CHAPTER 4

EVALUATION OF CUSTOMER REQUIREMENTS WITH ANALYZER AND SECURITY ORIGINATOR

4.1 USER REQUIREMENT INVESTIGATION IN SOA

Service Oriented Architecture is an emerging field that is used by different set of people. The users may vary from project leaders to end users. Each person is playing a vital role in the generation of combined service plug-ins in SOA. Especially, customers play a special role in service implementation. According to the user requirements, the agreements are made and the services are retrieved from a service broker. During end user requirement collection, the unsatisfied or non-reliable requirements make more issues. The focused issues include unclear representation, fragmentary requirements, non stable requirements, uncertain requirements and conflicting requirements.

4.1.1 Unclear Requirements

Generally, the project information is gathered from end users or the internal stakeholders. In some projects, before starting the process, the requirements are not properly analyzed by project leaders. Such an improper allocation may cause undefined scope over allocation of resources and may provide unpaid deliveries to the customers. To neglect such improper allocation of requirements, formal reviews and interviews are conducted with the options of one to one, one to many and many to many communications. The one to one communication is between a single customer and single project leader, one to many communication is between one customer and group of project members and many to many communication is between group of clients and group of project leaders. For example, if a customer wants to design a shopping web site for any required consulting service company, he has to state all the needed specifications like database, graphic options and shopping carts. While designing the web pages the unstated requirements defined by the customers are noted and added. Thus, it may increase the labor cost, time duration and wastage of resources. To reduce such inconvenience, a group review is conducted and all necessary requirements are collected and the feedback is updated frequently.

One more critical issue is that the non functional requirements are to be correctly defined and are matched for the stated project. In web services, multiple service providers sales types of services to the customers. The customers can choose the required services from the list of disclosed service list. Here, the service products differ but their functionality remains to be the same. Non functionalities like stability, reliability, availability, usability and maintainability are initially defined clearly by the customers. These non functionalities also serve as the major criteria to be analyzed in the product generation.

4.1.2 Fragmentary Requirements

The incomplete customer requirements are caused because of the improper communication between customers and project leaders. With these requirements, the project overruns the cost and schedule and thus results in less customer satisfaction. To overcome this issue, the requirements are executed with analysis models, meta-data requirement status, requirement repositories and the requirement base line. In analysis model, requirements are represented with the graphical view or text view or logical view. The customer requirements are primarily executed with different views. Analysis model is represented using use case model, data model, context model, formal model and event model. Use case model presents the business requirements with the set of actors and use cases, as a blue print, where the use case defines functionalities and actors represents the role of person. The actors are categorized as primary actor and secondary actor. The primary actor acts as a parent and secondary actor acts as subordinate. The actors to use case communications are defined with the relationship namely includes, extends, generalization and proceeds.

A data model also known as information model or class model is used to display the information and their relationships. For the given application, all major primary data classes are analyzed and drawn in the form of this model. This model describes the units of data, data components and values of data, where aggregation, generalization are represented as relationship between the components. The formal model provides sufficient information with mathematical logics. Here, the domain properties and functionalities are put forth to potential validation. The event model displays the models with the set of events. It is also used to display how the system functions based on the events. It is represented using decision trees in the form of forward decision trees and backward decision trees. This model frames a complete structure with the precondition, post condition, risk factor and event estimation. With such models, the incomplete requirements are elaborated easily, errors are reduced and feedbacks are obtained during analysis process.

4.1.3 Non Stable Requirements

The stability of a project depends upon the initial collection of project requirements while estimating the project needs. User specified information is gathered and unreachable information is translated with the help of development manager. During requirement consultation, group discussion or team discussion takes place in order to fix project cost and schedule. Also, previously evaluated project cost and estimation are matched with the proposed project.

Here, prototyping of software projects are maintained at some stage during development of the project. The preparation of regular stable rules may avoid and save the cost and duration time. Also, it avoids certain misconceptions and non valuable service agreements generated between the customers and development manager. Finally, the choices of framing the best solution are finalized with development manager and are discussed with the team members and end users. When the final report gets confirmed, then the software process is started for implementation.

4.1.4 Uncertainty Requirements

Quantity measurements for reasonable sample values are discussed with uncertain requirements. To measure uncertainty, some sample types of tests are carried out and their results are matched with the measured quantity. User requirements are estimated using functionalities of user needs then construct association with respect to the user needs, prepare documentation to address all customer requirements and finally make note of all updated frequent activities. For example, if a measurer component has unvaried air pressure and displays the alternative variation range values then to check the uncertainty for the range of values, the predicted range values are compared with existing measure values. Input requirements are gathered from the users and the estimation is measured with the existing results.

During requirement prediction analysis, the predicted results are affected with some risk factors. Normally uncertainty of requirements is used to measure the sample values from the quantity results. If the measured range values are fixed with certain limits, then it arises the chances of risk mitigation. If these fixing ranges get exceeded with the measured value, then the estimated value is neglected. True sample information are gathered and analyzed for the end range estimation. By doing this, will reduce the risk factors of the retrieved requirements.

Before doing the analysis, the executed requirements are calibrated and executed. The test can be conducted with the measured entry data and specifications. If both the estimation are defined correctly, then the predicted requirements are executed.

4.1.5 Conflicting Requirements

In a business process, when only one requirement specification document is generated rather than two then it, is called as conflict requirements. To avoid conflict, the system analyst first identifies the root source that produces conflicts. Conflict arises due to the improper understanding of project goals, improper communications and unpredicted features. To avoid such situations, in one way business goals are defined during the review meeting, reiterated and changes are made with the conflicting goals. Group discussion, open conversations are conducted to reduce the conflict situation. The work is assigned among various group leaders who control different groups. Each group has their own decisions and they collaborate when it comes to the final decision. Highly prioritized goal decisions are preferably selected during discussion and fixed for the project.In another way, the different stockholders understand the common goals and work separately. In such cases, both the results are measured and compare with old processing results. Also walkthroughs, story boards are conducted to identify the best requirements.

In the above requirement issue analysis how the user requirements are executed with multiple factors is discussed. Before processing the project analysis, the end user requirements are analyzed thoroughly with these different factors and the proposed Heterogeneous Offer Agreement Generation (HOAG) system collects user requirements with the above solicited factors for the first analysis. In this chapter, the layer 1 (Analyzer Security Originator) technique analyzes and filters the incoming user requirements. At first, the requirements are gathered and matched with the Extracted Database, then it is put forth for the Enforcer processing. In this secure algorithm, two types of encryption (Enforcer E1, Enforcer E2) and two types of decryption (Enforcer D3, Enforcer D4) are carried out to make the evaluation secure. Finally, the securely generated keys are matched with the Arbitrator and Centralizer.

4.2 WORKING PROCESS OF ANALYZER AND SECURITY ORIGINATOR

As stated in the previous analysis customer requirements, the incoming inputs are filtered and securely executed with the proposed first layer of Analyzer and Security Originator. This layer was designed with the components of Information Packages, Password Transaction – Service Identity (PT-SI) records, Extracted Database, Enforcer, Arbitrator and Centralizer. These components act as the basic progressing elements to generate security process. Figure 4.1. shows how the set of components are interconnected with their working functionality.

The actual processing of Layer1 started with the customer requirements. In a frequent time, if a single customer or group of customers invokes services, the provider’s services get delayed with respect to the availability. At the same time, the requirements also vary with the individual customers and they have their different choices to finalize the services. In the day to day business environment, service providers offer various services to the customers. Based on the customer requirements, service providers can provide prompt service with the parameters of reasonable costs, minimum duration and standard quality.

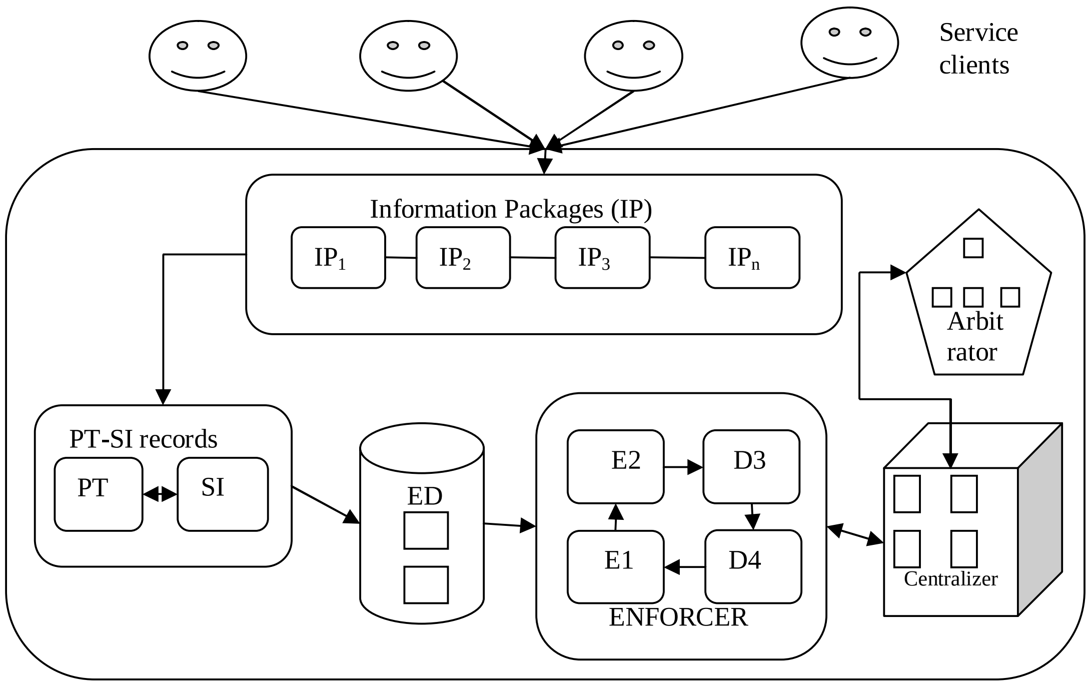


Figure 4.1 Analyzer and Security Originator

In the proposed system, customers can state their requirements with the preferable option of cost, duration and quality parameters with the indication of ‘High’, ‘medium’ and ‘low’. The indicated parameters are initially given to the customers, from that the customers can select their favored services. For first indication, the needed requirements are collected and the progress started with the Information Packages.

4.2.1 Information Packages

This will be the first active component to receive all required information from the customers. After having the necessary information from the customers, they are forwarded to the Information Package (IP) component. From the gathered requirements the package chooses the service priority list, customer transaction Id, password, service Id, shop code and total amount for the selected service. Here, the multi collective requirements are collected using small individual packages. The sub packages act as a collector and reduces the multi collective information load that received from various customers.

4.2.2 Password Transaction – Service Identity

Credential information received from the Information Package component are further passed on to PT-SI records. This package was designed with two sub packages called Password-Transaction (PT) and Service Identity (SI). The first PT package collects the password and Transaction Id. The second package collects the customer preferred service Identity. All the collected PT-SI records are verified with the Extracted Database. The purpose of matching with these databases is to verify the user accounts that are registered with the service provider. If any un-registered customer requirements are encountered during verification, they are discarded or returned back to the submitted customers. The registered user details are counted and the credential information is next put forth to the Enforcer component.

4.2.3 Extracted Database

Extracted Database is designed using the present and the last five years of the registered customer details. The database storage is grouped under registered storage and non registered storage. Information that matched in the PT-SI records are collected and stored under the registered storage and unmatched records are collected and stored under registered storage. Such as, if the customers have sufficient account balance with the selected list of service providers then they are grouped under registered storage and the non registered customers who have insufficient balance are grouped under non registered storage. The purpose of storing the unmatched data is to resend an acknowledgement to the non registered users. All the filtered information from the PT-SI records are generated using these two types of categorized repositories only.

4.2.4 Enforcer Algorithms

This is the heart of Analyzer Security Originator System. Each processing component in Enforcer unit can be defined with specific algorithms. The working functionality of Enforcer is elaborated with the four components. Figure 4.2 displays the internal processing of this secure generation algorithm.

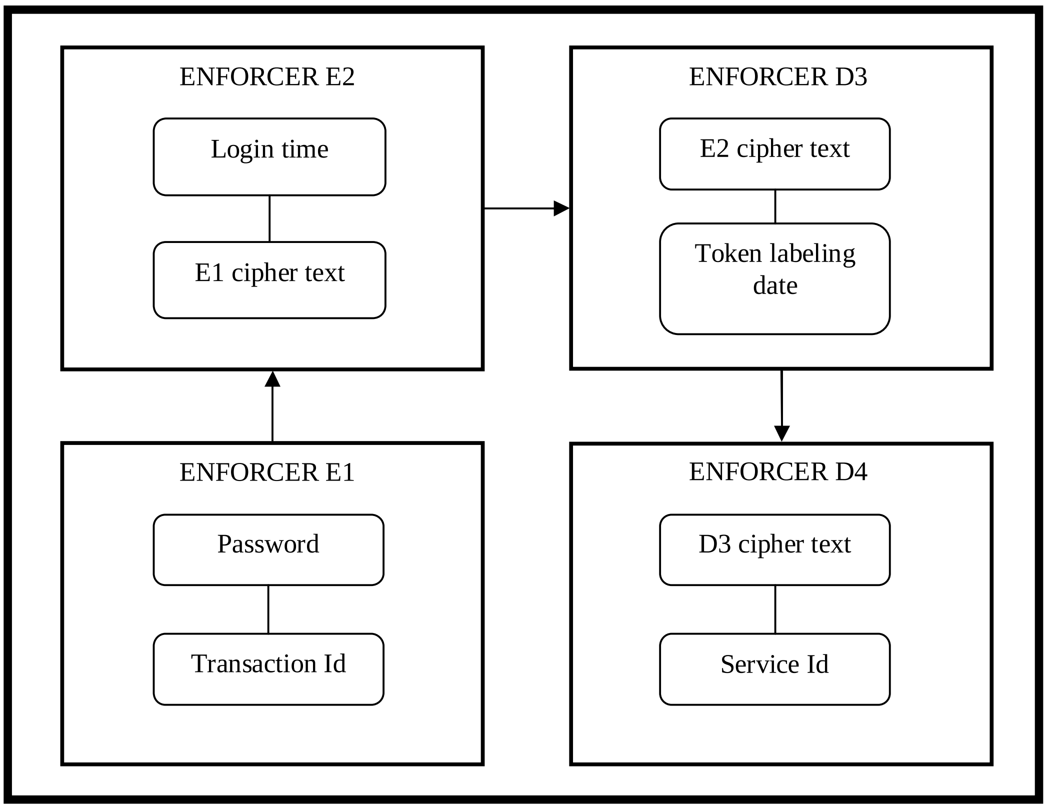


Figure 4.2 Enforcer Execution

4.2.4.1 Enforcer encryption E1 algorithm

Encryption E1 algorithm generates encryption using user password and transaction Id. User password is limited to 9 characters which must contain mixed symbols. While entering the password, user has to follow these specifications as a required status. For each password input, the user's preferred service transaction Id is generated. If the user selects more than one preferred service, then the user's login time and transaction Id is used to generate the password. Transaction Id is generated as S1-TID, S2-TID…Sn-TID. For example, if S1-TID service is accessed my numerous users then the users are serviced with sub ID's generated as $S1_{c1}$-TID, $S1_{c2}$-TID…$S1_{cn}$-TID. In this encryption, the user's password and transaction Id are converted to (8+8) =16 byte data and is stored in the memory stream. For the assigned 16 byte data encrypted with the fixed random string and the numbers get activated only. When, it receives the legitimate user password and transaction Id. The preferred random string must include letters, alpha numeric and special symbols and must be of 16 byte length. Figure 4.3 shows Enforcer Encryption E1 algorithm with processing steps.

1. Accept the authenticated user password and transaction Id.
2. When the user selects the service, preferred service transaction Id get generated.
3. Generated Id can be noted as S1-TID, S2-TID…Sn-TID.
4. If the same service accessed by more than one number of users then the service transaction defined as $S1_{c1}$-TID, $S1_{c2}$-TID…$S1_{cn}$-TID.
5. Convert the user password and transaction Id into 16 byte.
6. Encrypt the 16 byte with the fixed random numbers of (letters, alpha numeric and special symbols)

Figure 4.3 Enforcer Encryption E1 Algorithm

4.2.4.2 Enforcer Encryption E2 Algorithm

Encryption E2 algorithm begins its processing from the output of Enforcer Encryption E1. The second encryption is evaluated with the cipher text of Encryption E1 and the user login time. User's login time varies with the kind of services accessed and registration of service provider services. Based on the accessing time slots, the messages are forwarded to two types of containers called AM container and PM container. The AM container has the list of small packages called full cycle, half cycle and quarterly cycle. With hour base and minute base accessing, the AM timing services are retrieved and stored in either full cycle or half cycle or quarterly cycle. Similarly, the PM container has packages of full cycle, half cycle and quarterly cycle. As AM container, the PM container also has the retrieval of services with hour based and minute based. For the AM or PM cycle container, a 24 bit TIMING key is activated. With this timing key, the cipher text obtained from Encryption E1, gets encrypted once again in Enforcer E2 encryption. Figure 4.4 shows the encryption processing of Enforcer-Encryption E2 algorithm.

```
1. Receive Enforcer Encryption E1 output as an Input.
2. Encryption E2 happens with the Encryption E1 cipher text and user login
   time.
3. Two types of containers access the timing in AM container and PM
   container.
4. According to the user accessing time messages forwarded to the small
   packages of full cycle or half cycle or quarterly cycle.
5. Packages store the retrieved timing and generate a 24 bit keys to encrypt the
   encrypted cipher text.
       24 bit TIMING keys = AM container + PM container
   Where ..,
   AM container = full cycle (or) Half cycle (or) Quarterly cycle of AM
   PM container = full cycle (or) Half cycle (or) Quarterly cycle of PM
6. Finalized Encrypted E2 cipher text forwarded to Enforcer D3 Decryption.
```

Figure 4.4 Enforcer Encryption E2 Algorithm

4.2.4.3 Enforcer Decryption D3 Algorithm

The encrypted cipher text from Enforcer Encryption E2 moves to Enforcer Decryption D3, is it decrypted using the text with respect to date, where the user accessed date is interrelated with the TIMING key. Token labeling date is produced using the combination of user accessed date and the timing key. The token labeling date is the major concept for Enforcer Decryption D3 algorithm. In this decryption, the cipher text from Enforcer Encryption E2 is decrypted with the Token labeling date. During the decryption process the token labeling date generates 30 random keys to decrypt the accepted cipher text. Based on the user accessed date, any one key is assigned from the available 30 random keys. Random keys are a combination of alpha letters, special symbols and non numeric values. After decryption the decrypted D3 text moves on to Enforcer D4 decryption. Figure 4.5 shows the decryption processing of Enforcer Decryption D3 algorithm.

```
1. Decryption D3 starts with the cipher text of Enforcer E2 and Token labeling
   date.
2. Each generated date related with TIMING keys.
      Where..
      Token labeling date -> TIMING keys
3. Token Labeling date interrelated with TIMING keys.
4. 30 random keys are generated for the incoming request for the ratio of 30
   days.
5. Enforcer Decryption D3 happens from any one key from 30 keys.
```

Figure 4.5 Enforcer Decryption D3 Algorithm

4.2.4.4 Enforcer Decryption D4 Algorithm

This is the final step of decryption D4, which decrypt the received cipher text from Enforcer D3. For this decryption, it chooses service Id from the broker collective source of Arbitrator. It has more than 400 source of collection of service provider service Id that are re-matched or checked from the Extracted Database. Decryption D4 started when it retrieves the customer preferred service Id from the Arbitrator. The Enforcer D4 process started from the Id number and the cipher text of decryption D3.If any one particular service is chosen by multiple customers, then the service Id is categorized based on the timing consequences. It is denoted as $_{T1}$-S1Id$_1$, $_{T2}$-S2Id$_1$, $_{T3}$-S2Id$_{1...}$ $_{Tn}$-S2Id$_{1,.}$ After the decryption, the decrypted confidential string passes the authenticated data to authorized manager. Figure 4.6 shows the decryption of Enforcer- Decryption D4 algorithm.

```
1. Accept the Enforcer Decryption  D3 cipher text.
2. Service Id generated based on the customer selected service.
   It denoted as SId i.e... Service Identity.
3. Any particular service chosen from multiple customers  can be represented
   as
   $_{T1}$-S1Id$_1$, $_{T2}$-S2Id$_1$, $_{T3}$-S2Id$_{1...}$ $_{T3}$-S2Id……... $_{Tn}$-S2Id.
4. Final decrypted string passed to the authorized manager.
5. Authorized manager pass the confidential information to the concern
   service provider.
```

Figure 4.6 Enforcer Decryption D4 Algorithm

4.2.5 Centralizer

Centralizer has the collective set of encrypted and decrypted keys that needs to be stored from the Enforcer component. All the encrypted and decrypted enforcer keys are temporarily stored in this container because the Centralizer acts as the central mode to store or take a copy of the Enforcer information. During the transmission of data, if any data get lost or tarnished it can be re-processed from this component. Also, it stores the keys at a temporary period of time (2 days) after that it demolishes the present keys and arrived with the next keys. With the temporary storage, the hacker do not judge or make assumption about the keys. Hence, the Centralizer activates not only storage but it also provides security for stored keys.

4.2.6 Arbitrator

This is the component, which it stores the customer selected service Id in the repository. Centralizer component has inter-connectivity with this Arbitrator component. Mainly the Arbitrator is used to store the customer chosen service provider Ids. This customer selected service Id is accepted and used especially in Enforcer Decryption D4 algorithm. Also, the chosen service Id from this component is further rechecked with the Extracted Database in order to confirm the verification.

4.3 SAMPLE VALUE ANALYSIS OF LAYER 1

Service security is executed with the proposed Enforcer algorithms. From this algorithm, two types of encryptions and decryptions will provide better security for layer 1. The example analysis executes with these four sources namely Enforcer –Encryption E1, Enforcer- Encryption E2, Enforcer-Decryption D3 and Enforcer-Decryption D4. Initially the input strings are examined with 128 bit strings. The received inputs are to be matched with the 128 bits, in order to fix the string size. As per the string length, secured evaluation starts with the service Id, password, user login time and date among which service Id and password are verified with the Extracted Database.

4.3.1 Sample Value Execution of Enforcer Encryption E1

- Initially the service Id, password, login time and date is received from individual users.

- For example service Id is ‘HP200’,Transaction Id ‘2325’, password ‘12#77gg’, login time ‘8.45 PM’, login date ‘2.6.2014’ for the first customer. Likewise, the next customer also has these individual details.
- The service Id ‘HP200’ may be accessed with more than one number of customers, then it is denoted as ‘HP200$_{C1}$ - 2325’ -> S1$_{C1}$-TID.
- The user password ‘12#77gg’ and transaction Id ‘2325’ are converted to 16 byte.
- Where the 8 bits of password and 8 bits of the transaction Id is converted to 16 bytes.

```
protected void btn_Enforcer_E1_Click(object sender, EventArgs e)

{

txt_data2.Text = objSecurity.encryption (txt_Data1.Text.Trim());

}
```

- Such as ‘12#77gg2325’ this get encrypted with alpha numeric letters like 36 (A-Z (and) 0-9) or 62 (A-Z (and) a-z (and) 0-9) or the special symbols like +€£¥ćэþБҊ
- The conversion will happen in randomized way, hence it produce cipher text like ‘ ḉḉḉḉ±ḜḜḜḒ’
- This encrypted string acts as the next input to the second Enforcer Encryption E2.

4.3.2 Sample Value Execution of Enforcer Encryption E2

The output from Enforcer Encryption E1 acts as next input to Enforcer Encryption E2. For example…

- ‘ḉḉḉḉ±ḜḜḜḒ’ the Encrypted E1 cipher text is accepted and again encrypted with the user login time.
- User login time is separated as AM and PM containers. AM for morning and PM for evening.
- According to the time of accessing both the AM or PM containers are divided into full cycle or half cycle or Quarterly cycle.

  Full cycle = 12 hours accessing,

  Half cycle <=6 hours accessing,

  Quarterly cycle <= 3 hours accessing.

  For e.g… If customer 1 had accessed the ‘HP200’ service from ‘8.45 PM to 10.45 PM’ (2 hours) then with less than two hours of accessing the service comes under Quarterly cycle of PM.

```
protected void btn_Enforcer_E2_Click(object sender, EventArgs e)
    {
            txt_data3.Text=objSecurity.Encrypt(txt_data2.Text,
                        DateTime.Now.ToString());
            if (DateTime.Now.ToString().Contains("AM"))
                {
                lbl_Container.Text = "AM container";
                }
        else
```

```
                    {
                     lbl_Container.Text = "PM container";
                    }
txt_key.Text = objSecurity. TimingKeys (DateTime. Now. ToString ()); }
```

- The accessed two hours timing is stored in 24 bit keys. It is used to store either AM value or PM value. So that the second encryption happens with ‘ḉḉḉḉ±ḜḜḜḒ8.45:10.45’ to produce ‘RMP6hhzYLLWQW’ as the encrypted text.
- ‘RMP6hhzYLLWQW’ next moves to the third Enforcer Decryption D3.

4.3.3 Sample Value Execution of Enforcer Decryption D3

In the third Decryption D3, execution happens with the cipher text of Enforcer Encryption E2 and with token labeling date.

- Token labeling date = user accessed date + TIMING keys
- ‘RMP6hhzYLLWQW’ encrypted text is matched with ‘8.6.2014:8.45:10.45’ token labeling date.
- The generated token labeling date is assigned to any one of the 30 random keys.
- Here the 30 keys are assigned for thirty days per month.
- For ‘8.6.2014:8.45:10.45’, token labeling date component randomly assigns the 6th key of ‘52’. The code displays the execution of 30 key generations.

```
public string KeyMaster(string Date)
    {
int[]  Keyvalues={45,52,75,42,12,52,78,65,21,35,71,58,
        56,78,79,52,41,2,78,12,25,14,100,12,13,87,89,90,
        95,99,46,78,96 };
DateTime objDateTime = Convert.ToDateTime(Date);
string key= Convert .ToString (Keyvalues [ objDate
                                   Time.Day ]);
return key ;     }
```

- With ‘8.6.2014:8.45:10.45’ + ‘52’, Decryption D3 produces ‘ €þϾШ245^{UR}¥£ç ’ next input to the last Enforcer Decryption D4 component.

4.3.4 Sample Value Execution of Enforcer Decryption D4

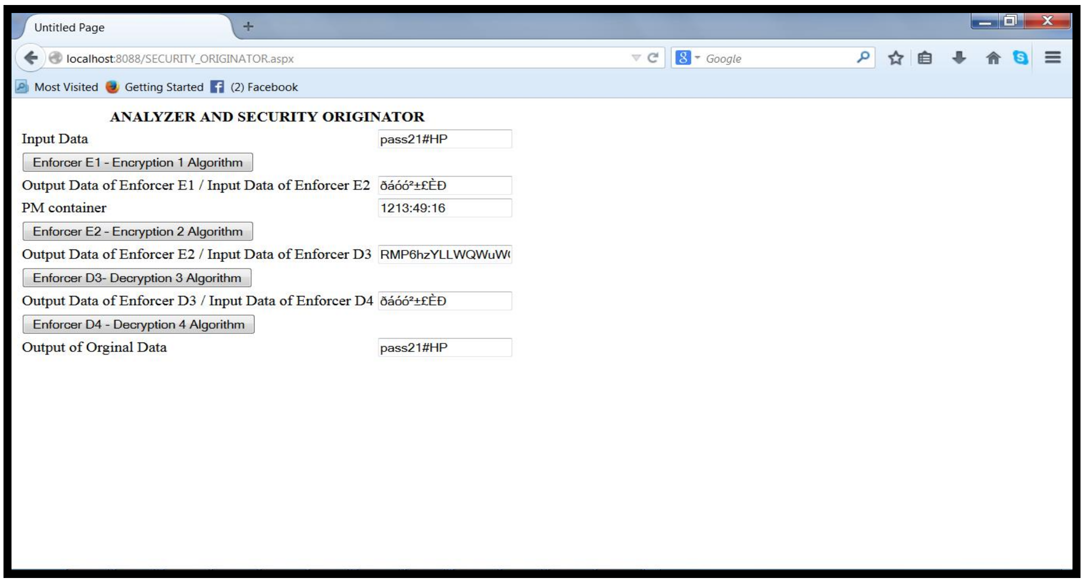
The output from the third Enforcer Decryption D3 component, finally moves to the Enforcer Decryption D4 algorithm. In this algorithm, secure evaluation is executed with the Arbitrator.

- The Arbitrator will check customer’s 1 preferred service Id ‘HP200’ and produce the service key ‘HP-2330021’.
- This service key will be generated at the time of execution by the Arbitrator, in order to avoid hacking. Also, the service key is generated by the particular service provider.
- The Decryption D4 happens with the token labeling date ‘€þϾШ245^{UR}¥£ç’ with ‘HP-2330021’.
- Such as ‘€þϾШ245^{UR}¥£ç’+‘HP-2330021’ will produce the actual string text ‘12#77gg2325’.
- Like ‘HP200’ service provider Id, the Arbitrator collects more number of provider Id services as storage. So that, the

required period of time the service key is accessed for Enforcer – Decryption D4.

- The produced decrypted strings next moves to layer 2 for classification. Like customer1, multiple customers can access the required services.

With all above process, the generated code from secured Enforcer next moves to layer 2 execution. Second layer will reduce the bulk arrival of requests with the proposed ETD classification algorithm. The proposed HOAG layer 1 execution is displayed in the following Figure 4.7, which displays sample value execution of Analyzer and Security Originator.



ANALYZER AND SECURITY ORIGINATOR

Input Data	pass21#HP
Enforcer E1 - Encryption 1 Algorithm	
Output Data of Enforcer E1 / Input Data of Enforcer E2	ðáóó²±£ÈÐ
PM container	1213:49:16
Enforcer E2 - Encryption 2 Algorithm	
Output Data of Enforcer E2 / Input Data of Enforcer D3	RMP6hzYLLWQWuW(
Enforcer D3- Decryption 3 Algorithm	
Output Data of Enforcer D3 / Input Data of Enforcer D4	ðáóó²±£ÈÐ
Enforcer D4 - Decryption 4 Algorithm	
Output of Orginal Data	pass21#HP

Figure 4.7 Sample Value Execution of Analyzer and Security Originator

4.4 EXPERIMENTAL TESTING OF LAYER 1

Customers can access service provider's services through multiple devices such as computer system or mobile devices. Various services can be offered with attractive offers to the customers. For the experimental analysis, 5 web service providers were included to test the timing sequence of the proposed system. The system received 15,000 transactions from multiple clients. Customers provide the necessary authentication to the first component of Information Packages. The submitted user information is verified with the service provider. Non registered user information is identified and sent back to the customers. Among the 15,000 requests initially received 7,000 requests were filtered and they were matched with the Extracted Database. Transaction Id is also generated for each filtered package data. For 15,000 customers the transaction Id is represented as T_1Id, T_2Id, T_3Id…..T_{15000}Id.

Generated Id next moves on to PT-SI records, where the password transaction moves to Password Transaction (PT) package and Service Identity moves to Service Identity (SI) package. The filtered user information is first passed on to the first container of PI to collect the secured password. Then it moves on to the SI, where it assigns the Service Identity according to the selected customer service. The PT-SI and Extracted Database interconnected 4,000 data sets are collected and stored in this database. Out of 7500 datasets, service transaction Id, password, user login date, user login time and service Id are passed on to the 4 types of Enforcers, where all the 7,500 requests are forsaken to this Enforcer authenticated processing and moves on to the Centralizer to store the scrutinized information. Information which is stored in the Centralizer can be recollected whenever it requests the necessary information from the Enforcers. After final scrutinizing, authenticated data moves to the authorized manager to generate the final document and move to Layer 2 execution.

4.4.1 Performance Analysis of Proposed Enforcer Algorithms

The performance analysis of Enforcer (E1, E2, D3 and D4) are discussed and compared with the legacy system component. Each Enforcer algorithm is experimentally analyzed and graphically discussed with the existing system.

4.4.1.1 Enforcer Encryption E1 Experiment

The first experiment begins with Enforcer Encryption E1 algorithm. In the experimental testing, samples of 15,000 transactions were carried out that includes both password and transaction Id. Comparing to the Enforcer E1 algorithm the legacy system was also tested with 15000 customer transactions from which 8500 datasets were filtered. For the legacy system, the filtered data is the combined ratio of password and transaction Id. Enforcer E1 and legacy system are tested using a timing frequency of 0.5 seconds. The above defined transaction details are listed in Table. 4.1 and the sequential graphical representation of Enforcer Encryption E1and legacy system is given in Figure 4.8

Table 4.1 Experimental Testing of Enforcer Encryption E1 and Legacy Component

Set of Information	Enforcer E1	Legacy	Enforcer E1 (filtered)	Legacy (filtered)	Timing freq tested
Password	15000	15000	7500	8500	0.5 sec
Transaction Id			7500		

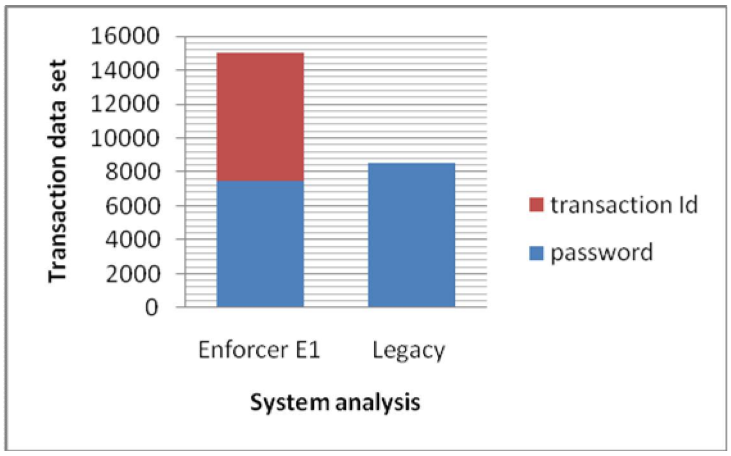


Figure 4.8 Comparative Analysis of Enforcer Encryption E1 and Legacy System

Figure 4.8 displays the graphical representation of Enforcer Encryption E1’s processing as compared with the legacy system. The bar chart comparison shows red and blue vertical bars with the increased view of Enforcer Encryption E1 algorithm. Compared to the proposed Enforcer Encryption E1 algorithm, the legacy system has a decreased range of vertical bar. The ‘X’ axis shows system analysis of the proposed Enforcer Encryption E1 and legacy system comparison. ‘Y’ axis shows the total number of transactions that are initially input to Layer1. From the analysis, filtering ratio is separately divided for Enforcer Encryption E1 algorithm. Simultaneously, it generates password and transaction Id of the received customers. In the analysed legacy system it includes both password and transaction Id. From the 15,000 transactions it filters about 8,500 transactions.

4.4.1.2 Enforcer Encryption E2 Experiment

Enforcer Encryption E2 algorithm is generated with the encrypted cipher text of E1 with user login time. The Enforcer Encryption E2 experiment is executed with respect to the user login time which is based on AM container or PM containers. Both the containers have the option of full cycle or half cycle or quarterly cycle. In experimental testing, it takes 0.10 secs to evaluate the timing option of AM and PM containers. AM container having timing of 6.00 am to 7.00 am (1 hour) is categorized quarterly cycle, 9.05 am to 11.30 am (2 ½ hours) is categorized under half cycle. Individual transactions executed for AM Quarterly cycle is 2,950 transactions and for PM – Half cycle is 2,550. So, totally it consists of 5500 data transactions.

Similarly, PM container is tested with 0.5 sec with quarterly cycle and half cycle. For Quarterly cycle, it takes 8.30 am to 9.30 am (1 hour) with the filtered 2000 sets of user data. And 10.05pm to 4.30 pm (6 ½ hours) filter 2500 set of data. In total it produces 4,500 data. From the 15,000 data it filters 10,000 with AM and PM containers. The legacy system totally produces 9,200 sets of user data. The second encryption dataset takes 0.10 secs for

Enforcer Encryption E2 and 2.86 secs for legacy systems. The prescribed information is depicted in Table 4.2.

Table 4.2 Experimental Testing of Enforcer Encryption E2 with Legacy Component

				Enforcer E2 (filtered)					
Set of Information	**Timing freq Input tested**	**Enforcer E2**	**Legacy system**	**AM container**	**PM container**	**Enforcer E2 (total filtered)**	**Legacy (filtered)**	**Timing freq Output tested**	
Enforcer E1 Encryption	-	15000	15000	-	-		-	**Enforcer E2**	**Legacy system**
login time	0.05 sec	6:00 am	7.00 am	2950	-	5500 + 4500 = 10,000	9200	0.05 Sec	0.10 sec
		09.05 am	11.30 am	2550					
	0.05 sec	8.30 pm	9.30 pm	-	2000			0.05 sec	
		10.05 pm	4.30 pm		2500				

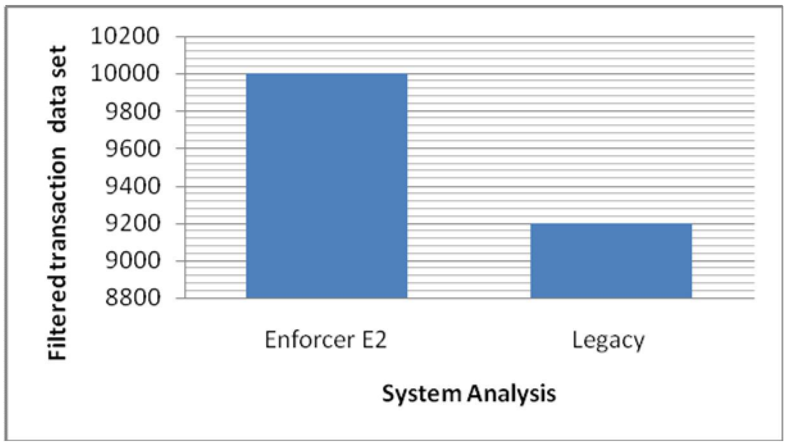


Figure 4.9 Comparative Analysis of Enforcer Encryption E2 and Legacy System

Enforcer Encryption E2 analysis from Figure 4.9 shows graphical comparison between Enforcer Encryption E2 algorithm and legacy system, where ‘X’ axis denotes filtered transaction data set and ‘Y’ axis denotes system analysis. The blue vertical bar represents an increased ratio of

Enforcer Encryption E2 and Legacy system. From the graph analysis, it can be seen that Enforcer Encryption E2's range is higher than the legacy system. The proposed algorithm produces the resultant output with a timing of 0.5 secs rather than 0.10 sec that the legacy system took. The produced filtered output is then passed to Enforcer Decryption D3 algorithm.

4.4.1.3 Enforcer Decryption D3 Experiment

After two set of encryptions, decryption starts with cipher text of Enforcer Encryption E2 with the token labeling date. Table 4.3 displays the count values of Enforcer Decryption D3 process. Here the user access date is correlated with timing. For this decryption, the total amount of output data is fed for decryption3. 5,500 datasets are stipulated from Enforcer Encryption E2 and 4500 datasets for token labeling date. In comparison, Enforcer Decryption D3 component outputs 10,000 decrypted data and legacy system outputs 6,500 filter data set. This execution took a timing of 0.5 secs.

Table 4.3 Experimental Testing of Enforcer Decryption D3 with Legacy Component

Set of Information	Enforcer D3 (Decrypted)	Legacy (Decrypted)	Timing freq tested
Cipher text of Enforcer E2	5500	6500	0.5 sec
Token labeling date	4500		

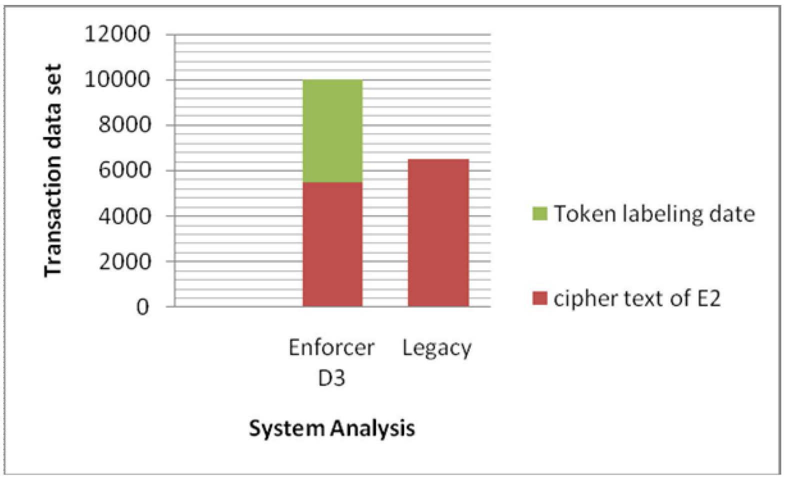


Figure 4.10 Comparative Analysis of Enforcer Decryption D3 and Legacy System

The above Figure 4.10 represents comparative analysis of Enforcer Decryption D3 algorithm with the legacy system. 'X' axis displays the transaction set data and 'Y' axis represents with system analysis of Enforcer Decryption D3 and Legacy system. The vertical red with green bar shows the increasing ratio of Enforcer Decryption D3 algorithm and red bar shows ratio of filtered data for legacy system. Here the next progressing output to be sent to the last decryption of Enforcer Decryption D4 algorithm.

4.4.1.4 Enforcer Decryption D4 Experiment

The last decryption component executes the cipher text of Enforcer Decryption D3 along with customer chosen service Id. Table 4.4 displays count values that are to be executed with the Enforcer Encryption E4 algorithm. Cipher text taken as 5,500 data and service Id of 4,500 data, totally it decrypts 10,000 data. The legacy system decrypts 4,350 data among 10,000 data and it discards 5650 data. These set of transaction occurs at the timing of 0.5 sec. Finally, these decrypted data is sent to layer 2 for classification.

Table 4.4 Experimental Testing of Enforcer Decryption D4 with Legacy Component

Set of Information	Enforcer D4 (Decrypted)	Legacy (Decrypted)	Timing freq tested
Cipher text of Enforcer D3	5500	4350	0.5 sec
Service Id	4500		

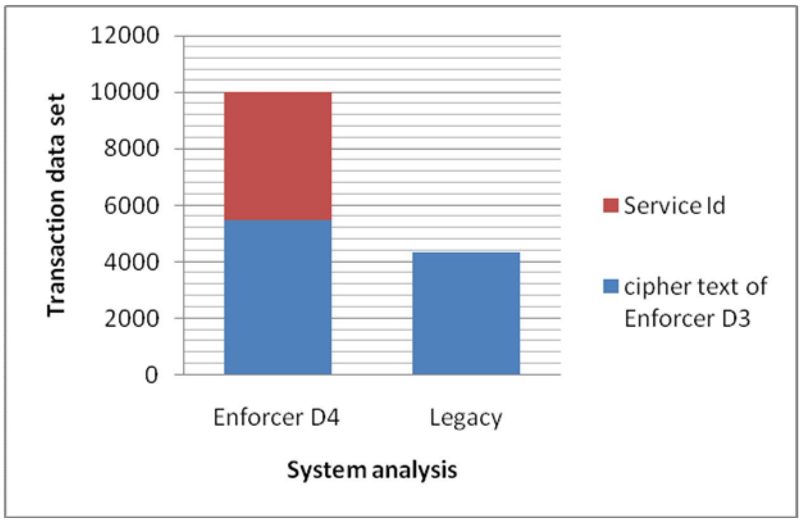


Figure 4.11 Comparative Analysis of Enforcer Decryption D4 and Legacy System

Figure 4.11 represents the graph study between Enforcer Decryption D4 and legacy system. 'X' axis denotes Enforcer Decryption D4 and legacy system and 'Y' axis denotes filtered transaction data set. By comparing the two studies, Enforcer Decryption D4 shows the red with blue bar and legacy system presents with the decreased blue bar. From the analysis of the four Enforcer algorithms (E1, E2, D3 and D4) done, it is understood that the proposed Enforcer algorithm reduces time complexity when compared with legacy system.

Therefore in Layer 1 the total numbers of transactions retrieved from the total number of customers are 15,000 and it evaluated with the four enforcer algorithms and produces the secured filtered output of 10,000 transactions. So, the final efficient rate produced in layer 1 is 67 %.

4.5 SUMMARY

In this chapter, experimental results of layer1 are discussed. Secure analysis takes place for the incoming transaction data set with the proposed Enforcer algorithm. The four types of proposed Enforcer algorithms

Encryption E1, Encryption E2, Decryption D3 and Decryption D4 are experimented and compared with the legacy system. Two types of encryptions and two types of decryptions are evaluated with respect to user login, login time, date, password and transaction Id.

The major contribution of this Enforcer algorithm is it prevents hacking of credential information from third party analysis. Also, the Arbitrator will act as a temporary storage to store and discard the keys in a certain period of time. This will improve on more secure transmission of information. Moreover, the implemented results are experimented and reduce judgment of the resultant keys in each experiment. Finally, the proposed Enforcer algorithm reduces the vulnerabilities and improves secure transactions.