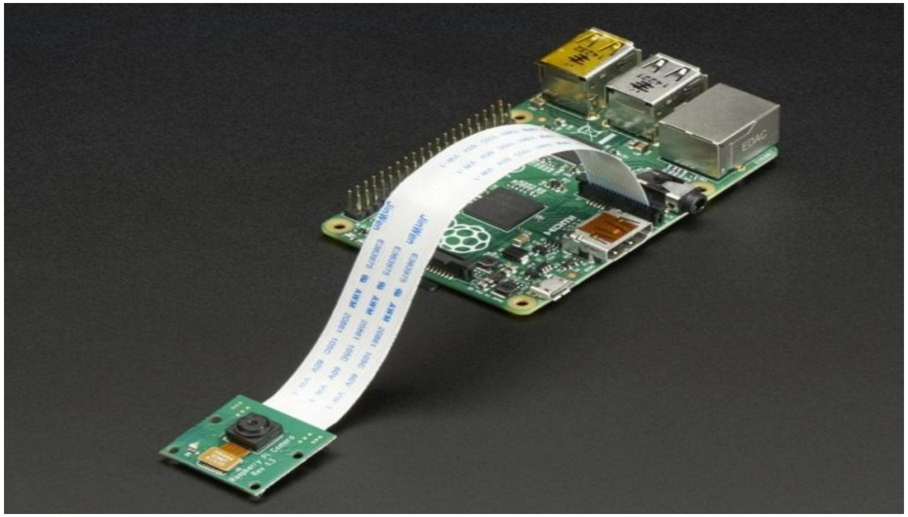
Technologies used

Raspberry Pi

- single board computer with 4x ARM Cortex A53 processor
- has an OS to support it and a variety of peripherals

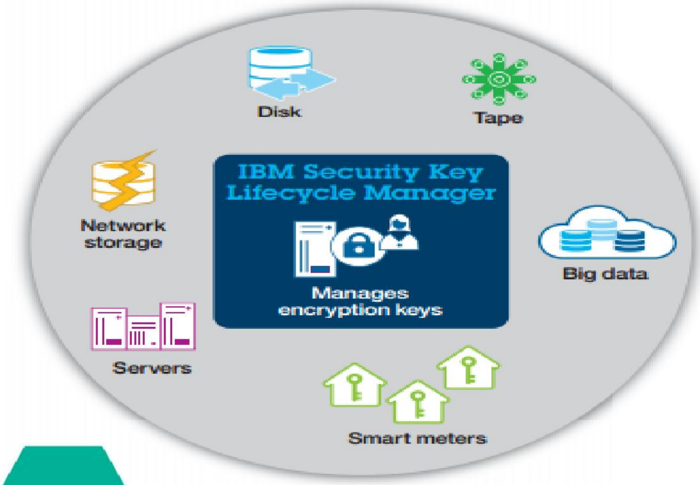
Camera sensor

- captures HD videos(H.264 format) and still photographs
- 15cm ribbon cable to the CSI port on the Raspberry Pi.



IBM Security Key Lifecycle Manager

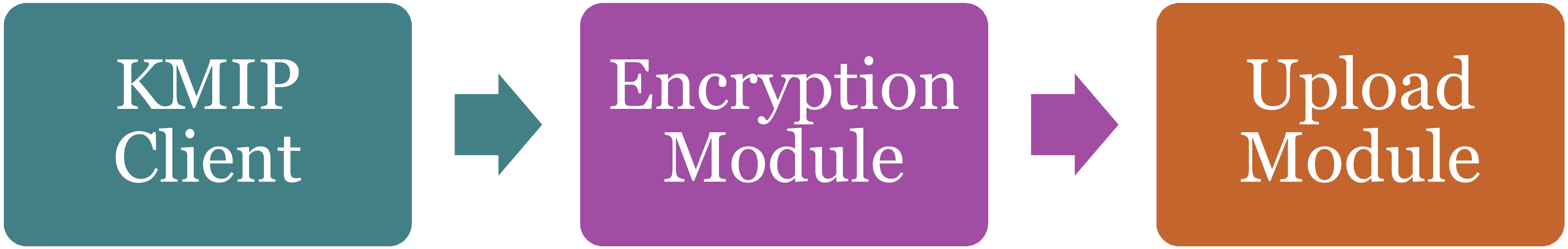
- simplifies, centralizes and automates the encryption-key management process[4].
- serves keys at the time of use and allows centralized storage of key in a secure location.



IBM Bluemix

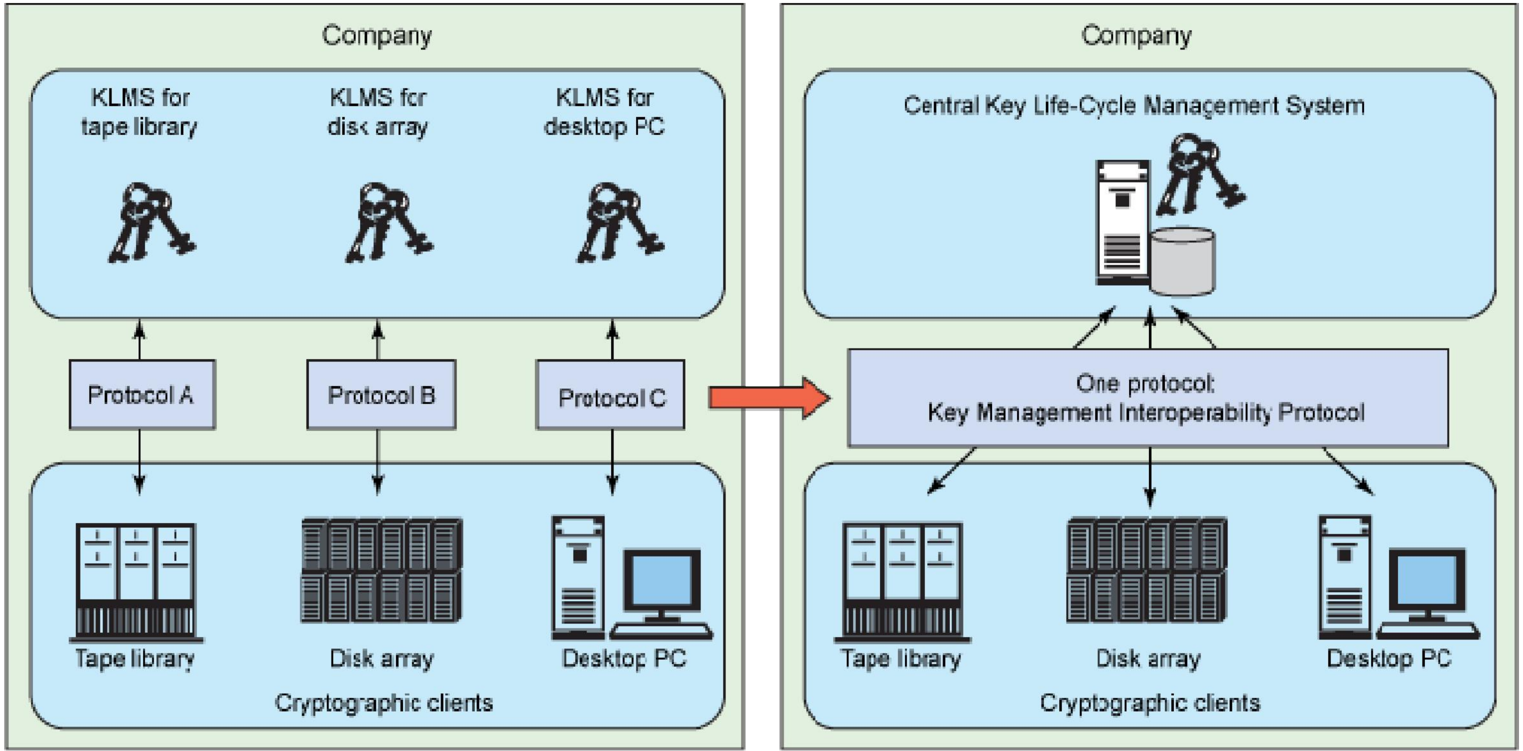
- Platform as a service
- Used to host application server

System Modules

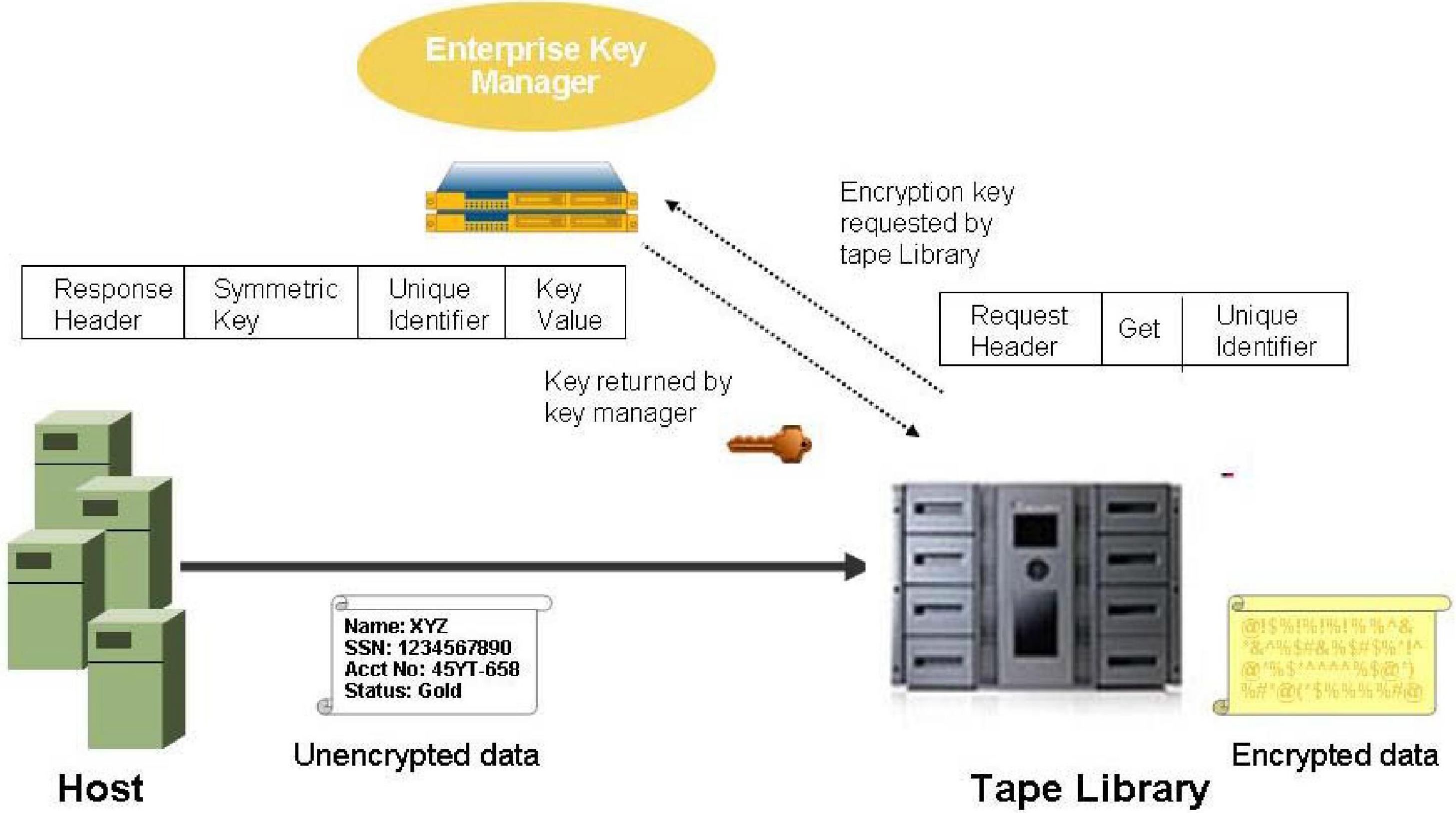


KMIP Client

- KMIP client is used to access cryptographic objects from KLMS using KMIP protocol
- Key Management Interoperability Protocol(KMIP) is a standard protocol for communication between cryptographic clients that need the keys to KLMS server [3].



KMIP Request response mechanism

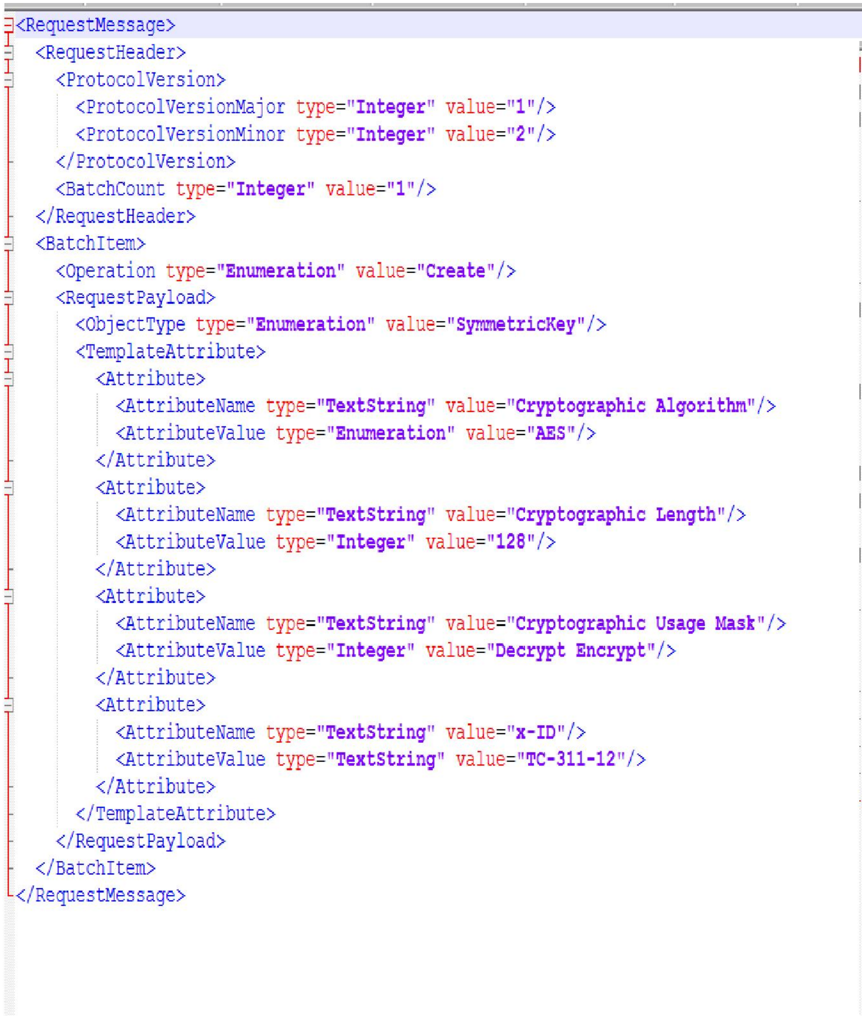


Response Header	Symmetric Key	Unique Identifier	Key Value

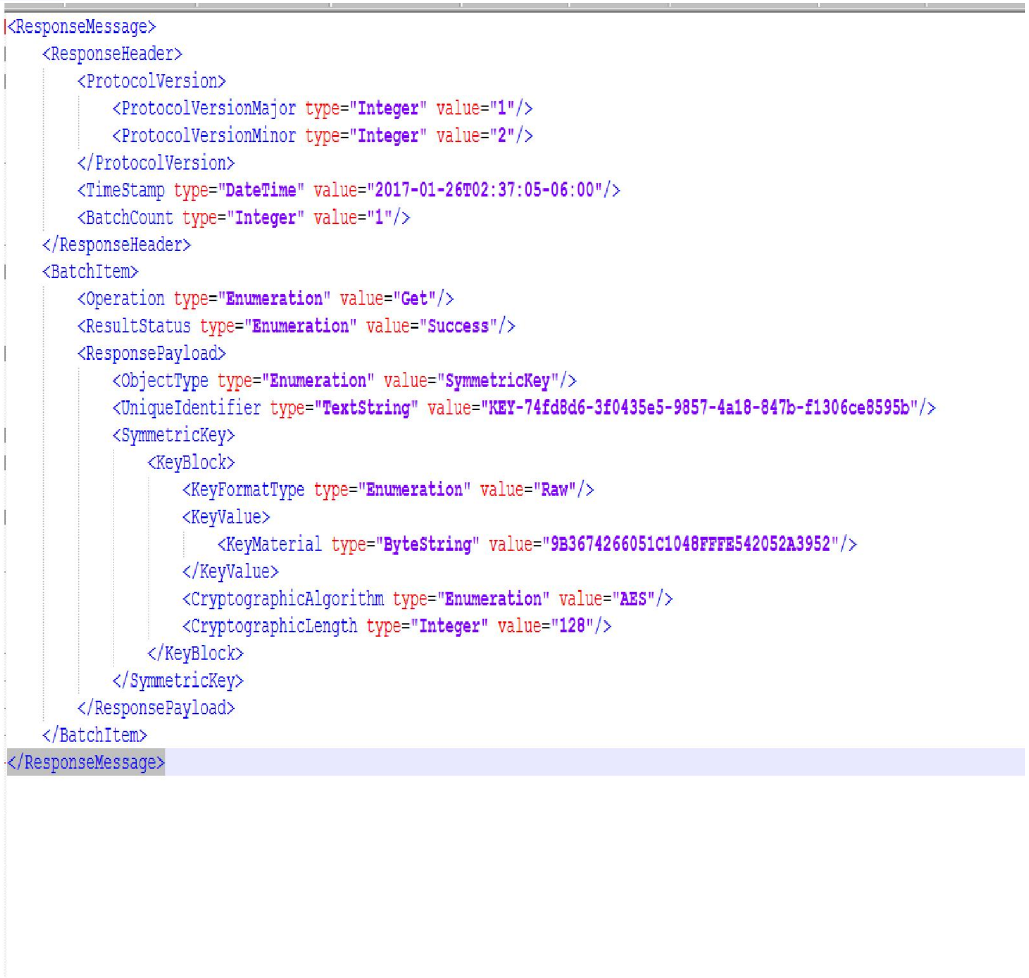
Request Header	Get	Unique Identifier

Name: XYZ
SSN: 1234567890
Acct No: 45YT-658
Status: Gold

KMIP Request – Response headers



```
<RequestMessage>
  <RequestHeader>
    <ProtocolVersion>
      <ProtocolVersionMajor type="Integer" value="1"/>
      <ProtocolVersionMinor type="Integer" value="2"/>
    </ProtocolVersion>
    <BatchCount type="Integer" value="1"/>
  </RequestHeader>
  <BatchItem>
    <Operation type="Enumeration" value="Create"/>
    <RequestPayload>
      <ObjectType type="Enumeration" value="SymmetricKey"/>
      <TemplateAttribute>
        <Attribute>
          <AttributeName type="TextString" value="Cryptographic Algorithm"/>
          <AttributeValue type="Enumeration" value="AES"/>
        </Attribute>
        <Attribute>
          <AttributeName type="TextString" value="Cryptographic Length"/>
          <AttributeValue type="Integer" value="128"/>
        </Attribute>
        <Attribute>
          <AttributeName type="TextString" value="Cryptographic Usage Mask"/>
          <AttributeValue type="Integer" value="Decrypt Encrypt"/>
        </Attribute>
        <Attribute>
          <AttributeName type="TextString" value="x-ID"/>
          <AttributeValue type="TextString" value="TC-311-12"/>
        </Attribute>
      </TemplateAttribute>
    </RequestPayload>
  </BatchItem>
</RequestMessage>
```



```
<ResponseMessage>
    <ResponseHeader>
        <ProtocolVersion>
            <ProtocolVersionMajor type="Integer" value="1"/>
            <ProtocolVersionMinor type="Integer" value="2"/>
        </ProtocolVersion>
        <TimeStamp type="DateTime" value="2017-01-26T02:37:05-06:00"/>
        <BatchCount type="Integer" value="1"/>
    </ResponseHeader>
    <BatchItem>
        <Operation type="Enumeration" value="Get"/>
        <ResultStatus type="Enumeration" value="Success"/>
        <ResponsePayload>
            <ObjectType type="Enumeration" value="SymmetricKey"/>
            <UniqueIdentifier type="TextString" value="KEY-74fd8d6-3f0435e5-9857-4a18-847b-f1306ce8595b"/>
            <SymmetricKey>
                <KeyBlock>
                    <KeyFormatType type="Enumeration" value="Raw"/>
                    <KeyValue>
                        <KeyMaterial type="ByteString" value="9B3674266051C1048FFFE542052A3952"/>
                    </KeyValue>
                    <CryptographicAlgorithm type="Enumeration" value="AES"/>
                    <CryptographicLength type="Integer" value="128"/>
                </KeyBlock>
            </SymmetricKey>
        </ResponsePayload>
    </BatchItem>
</ResponseMessage>
```

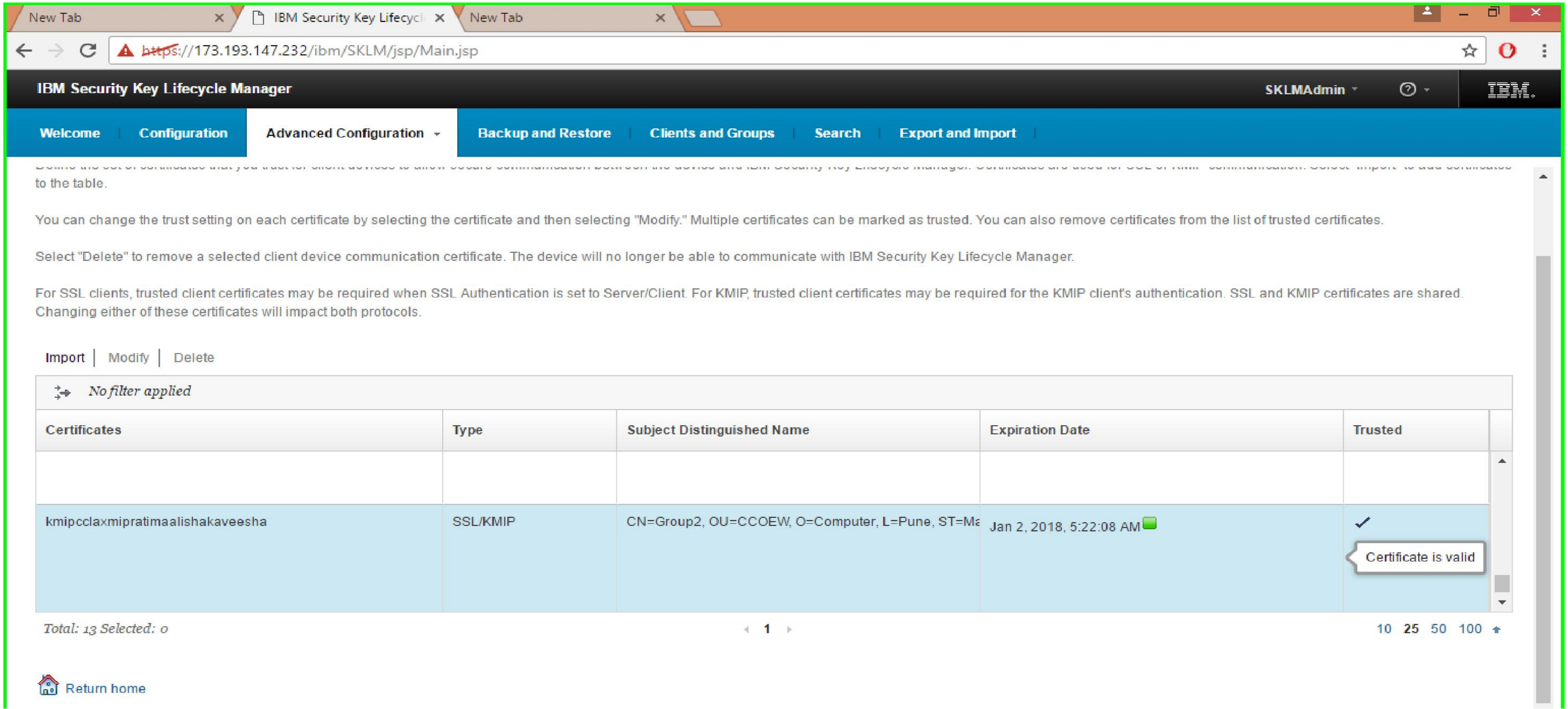
Implementation details

- Client Side : Raspberry Pi

1. Creating keyStore and trustStore on Raspberry Pi
2. Script for booting Pi as a camera and start recording
3. Encrypt the recorded video using key fetched from IBM Security Key Lifecycle Management server via KMIP
4. Upload encrypted data on FTP server

Initial Setup

- Setting up IoT device (Raspberry Pi + camera module)
- Setting up keystore for SSL handshake



to the table.

You can change the trust setting on each certificate by selecting the certificate and then selecting "Modify." Multiple certificates can be marked as trusted. You can also remove certificates from the list of trusted certificates.

Select "Delete" to remove a selected client device communication certificate. The device will no longer be able to communicate with IBM Security Key Lifecycle Manager.

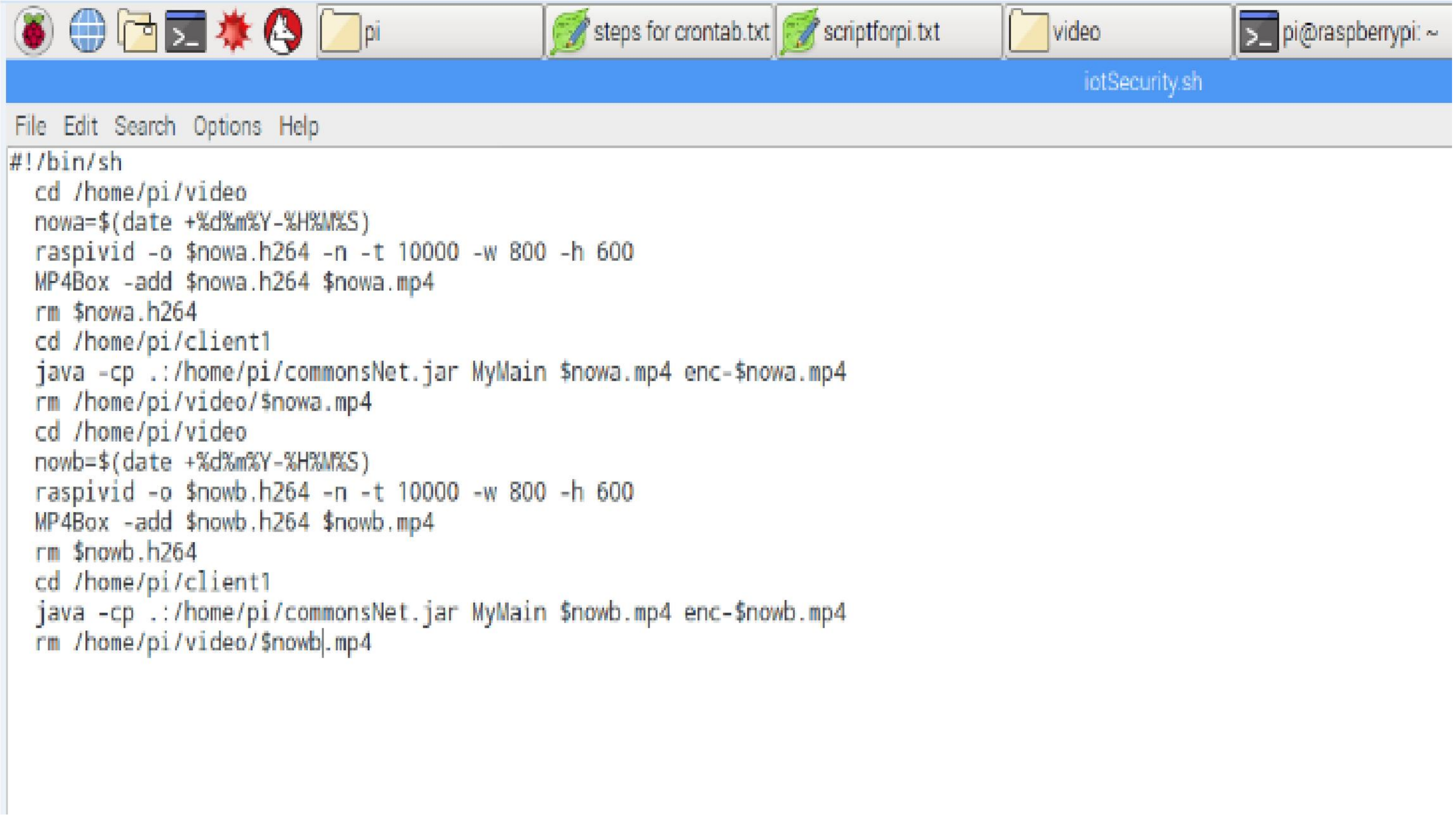
For SSL clients, trusted client certificates may be required when SSL Authentication is set to Server/Client. For KMIP, trusted client certificates may be required for the KMIP client's authentication. SSL and KMIP certificates are shared. Changing either of these certificates will impact both protocols.

No filter applied

Certificates	Type	Subject Distinguished Name	Expiration Date	Trusted
kmipcclaxmipratimaalishakaveesha	SSL/KMIP	CN=Group2, OU=CCOEW, O=Computer, L=Pune, ST=Ma	Jan 2, 2018, 5:22:08 AM 🟩	✓ Certificate is valid

Total: 13 Selected: 0

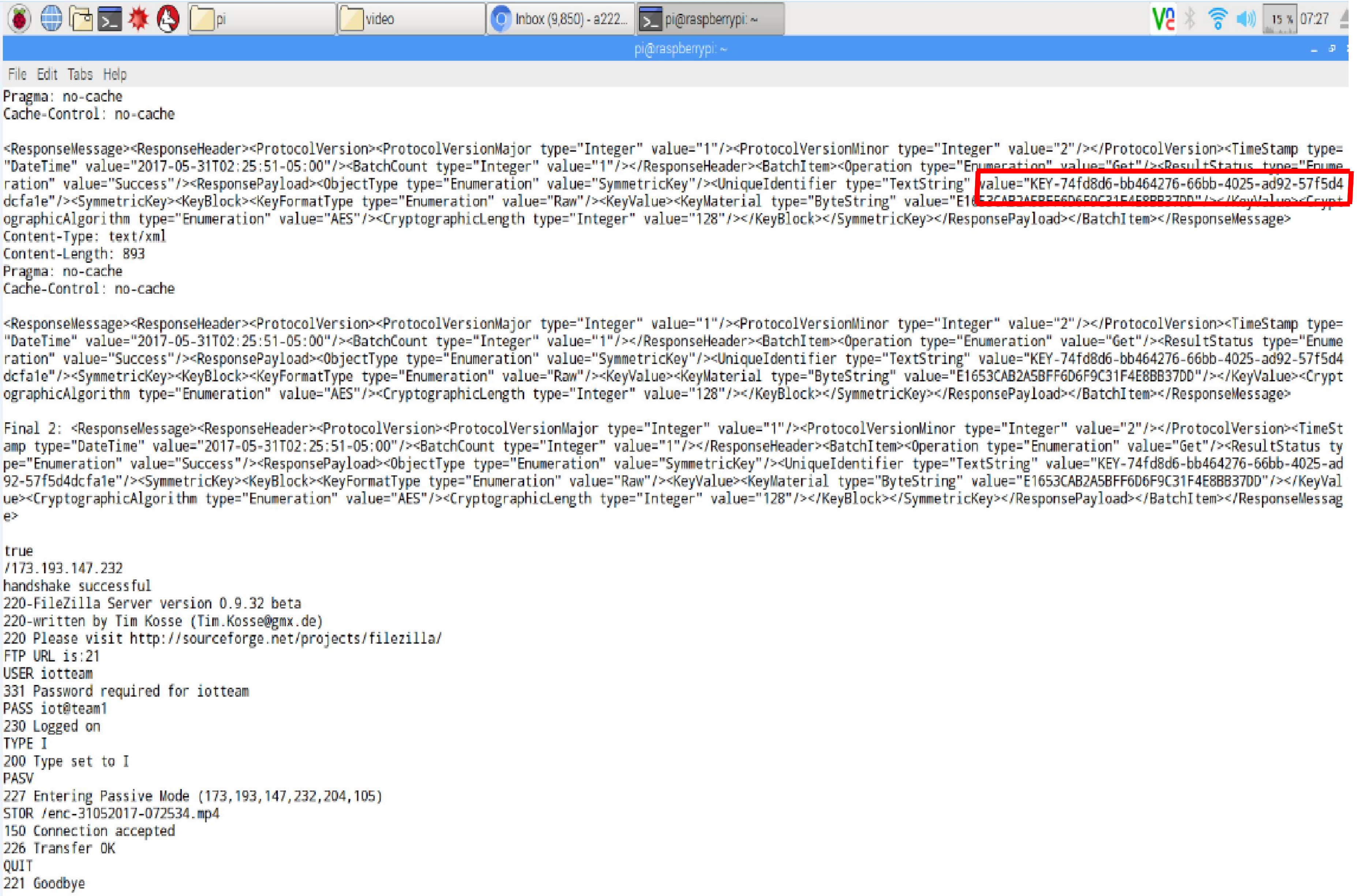
- Booting Pi as a camera and recording video continuously



```
#!/bin/sh
  cd /home/pi/video
  nowa=$(date +%d%m%Y-%H%M%S)
  raspivid -o $nowa.h264 -n -t 10000 -w 800 -h 600
  MP4Box -add $nowa.h264 $nowa.mp4
  rm $nowa.h264
  cd /home/pi/client1
  java -cp .:/home/pi/commonsNet.jar MyMain $nowa.mp4 enc-$nowa.mp4
  rm /home/pi/video/$nowa.mp4
  cd /home/pi/video
  nowb=$(date +%d%m%Y-%H%M%S)
  raspivid -o $nowb.h264 -n -t 10000 -w 800 -h 600
  MP4Box -add $nowb.h264 $nowb.mp4
  rm $nowb.h264
  cd /home/pi/client1
  java -cp .:/home/pi/commonsNet.jar MyMain $nowb.mp4 enc-$nowb.mp4
  rm /home/pi/video/$nowb.mp4
```

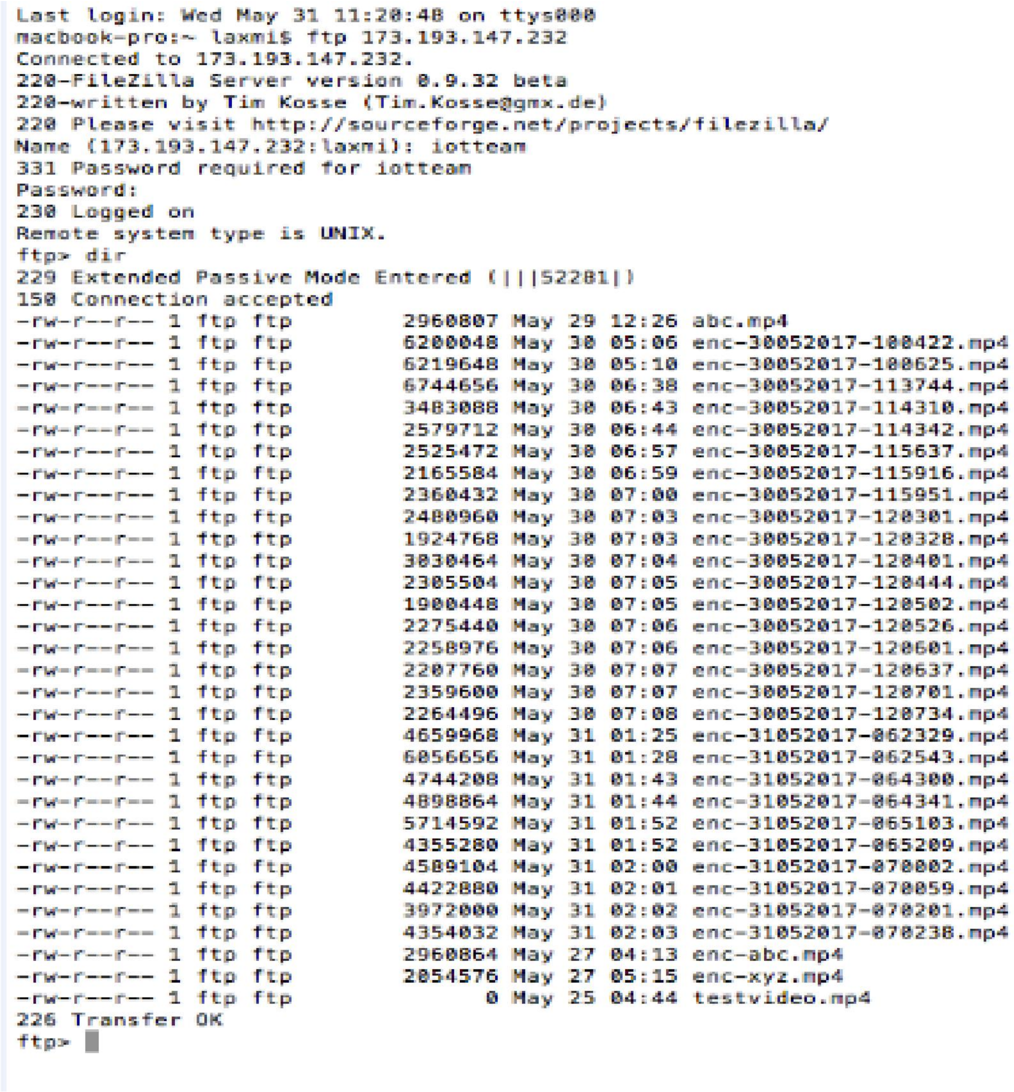
Script for continuous recording

- KMIP Client to create and get keys for encryption from IBM Security Key Lifecycle Management server. Server returns UUID of key
- Encrypt the recorded video using Cipher operations
- Append the UUID of key with the encrypted bytes for uploading data as a file
- Upload the encrypted file on FTP server



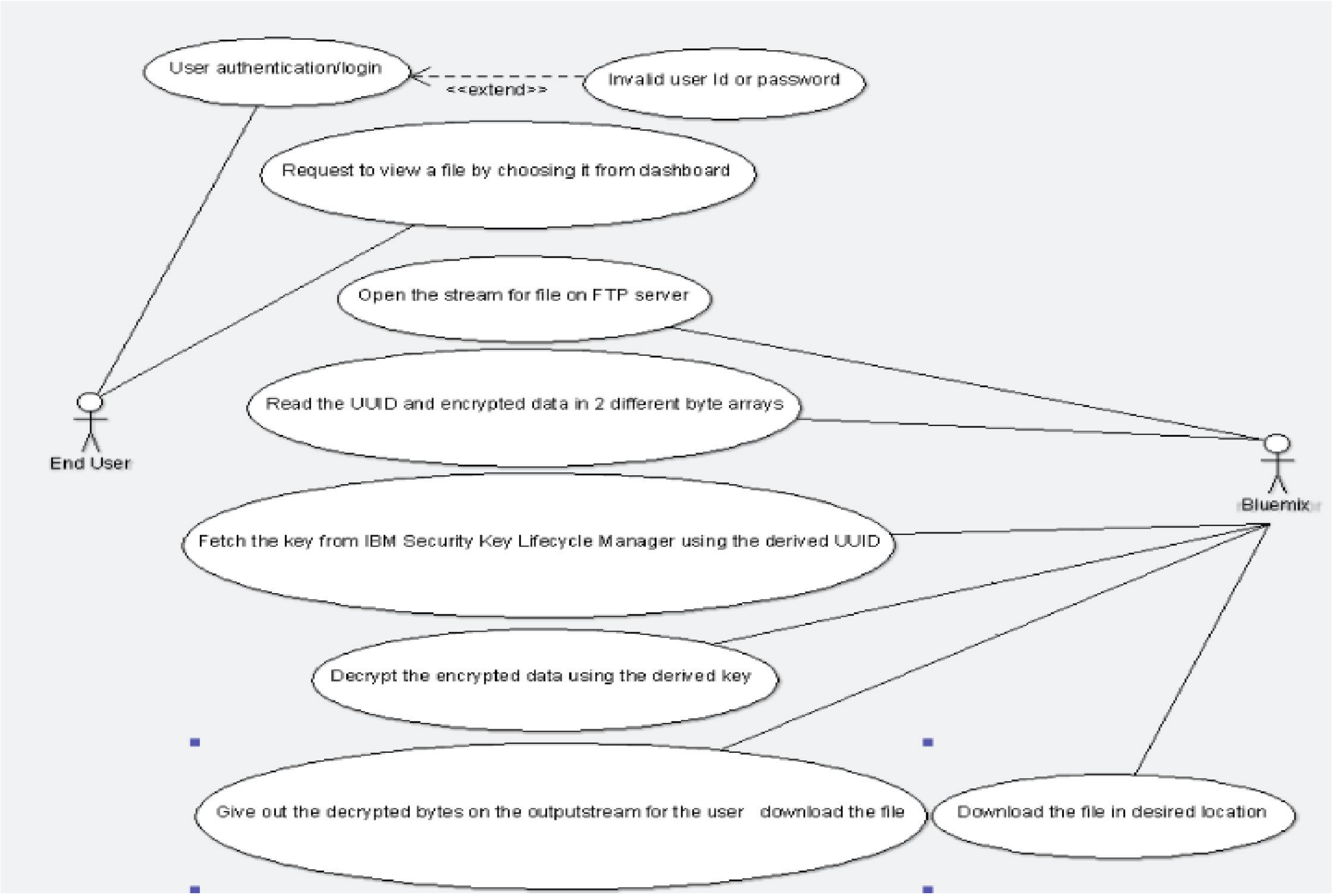
```
Pragma: no-cache
Cache-Control: no-cache

<ResponseMessage><ResponseHeader><ProtocolVersion><ProtocolVersionMajor type="Integer" value="1"/><ProtocolVersionMinor type="Integer" value="2"/></ProtocolVersion><TimeStamp type=
"DateTime" value="2017-05-31T02:25:51-05:00"/><BatchCount type="Integer" value="1"/></ResponseHeader><BatchItem><Operation type="Enumeration" value="Get"/><ResultStatus type="Enume
ration" value="Success"/><ResponsePayload><ObjectType type="Enumeration" value="SymmetricKey"/><UniqueIdentifier type="TextString" value="KEY-74fd8d6-bb464276-66bb-4025-ad92-57f5d4
dcfa1e"/><SymmetricKey><KeyBlock><KeyFormatType type="Enumeration" value="Raw"/><KeyValue><KeyMaterial type="ByteString" value="E1653CAB2A5BFF6D6F9C31F4E8BB37DD"/></KeyValue><Crypt
ographicAlgorithm type="Enumeration" value="AES"/><CryptographicLength type="Integer" value="128"/></KeyBlock></SymmetricKey></ResponsePayload></BatchItem></ResponseMessage>
Content-Type: text/xml
Content-Length: 893
Pragma: no-cache
Cache-Control: no-cache

<ResponseMessage><ResponseHeader><ProtocolVersion><ProtocolVersionMajor type="Integer" value="1"/><ProtocolVersionMinor type="Integer" value="2"/></ProtocolVersion><TimeStamp type=
"DateTime" value="2017-05-31T02:25:51-05:00"/><BatchCount type="Integer" value="1"/></ResponseHeader><BatchItem><Operation type="Enumeration" value="Get"/><ResultStatus type="Enume
ration" value="Success"/><ResponsePayload><ObjectType type="Enumeration" value="SymmetricKey"/><UniqueIdentifier type="TextString" value="KEY-74fd8d6-bb464276-66bb-4025-ad92-57f5d4
dcfa1e"/><SymmetricKey><KeyBlock><KeyFormatType type="Enumeration" value="Raw"/><KeyValue><KeyMaterial type="ByteString" value="E1653CAB2A5BFF6D6F9C31F4E8BB37DD"/></KeyValue><Crypt
ographicAlgorithm type="Enumeration" value="AES"/><CryptographicLength type="Integer" value="128"/></KeyBlock></SymmetricKey></ResponsePayload></BatchItem></ResponseMessage>

Final 2: <ResponseMessage><ResponseHeader><ProtocolVersion><ProtocolVersionMajor type="Integer" value="1"/><ProtocolVersionMinor type="Integer" value="2"/></ProtocolVersion><TimeSt
amp type="DateTime" value="2017-05-31T02:25:51-05:00"/><BatchCount type="Integer" value="1"/></ResponseHeader><BatchItem><Operation type="Enumeration" value="Get"/><ResultStatus ty
pe="Enumeration" value="Success"/><ResponsePayload><ObjectType type="Enumeration" value="SymmetricKey"/><UniqueIdentifier type="TextString" value="KEY-74fd8d6-bb464276-66bb-4025-ad
92-57f5d4dcfa1e"/><SymmetricKey><KeyBlock><KeyFormatType type="Enumeration" value="Raw"/><KeyValue><KeyMaterial type="ByteString" value="E1653CAB2A5BFF6D6F9C31F4E8BB37DD"/></KeyVal
ue><CryptographicAlgorithm type="Enumeration" value="AES"/><CryptographicLength type="Integer" value="128"/></KeyBlock></SymmetricKey></ResponsePayload></BatchItem></ResponseMessag
e>

true
/173.193.147.232
handshake successful
220-FileZilla Server version 0.9.32 beta
220-written by Tim Kosse (Tim.Kosse@gmx.de)
220 Please visit http://sourceforge.net/projects/filezilla/
FTP URL is:21
USER iotteam
331 Password required for iotteam
PASS iot@team1
230 Logged on
TYPE I
200 Type set to I
PASV
227 Entering Passive Mode (173,193,147,232,204,105)
STOR /enc-31052017-072534.mp4
150 Connection accepted
226 Transfer OK
QUIT
221 Goodbye
```

- List of videos on FTP server



```
Last login: Wed May 31 11:20:48 on ttys000
macbook-pro:~ laxmi$ ftp 173.193.147.232
Connected to 173.193.147.232.
220-FileZilla Server version 0.9.32 beta
220-written by Tim Kosse (Tim.Kosse@gmx.de)
220 Please visit http://sourceforge.net/projects/filezilla/
Name (173.193.147.232:laxmi): iotteam
331 Password required for iotteam
Password:
230 Logged on
Remote system type is UNIX.
ftp> dir
229 Extended Passive Mode Entered (|||52281|)
150 Connection accepted
-rw-r--r-- 1 ftp ftp        2960807 May 29 12:26 abc.mp4
-rw-r--r-- 1 ftp ftp        6200048 May 30 05:06 enc-30052017-100422.mp4
-rw-r--r-- 1 ftp ftp        6219648 May 30 05:10 enc-30052017-100625.mp4
-rw-r--r-- 1 ftp ftp        6744656 May 30 06:38 enc-30052017-113744.mp4
-rw-r--r-- 1 ftp ftp        3483088 May 30 06:43 enc-30052017-114310.mp4
-rw-r--r-- 1 ftp ftp        2579712 May 30 06:44 enc-30052017-114342.mp4
-rw-r--r-- 1 ftp ftp        2525472 May 30 06:57 enc-30052017-115637.mp4
-rw-r--r-- 1 ftp ftp        2165584 May 30 06:59 enc-30052017-115916.mp4
-rw-r--r-- 1 ftp ftp        2360432 May 30 07:00 enc-30052017-115951.mp4
-rw-r--r-- 1 ftp ftp        2480960 May 30 07:03 enc-30052017-120301.mp4
-rw-r--r-- 1 ftp ftp        1924768 May 30 07:03 enc-30052017-120328.mp4
-rw-r--r-- 1 ftp ftp        3030464 May 30 07:04 enc-30052017-120401.mp4
-rw-r--r-- 1 ftp ftp        2305504 May 30 07:05 enc-30052017-120444.mp4
-rw-r--r-- 1 ftp ftp        1900448 May 30 07:05 enc-30052017-120502.mp4
-rw-r--r-- 1 ftp ftp        2275440 May 30 07:06 enc-30052017-120526.mp4
-rw-r--r-- 1 ftp ftp        2258976 May 30 07:06 enc-30052017-120601.mp4
-rw-r--r-- 1 ftp ftp        2207760 May 30 07:07 enc-30052017-120637.mp4
-rw-r--r-- 1 ftp ftp        2359600 May 30 07:07 enc-30052017-120701.mp4
-rw-r--r-- 1 ftp ftp        2264496 May 30 07:08 enc-30052017-120734.mp4
-rw-r--r-- 1 ftp ftp        4659968 May 31 01:25 enc-31052017-062329.mp4
-rw-r--r-- 1 ftp ftp        6056656 May 31 01:28 enc-31052017-062543.mp4
-rw-r--r-- 1 ftp ftp        4744208 May 31 01:43 enc-31052017-064300.mp4
-rw-r--r-- 1 ftp ftp        4898864 May 31 01:44 enc-31052017-064341.mp4
-rw-r--r-- 1 ftp ftp        5714592 May 31 01:52 enc-31052017-065103.mp4
-rw-r--r-- 1 ftp ftp        4355280 May 31 01:52 enc-31052017-065209.mp4
-rw-r--r-- 1 ftp ftp        4589104 May 31 02:00 enc-31052017-070002.mp4
-rw-r--r-- 1 ftp ftp        4422880 May 31 02:01 enc-31052017-070059.mp4
-rw-r--r-- 1 ftp ftp        3972000 May 31 02:02 enc-31052017-070201.mp4
-rw-r--r-- 1 ftp ftp        4354032 May 31 02:03 enc-31052017-070238.mp4
-rw-r--r-- 1 ftp ftp        2960864 May 27 04:13 enc-abc.mp4
-rw-r--r-- 1 ftp ftp        2054576 May 27 05:15 enc-xyz.mp4
-rw-r--r-- 1 ftp ftp              0 May 25 04:44 testvideo.mp4
226 Transfer OK
ftp> 
```

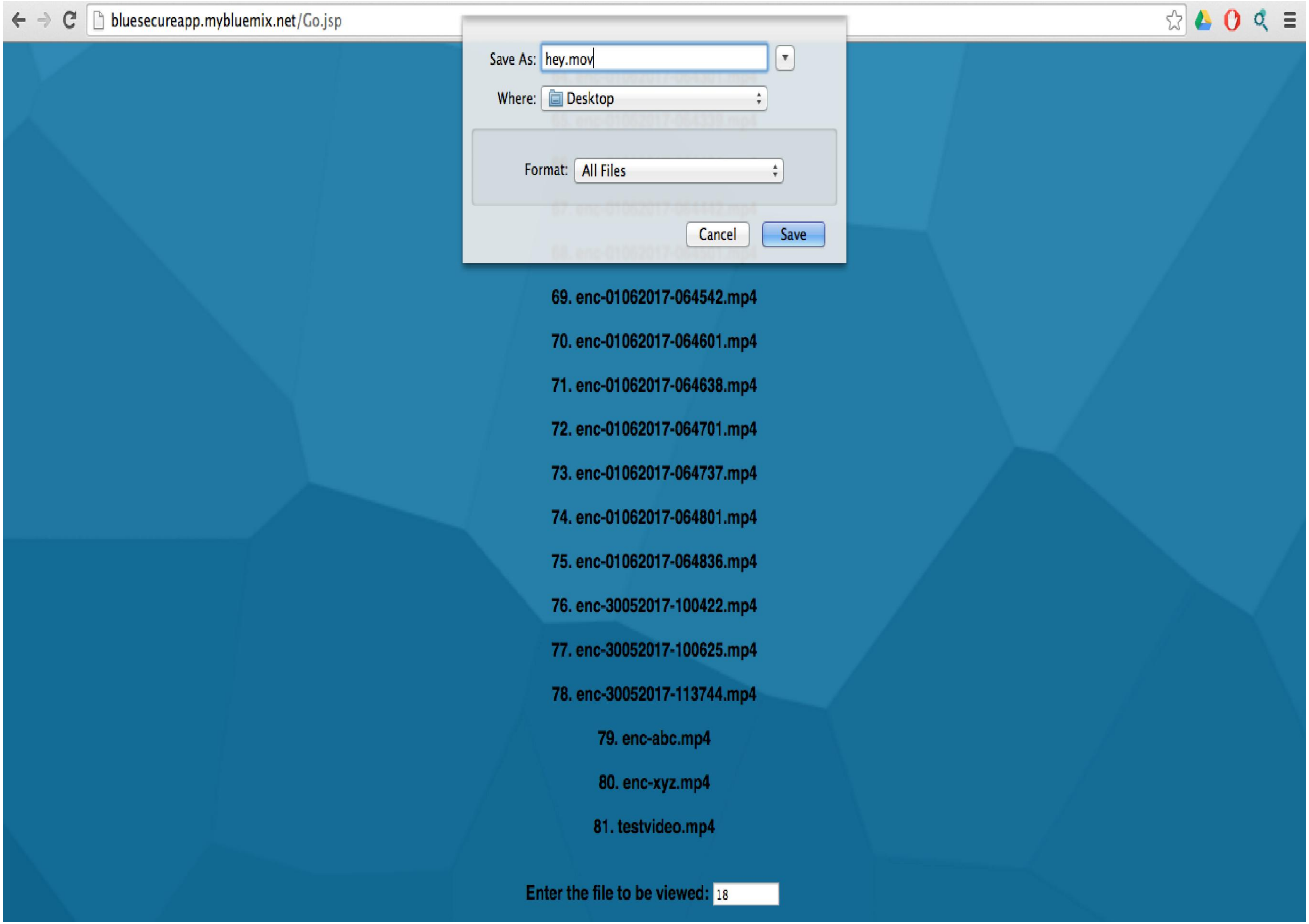
Server side : Bluemix


User authentication/login
<<extend>>
Invalid user Id or password
Request to view a file by choosing it from dashboard
Open the stream for file on FTP server
Read the UUID and encrypted data in 2 different byte arrays
End User
Bluemix
Fetch the key from IBM Security Key Lifecycle Manager using the derived UUID
Decrypt the encrypted data using the derived key
Give out the decrypted bytes on the outputstream for the user download the file
Download the file in desired location

Bluemix Use Case

The deployed application has following functionalities:

- Downloads the encrypted file from FTP server
- Segregate the UUID and encrypted bytes from the file fetched from FTP server
- KMIP Client gets the key from IBM Security Key Lifecycle Management server and decrypts the data
- The decrypted data can then be downloaded and the video can be viewed.



bluesecureapp.mybluemix.net/Go.jsp

Save As: hey.mov

Where: Desktop

Format: All Files

69. enc-01062017-064542.mp4

70. enc-01062017-064601.mp4

71. enc-01062017-064638.mp4

72. enc-01062017-064701.mp4

73. enc-01062017-064737.mp4

74. enc-01062017-064801.mp4

75. enc-01062017-064836.mp4

76. enc-30052017-100422.mp4

77. enc-30052017-100625.mp4

78. enc-30052017-113744.mp4

79. enc-abc.mp4

80. enc-xyz.mp4

81. testvideo.mp4

Enter the file to be viewed: 18

Deploying application on Bluemix

- Download Maven, Eclipse tools for IBM Bluemix and cloud foundry CLI
- Deploy the Maven Eclipse Application on Bluemix using following steps:

  mvn clean install

  specify API Endpoint

  cf login and enter user credentials

  cf push

Applications

Healthcare

Military Drones

Home Automation

Google+
FaceTime
skype
ooVoo
qik
business Bee

Real Time Streaming

GPS Trackers

Baby Monitors

Future scope

- System can be extended to multi-user security with multiple devices

- It can be generalized for any type of sensor data-file (.txt, .csv, .docx), audio, video.

- Making encrypted and signed firmware updates as per the requirement stated in OWASP[1].

Conclusion

- The system ensures security of data captured from IoT devices by using a suitable encryption algorithm along with KLMS and KMIP. The encrypted data can be accessed by authenticated user via an application server.

Literature Review

Papers/Survey Referred	Summary
[1] OWASP Internet of Things Top 10 vulnerabilities	Highlights all existing problems in IoT domain and provides solutions
[2] IBM Security Key Lifecycle Manager Datasheet	To simplify, centralize and automate encryption key management process and support for key-management standard--KMIP
[3]KMIP white paper	Explains how clients interact with enterprise servers using KMIP Protocol.
[4] IBM Bluemix website https://www.ibm.com/cloud-computing/bluemix/	Solves real problems and drives business value with applications, infrastructure and services.

Q
A

Thank
You