01 Jan 2022

Targeted Content-Sharing in a Multi-Group Dtn Application using Attribute-Based Encryption

Xiaofei Cao

Shudip Datta

Ram Charan Bolla

Sanjay Kumar Madria
Missouri University of Science and Technology, madrias@mst.edu

Recommended Citation

X. Cao et al., "Targeted Content-Sharing in a Multi-Group Dtn Application using Attribute-Based Encryption," *Proceedings - IEEE International Conference on Mobile Data Management*, pp. 306 - 309, Institute of Electrical and Electronics Engineers, Jan 2022.
The definitive version is available at https://doi.org/10.1109/MDM55031.2022.00067

Targeted Content-Sharing in a Multi-group DTN Application using Attribute-Based Encryption

Xiaofei Cao
Dept of Computer Science
Missouri University of Science and Technology, USA
xiaofeicao0@gmail.com

Shudip Datta
Dept of Computer Science
Missouri University of Science and Technology, USA
sdnv5@mst.edu

Ram Charan Bolla
Dept of Computer Science
Missouri University of Science and Technology, USA
rbcn8@mst.edu

Sanjay Madria
Dept of Computer Science
Missouri University of Science and Technology, USA
madrias@mst.edu

2022 23rd IEEE International Conference on Mobile Data Management (MDM) | 978-1-6654-5176-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/MDM55031.2022.00067

***Abstract*—In a battlefield, multiple groups operate with different missions, but their missions and groups can dynamically change based on the evolving situation. Due to the unavailability of network infrastructure after deployment, group members form a Delay Tolerant Network (DTN) which is prone to security attacks. Hence, based on the mission attributes, group memberships, nodes' interests, and data tags determination, targeted contents need to be distributed in a secure fashion to different users. Though existing Attributes Based Encryption (ABE) can provide security of information, revoking a member from a group is always an issue in DTN as the Attribute Authority (*AA*) is unavailable to the DTN nodes. Therefore, we adopt the ReVO-ABE algorithm for a battlefield DTN application for targeted data forwarding based on mission attributes and content interests.**

***Index Terms*—Delay Tolerant Network, Attribute based Encryption**

I. Introduction

In a battlefield, multiple groups are assigned with multiple missions which evolves based on different circumstance. To handle the evolving situation, a member from some specific mission needs to be revoked or added to maintain the group. However, in a DTN network user revocation is a challenge due to high communication overhead and non-availability of *AA*. Previous asymmetrical encryption algorithms such as RSA and other *ABE* are not able to revoke users without regenerating and disseminating the security key for all the other users. To solve this problem, ReVO-ABE, an attribute-based asymmetrical encryption algorithm with revocation capability is proposed [1]. Different from the classic ABE scheme, it contains a new data structure called Revocation Tree (*RT*). The path from the root of the *RT* to the nodes that need to be revoked is called the Revocation List (*RL*). Using the *RL*, any user with the *RT* can revoke other users' ability to decrypt the data even if the data has been encrypted using the attributes of the users in the *RL*. Besides, the decryption ability of different data by the non-revoked users can also be controlled by encrypting the data using different sets of users-attributes as needed.

Funded by Air Force Research Lab (AFRL)

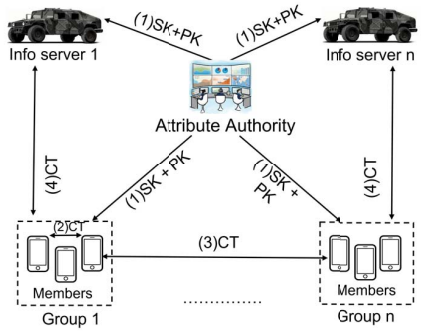

Fig. 1. System Architecture.
Here, $PK \rightarrow$ Public Key, $SK \rightarrow$ Security Key, $CT \rightarrow$ Cipher Text.
In step 1, IS and the nodes receive the PK and SK from the AA at bootstrap. Step 2, 3, and 4 show the CT forwarding between two nodes in a group, two nodes in two different groups, and one node & one IS respectively.

Based on the mission attributes, group memberships, nodes' interests, and data tags, targeted contents can be distributed securely using ReVO-ABE. First, DTN nodes need to fetch the necessary keys and *RT* from the *AA* at the bootstrap. Thus, DTN nodes can share the data in a secure fashion with the members of the same or different groups dynamically. While sharing data, DTN nodes can decide on users to be revoked for targeted data. Besides, nodes also consider sending meaningful data related to the missions by expressing the content of the data as captions to reduce the consumption of limited bandwidth. To generate the captions of data i.e. images, an object detection model using Tensorflow Lite is implemented at the DTN nodes. The captions of the data are also expressed as the probabilities of different objects' existence in the data so that the objects can be matched with the interests of the receiver node to decide which data to send first.

II. Architecture

Consider a battlefield scenario where three teams are working on different missions. $Team_1$ is scanning one part of the area for IED (Improvised Explosive Devices) detection,

2375-0324/22/$31.00 ©2022 IEEE
DOI 10.1109/MDM55031.2022.00067

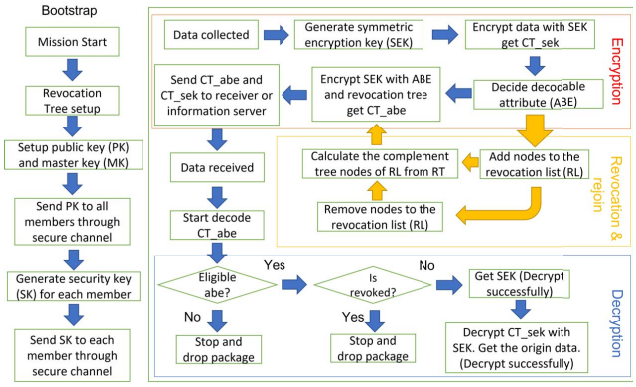

Fig. 2. Secure sharing and revocation flowchart

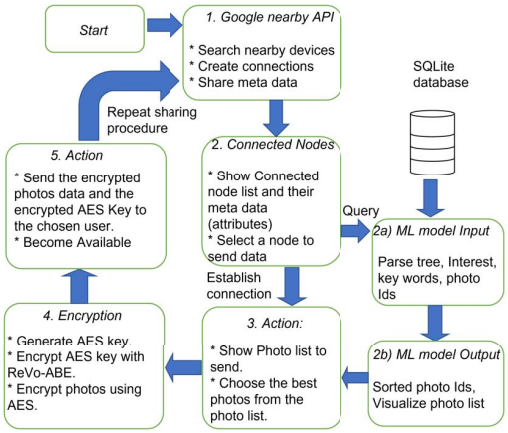

Fig. 3. Data sharing flow chart

$Team_2$ is monitoring the enemy territory, and $Team_3$ is searching for injured survivors. A Humvee, a moving node patrolling the region continuously collects and forwards content and can be connected to the command-and-control center (CC) when required. Thus, we designed the architecture as shown in figure 1. The CC acts as the Attribute Authority (*AA*), the Humvees act as the moving information server (*IS*) as they have a higher communication range and storage, and the team members of the three teams act as the DTN nodes. This architecture supports maximum reliability of information sharing by using peer-to-peer secure forwarding while reducing the latency by introducing multiple local and mobile *IS*.

A. Attribute Authority (AA)

We design the *AA* using ReVO-ABE [1] scheme for data security and user revocation. In the bootstrap phase, the *AA* sets up the public key (*PK*) and the master key (*MK*). The *AA* then generates the security key (*SK*) for each member. The *AA* sends PK and SK to each user through a secure channel. In the encryption phase, the user who has the *RT* can revoke/rejoin a user by adding or removing him from the *RL*. The decryption of ReVO-ABE needs to verify both the attributes security and the revocation security. The flowchart of the method is shown in figure 2. In the *Encryption* part, a node generates a symmetric encryption key (*SEK*) and encrypt the data (CT_{sek}) with it. The *SEK* is then encrypted (CT_{abe}) using *ABE* and *RT*. In the *Revocation and Rejoin* part, revoked nodes are added to the *RL* and and complement tree of *RL* from *RT* is stored for the subsequent encryption phases. In the *Decryption* part, if the receiver is not revoked and authorized to decrypt CT_{sek} and CT_{abe}, it can regenerate the original sent data.

B. DTN node

Upon receiving the *PK* and *SK* from the *AA*, DTN nodes share data among them when in communication range by encrypting the messages using necessary attributes which ensures the security of the information. While selecting a recipient and the message to send, a node follows the procedures of the scheme [2]. Hence, the nodes calculate the interest matching of the recipient with the messages so that they can forward the messages that add maximum value to the recipients. The interest matching is done by a trained object detection model which is applied to the sender's messages. Data forwarding between two nodes is shown in figure 3. In step 1, nodes connect and share metadata. In step 2, the sender selects a receiver. In steps 2a and 2b, an ML model calculates the interest match of the sender photos with the selected receiver. In step 3, the sender selects the best photo suggested by the ML model. In steps 4 and 5, the selected photo is encrypted and sent.

C. Information Server IS

IS are the DTN nodes described above that have a higher connection range, storage, and mobility to increase the number of connections and delivery in the network. They are used as the intermediate nodes to carry information among other DTN nodes and report the revoked users.

III. Implementation

Our application has two parts: a server and multiple clients. The server's (ReVO-ABE) task is to manage missions, users, master key, users' public key, and security key. The client's (Android device) task is to register to the server at bootstrap and then connect to other clients to securely forward information in a DTN.

A. ReVO-ABE Implementation

The front end of the *AA* is implemented with React which enables the users to view and request for user and mission modification. The backend of the *AA* is a Restful API built with java servlet and MySQL. The detailed implementation are as follows:

Fig. 4. Interface of the ReVO-ABE application

1) Frontend Implementation: The welcome page with the title "Central authority Demo" shows how many missions and how many users are in the system. It also has a sidebar to the "manage and add users" and "manage and missions" page. The "add user" and "manage user" pages are for adding and updating a user with his name and attributes. The "add mission" page is to generate a new mission with different properties. The "manage mission" page as shown in figure 4 displays the mission detail including the stacked linear barcode as the mission code. We can modify this mission as well as add new users to that mission. It also shows a list of users already joined the mission. It allows to change the mission's name, mission capacity, start/end date and delete/add users. When adding users, it allows us to search the users with the user's name. The newly added user will be assigned a tree ID which will be stored in the database. The user's private key will also be generated and stored in the database. When the user requests registration at bootstrap, the *AA* will fetch the keys from the database instead of generating them in real-time which may cause a delay.

2) Backend Implementation: The backend is implemented as a RESTful API that takes an HTTP request and returns the required information as a JSON file with the HTTP response. The API is implemented with Java. The database of the API is MySql. The API provides the interface to the following functionalities:

- Get all the user information as JSON by request to (URL/Users and URL/GetUsersOfAMission)
- Add user to the system by request to (URL/AddUser)
- Add mission to the system by request to (URL/AddMission)
- Add user to a specific mission by request to (URL/AddUserToMission)
- Get user or mission count by request to (URL/GetUserCount and URL/GetMissionCount)
- Get mission code as the pdf417 bar code by request to (URL/MissionQRCode)
- Get all mission as JSON object array by request to (URL/Missions)
- Search a specific User by request to (URL/SearchUser)
- Update mission by request to (URL/UpdateMission)
- Update user by request to (URL/UpdateUser)
- Setup and transfer key for the user by request to (URL/Bootstrap)

B. Secure and Prioritize Content Dissemination

An Android application has been created to enable nodes forwarding data constrained by their attributes and interests. This application uses HTTP requests and response for connecting with the server, and *Google Nearby API* for connecting multiple devices. *RoomDatabase* which is a database layer on top of an SQLite is used for storing all metadata and messages. The main components of this application are as follows:

1) Setup and Login: At bootstrap, a node is required to fetch its attributes, mission, public key, and security key from our ReVO-ABE application. Therefore, a node registers to a mission by connecting to the server using its unique username, password, and scanned mission code. If the setup is successful, a node can later log in using the same username and password and take part in the registered mission.

2) Member Revocation: A node registered to a mission can revoke any member (including itself) of that mission. A revoked member will not be able to decrypt a received message even if the attributes used to encrypt the message match with its corresponding attributes.

3) Policy and Receiver Selection: To send a message, a node needs to select a member of the mission and also set the policy to encrypt the message before sending (See figure 5). We use Google Nearby API to connect multiple nodes when they are in communication range (i.e. Bluetooth, NFC, WiFi). When a connection is established, they instantly send their initial metadata such as name, attributes, interest, etc. The user can then select a node from the connections to which he wishes to send the message.

4) Photo Transfer: When a user selects a node and sets a policy to send a photo, he is given the choice of selecting a photo from the pool of self-captured and collected photos

ordered by the match of interests of the receiver (See figure 6). The match of interests is measured using our ML model described in the following subsection. A user can also capture a photo using the camera and send it. Note that, the "Collected Photos" part may contain special images such that 'invalid attributes', 'invalid permission', and 'user revoked'. This occurs when a node is not allowed to decrypt a received photo and only carries the photo to forward to the appropriate nodes.

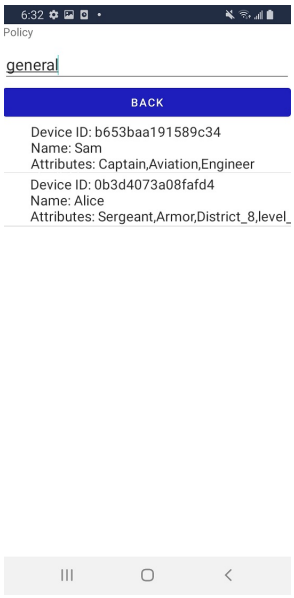

Fig. 5. A policy textfield to define encryption attributes and a list of connected nodes to select before sending a message

Fig. 6. Photos suggested by the ML model based on the interest of the selected receiver. Few photos may shown as "error" as the sender is not authorized to decrypt them yet carrying for others

5) Photo Caption Generator: When a node captures a photo, a caption describing the photo needs to be generated to find its match with different nodes' interests. In figure 7, we can see a caption is generated for each photo where the probabilities of the different words/objects are also included. To create a caption, we used an object detection model named quantized Mobile Net SSD and a TensorFlow Object Detection API which is an open-source framework built on top of TensorFlow that makes it easy to construct, train and deploy object detection models trained on Microsoft COCO dataset [3]. There are two parts: creating and preparing the TensorFlow model and accessing the model inside an Android app. First, we create a simple model and save its computation graph as a serialized GraphDef file. After training the model, we then save the values of its variables in a checkpoint file and convert it to a TensorFlow-Lite file. This is used for object detection and generating caption.

C. Demonstration

The demo will be shown with one remote server as *AA* and five Android devices i.e. one as *IS*, two in one mission

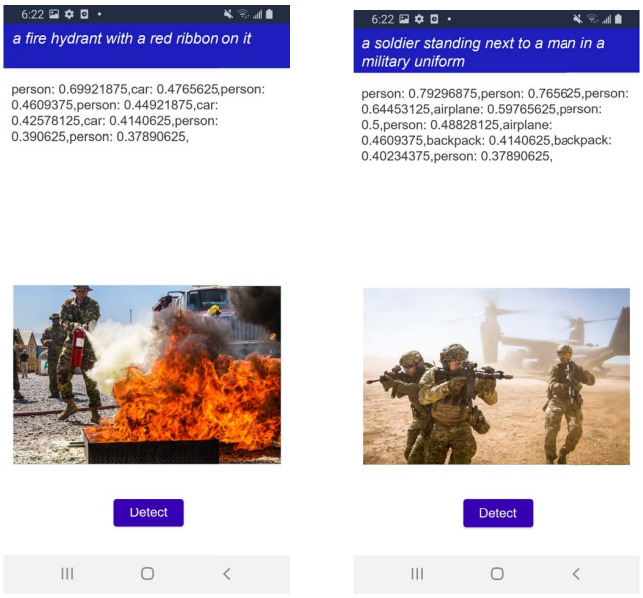

Fig. 7. Generated Captions of Photos. The probabilities of different detected objects are also shown which may or may not contribute to generate the captions.

group, and the other two in another mission group. The steps described in III-B will be shown with the following setup.

At the bootstrap, all the Android devices/nodes receive their missions, keys, and attributes using HTTP requests from the remote server. Nodes then disconnect from the server as if they are deployed in the field.

The nodes use Google Nearby Connection API to connect with each other when in communication range. As the devices will be kept nearby during the demonstration, a button to enable and disable connection has been implemented to simulate connection between node(s) when in physical proximity in a real DTN scenario.

D. Conclusion

In this demo paper, we have implemented an Android application in a DTN environment to apply the attribute-based access control scheme ReVo-ABE [1] to support efficient user revocations along with the advantage of prioritizing content dissemination based on image captions generated by DTN nodes. Based on the mission, DTN nodes can decide dynamically with whom to share the targeted contents and then only encrypt the data with those ABE keys.

REFERENCES

[1] M. A. Islam and S. Madria, "Attribute-based encryption scheme for secure multi-group data sharing in cloud," *IEEE Transactions on Services Computing*, 2020.

[2] S. Datta and S. Madria, "Prioritized content determination and dissemination using reinforcement learning in dtns," *IEEE Transactions on Network Science and Engineering*, 2021.

[3] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.