Implementation and Performance Analysis of Attribute-Based Encryption(ABE) in ICN

Nurefşan Sertbaş and Samet Aytaç
Bogazici University, Department of Computer Engineering
34342, Bebek, Istanbul, Turkey

***Abstract*—Information-centric networking(ICN) is a new architecture which uses named information (data object) instead of end-to-end communication. In ICN, data objects are cached in ICN routers and it can be retrieved from this in-network storage whenever there is a request for that content. This architecture decreases the traffic load and delay. From the security perspective, ICN concentrates on securing data objects instead of ensuring the security of end-to-end communication link. Although this mechanism enables efficient content delivery for consumers, it may cause some security vulnerabilities such as restricting unauthorized user to gain access to the content. From security point-of-view, it is important that unauthorized users should not be able to access the content during the retrieval phase. On the other hand, the owner of the content is not responsible from the access control since he does not have control over distributed caches.Thus, there should be an efficient access control policy in order to provide secure information centric operation. In this study, we present Attribute Based Encryption (ABE) as an access control mechanism for ICNs as defined in [1]. Moreover, we experiment how our system performs with different attribute numbers and file sizes in this paper.**

I. Introduction

Fast delivery of content is one of the essential necesities of Internet of Things(IoT) by not decreasing the security level of the data. In IoT environment, it is typical that same data can be requested more than once. Therefore, caching is instrumental to eliminate extra transmissions. ICN can be used for this purpose effectively over battery constraint IoT devices. Moreover, it is needed to make the data secure before caching process. ABE is promising in that regard since it allows several different group of users by defining only attributes, not the identities [2].

In [2], energy related problems are partially solved using ICN approach to overcome extra transmissions but it is still problematic because of the memory usage and computational load of the ABE implementation. In project we will work on implementing ABE as an access control mechanism for ICN.

The remainder of this paper is organized as follows. Section II goes through the related works. Section III presents the system models. We evaluate the performance of proposed solution in Section IV and conclude this paper in Section V.

II. Related Work

In this section, we will provide background information about IoT, ICN and ABE. Then, we will introduce related research results about usage of ABE in ICN.

A. Internet of Things

As Internet technologies are expanding and integrating itself more into our lives, new paradigms such as IoT are emerged in daily use. IoT is network connectivity that enables things or objects such as -tags,sensors,mobile phones- to interact with each other.

Although IoT is seen as a part of future Internet, it already has many different application areas. Smart home concept is the largest application area of IoT. Smart home gives owner the ability to customize and control home environment for increased security and efficient energy management. There are hundreds of IoT technologies available for monitoring and building smart homes. Moreover, IoT can be used for wearable technologies, health-care, transportation and logistic and futuristic technologies like self driving cars[3].

IoT needs to supply full interoperability of interconnected devices while guaranteeing trust security and privacy. Moreover, unlikely traditional networks, IoT devices has low computational and energy resources. Thus, IoT solution has to consider about low resource capacity. According to [4], there is 30 % increase in the number of connected devices in 2016 as compared to 2015 with 6.4 billion IoT devices entering the realm of IoT. The number is further expected to increase to 26 billion by 2020. This enormous number of devices will lead scalability issues too.

With the rise of IoT concept, security of IoT devices becomes more critical issue. When it is considered the biggest DDos attack of history is made by IoT devices[5],someone can understand how important to ensure security of IoT devices . Due to the it's nature, most of the IoT devices use wireless communication. Thus, they are vulnerable against eavesdropping attacks.

Furthermore, most of the IoT devices has low computation power and and fast battery depletion problem. It makes impossible to implement complex security schemes on devices. Any security solutions which will be implemented over IoT architecture should be aware of energy and computation power constrains.

B. Information-Centric Networks

Information Centric Network is one of the most promising approaches for future Internet since estimations shows that video traffic will reach 79% of the Internet traffic by 2018[6]. ICN architecture leverages in-network storage. In ICN, content owner publish the content and contents are stored in caches.

When a host wants to reach a content, it fetch the content directly from cache instead of communication with content owner. With content caching, there will be reduce in congestion and improve in delivery speed.

With all its benefits, ICN architecture comes with many drawbacks. First of all, number of contents in Internet is much more greater than number of hosts. For this reason, ICN routing system has a harder job than todays global IP routing. An intelligent solution to detect duplicated contents may be needed[7]. Moreover, switching from host centric network to ICN network will be incredible expensive issue. All infrastructures are needed to be ICN aware to employ ICN.

Security of ICN is one of the most important research topics. Unlike host-centric security , ICN requires location independent security mechanism to enable ubiquitous in-network caching system. Endpoint security is not sufficient for ICN. It needs to be sure contents are also secure itself. Therefore, security model for ICN should provide an information oriented data integrity and authenticity check mechanism. Moreover, there are ICN specific attacks like content poisoning and cache pollution attacks[8]. While the goal of content poisoning attack is to fill the routers caches with fake contents, the goal of cache pollution attack is degrade cache effectiveness and increase the content retrieval latency.

C. Attribute Based Encryption

In traditional network architecture, confidentiality is provided by the owner of the data or trusted third parties. However, it is nearly impossible for the owner to control the content in ICN [9]. At that point, ABE is proposed which relies on attaching access control policies to the data content itself. There is no need to share common secret keys with each other. It makes easy to perform key management and provides highly scalable structure. Moreover, it eliminates the necessity for encrypting data for each user type separately by giving access to clients that have the correct set of attributes.

There exist four phases of ABE whose block diagram is given in Fig. 3 [10]. In the first phase of ABE, master and public keys which will be used in encrypt and decrypt operations in the future are generated and published. Then, message is encrypted with published public key and given access policy. In addition to public and master keys, a set of attributes are used to produce decryption key in the third phase. In the decryption phase, the ciphertext is decrypted by using decryption key and master key. If the user satisfies the access policy, he/she is able to reproduce the original message.

There are two different widely used ABE schemes:

1) Key Policy ABE: Key Policy ABE is introduced in [11]. In KP-ABE ciphertexts are labeled with sets of attributes and private keys are associated with access structures that control which ciphertexts a user is able to decrypt. Scheme consist of four algorithms.

- **Setup:** Master key MK and public parameters PK are generated.
- **Encrypt:** This algorithm takes a message M, a set of attributes γ and public parameters PK as input. It generates ciphertext C.
- **Key Generation:** This algorithm takes access structure A,public parameters PK and master key MK as input. It generates user's private key SK.
- **Decrypt:** This algorithm takes ciphertext C that was encrypted under the set of attributes γ, private key SK for access control structure A and public parameters PK.If γ satisfies A, algorithm decrypt ciphertext and outputs message M.

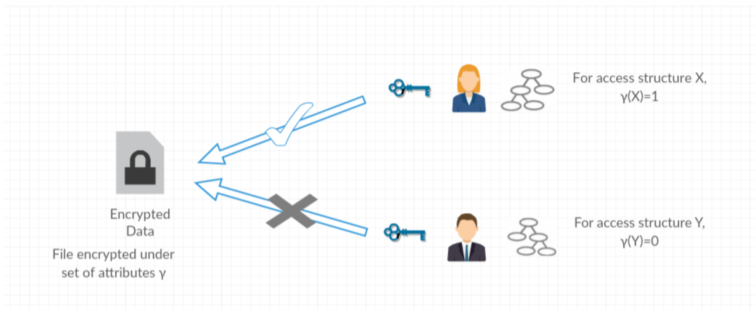


Fig. 1. KP-ABE in practice.

2) Ciphertext Policy ABE: Ciphertext Policy ABE is introduced in [12]. The concept is very similar with KP-ABE but unlikely KP-ABE, in CP-ABE, ciphertexts are associated with an access structure and attributes are used to construct private keys. The scheme consist of four algorithms. In scheme, master key(MK) represents key of authentication center and public parameters(PK) represents key of each attribute.

- **Setup:** MK and PK are generated.
- **Encrypt:** This algorithm takes a message M, an access structure A and PK as input. It generates ciphertext C.
- **Key Generation:** This algorithms takes set of attributes γ and master key MK as an input. In generates user's private key SK.
- **Decrypt:** This algorithm takes ciphertext C that contains access structure A, private key SK for set of attributes γ and public parameters PK. If γ satisfies A, algorithm decrypt ciphertext and outputs message M.

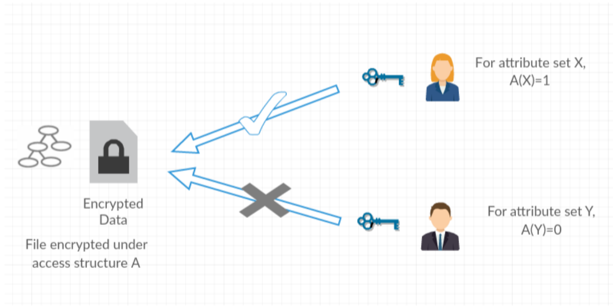


Fig. 2. CP-ABE in practice.

D. ABE in ICN

In ICN, objects are cached in different locations and they can be easily retrieved whenever there is a request for that content.

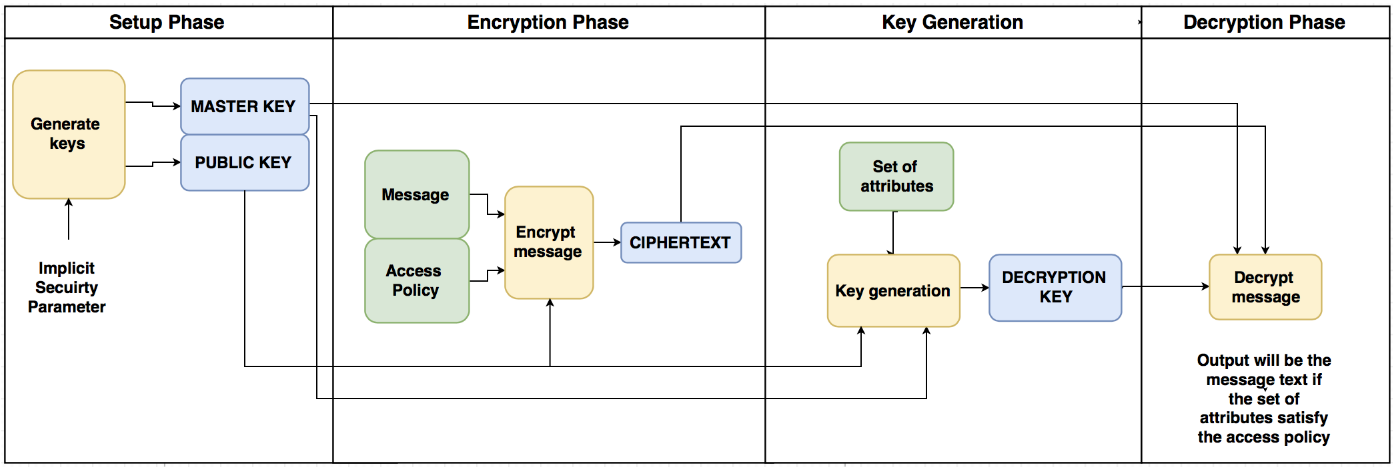


Fig. 3. Block diagram for ABE functions.

Then, ABE proposes a practical approach for that architecture although it reveals some important issues related with management and security. First of all, due to the distributed nature of the system attribute management is difficult. Moreover, the owner of the content is not responsible from the access control because he does not have control over distributed caches. Thus, there should be an access control policy.

In [1], they used modified version of ABE for ICN naming scheme in order to preserve privacy in access control and provide flexible attribute management solution. In their ontology based attribute management approach, the content owner pushes his data by specifying which attributes are used for access control and which ones are used for content search. Therefore, content names are protected based on attributes and attribute combination operations in the access control policies are supported with reduced costs. It can be seen as privacy preserving access policy and the flexible attribute management solution through the ICN naming scheme. Also in [9], they use combination of ICN names with Boolean expressions of constraints on attributes to give users opportunity to express more complex requests.

Other significant function of ICN is name based routing which can be too restrictive at times. In [9], they use attribute based routing in ICN environment. The constraints are expressed on attributes of data and it is forwarded in the case of matching constraints.

In [2], ABE is used at sensor nodes before pushing sensing data to the ICN environment. They point out that ABE can be implemented for resource constrained devices that have low memory and battery power although it is a processing intensive task. The significant factor is choosing right level of security and the number of levels will be used .

Ubiquitous adaption of mobile and resource constrained devices lead to research over performance of cryptographic algorithms on these devices. In [10], ABE has been applied to smart phone devices by considering some constraints such as battery and memory. Similarly, a Java-based implementation of ABE on Android smart phones has been carried out in [13]. Another IoT focused paper describes a new technique that applied on ABE algorithm to overcome the computational load for resource-constrained devices [14]. Also, feasibility of ABE on sensors and several ways of improving performance of ABE in such devices are examined in [15].

III. Project Work

First of all we decided to use CCN-lite which is an implementation of the Content Centric Networking protocol CCNx in the project [16]. We have planned to build to network topology on CCN-lite as given in Fig. 4. Also, we have planned to use either simulation or hardware environment as a test setup.

A. Real-time Implementation Attempt

Our first attempt was to implement the system with hardware environment. There are two main blocks in the setup: CCN client and sensors acting as data sources.

In this scenario, sensors produce data which users will request to access. These contents are encrypted on sensors using ABE. Then, the encrypted data are passed by the CCN gateway to client Android devices. The CCN client which is an Android phone will decrypt that data by using its access policy.

Firstly, we planned to build sensor environment by using Arduino Mega 2560 boards for sensors and XBee modules for communication of sensors. Our plan was to build RIOT OS environment on sensors and by using CCN-lite, we would establish communication between these sensors. We prefered to use RIOT OS, it is optimized for resource constrained devices [17]. We managed to configure communication between sensors by using X-CTU software. Our test application for communication which is written in Arduino runs perfectly. Moreover, we built RIOT OS environment in our Linux machine.

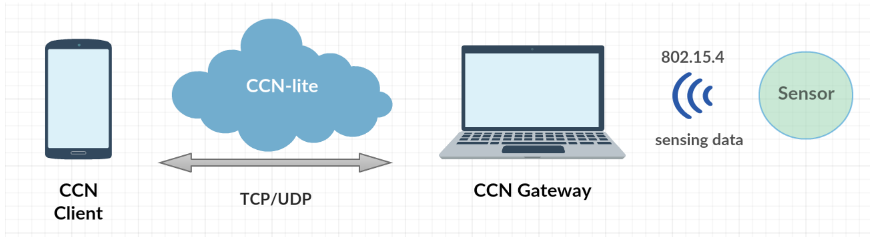


Fig. 4. Block diagram for the implementation adapted from [2].

Second part of the implementation includes setting up CCN-lite environment into both the computer that will serves as a CCN gateway and an Android phone that will serve as a CCN client. We have used CCN-lite 0.3.0 version to setup an environment in Linux computer. In order to test the installation, we have followed the given tutorial to build a simple content look-up [18].

However, we have experienced problems about running RIOT OS application in our boards. We have examined all possible sources but we could not solve the problem. In official website of RIOT OS, it is said that RIOT OS has full support to Arduino Mega 2560 boards. But it turns out RIOT OS has limited support to Arduino Mega 2560 boards and it does not include CCN-lite functionalities. Thus, it becomes impossible to implement the project with using Arduino Mega 2560 boards. Therefore; we choose to proceed with the simulation instead of the hardware environment.

B. Simulation Attempt

1) Simulation Environment Setup: In simulation part, we have used CCN-lite environment which is already installed on a MacOS machine. Then, we aimed to simulate the scenario given in Fig. 5. In this topology, we have two clients in order to test the basic operations of ICN. We have used Client1 as a data owner who pushes his data to the ICN network and it will be cached in the network nodes. Client2 is used as a requester for that data object.

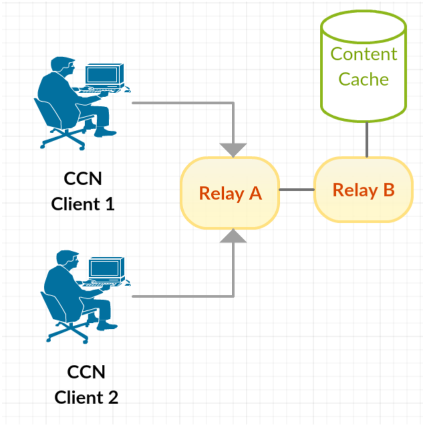
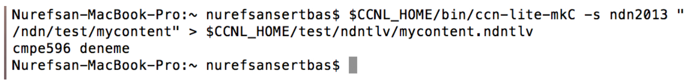


Fig. 5. Block diagram for the implementation.

Note that in the second part of the simulation this push-request operations will be done based on the attributes with the help of ABE. However, it is assumed that any one can access the content for the first part. In order to establish mentioned communication between clients in ICN architecture we have carried out following steps [18].

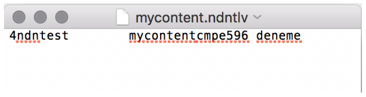
Producing content: We have used ndn2013 as a wire fire format. It is one of the supported formats in CCN-lite. Then, we have created an content object by using ccn-lite-mkC function of the library. The command and its usage is given in Fig. 6.



```
Nurefsan-MacBook-Pro:~ nurefsansertbas$ $CCNL_HOME/bin/ccn-lite-mkC -s ndn2013 "
/ndn/test/mycontent" > $CCNL_HOME/test/ndntlv/mycontent.ndntlv
cmpe596 deneme
Nurefsan-MacBook-Pro:~ nurefsansertbas$
```

Fig. 6. Producing content.

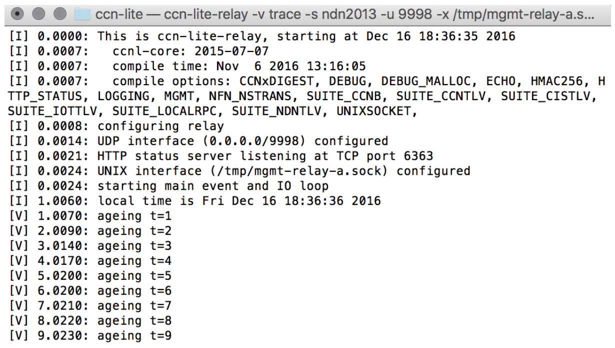
After executing above command, we have create a content with "cmpe596 deneme" with prefix and it is stored in the file which is given in Figure 7.



```
4ndntest          mycontentcmpe596 deneme
```

Fig. 7. Cached content

Starting relay A and B: We have started relay A with following command


ccn-lite — ccn-lite-relay -v trace -s ndn2013 -u 9998 -x /tmp/mgmt-relay-a.s...

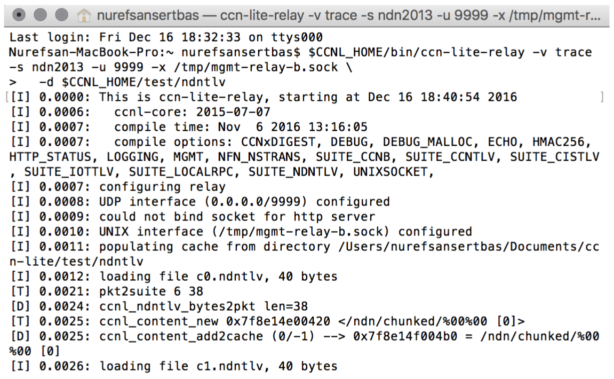
```
[I] 0.0000: This is ccn-lite-relay, starting at Dec 16 18:36:35 2016
[I] 0.0007:   ccnl-core: 2015-07-07
[I] 0.0007:   compile time: Nov  6 2016 13:16:05
[I] 0.0007:   compile options: CCNxDIGEST, DEBUG, DEBUG_MALLOC, ECHO, HMAC256, H
TTP_STATUS, LOGGING, MGMT, NFN_NSTRANS, SUITE_CCNB, SUITE_CCNTLV, SUITE_CISTLV,
SUITE_IOTTLV, SUITE_LOCALRPC, SUITE_NDNTLV, UNIXSOCKET,
[I] 0.0008: configuring relay
[I] 0.0014: UDP interface (0.0.0.0/9998) configured
[I] 0.0021: HTTP status server listening at TCP port 6363
[I] 0.0024: UNIX interface (/tmp/mgmt-relay-a.sock) configured
[I] 0.0024: starting main event and IO loop
[I] 1.0060: local time is Fri Dec 16 18:36:36 2016
[V] 1.0070: ageing t=1
[V] 2.0090: ageing t=2
[V] 3.0140: ageing t=3
[V] 4.0170: ageing t=4
[V] 5.0200: ageing t=5
[V] 6.0200: ageing t=6
[V] 7.0210: ageing t=7
[V] 8.0220: ageing t=8
[V] 9.0230: ageing t=9
```

Fig. 8. Starting relay A.

We have used the following options.

-s for specifying ndn2013 format
-v for setting DEBUGLEVEL as trace
-u 9998 for setting listening port of the relay
-x $/tmp/mgmt-relay-a.sock$ for setting an Unix socket which will be used later

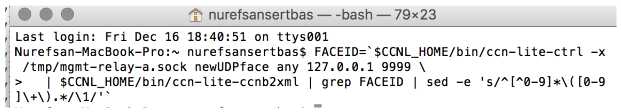
Note that we will use node A as a forwarding node and node B as a final/cache node. Therefore, we have started relay B with following command by adding -d option which adds all content objects from a directory to the cache of the relay.



```
Last login: Fri Dec 16 18:32:33 on ttys000
Nurefsan-MacBook-Pro:~ nurefsansertbas$ $CCNL_HOME/bin/ccn-lite-relay -v trace
-s ndn2013 -u 9999 -x /tmp/mgmt-relay-b.sock \
>   -d $CCNL_HOME/test/ndntlv
[I] 0.0000: This is ccn-lite-relay, starting at Dec 16 18:40:54 2016
[I] 0.0006:   ccnl-core: 2015-07-07
[I] 0.0007:   compile time: Nov  6 2016 13:16:05
[I] 0.0007:   compile options: CCNxDIGEST, DEBUG, DEBUG_MALLOC, ECHO, HMAC256,
HTTP_STATUS, LOGGING, MGMT, NFN_NSTRANS, SUITE_CCNB, SUITE_CCNTLV, SUITE_CISTLV
, SUITE_IOTTLV, SUITE_LOCALRPC, SUITE_NDNTLV, UNIXSOCKET,
[I] 0.0007: configuring relay
[I] 0.0008: UDP interface (0.0.0.0/9999) configured
[I] 0.0009: could not bind socket for http server
[I] 0.0010: UNIX interface (/tmp/mgmt-relay-b.sock) configured
[I] 0.0011: populating cache from directory /Users/nurefsansertbas/Documents/cc
n-lite/test/ndntlv
[I] 0.0012: loading file c0.ndntlv, 40 bytes
[T] 0.0021: pkt2suite 6 38
[D] 0.0024: ccnl_ndntlv_bytes2pkt len=38
[T] 0.0025: ccnl_content_new 0x7f8e14e00420 </ndn/chunked/%00%00 [0]>
[D] 0.0025: ccnl_content_add2cache (0/-1) --> 0x7f8e14f004b0 = /ndn/chunked/%00
%00 [0]
[I] 0.0026: loading file c1.ndntlv, 40 bytes
```

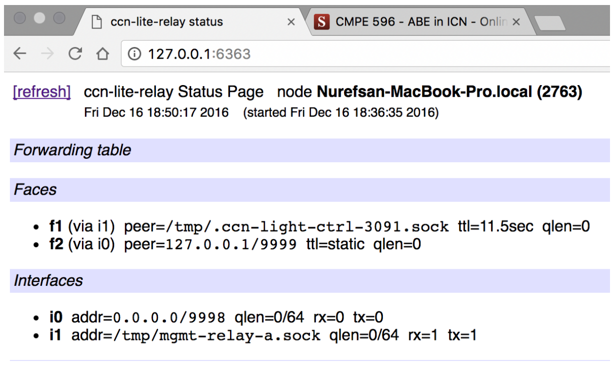
Fig. 9. Starting relay B.

Adding forwarding rule between A and B: We should add forwarding rule from A to B for prefix. Therefore, when an interest comes to the node A with prefix, it knows that which interface to forward. For that purpose, we have created an interface at node A with following command in Figure 10.



```
Last login: Fri Dec 16 18:40:51 on ttys001
Nurefsan-MacBook-Pro:~ nurefsansertbas$ FACEID=`$CCNL_HOME/bin/ccn-lite-ctrl -x
 /tmp/mgmt-relay-a.sock newUDPface any 127.0.0.1 9999 \
>   | $CCNL_HOME/bin/ccn-lite-ccnb2xml | grep FACEID | sed -e 's/^[^0-9]*\([0-9
]\+\).*/\1/'`
```

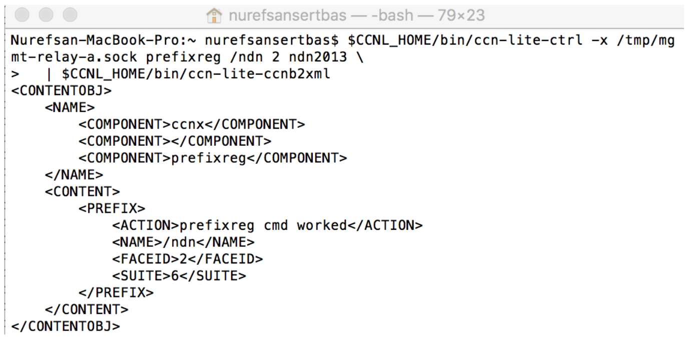
Fig. 10. Creating an interface at relay A.



```
[refresh]  ccn-lite-relay Status Page   node Nurefsan-MacBook-Pro.local (2763)
           Fri Dec 16 18:50:17 2016    (started Fri Dec 16 18:36:35 2016)

Forwarding table

Faces

  • f1 (via i1)  peer=/tmp/.ccn-light-ctrl-3091.sock  ttl=11.5sec  qlen=0
  • f2 (via i0)  peer=127.0.0.1/9999  ttl=static  qlen=0

Interfaces

  • i0  addr=0.0.0.0/9998  qlen=0/64  rx=0  tx=0
  • i1  addr=/tmp/mgmt-relay-a.sock  qlen=0/64  rx=1  tx=1
```

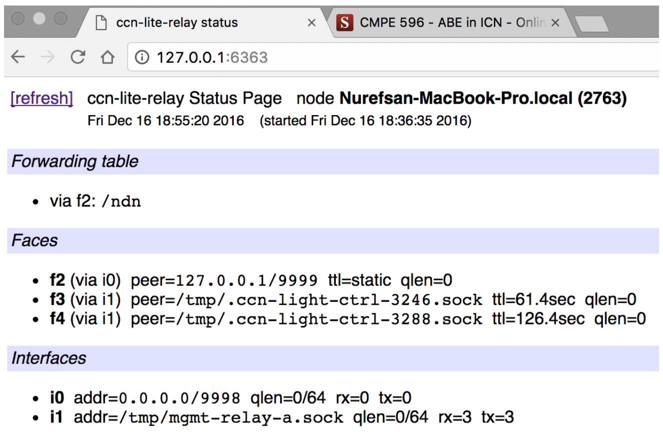
Fig. 11. Checking status of an interface from HTTP browser.

We have created an interface at node A but we did not connect it to the node B. Therefore, we have used following command to create forwarding rule for prefix.



```
Nurefsan-MacBook-Pro:~ nurefsansertbas$ $CCNL_HOME/bin/ccn-lite-ctrl -x /tmp/mg
mt-relay-a.sock prefixreg /ndn 2 ndn2013 \
>   | $CCNL_HOME/bin/ccn-lite-ccnb2xml
<CONTENTOBJ>
    <NAME>
        <COMPONENT>ccnx</COMPONENT>
        <COMPONENT></COMPONENT>
        <COMPONENT>prefixreg</COMPONENT>
    </NAME>
    <CONTENT>
        <PREFIX>
            <ACTION>prefixreg cmd worked</ACTION>
            <NAME>/ndn</NAME>
            <FACEID>2</FACEID>
            <SUITE>6</SUITE>
        </PREFIX>
    </CONTENT>
</CONTENTOBJ>
```

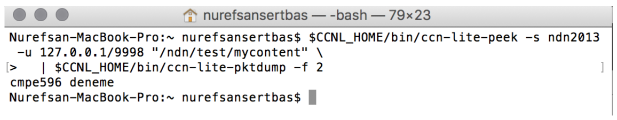
Fig. 12. Creating a forwarding rule from A to B.



```
[refresh]  ccn-lite-relay Status Page   node Nurefsan-MacBook-Pro.local (2763)
           Fri Dec 16 18:55:20 2016    (started Fri Dec 16 18:36:35 2016)

Forwarding table

  • via f2: /ndn

Faces

  • f2 (via i0)  peer=127.0.0.1/9999  ttl=static  qlen=0
  • f3 (via i1)  peer=/tmp/.ccn-light-ctrl-3246.sock  ttl=61.4sec  qlen=0
  • f4 (via i1)  peer=/tmp/.ccn-light-ctrl-3288.sock  ttl=126.4sec  qlen=0

Interfaces

  • i0  addr=0.0.0.0/9998  qlen=0/64  rx=0  tx=0
  • i1  addr=/tmp/mgmt-relay-a.sock  qlen=0/64  rx=3  tx=3
```

Fig. 13. Checking status of forwarding rule from HTTP browser.

Sending an interest to A for an interest: We have used $ccn-lite-peek$ utility of the library to send an interest to Relay A. Relay A receives the interest and forwards it to the Relay B. Since the Relay B has that content, it will respond with the object for that request.



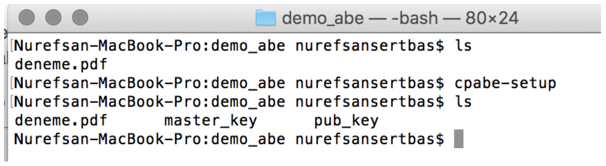
```
Nurefsan-MacBook-Pro:~ nurefsansertbas$ $CCNL_HOME/bin/ccn-lite-peek -s ndn2013
 -u 127.0.0.1/9998 "/ndn/test/mycontent" \
>   | $CCNL_HOME/bin/ccn-lite-pktdump -f 2
cmpe596 deneme
Nurefsan-MacBook-Pro:~ nurefsansertbas$
```

Fig. 14. Sending an interest.

Also, we used $ccn-lite-pktdump$ command as an additional option to change ndn2013 format to the readable format at the output.

2) ABE Phase: We have used cpabe toolkit[19] which provides a set of programs implementing a ciphertext-policy attribute-based encryption scheme [12]. This toolkit is developed by John Bethencourt, Amit Sahai and Brent Waters who are creator of CP-ABE scheme. We aimed to show its usage in a simple scenario as following.

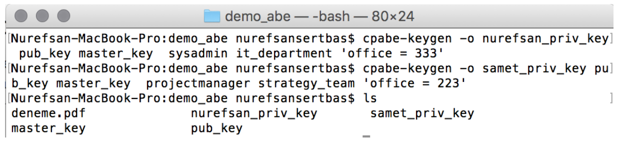
First of all, $cpabe - setup$ function is called in order to generate the required public parameters as represented in Setup phase of Fig. 3.



```
Nurefsan-MacBook-Pro:demo_abe nurefsansertbas$ ls
deneme.pdf
Nurefsan-MacBook-Pro:demo_abe nurefsansertbas$ cpabe-setup
Nurefsan-MacBook-Pro:demo_abe nurefsansertbas$ ls
deneme.pdf      master_key      pub_key
Nurefsan-MacBook-Pro:demo_abe nurefsansertbas$
```

Fig. 15. Setup phase of ABE.

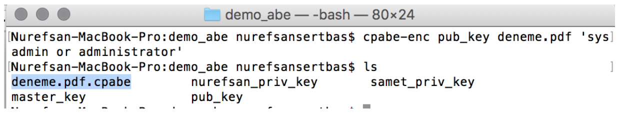
In the second phase, we have created private keys for clients by specifying their attributes. As given in Fig. 16, we have created two clients with different attributes.



```
Nurefsan-MacBook-Pro:demo_abe nurefsansertbas$ cpabe-keygen -o nurefsan_priv_key
 pub_key master_key  sysadmin it_department 'office = 333'
Nurefsan-MacBook-Pro:demo_abe nurefsansertbas$ cpabe-keygen -o samet_priv_key pu
b_key master_key  projectmanager strategy_team 'office = 223'
Nurefsan-MacBook-Pro:demo_abe nurefsansertbas$ ls
deneme.pdf              nurefsan_priv_key       samet_priv_key
master_key              pub_key
```

Fig. 16. Key generation phase of ABE.

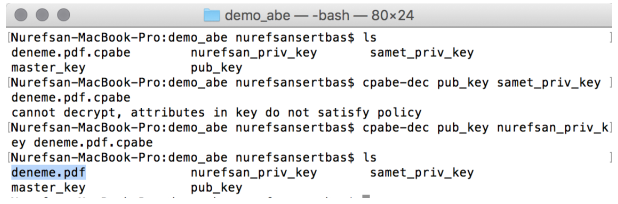
Then, we encrypt the file named $deneme.pdf$ with a given access policy. It means that only the users whose attributes are sysadmin or administrator are able to decrypt the file. After this step, we produce encrypted file $deneme.pdf.cpabe$



```
Nurefsan-MacBook-Pro:demo_abe nurefsansertbas$ cpabe-enc pub_key deneme.pdf 'sys
admin or administrator'
Nurefsan-MacBook-Pro:demo_abe nurefsansertbas$ ls
deneme.pdf.cpabe        nurefsan_priv_key       samet_priv_key
master_key              pub_key
```

Fig. 17. Encryption phase of ABE.

As a last step, we have tried to decrypt the file with two different users. In the first attempt, the file does not decrypted since the user does not satisfy the access policy. In the second attempt, the user is able to decrypt the file and get the original content as given in Fig. 18.



```
Nurefsan-MacBook-Pro:demo_abe nurefsansertbas$ ls
deneme.pdf.cpabe        nurefsan_priv_key       samet_priv_key
master_key              pub_key
Nurefsan-MacBook-Pro:demo_abe nurefsansertbas$ cpabe-dec pub_key samet_priv_key
deneme.pdf.cpabe
cannot decrypt, attributes in key do not satisfy policy
Nurefsan-MacBook-Pro:demo_abe nurefsansertbas$ cpabe-dec pub_key nurefsan_priv_k
ey deneme.pdf.cpabe
Nurefsan-MacBook-Pro:demo_abe nurefsansertbas$ ls
deneme.pdf              nurefsan_priv_key       samet_priv_key
master_key              pub_key
```

Fig. 18. Decryption phase of ABE.

IV. Experimental Results

In this section, we will show and discuss about our experiment results. We run our experiments on MacOS with the processor 2,9 GHz Intel Core i5 and 16 GB 1867 MHz DDR3 RAM.

A. *Effect of number of attributes*

As a first experiment, we measure effect of number of attributes on performance. We use 6byte fixed file size for this experiment. Our results can be seen in Table 1 and Fig20. It is obvious that with the increase of number of attributes, object encrypt and pushing to the cache time is increased. Our test results are correlated with the results of [12] which can be seen in Fig 19. The results of [12] shows that there is linear correlation between number of attributes and object encrypt time. Moreover, our results shows that there is no strongly correlation between number of attributes and object decrypt time. Our results for decryption time also corresponds with [12].

TABLE I. Processing time with various number of attributes

	Depth of tree	**Num of attributes**	**Object Encrypt & Push**	**Object Request & Decrypt**
1	0	1	0.162 sec	0.156 sec
2	1	2	0.166 sec	0.143 sec
3	2	4	0.190 sec	0.144 sec
4	3	8	0.207 sec	0.166 sec
5	4	16	0.290 sec	0.165 sec
6	5	32	0.406 sec	0.168 sec

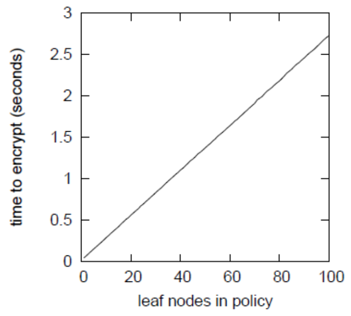


Fig. 19. Experiment results of [12]

B. *Effect of file size*

As second experiment, we measure effect of number of file size on performance. We use fixed access policy with 16 attributes. Our results can be seen in Table 2 and Fig21. We are aware that the file sizes which we use in our experiment is not realistic for IoT environment but to show the effect of file size on computation time, we need to use exaggerated file sizes. It is obvious that with the increase of file size, object encrypt and pushing to the cache time is increased. This behaviour is expected since both object encrypt and pushing object to the cache time is related with file size.

TABLE II. Processing time with various file sizes

	File Size	**Object Encrypt & Push**
1	1Kb	0.317 sec
2	5 Mb	0.383 sec
3	10 Mb	0.449 sec
4	50 Mb	0.946 sec
5	100 Mb	1.557 sec

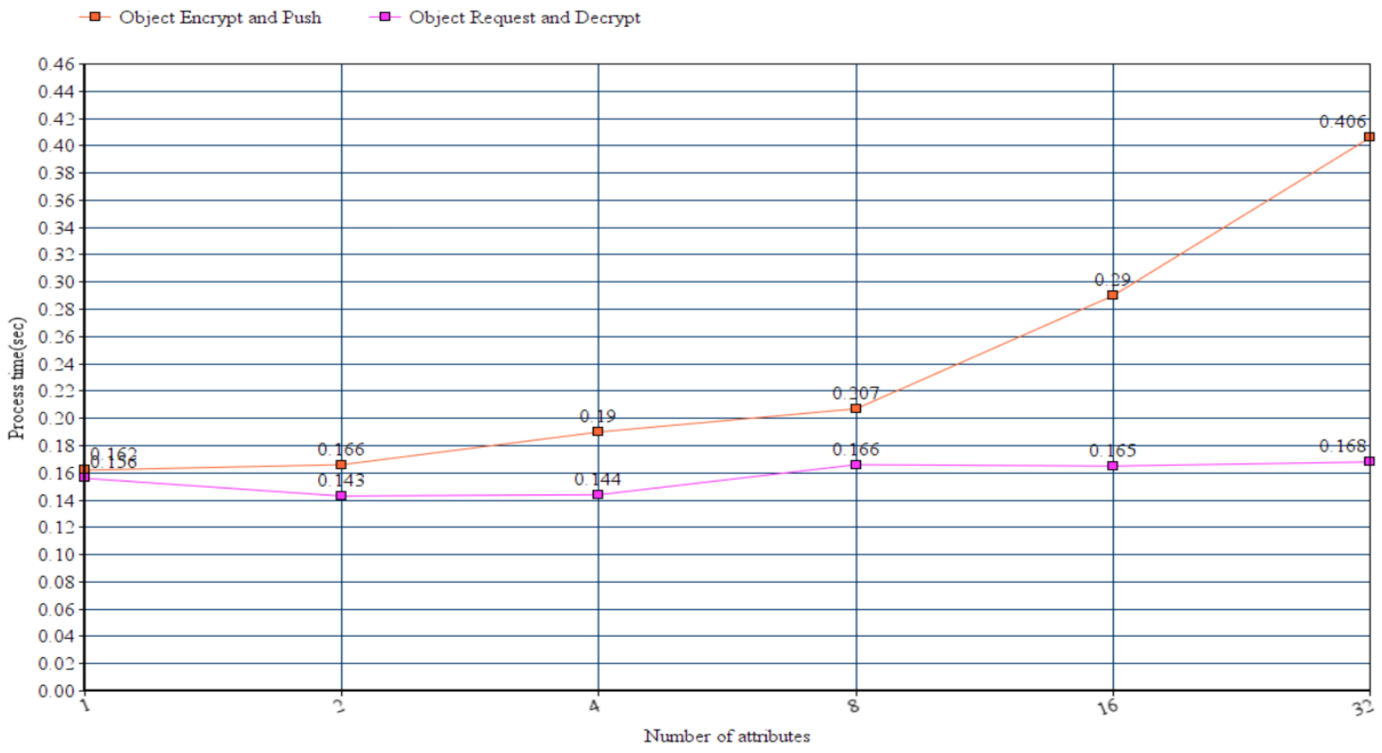


Fig. 20. Processing time with various number of attributes

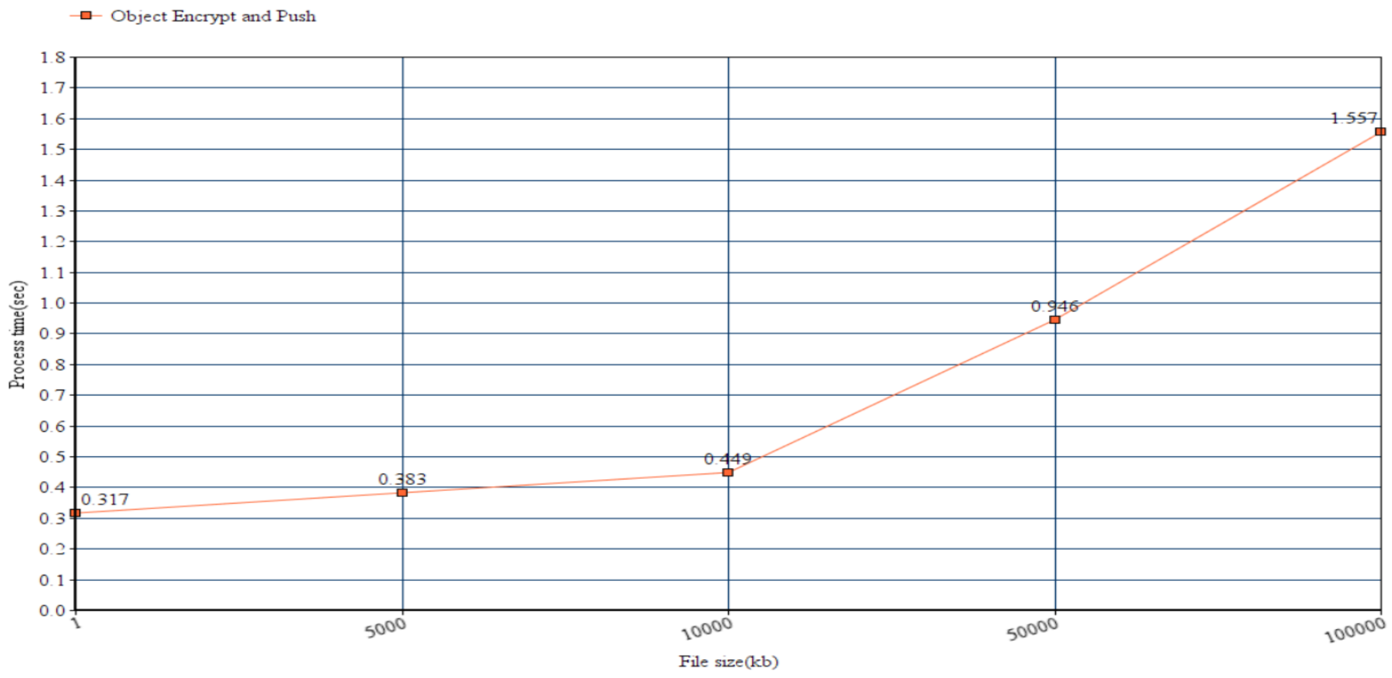


Fig. 21. Processing time with various file sizes

V. Conclusion

In this project, we have successfully simulated the ICN environment. Moreover, we have implemented CP-ABE scheme over ICN to ensure object security. We have performed some experiments on our setup to measure the effect of attribute number and file size to the performance. The project source code is available at https://github.com/sertbasn1/ABE_ICN_Project.git.Implementation and energy consumption analysis of ABE on a real-time ICN is left as a future work.

References

[1] B. Li, A. P. Verleker, D. Huang, Z. Wang, and Y. Zhu, "Attribute-based access control for icn naming scheme," in *Communications and Network Security (CNS), 2014 IEEE Conference on*. IEEE, 2014, pp. 391–399.

[2] A. M. Malik, J. Borgh, and B. Ohlman, "Attribute-based encryption on a resource constrained sensor in an information-centric network," in *Proceedings of the 2016 conference on 3rd ACM Conference on Information-Centric Networking*. ACM, 2016, pp. 217–218.

[3] L. Atzoria, A. Ierab, and G. Morabitoc, "The internet of things: A survey," *Future Generation Computer Systems*, vol. 54, 2010.

[4] R. van der Meulen. (2015) Gartner says 6.4 billion connected "things" will be in use in 2016, up 30 percent from 2015. [Online]. Available: http://www.gartner.com/newsroom/id/3165317

[5] N. Woolf. (2016) Ddos attack that disrupted internet was largest of its kind in history, experts say. [Online]. Available: https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet

[6] Cisco. (2016) White paper: Cisco vni forecast and methodology, 2015-2020. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html

[7] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Communications Magazine*, 2012.

[8] R. Tourani, T. Mick, S. Misra, and G. Panwar, "Security, privacy, and access control in information-centric networking: A survey," 2016.

[9] M. Ion, J. Zhang, and E. M. Schooler, "Toward content-centric privacy in icn: Attribute-based encryption and routing," in *Proceedings of the 3rd ACM SIGCOMM workshop on Information-centric networking*. ACM, 2013, pp. 39–40.

[10] M. Ambrosin, M. Conti, and T. Dargahi, "On the feasibility of attribute-based encryption on smartphone devices," in *Proceedings of the 2015 Workshop on IoT challenges in Mobile and Industrial Systems*. ACM, 2015, pp. 49–54.

[11] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM Conference on Computer and Communications Security*, ser. CCS '06. New York, NY, USA: ACM, 2006, pp. 89–98. [Online]. Available: http://doi.acm.org/10.1145/1180405.1180418

[12] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *2007 IEEE symposium on security and privacy (SP'07)*. IEEE, 2007, pp. 321–334.

[13] X. Wang, J. Zhang, E. M. Schooler, and M. Ion, "Performance evaluation of attribute-based encryption: Toward data privacy in the iot," in *2014 IEEE International Conference on Communications (ICC)*. IEEE, 2014, pp. 725–730.

[14] N. Oualha and K. T. Nguyen, "Lightweight attribute-based encryption for the internet of things," in *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, Aug 2016, pp. 1–6.

[15] J. Borgh, "Attribute-based encryption in systems with resource constrained devices in an information centric networking context," Master's thesis, Uppsala University, 2016.

[16] "Ccn lite: Lightweight implementation of the content centric networking protocol." [Online]. Available: http://ccn-lite.net

[17] E. Baccelli, O. Hahm, M. Gunes, M. Wahlisch, and T. Schmidt, "Riot os: Towards an os for the internet of things," *2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, Apr 2013.

[18] cn uofbasel, "Cn-uofbasel/ccn-lite." [Online]. Available: https://github.com/cn-uofbasel/ccn-lite/blob/master/tutorial/tutorial.md

[19] "Ciphertext-policy attribute-based encryption." [Online]. Available: http://acsc.cs.utexas.edu/cpabe