ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
Regular Issue, Vol. 9 N. 3 (2020), 105-117
eISSN: 2255-2863
DOI: https://doi.org/10.14201/ADCAIJ202093105117

Secure Data Transmission in BPEL (Business Process Execution Language)

Satya Bhushan Verma[a], Shashi Bhushan Verma[b]

[a] Computer Science and Engineering, Goel Institute of Technology & Management Lucknow India
[b] Tech Mahindra Limited, Pune, India
satyabverma1@gmail.com, shbh1991@gmail.com

KEYWORD

BPEL; Service-Oriented Architecture (SOA); Cryptography; WS-BPEL.

ABSTRACT

In the world of computation, the encryption is a technique by which the plaintext or any type of data which is converted from the readable form is transformed into an encoded form. That encoded form can only be read by another entity if they have corrected key for decryption. The proposed technique providing the security to the data in inefficient way that can be further use in implementation in new upcoming task and enhancement in current running projects of SOA (Service Oriented Architecture) BPEL (Business Process Execution Language). The importance of proposed method is, if client would like to send any data that via any communication medium, it make sure to safe from the outer world. The client wants that data must be in a non-readable format on the open network.

1. Introduction

The need for this application arrives when a client wants to sure that data that he is sending via any communication medium, needs to be secured from the outer world. The client (Sender) wants data should be in a non-readable format on the network. This requirement leads to getting this done which include an encryption and decryption service with unique Key and This whole application contains BPEL and JAVA technologies for the final desired requirement (Java Cryptography 2020).

When server A wants to send the data to server B there was the problem of security issues, during the transmission of data. When server A sends the data to server B, data is fully exposed there is no encryption involved, which can be easily readable, so by using this proposed approach solves the security issues.

Service-Oriented Architecture (SOA): SOA is an architectural method, in which applications make use of services accessible within network. This type of architecture, applications provides the services,

through the communication call over internet (Liao, Ziqi & Shi, Xinping, 2017) The SOA permits users to associates a huge number of services from the current services to form applications. The main concepts behind the SOA are established before the Web Services came along. A service inside SOA is totally autonomous of the concept of Web Service (Oldooz Karimi 2011).

SOA includes a set of design principles that structure system development and provide means for participating the components into an intelligible and decentralized system. Packages of computing based on SOA functionalities into a set of interoperable services, which are incorporated into various software systems be appropriate to distinct business domains.

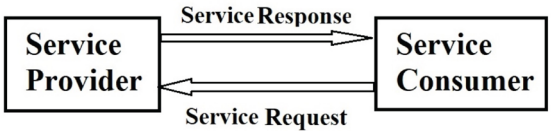

Figure 1: Service-Oriented Architecture (SOA).

There are two main roles within the SOA (Service-oriented Architecture):

Service provider: Service provider act as a maintainer of the services and organization that makes all time accessible one or more services for others to use. To advertise services, the provider can publish them in a registry, together with a service contract that specifies the nature of the service, how to use it, the requirements for the service, and the fees charged (Martin Keen, et al. 2004).

Service consumer: Service consumer can find the metadata of service in registry and then create the essential components of client to bind and use these services.

Business Process Execution Language (BPEL): The WS-BPEL (Web Services Business Process Execution Language) usually known as BPEL (Business Process Execution Language), is an OASIS (A handbook (2020)) standard executable language for specifying actions within business processes with web services.

BPEL (Business Process Execution Language) is language based on XML that permits web services in SOA to connect and allows to share data. The programmers can use BPEL to describe how to business process that involves web services will be executed. Messages of BPEL typically used to invoke remote services, rearrange process execution and the manage events and the exceptions ((M. Tian, et al. 2014) (Michele Bugliesi, et al. 2016)) BPEL is frequently connected with the BPMN (Business Process Management Notation), it is a standard for graphically representing the business processes.

2. Related Work

2.1. Encryption

In the computing world, encryption is a technique by which any type of data or plaintext is converted from a readable form to an encoded version that can only be read by another entity if they have correct key for decrypting. Encryption is most important and widely used in providing data security, especially for end-to-end protection of data transmitted across networks (Deven Shah et al. 2008). Proving security in data by the encryption is widely used on the network to prevent user information

being sent between a browser and a server, including passwords, payment information and other information like card details for money transfer that should be considered private. Organizations and individuals frequently use encryption to protect sensitive data stored on various devices like computers, servers, and mobile devices (AES (2020)).

2.2. Working of Encryption

Non-Encrypted data often referred to as plaintext, is coded in non-meaningful words using an encryption algorithm and an encryption key. This process generates encrypted text that can only be viewed in its original form if decrypted with the correct key. We can divide the Encryption algorithm into two categories: symmetric and asymmetric [6].

Symmetric-key ciphers, also said as "secret key," uses one key, generally said as a shared secret because system doing the cryptography should share it with any entity it intends to be able to decode the encrypted knowledge. The most used symmetric-key cipher is that the Advanced Encryption Standard (AES), that was designed to shield government-classified data ((Deven Shah et al. 2008) (Giorgia Gazzarataa et al. 2015)).

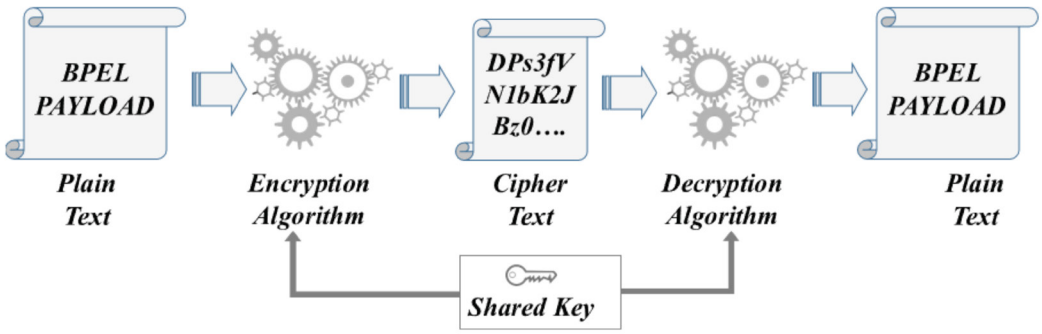

Figure 2: Symmetric-key encryption.

Symmetric-key encryption [Figure.2] is much faster than other encryption (Asymmetric), but the sender has to share the key used to encrypt the information with the destination receiver before the recipient can perform decryption on the cipher text. This process of converting information or data into a Cipher code, especially to prevent unauthorized access, needs to securely distribute and manage large numbers of keys means most cryptographic processes use a symmetric algorithm to efficiently encrypt data but use an asymmetric algorithm to securely exchange the secret key (Oldooz Karimi et al. 2011).

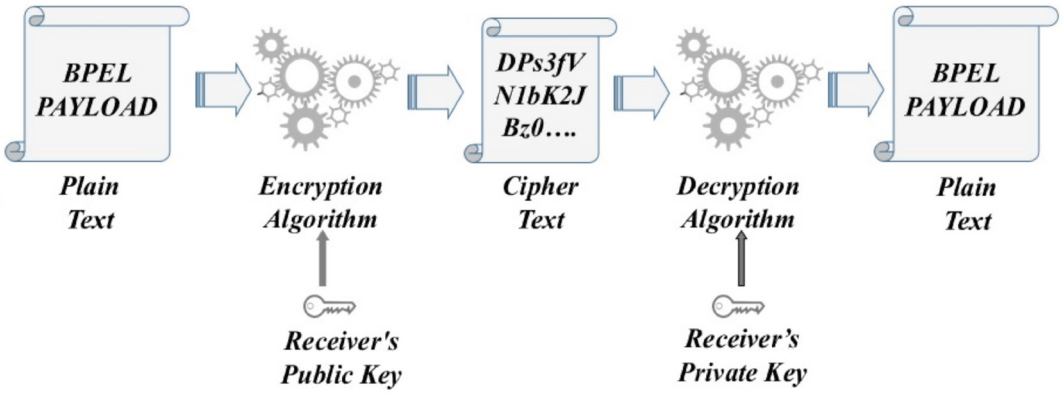

Figure 3: Asymmetric key encryption.

Asymmetric cryptography, also known as public key cryptography, uses two different but mathematically linked keys, one public and one private [Figure. 3]. The public cryptographic key (16 Character hexadecimal key) can be shared with everyone, whereas the private key must be kept secret. The RSA (Encryption Algorithm) is the common and preferred public key algorithm, partly because both the public cryptographic Key and the private cryptographic keys can encrypt a message. The opposite key (in private and public keys) from the one used to encrypt a message, is used to interpret or decode it. This technique provides a method of assuring not only strict privacy or secrecy, but also that data will in perfect condition, genuine and no reputability of electronic communications and data at rest with E-signatures (Bo Zhou et al. 2017).

3. Description

3.1. BPEL (Business Process Execution Language)

The Web Services Business Process Execution Language (WS-BPEL), also known as BPEL (Business Process Execution Language), is an OASIS [Organization for the Advancement of Structured Information Standards] standardizes an executable language for describing the actions within business computing with web services. Processes in BPEL exports and imports information by using web service interfaces exclusively. The figure displayed above is a sample BPEL process (figure 5), in which there is one client that invokes the process and in between processing BPEL process invokes two more external processes linked by WSDL. BPEL (Business Process Execution Language) is a language which is helps to define and execution of business processes using Web services; it enables the top-down realization of Service Oriented Architecture (SOA) through the orchestration, composition, and coordination of the Web services. Business Process Execution Language provides a comparatively easy and direct way to write numerous Web services into new complex services that is called business processes.

3.2. Need of Cryptography in BPEL process

Data security in the BPEL processes is concerned with the protection of data/information against alteration, destruction, and unauthorized use. Cryptography and encryption are the most critical components of data security. While transferring data different types of modes being used and that belongs to Network. The network used in data transfer take all kind of sensitive data and Security plays a vital role in any wireless network system. Security certifies the level of data integrity and data confidentiality as it maintained in a wired network, without accurately implementing security measures and Wireless network adapter comes within range of the network adapter. For using Cryptography in BPEL, a web service being introduced within BPEL Process that takes the plaintext or original non-encrypted payload passes to that service and that service perform the encryption on the payload, and transfers result to calling BPEL.

4. Experimental Evaluation

4.1. Encryption Phase in BPEL

4.1.1. Algorithm

Process of the encryption in the BPEL

1: Invoke the BPEL Service and provide plain text Payload
2: BPEL Process invokes the Java Encryption Web Service and pass the data from payload as arguments.
3: In Java, there is a method of encryption that called and with the help of a Key and Initialization Vector; it converts the Plain text into cipher text.
4: When BPEL got the output from Java web service it transforms the delimited string to final output payload by calling separate BPEL service and the out from that service responds back to its calling module or user.
5: output received at calling module or to the user.

4.1.2. Implementation of the Encryption Service in BPEL

Proposed method of encryption and decryption service in BPEL is successfully tested in reputed multinational company. Complete process of encryption in BPEL is given below. The graphical view of complete BPEL Service for Encryption represented in the following figure 4.

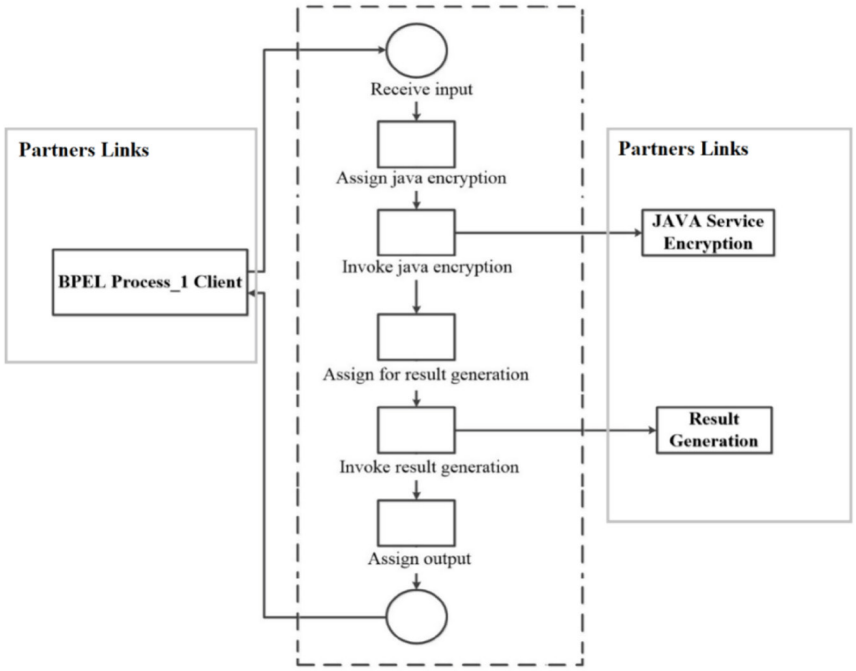

Figure 4: The process of the encryption in BPEL.

Step 1: Invoke the BPEL Service and provide plain text Payload. Below is the sample Payload which process takes to process the encryption.

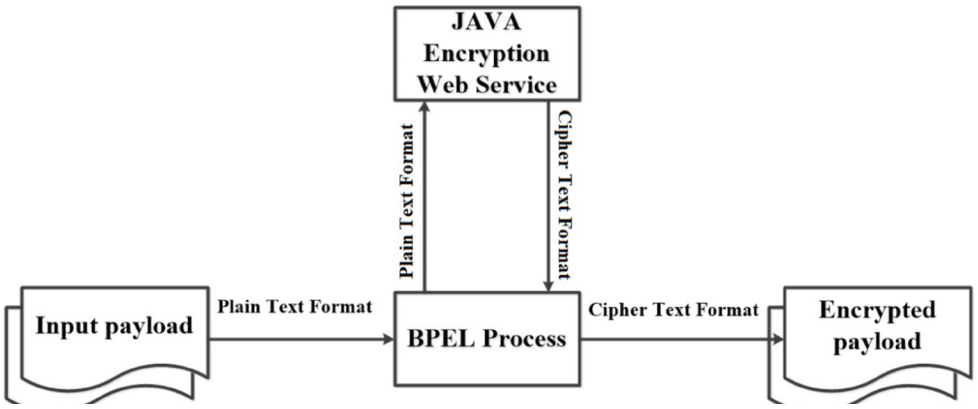

Figure 5: Encryption BPEL Service Flow Diagram.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
<soap:Body>
<ns1:process
xmlns:ns1="http://xmlns.oracle.com/Generic_App/Secure_Encryption/BPELProcess1">
<ns1:NAME>SHASHI BHUSHAN VERMA</ns1:NAME>
<ns1:AGE>27</ns1:AGE>
<ns1:CITY>PUNE</ns1:CITY>
</ns1:process>
</soap:Body>
</soap:Envelope>
```

Step 2: After successful loading of input payload, BPEL Process invoke the Java Encryption Web Service and pass the data from payload as arguments.

```
<in-
vokename="Invoke_Java_Encryption"partnerLink="Java_Service_Encrypt"portType="ns4:ProcessIT
"opera-
tion="encryptPayload"inputVariable="Invoke1_encryptPayload_InputVariable"outputVariable="In
voke1_encryptPayload_OutputVariable"bpelx:invokeAsDetail="no" />
```

In the above script, the *inputVariable* contains the data from payload that we use to process the service as mention below:

```
<?xml version="1.0" encoding="UTF-8"?>
<Invoke1_encryptPayload_InputVariable>
<part name="parameters">
<encryptPayload>
        <arg0>SHASHI BHUSHAN VERMA</arg0>
        <arg1>27</arg1>
        <arg2>PUNE</arg2>
</encryptPayload>
</part>
</Invoke1_encryptPayload_InputVariable>
```

Step 3: In Java, there is a method of encryption that called and with the help of a Key and Initialization Vector; it converts the Plain text into cipher text.

```
private static String encryptData(String text) {try {
    byte[] key = new BigInteger(strToHex("ddafXA1afcf6b1cb"), 16).toByteArray();
    byte[] iv = new BigInteger(strToHex("a33dc00670g13ed7"), 16).toByteArray();
    Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
    SecretKeySpec keySpec = new SecretKeySpec(key, "AES");
    IvParameterSpec ivSpec = new IvParameterSpec(iv);
    cipher.init(Cipher.ENCRYPT_MODE, keySpec, ivSpec);
    byte[] results = cipher.doFinal(text.getBytes("UTF-8"));
    BASE64Encoder encoder = new BASE64Encoder();
    return encoder.encode(results);
   } catch (Exception e) {
    return "Process Error";
   }
   }
```

As we can see in the code there is two underlined text that text acts as key and Initialization Vector(IV) that same key and IV used to decrypt the data also, because here symmetric key encryption is used, and we can use any 16 characters alphanumeric string for key and IV. After processing each element of payload, one more java method is called to collect all elements in a single unit separated by a single keyword (delimited String) and return that combined output to BPEL service.

Step 4: When BPEL got the output from Java web service it transforms the delimited string to final output payload by calling separate BPEL service and the out from that service responds to its calling module or user.

```
  INPUT PAYLOAD to generate results:
  <input>
  pUw+4WuO8phIeuu/ZxVE/9QmaBPAFGc+SwBc1/DVIT0=,8CnjLnfA/pFTpM5dzwvlGg==,m07C2a
3Kg5FTVA1yAg5BXQ==
  </input>
```

Step 5: Below is the final output received at calling module or to the user.

```
   <outputVariable>
   <part xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="payload">
   <processResponse
xmlns="http://xmlns.oracle.com/Generic_App/Secure_Encryption/BPELProcess1">
   <NAME>pUw+4WuO8phIeuu/ZxVE/9QmaBPAFGc+SwBc1/DVIT0=</NAME>
   <AGE>8CnjLnfA/pFTpM5dzwvlGg==</AGE>
   <CITY>m07C2a3Kg5FTVA1yAg5BXQ==</CITY>
   </processResponse>
   </part>
   </outputVariable>
```

4.2. Decryption Phase in BPEL

Decryption is a procedure of converting the encoded or encrypted text into normal text that is understandable for human or computer.

4.2.1. Algorithm

Process of the Decryption in the BPEL

1: Invoke the BPEL Service and provide Encrypted Payload
2: BPEL Process invoke the Java Decryption Web Service and pass the data from payload as arguments.
3: In Java, there is a method of decryption that called and with the help of a key and Initialization Vector it converts the cipher text into plain text.
4: When BPEL got the output from Java web service it transforms the delimited string to final output payload by calling separate BPEL service and the out from that service responds back to its calling module or user.
5: The final output received at calling module or to the user.

4.2.2. Implementation of the Decryption Service in BPEL

The graphical view of complete BPEL Service for Decryption represented in the following figure 6.

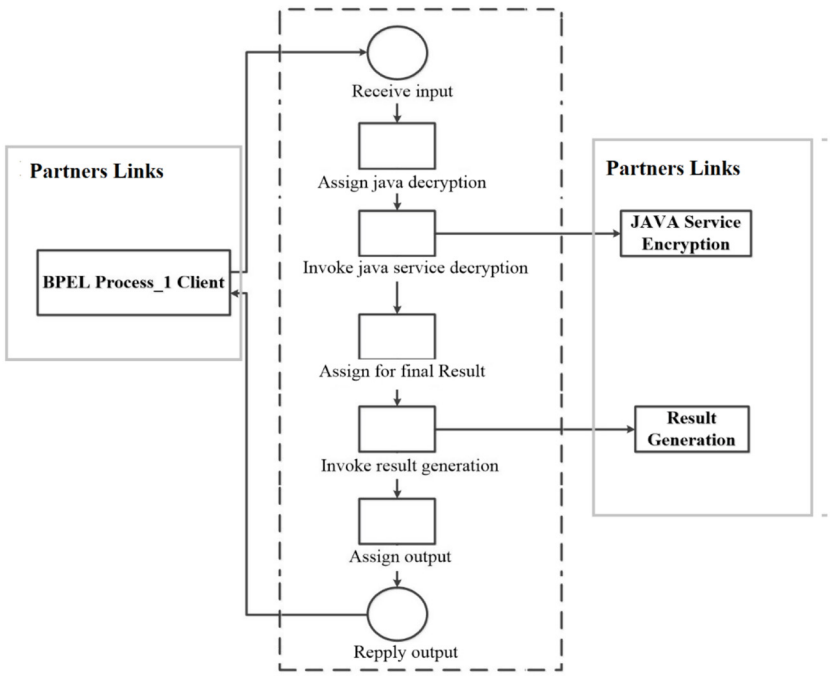

Figure 6: The process of Decryption in BPEL.

Step 1: Invoke the BPEL Service and provide Encrypted Payload. Below is the sample Payload which process takes to process the Decryption.

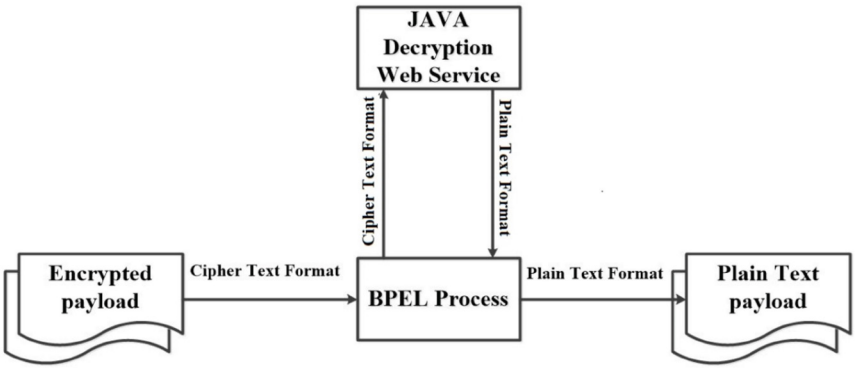

Figure 7: Decryption BPEL Flow Diagram.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
<soap:Body>
<ns1:process xmlns:ns1="http://xmlns.oracle.com/Generic_App/Secure_Decryption/
BPELProcess1">
<ns1:NAME>pUw+4WuO8phIeuu/ZxVE/9QmaBPAFGc+SwBc1/DVIT0=</ns1:NAME>
<ns1:AGE>8CnjLnfA/pFTpM5dzwvlGg==</ns1:AGE>
<ns1:CITY>m07C2a3Kg5FTVA1yAg5BXQ==</ns1:CITY>
</ns1:process>
</soap:Body>
</soap:Envelope
```

Step 2: After successful loading of input payload, BPEL Process invoke the Java Decryption Web Service and pass the data from payload as arguments.

```
<invoke name="Invoke_Java_Service_Decryption" bpelx:invokeAsDetail="no"
partnerLink="Java_Service_Decrypt"
portType="ns1:DecryptIt" operation="decryptPayload" inputVariable="Invoke1_
decryptPayload_InputVariable" outputVariable="Invoke1_decryptPayload_OutputVariable"/>
```

Step 3: In Java, there is a method of decryption that is known and with the help of a key and Initialization Vector it converts the cipher text into plain text.

```
    private static String decryptIt(String text){
    try{
    byte[] key = new BigInteger(strToHex("ddafXA1afcf6b1cb"),
16).toByteArray();
    byte[] iv =new BigInteger(strToHex("a33dc00670g13ed7"),
16).toByteArray();
    Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
    SecretKeySpec keySpec = new SecretKeySpec(key, "AES");
    IvParameterSpec ivSpec = new IvParameterSpec(iv);
    cipher.init(Cipher.DECRYPT_MODE, keySpec, ivSpec);
    BASE64Decoder decoder = new BASE64Decoder();
    byte[] results = cipher.doFinal(decoder.decodeBuffer(text));
    return new String(results, "UTF-8");
    }catch(Exception e){
    return "Error While Processing!!";
    }
    }
```

Here we can check that the 16 characters key and Initialization Vector (IV) used to decrypt the payload is the same as earlier we use in the encryption process. After processing each element of payload, one more java method is called to collect all elements in single unit separated by a single keyword (delimited String) and return that combined output to BPEL service.

Step 4: When BPEL got the output from Java web service it transforms the delimited string to final output payload by calling separate BPEL service and the out from that service responds back to its calling module or user.

```
INPUT PAYLOAD to generate results:
<ns2:decryptPayloadResponse>
   <return>SHASHI BHUSHAN VERMA,27,PUNE</return>
</ns2:decryptPayloadResponse>
```

Step 5: Below is the final output received at calling module or to the user.

```
<processResponse
    xmlns="http://xmlns.oracle.com/Generic_App/Secure_Decryption/BPELProcess1">
<NAME>SHASHI BHUSHAN VERMA</NAME>
<AGE>27</AGE>
<CITY>PUNE</CITY>
</processResponse>
```

5. Result and Analysis

The Data transferred through the services uses this concept and technique are safer than earlier used data transferred without any encryption security. If the data leaks in between transformation that data will be garbage without the key and Initialization Vector (IV) that we used to encrypt. In addition, these keys and IV is a 16-character alphanumeric keyword, which is not so easy to crack.

6. Conclusion

Service-Oriented Architecture is an architectural approach in which applications make use of services available in the network, and BPEL (Business Process Execution Language), is a standard executable language for specifying actions within business processes with web services. Processes in BPEL export and import information by using web service interfaces exclusively. Based on the generalization of Cryptography to BPEL services, we can implement this kind of security each and every process of BPEL regardless of whether it for FTP, SFTP or a simple text payload. Whatever the data transferred through BPEL have to pass cryptographic process.

7. References

A handbook (2020), A Hands-on Introduction to BPEL, Retrieved from 18/06/2020 https://www.oracle.com/technetwork/articles/matjaz-bpel1-090575.html

AES (2020), Advanced Encryption Standard (AES) Retrieved from 18/06/2020 https://searchsecurity.techtarget.com/definition/Advanced-Encryption-Standard.

Bo Zhou et al. 2017, Bo Zhou, Quan Zhang, Qi Shi, Qiang Yang, 2017, Measuring web service security in the era of Internet of Things, Computers and Electrical Engineering.

Deven Shah et al. 2008, Deven Shah, Dr. Dhiren Patel, Dynamic and Ubiquitous Security Architecture for Global SOA, 2008, The Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, IEEE, DOI 10.1109/UBICOMM.2008.68

Giorgia Gazzarataa et al. 2015, Giorgia Gazzarataa, Roberta Gazzarataa, Mauro Giacominia, A standardized SOA based solution to guarantee the secure access to EHR, International Conference on Project Management / Conference on Health and Social Care Information Systems and Technologies, Procedia Computer Science 641124 – 1129.

Hariharan et al. 2014, Hariharan, Chitra Babu, Security Testing of Orchestrated Business Processes in SOA, International Conference on Advanced Communication Control and Computing Technologies (lCACCCT), IEEE.

Hua Yue et al. 2012, Hua Yue, Xu Tao, Web Services Security Problem in Service-oriented Architecture, International Conference on Applied Physics and Industrial Engineering.

Liao, Ziqi & Shi, Xinping, 2017, Web functionality, web content, information security, and online tourism service continuance. Journal of Retailing and Consumer Services. 39. 258-263.

Java Cryptography 2020, Retrieved from 18/06/2020 Java Cryptography http://tutorials.jenkov.com/java-cryptography/index.html

Java Symmetric AES 2020, Java Symmetric AES Encryption Decryption using JCE, Retrieved from 18/06/2020 https://javapapers.com/java/java-symmetric-aes-encryption-decryption-using-jce/

Oldooz Karimi et al. 2011, Oldooz Karimi, Security Model For Service-Oriented Architecture, Advanced Computing: An International Journal (ACIJ), Vol.2, No.4.

Martin Keen, et al. 2004, Martin Keen, Amit Acharya, Susan Bishop, Alan Hopkins, Sven Milinski, Patterns: Implementing an SOA Using an Enterprise Service Bus, IBM Redbooks, 2004.

M. Tian, et al. 2014, M. Tian, Y. Feng, X. Jin and Y. Zhao, "A security model of command and control system based on SOA," 2014 IEEE 5th International Conference on Software Engineering and Service Science, Beijing, pp. 784-787.

Michele Bugliesi, et al. 2016, Michele Bugliesi, StefanoCalzavara, RiccardoFocardi, Formal methods for web security, Journal of Logical and Algebraic Methods in Programming, pp-2352-2208 http://dx.doi.org/10.1016/j.jlamp.2016.08.006

Cryptography 2020, What Is Cryptography? By Jorn van Zwanenburg, Retrieved from 18/06/2020 https://www.investinblockchain.com/what-is-cryptography/

Zhengan Huang et al. 2018, Zhengan Huang, Junzuo Lai, Wenbin Chen, Tong Li, Yang Xiang, Data Security against Receiver Corruptions: SOA Security for Receivers from Simulatable DEMs, Information Sciences (2018), doi: https://doi.org/10.1016/j.ins.2018.08.059