Michael Risher
GMU - ECE 646
Fall 2015

# Message Confidentiality in Enterprise Messaging Systems

Enterprise messaging systems (EMS) are widely used in many applications to route work and information between users and applications. Messages are published to either topics or queues. Messages passing through queues have one consumer while topics typically have multiple consumers. EMSs have security controls that prevent unauthorized users from consuming messages from certain queues or topics. However, no mechanisms exist to control access on a message level. Most of the time this isn't an issue since users would just be denied consume privileges for topics where those messages would appear. When using an EMS to pass classified information, this can lead to access control problems. If messages were sent to different topics based on their classification, there would need to be an estimated $10^{68}$ number of topics per message type (see calculations below) and an EMS typically handles numerous message types. Granted, most of the topics would have very little or no traffic while only a handful would contain most of the messages. One possible solution would be to implement an EMS that supported this security model, but given the time constraints of implementation and deployment of a solution this is also not practical.

This project seeks to create an approach that can be implemented in a practical time frame and using a feasible number of topics. The goal is to maintain the current message distribution model while keeping the message confidentiality.

The DoD utilizes a version of the Bell-LaPadula access control model[1]. An access control rule has 3 components (with some exceptions). The first component can be one of 4 values; Unclassified, Confidential, Secret, Top Secret. This first section is hierarchical where a user may view anything at or below the user's clearance level. The second section includes a list of compartments, or programs, to which the data belongs. A user must have every compartment listed on the data. The third section is a list of country codes or special country groups. A user must only have one of the countries listed. There are 4 different levels of classification; there are 196 countries; few if any one person knows the exact

number of compartments, but it is probably between 30 - 60. We'll use 30 for estimation. Therefore we have a total number of possible classifications expressed as $\sum_{r=0}^{30} \frac{31!}{r!(30-r)!} \times 4 \times \sum_{r=0}^{196} \frac{196!}{r!(196-r)!}$ which is equivalent to $1073741824 \times 4 \times 1.004e59 \approx 4.31359e68$ . However, for a practical estimation, only a handful of countries are ever really used and not many compartments per classification either. Let us assume that only 1-5 countries are used and no more than 4 compartments. Then we have $\frac{31!}{5!25!} \times 4 \times \frac{196!}{6!190!} \approx 1.28e18$ . Which is still far too many to represent in a practical system.

## *Language and Environment*

The solution will be implemented in Java so it will be capable of being deployed in multiple environments. The solution will consist of three components: the message publisher, the messaging system, and the consumer. Additional support infrastructure may be added as needed. BouncyCastle and RabbitMQ are planned to be utilized as third party software. Maven and JUnit will be utilized to compile and test the environment. There will need to be a key distribution component that will coordinate the encryption and decryption keys between the sender and authorized receivers. Additionally the Java extended crypto package will be needed to perform some of the extended cryptographic functions.

## *Assumptions*

- The system knows the clearance level of every user connecting to it.
- No decrypted message can be handled by any component of the system. Therefore the generic publisher component will always be authorized for any message it handles in this theoretical system.
- There are some specialized rules for translating country groups into specific country codes. This will not handle country groups, but only specific country codes or lists of country codes.
- This model can be adapted to binary data models, but will be limited to text only. This limitation is only due to implementation and testing time requirements. This feature may be added if time permits.

- Computational power is not a limitation. While some devices have limited computational resources, this solution was not designed specifically for environments where resources are limited. While it should work in such environments, no design considerations or testing will be performed on these devices.
- An AES key is unbreakable within any reasonable timeframe. The message shall not expose any information about the key or message that would make it easier than a brute force attack on an AES key to obtain the data of the message.

## Detailed Specification

All users of the program will need to obtain the encryption/decryption keys. Users will authenticate to a key server utilizing a PKI which would have been issued to the user or service. This issuance of identity is common and may be assumed completed outside the scope of this project. The key server will then issue the keys to the user via encrypted channels. The 'publisher' user will be issued all keys, but only for proof of concept. All subscriber users will be issued keys that match their security clearance. So each component of a clearance specification will have its own encryption key. Thus SECRET has key $K_{SECRET}$ and CONFIDENTIAL has $K_{CONFIDENTIAL}$and so on. There are many fewer keys to create and issue by having one unique one per access component. That gives us a total number of keys of 4+30+196=230 which is far more manageable than $10^{68}$.

Both the subscriber and publisher will connect to the messaging system where the subscribers will all listen to a single messaging topic. While multiple topics are possible, they are not necessary for the demonstration of this solution. All users will have access controls implemented on the messaging system's internal configuration. The publisher will only be allowed to publish messages. The consumers will only be allowed to consume messages.

Messages of different sensitivity levels will be passed into the publisher via REST interface. The messages will be POSTed to the endpoint in the format:

*S//COM-GAMMA//US/UK/CA*

*Ad novum legere propriae est. Has prima vituperata an. Ius ad choro aliquando deseruisse, maluisset instructior est at. Usu eius docendi et, erant mundi honestatis id mel. Eleifend*

*persecuti consequuntur his te. Laoreet explicari vim ut, vix iisque viderer ullamcorper ei, feugait concludaturque no mel. Quo ad phaedrum vituperata, solet nemore deleniti sed at.*

*Eu autem inermis sensibus vis, eam ut expetendis definitiones interpretaris. In vis latine corpora definitiones, scripta volumus deserunt mel ut. Inani perpetua conclusionemque te cum, nam ea augue debet. Sea ex tempor tamquam. Cu veniam comprehensam pri, an vel molestie rationibus, mei decore argumentum at. Ne cum blandit sapientem, te lorem scribentur sed, magna munere ei vel.*

The first line is a valid security marking, either in full or abbreviated format (as defined by this project; Real security markings are not used). Followed by a blank line, then the message will continue until the end of input.

The message will be parsed by the publisher to determine how to encrypt the message based on the security markings. The output will be in the generic format of:

*S//COM-GAMMA/TEL-REC//US/UK/CA*

$E(K_{US} \wedge K_{UK} \wedge K_{CA}, E(K_{COM\text{-}GAMMA} \,\&\&\, K_{TEL\text{-}REC}, E(K_{SECRET}, K))) \,||\, h(K)$

*IV*

*E(K, M) || h(IV||E(K, M))*

Where

S//COM-GAMMA/TEL-REC//US/UK/CA

- S - Secret classification
- COM-GAMMA - a compartment name (name carries no significance)
- TEL-REC - a second compartment name (name carries no significance)
- US - United States
- UK - United Kingdom
- CA - Canada

E(X, Y) - Encryption function of message Y using encryption key X

IV - Encryption algorithm’s Initialization Vector

h(M) - One way hash function of message M

|| - string concatenation

E(A^B, C) - Encrypt message C such that either key A or key B may be used to decrypt it

E(A&&B, C) - Encrypt message C such that both keys A and B must be used to decrypt it

The security marking remains as the first line of the message to inform receivers of the keys needed to unlock the decryption key. The message is encrypted with an AES key and a hash is appended to confirm that the message integrity is preserved and has been decrypted properly. The AES key is encrypted multiple times based on the security markings. The key itself will be randomly generated using a secure PRNG and will be unique for every message sent. While a TRNG would be preferable, it lacks the bandwidth that would be needed to scale the solution to an enterprise level. Cryptographically secure PRNG will be more than sufficiently secure to generate keys as Java will sometimes utilize a TRNG when available via the OS. The open source TRNG implementation for Java utilizes third party online services with daily caps. Future adaptations of this solution may choose to take advantage of TRNGs when available in hardware, but since this is a multi platform software solution, no such assumptions are made.

The subscribers will pull all messages off of the topic. Subscribers will all misbehave and attempt to decrypt messages even if they are not authorized to do so. Messages that cannot be decrypted will be discarded.

There is no limitations on the number of subscribers or publishers in this proof-of-concept model or a more realistic model. The custom code could be deployed into any number of publishers/subscribers and the EMS would have no problems handling thousands of users.

There are a few known concerns from a information security standpoint. The first is that the entire message is not encrypted. The security markings on the message provide some level of information leakage to parties that are not permitted to view the contents of the message. While this is of minimal utility it would be best avoided if possible. Another concern is the sharing of symmetric keys among multiple users. While no single key is capable of exposing a message, the compromising of a single user would leak all of the keys they hold. Key rotation is certainly a consideration, but with the asymmetric nature of message delivery, a published message may wait to be read for an indeterminate amount of time, it may be difficult to synchronize key rotation among all the users.

## Documentation of functions

Documentation of AES can be found under FIPS 197 specification at
http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

## Testing Procedures

In order to test there will need to define several users of the system.

- User A - UNCLASSIFIED//US
- User B - SECRET//COM-GAMMA/TEL-REC//US
- User C - SECRET//TEL-REC//CA
- User D - TOP SECRET//COM-GAMMA/TEL-REC//US
- User E - TOP SECRET//COM-GAMMA/TEL-REC//UK
- User F - TOP SECRET//COM-GAMMA//US

Then a series of messages will be handed off to the publisher component each with a different level of classification. The subscribers for each user will be checked if the message can be decrypted if their clearance dominates the classification of the message.

## Milestones

**October 26**

Development environment setup. This includes development IDE, installation and general configuration of messaging system and any authentication and key distribution systems.

**November 9**

Development of the publisher and subscribers nearly complete. The design should be mostly complete at this point.

**November 23**

Final report started. Development should be in the debugging and testing phase where final results can be compiled and inserted into the report.

**December 7**

Development and testing is complete. The report of findings and description of the approach should be in the final touch up stages.

## Possible areas of change

- The scheme for message encryption may need to change as implementation may reveal flaws in initial approach.
- The input and output of the program may vary to match any changes in data encryption approach.
- How the publisher will receive messages may change from one of many different options. It may be a REST interface, it may be a drop box type folder, or may be a command line interface.
- The manner in which encryption keys are retrieved from the key server may involve generating pub/priv key pairs for authentication and encryption.

## Table of Contents



## Related Literature


[1] J. Hur and D. Noh, "Attribute-Based Access Control with Efficient Revocation in Data Outsourcing Systems", *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 7, pp. 1214-1221, 2011.

[2] V. Goyal, O. Pandey, A. Sahai and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data", *Proceedings of the 13th ACM conference on Computer and communications security - CCS '06*, 2006.

[3] S. Jahid, P. Mittal and N. Borisov, "EASiER", *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security - ASIACCS '11*, 2011.

[4] R. Ostrovsky, A. Sahai and B. Waters, "Attribute-based encryption with non-monotonic access structures", *Proceedings of the 14th ACM conference on Computer and communications security - CCS '07*, 2007.

## References

[1] Wikipedia, 'Bell–LaPadula model', 2015. [Online]. Available: https://en.wikipedia.org/wiki/Bell%E2%80%93LaPadula_model. [Accessed: 14- Oct- 2015].