Security in services for central cloud services to various cloud services

[1] A.Yamuna

1(Final Year M.Tech Student, Dept. of CSE, Aditya Institute of Technology and Management (AITAM), Tekkali, Srikakulam, Andhra Pradesh, yamuna505@gmail.com

Abstract:

Basically services will be having the storage areas where the clients can throw and preserve the data in bulk storage. To encrypt the data we propose the BASE-64 coz some other algorithms will also encrypt the data but not up to the level of 100% encryption coz of some strange formats of the data. Though the encryption is happening clients can directly interact with services in a distributed way. So this is very sensitive to services to restrict the confidentiality. So root services should be having the authority hierarchy to decrypt according to the users/clients categories. So we propose ROTA and without any further queries like percentage of compression, without any loss of data further in the distributed form. So the confidentiality architecture will be maintained throughout the hierarchy. The log will be maintained by the services , how many requests and types of requests coming from various clients. Once the client(s) started sending the request the monitor will monitor all the request cycle till it reach to service side. So all the request process will be monitored in the proper log file at stub side. The whole architecture is based on SAO (service oriented architecture) , where clients can have continuous services from service. By encrypting the data and the key will be ceiled into the request data. After that the data will be encrypted. So the security will be maintained through clients stub and service skeleton level(proxy structure). At service side the service will not have direct interaction , the reason is the security. Either stub or some intermediately program will accept the client's request and that will be redirected to service. So service will process the request and the request which is processed at the service in secured way.

(index terms: SOA , confidentiality ,stub,proxy)

Introduction:

We are proposing a new SOA architecture in this paper. The regular way is client(s) can directly interact with service as this is very sensitive and the client can be intruder to hack/capture the sensitive information from service. And ancient cloud services will be having direct interaction though they have access to third party services. But in this architecture we propose an LDAP request dispatcher to route the requests to valid service indirectly. So the request will not go directly to service.

LDAP (Light weight directory access protocol is the protocol which allows in this architecture to route the calls to valid services (own services or third party services). In this work LDAP receives the request at skeleton side and does categorization to send to the particular service.

The requested data also will be encrypted/ the key will be concealed and routed to services side. Here the key will be saved by the tracker to bring the response from service to normal form. So we put secret data structure at client side to maintain the keys which are random (unique) generated and will be compared with returned key from service. If that matches the data will be presented to normal form to clients.

Related work:

Normally SOA(service oriented architectures) are so strong in maintaining the concurrent client(s) as a services. Once the data is sent by client as request.

LDAP: Light weight directory access protocol will play key role in this work. Regular data will be stored in segregated directories whether they are in client side or service side. In this work the client will send a request for particular topic which has to be routed to valid directory where the data is saved and that data can local service or remote service. As in our work there is no direct interaction with the service LDAP will segregate that request and route to valid service to send back the response to client.

Terminology:

Si $\longrightarrow$ Main Service

rd $\longrightarrow$ Request Dispatcher

rh $\longrightarrow$ request Handler

total No. of clients

t $\longrightarrow$ Tracker (data Structure)

```
Sd  ⟶ Data source
K   ⟶ Key data structurer
        Set of all requests to clinet
        Set of all responses
k   ⟶ Random Key
```

Service creation: (Skelton)

```
Initialization
Sd ⟶ 0
Kd ⟶ 0      // where set of all keys will be observed
I ⟶ index
For all services s in S
        Loop
        50
Sd(s) = Space () // space for each service in
        i++
        Data S  Access(s,i)
        Rd = data will be send data re To request description to
S ⟶ data
End loop
Request Dispatches
Input: request/response
Output: reduction
Loop for each r in R || RS
 Count = 0
 If r ⟶ R
        R=Modify ( r)
        Else if rRs
        Rs = Modify (r,Ci)
End Loop
```

Client Callers(Stub)

```
Loop:  for   each  c    in    C
        c = { r1,r2,r3........ $r_n$} →
SEND ($r_d$,r)
        ENCR (r,k)
        R r
        K k
```

Rota Encrypt/Decrypt Algorithm:

Encryption:

```
Input: Plain text
Output: CipherText
Initialization:
N ⟶ 0          //total number of characters.
∑D ⟶ 0         //total data
Cipher ⟶ 1
∑E ⟶ 0         //total Encrypted data
t1 ⟶ null      //temp variable
t2 ⟶ null      //temp variable
Loop for each c in D
        n=0
        t1 ⟶ LSHIFT(c,n,CIPHER)
```

```
    t2 ⟶ RSHIFT(c,n+1,CIPHER)
              n+1
    E ⟶ APPEND(t1)
    E ⟶ APPEND(t2-1)
End loop
```

Decryption:

Input: Cipher text

```
End loop
For each $r_S$ in RS
    RECEIVE (r,$r_S$)
    DECR ( r,k)
    RS=r
End loop
```

Output: Plain text

```
Initialization:
N ⟶ 0 // initialization for total number of characters.
∑E ⟶ 0 //Cipher texts
∑N ⟶ 0      // plain text
T1 ⟶ null
T2 ⟶ null
Loop for each c in E
n=0
t1 ⟶ RSHIFT(c,n,CIPHER)
t2 ⟶ LSHIFT(c,n+1,CIPHER)
          n+1
    D ⟶ APPEND(t1)
    D ⟶ APPEND(t2+1)
End loop
```

In Rota Encryption Algorithm the Data Files will be encrypted in the form of cipher text. When transmitting the data from node to node the source file data was encrypted by the sender using Rota Encryption Algorithm. Here the Data Characters will be converted into based ASCII Values. These values are internally turned into binary bits. We perform operations such as Left Shift and Right Shift on Binary Data. If the ASCII values are Even number digit then perform 1LeftShift operation on Even ASCII value of given character (B<<1).its generate Cipher text, and then we again perform 1RightShift operation on cipher text.noe we got Plane text. If the ASCII values are odd number digit then perform 1RightShift operation on odd ASCII value of given character (A>>1). Its generate Cipher text, and then we again perform 1LeftShift operation on cipher text. Now we got Plane text.

Example1:

Enter Your String

Oppengangam style

Original Test is

Oppengangam style

Encrypt Test is

&àà1Ü@2/Ü2/5@8è;Ø1

Decryption Test is

Oppengangam style

Example2:

Enter Your String

jampingjapangujampakjampakjampingjam
p agujampakjampakgiligili ah

Original Test is

jampingjapangujampakjampakjampingjam
p agujampakjampakgiligili ah

Encrypt Test is

Ô/5à3Ü2@Ô/à/Ü29@Ô/5à/4@Ô/5à/
4@Ô/5à3Ü2@Ô/5à/29@Ô/5à/4@Ô/5
à/ 4@23Ø3@23Ø3@/Đ

Decryption Test is

jampingjapangujampakjampakjampingjam
p agujampakjampakgiligili ah

Example3:

Enter Your String

pleasedonot touch steves pet aligator

Original Test is

pleasedonot touch steves pet aligator

Encrypt Test is

àØ1/81@È6Ü6è@è690Đ@8è1ì18@à1è@
/ Ø32/è6ä

Decryption Test is

pleasedonot touch steves pet aligator

Advantages:

1. Rota encryption provides high security for data files while transmitting data from node to node in manet.
2. The main advantage of sha-2 algorithm is it generates same size of key for any size of plain text. so we can specify the size for key.
3. In SNMP multiple peers can be managed but very few adhoc networks have the following features: (Dynamic peer generation with static peer network, data sharing among unlimited casting peers as destination group, SHA-2 for secure key generation).
4. Logs will be maintained instantly with micro level information.

CONCLUSION:

Essentially facilities will be getting the hard drive locations where the consumers can certainly put in addition to protect the results throughout bulk hard drive. For you to encrypt the results we all offer this BASE-64 coz other sorts of algorithms will likely encrypt the results but not nearly the amount of 100% encryption coz involving many odd formats from the data. Although the encryption is going on consumers can certainly immediately interact with companies inside a distributed way. So this is sensitive to companies to reduce this secrecy. So origin companies must be getting the authority structure to decrypt good users/ clients classes. So we all offer ROTA in addition to with no even more inquiries just like proportion involving data compresion, devoid of just about any loss in data even more in the distributed kind. And so the secrecy architecture will be preserved during this structure. The log will be preserved by the companies, the amount of asks in addition to types of asks coming from different consumers. In the event the client(s) started off sending this obtain this keep track of will certainly keep track of the many obtain routine till it achieve to assistance aspect. So the many obtain method will be watched in the right log file on stub aspect. The full architecture will be based upon SAO(service driven architecture), wherever consumers will surely have continual companies from assistance. By means of encrypting the results plus the critical will be ceiled into your obtain data. After that the results will be encrypted. And so the protection will be preserved via consumers stub in addition to assistance skeleton level(proxy structure). With assistance aspect this assistance won't have strong conversation, the reason is this protection. Often stub as well as many intermediately method need this client’s obtain in addition to which will be rerouted to assistance. So assistance will certainly method this obtain plus the obtain that is ready-made on the assistance throughout secured way.

REFERENCES:

[1.] K. D. Bowers, A. Juels, and A. Oprea, "*Proofs of retrievability: Theory and implementation*," Cryptology ePrint Archive, Report 2008/175, 2008.

[2.] Q. Wang, K. Ren, W. Lou, and Y. Zhang, "*Dependable and secure sensor data storage with dynamic integrity assurance*," in Proc. of IEEE INFOCOM'09, Rio de Janeiro, Brazil, Appril 2009.

[3.] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "*Provable data possession at untrusted stores*," in Proc. of CCS'07. New York, NY, USA: ACM, 2007, pp. 598–609.

[4.] A. Juels and B. S. Kaliski, Jr., "*Pors: proofs of retrievability for large files*," in Proc. of CCS'07. New York, NY, USA: ACM, 2007, pp. 584–597.

[5.] M. Naor and G. N. Rothblum, "*The complexity of online memory checking*," in Proc. of FOCS'05, 2005, pp. 573–584.

[6.] H. Shacham and B. Waters, "*Compact proofs of retrievability*," in Proc. of ASIACRYPT'08. Springer-Verlag, 2008, pp. 90–107.

[7.] T. Schwarz and E. L. Miller, "*Store, forget, and check: Using algebraic signatures to check remotely administered storage*," in Proc. of ICDCS'06, 2006.

[8.] E.-C. Chang and J. Xu, "*Remote integrity check with dishonest storage server*," in Proc. of ESORICS'08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 223–237.

[9.] M. A. Shah, R. Swaminathan, and M. Baker, "*Privacy-preserving audit and extraction of digital contents*," Cryptology ePrint Archive, Report 2008/186, 2008.

[10.] A. Oprea, M. K. Reiter, and K. Yang, "*Space-efficient block storage integrity*," in Proc. of NDSS'05, 2005.