# ABE: Attribute-Based Encryption of Encrypted Data in Wireless Sensor Networks

Komali Pradeep Kumar #[1], CH.Satyananda Reddy*[2]
Department of Computer Science & Systems Engineering,
Andhra University College of Engineering,
Visakhapatnam, AP, India.

## Abstract

**A WSN usually consists of a large number of sensor nodes that can be easily deployed to various terrains of interest to sense the environment. As more sensitive data is shared and stored by third-party sites on the Internet, there will be a need to encrypt data stored at these sites. One drawback of encrypting data, is that it can be selectively shared only at a coarse-grained level (i.e., giving another party your private key). We develop a new cryptosystem for fine-grained sharing of encrypted data that we call Key-Policy Attribute-Based Encryption (KP-ABE). In our cryptosystem, cipher texts are labeled with sets of attributes and private keys are associated with access structures that control which cipher texts a user is able to decrypt. We demonstrate the applicability of our construction to sharing of audit-log information and broadcast encryption. Our construction supports delegation of private keys which subsumes Hierarchical Identity-Based Encryption (HIBE).**

**Keywords:** Attribute-based encryption, access control, audit logs, broadcast encryption, delegation, hierarchical identity based encryption.

## 1. Introduction

There is a trend for sensitive user data to be stored by third parties on the Internet. For example, personal email, data, and personal preferences are stored on web portal sites such as Google and Yahoo. The attack correlation center, dshield.org, presents aggregated views of attacks on the Internet, but stores intrusion reports individually submitted by users. Given the variety, amount, and importance of information stored at these sites, there is cause for concern that personal data will be compromised. This worry is escalated by the surge in recent attacks and legal pressure faced by such services.

Wireless sensor networks (WSN) have been an area of significant research in recent years .A WSN usually consists of a large number of sensor nodes that can be easily deployed to various terrains of interest to sense the environment. WSN's have found their wide applications in both civilian and military domains. To accomplish the targeted application and fulfill its functionalities, a WSN usually generates a large amount of data continuously over its lifetime. One of the biggest challenge then is how to store and access these sensed data.

One method for alleviating some of these problems is to store data in encrypted form. Thus, if the storage is compromised the amount of information loss will be limited. One disadvantage of encrypting data is that it severely limits the ability of users to selectively share their encrypted data at a Fine-grained level. Suppose a particular user wants to grant decryption access to a party to all of its Internet traffic logs for all entries on a particular range of dates that had a source IP address from a particular subnet. The user either needs to act as an intermediary and decrypt all relevant entries for the party or must give the party its private decryption key, and thus let it have access to *all* entries. Neither one of these options is particularly appealing. An important setting where these issues give rise to serious problems is **audit logs**.

Sahai and Waters [1] made some initial steps to solving this problem by introducing the concept of Attributed-Based Encryption (ABE). In an ABE system, a user's keys and cipher texts are labeled with sets of descriptive attributes and a particular key can decrypt a particular cipher text only if there is a match between the attributes of the cipher text and the user's key. The cryptosystem of Sahai and Waters allowed for decryption when at least *k* attributes overlapped between a cipher text and a private key. While this primitive was shown to be useful for error-tolerant encryption with biometrics, the lack of expressibility seems to limit its applicability to larger systems.

**Our Contribution:** We develop a much richer type of attribute-based encryption cryptosystem and demonstrate its applications. In our system each cipher text is labeled by the encryptor with a set of descriptive attributes. Each private key is associated with an access structure that specifies which type of cipher texts the key can decrypt. We call such a scheme a Key-Policy Attribute-Based Encryption (KPABE), since the access structure is specified in the private key, while the cipher texts are simply labeled with a set of descriptive attributes.

We note that this setting is reminiscent of secret sharing schemes (see, e.g., [3]). Using known techniques one can build a secret-sharing scheme that specifies that a set of parties must cooperate in order to reconstruct a secret. For example, one can specify a tree access structure where the interior nodes consist of **AND** and **OR** gates and the leaves consist of different parties. Any set of parties that satisfy the tree can reconstruct the secret.

In our construction each user's key is associated with a tree-access structure where the leaves are associated with attributes. A user is able to decrypt a cipher text if the attributes associated with a cipher text satisfy the key's access structure. The primary difference between our setting and secret-sharing schemes is that while secret-sharing schemes allow for cooperation between different parties, in our setting, this is expressly forbidden. For instance, if Alice has the key associated with the access structure "X **AND** Y", and Bob has the key associated with the access structure "Y **AND** Z", we would not want them to be able to decrypt a ciphertext whose only attribute is Y by colluding. To do this, we adapt and generalize the techniques introduced by [1] to deal with more complex settings. We will show that this cryptosystem gives us a powerful tool for encryption with fine-grained access control for applications such as sharing audit log information.

In addition, we provide a delegation mechanism for our construction. Roughly, this allows any user that has a key for access structure X to derive a key for access structure Y, if and only if Y is more restrictive than X. Somewhat surprisingly, we observe that our construction with the delegation property subsumes Hierarchical Identity-Based Encryption [2, 20] and its derivatives [4].

## 2. Related Work

**Fine-grained Access Control:** Fine-grained access control systems facilitate granting differential access rights to a set of users and allow flexibility in specifying the access rights of individual users. Several techniques are known for implementing fine grained access control.

Common to the existing techniques (see, e.g., [5, 6,] and the references therein) is the fact that they employ a trusted server that stores the data in clear. Access control relies on software checks to ensure that a user can access a piece of data only if he is authorized to do so. This situation is not particularly appealing from a security standpoint. In the event of server compromise, for example, as a result of software vulnerability exploit the potential for information theft is immense. Furthermore, there is always a danger of "insider attacks" wherein a person having access to the server steals and leaks the information, for example, for economic gains. Some techniques (see, e.g., [2]) create user hierarchies and require the users to share a common secret key if they are in a common set in the hierarchy. The data is then classified according to the hierarchy and encrypted under the public key of the set it is meant for. Clearly, such methods have several limitations. If a third party must access the data for a set, a user of that set either needs to act as an intermediary and decrypt all relevant entries for the party or must give the party its private decryption key, and thus let it have access to all entries. In many cases, by using the user hierarchies

 

it is not even possible to realize an access control equivalent to monotone access trees.

In this paper, we introduce new techniques to implement fine grained access control. In our techniques, the data is stored on the server in an encrypted form while different users are still allowed to decrypt different pieces of data per the security policy. This effectively eliminates the need to rely on the storage server for preventing unauthorized data access.

**Secret-Sharing Schemes:** Secret-sharing schemes (SSS) are used to divide a secret among a number of parties. The information given to a party is called the share (of the secret) for that party. Every SSS realizes some access structure that defines the sets of parties who should be able to reconstruct the secret by using their shares. Shamir [7] and Blakley [6] were the first to propose a construction for secret-sharing schemes where the access structure is a threshold gate. That is, if any *t* or more parties come together, they can reconstruct the secret by using their shares; however, any lesser number of parties do not get any information about the secret. Benaloh [5] extended Shamir's idea to realize any access structure that can be represented as a tree consisting of threshold gates. Other notable secretsharing schemes are [10, 11].

In SSS, one can specify a tree-access structure where the interior nodes consist of AND and OR gates and the leaves consist of different parties. Any set of parties that satisfy the tree can come together and reconstruct the secret. Therefore in SSS, collusion among different users (or parties) is not only allowed but required.

**Identity-Based Encryption and Extensions**: The concept of Attribute-Based Encryption was introduced by Sahai and Waters [1], who also presented a particular scheme that, they called Fuzzy Identity-Based Encryption (FIBE). The Fuzzy-IBE scheme builds upon several ideas from Identity-Based Encryption [9, 13]. In FIBE, an identity is viewed as a set of attributes. FIBE allows for a private key for an identity, $\omega$, to decrypt to a ciphertext encrypted with an identity, $\omega$,[1] if and only if the identities $\omega$ and $\omega^1$ are close to each other as measured by the "set overlap" distance metric. In other words, if the message is encrypted with a set of attributes $\omega^1$, a private key for a set of attributes $\omega$ enables decrypting that message, if and only if $|\omega \cap \omega^1| \geq d$, where $d$ is fixed during the setup time. Thus, FIBE achieves error tolerance making it suitable for use with biometric identities. However, it has limited applicability to access control of data, our primary motivation for this work. Since the main goal in FIBE is error tolerance, the only access structure supported is a threshold gate whose threshold is fixed at the setup time.

We develop a much richer type of attribute-based encryption. The private keys of different users might be associated with different access structures. Our constructions support a wide variety of access structures (indeed, in its most general form, every LSSS realizable access structure), including a tree of threshold gates. Yao et. al. [14] show how an IBE system that encrypts to multiple hierarchical identities in a collusion-resistant manner implies a forward secure Hierarchical IBE scheme. They also note how their techniques for resisting collusion attacks are useful in attribute-based encryption. However, the cost of their scheme in terms of computation, private key size, and ciphertext size increases exponentially with the number of attributes. We also note that there has been other work that applied IBE techniques to access control, but did not address our central concern of resisting attacks from colluding users [11].

## 3. Background Theory

### 3.1 Bilinear Maps

We present a few facts related to groups with efficiently computable bilinear maps. Let G1 and G2 be two multiplicative cyclic groups of prime order *p*. Let *g* be a generator of G1 and *e* be a bilinear map, i.e*:* G1 × G1 → G2. The bilinear map *e* has the following properties:

1. Bilinearity: for all $u, v \in G_1$ and $a, b \in Z_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$
2. Non-degeneracy: $e(g, g) \neq 1$.

We say that G1 is a bilinear group if the group operation in G1 and the bilinear map $e : G1 \times G1 \rightarrow G2$ are both efficiently computable. Notice that the map *e* is symmetric since

$e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

### 3.2 The Decisional Bilinear Diffie-Hellman (BDH) Assumption

Let $a, b, c, z \in Z_p$ be chosen at random and $g$ be a generator of G1. The decisional BDH assumption is that no probabilistic polynomial-time algorithm $B$ can distinguish the tuple $(A = g^a, B = g^b, C = g^c, e(g, g)^{abc})$ from the tuple $(A = g^a, B = g^b, C = g^c, e(g, g)^z)$ with more than a negligible advantage. The advantage of $B$ is

$$\left| \Pr[B(A, B, C, e(g,g)^{abc}) = 0] - \Pr[B(A, B, C, e(g,g)^z) = 0] \right|$$

Where the probability is taken over the random choice of the generator $g$, the random choice of $a, b, c, z$ in $Z_p$, and the random bits consumed by $B$.

## 4. Construction for Access Trees

In the access-tree construction, ciphertexts are labeled with a set of descriptive attributes. Private keys are identified by a tree-access structure in which each interior node of the tree is a threshold gate and the leaves are associated with attributes. (We note that this setting is very expressive. For example, we can represent a tree with "AND" and "OR" gates by using respectively 2 of 2 and 1 of 2 threshold gates.) A user will be able to decrypt a ciphertext with a given key if and only if there is an assignment of attributes from the ciphertexts to nodes of the tree such that the tree is satisfied.

### 4.1 Access Trees

**Access tree** *T*. Let *T* be a tree representing an access structure. Each non-leaf node of the tree represents a threshold gate, described by its children and a threshold value. If $num_x$ is the number of children of a node $x$ and $k_x$ is its threshold value, then $0 < k_x \leq num_x$. When $k_x = 1$, the threshold gate is an OR gate and when $k_x = num_x$, it is an AND gate. Each leaf node $x$ of the tree is described by an attribute and a threshold value $k_x = 1$. To facilitate working with the access trees, we define a few functions. We denote the parent of the node $x$ in the tree by parent($x$). The function att($x$) is defined only if $x$ is a leaf node and denotes the attribute associated with the leaf node $x$ in the tree. The access tree $T$ also defines an ordering between the children of every node, that is, the children of a node are numbered from 1 to *num*. The function index(x) returns such a number associated with the node x. Where the index values are uniquely assigned to nodes in the access structure for a given key in an arbitrary manner.

### 4.2 Our Construction

Let $G_1$ be a bilinear group of prime order $p$, and let $g$ be a generator of $G_1$. In addition, let $e : G_1 \times G_1 \to G_2$ denote the bilinear map. A security parameter, $\kappa$, will determine the size of the groups. We also define the Lagrange coefficient $\Delta_{i,S}$ for $i \in Z_p$ and a set, $S$, of elements in $Z_p$: $\Delta_{i,S}(x) = \prod_{j \in S, j \neq i} \frac{x-j}{i-j}$. We will associate each attribute with a unique element in $Z_p^*$. Our construction follows.

**Setup** Define the universe of attributes $U = \{1, 2, \ldots, n\}$. Now, for each attribute $i \subset U$, choose a number $t_i$ uniformly at random from $Z_p$. Finally, choose $y$ uniformly at random in $Z_p$. The published public parameters PK are

$$T_1 = g^{t_1}, \ldots, T_{|U|} = g^{t_{|U|}}, Y = e(g,g)^y .$$

The master key MK is:

$$t_1, \ldots, t_{|U|}, y .$$

**Encryption** $(M, \gamma, \mathrm{PK})$ To encrypt a message $M \in G_2$ under a set of attributes $\gamma$, choose a random value $s \subset Z_p$ and publish the ciphertext as:

$$E = (\gamma, E' = MY^s, \{E_i = T_i^s\}_{i \in \gamma}) .$$

Key Generation $(T, MK)$ The algorithm outputs a key that enables the user to decrypt a message encrypted under a set of attributes $\gamma$ if and only if $T(\gamma) = 1$. The algorithm proceeds as follows. First choose a polynomial $q_x$ for each node $x$ (including the leaves) in the tree $T$. These polynomials are chosen in the following way in a top-down manner, starting from the root node $r$.

For each node $x$ in the tree, set the degree $d_x$ of the polynomial $q_x$ to be one less than the threshold value $k_x$ of that node, that is, $d_x = k_x - 1$. Now, for the root node $r$, set $q_r(0) = y$ and $d_r$ other points of the polynomial $q_r$ randomly to define it completely. For any other node $x$, set $q_x(0) = q_{parent(x)}(\text{index}(x))$ and choose $d_x$ other points randomly to completely define $q_x$.

Once the polynomials have been decided, for each leaf node $x$, we give the following secret value to the user:

$$D_x = g^{\frac{q_x(0)}{t_i}} \text{ where } i = \text{att}(x).$$

The set of above secret values is the decryption key $D$.

Decryption $(E, D)$ We specify our decryption procedure as a recursive algorithm. For ease of exposition we present the simplest form of the decryption algorithm and discuss potential performance improvements in the next subsection.

We first define a recursive algorithm DecryptNode$(E, D, x)$ that takes as input the ciphertext $E = (\gamma, E', \{E_i, E_i'\}_{i \in \gamma})$, the private key $D$ (we assume the access tree $T$ is embedded in the private key), and a node $x$ in the tree. It outputs a group element of $G_2$ or $\perp$.

Let $i = \text{att}(x)$. If the node $x$ is a leaf node then:

$$\text{DecryptNode}(E, D, x) = \begin{cases} e(D_x, E_i) = e(g^{\frac{q_x(0)}{t_i}}, g^{s \cdot t_i}) \\ = e(g, g)^{s \cdot q_x(0)} & \text{if } i \in \gamma \\ \perp \text{ otherwise} \end{cases}$$

We now consider the recursive case when $x$ is a non-leaf node. The algorithm DecryptNode$(E, D, x)$ then proceeds as follows: For all nodes $z$ that are children of $x$, it calls DecryptNode$(E, D, z)$ and stores the output as $F_z$. Let $S_x$ be an arbitrary $k_x$-sized set of child nodes $z$ such that $F_z \neq \perp$. If no such set exists then the node was not satisfied and the function returns $\perp$.

Otherwise, we compute:

$$\begin{aligned} F_x &= \prod_{z \in S_x} F_z^{\Delta_{i,S_x'}(0)}, \quad \text{where } S_x' = \{\text{index}(z) : z \in S_x\}^{i = \text{index}(x)} \\ &= \prod_{z \in S_x} \left(e(g,g)^{s \cdot q_z(0)}\right)^{\Delta_{i,S_x'}(0)} \\ &= \prod_{z \in S_x} \left(e(g,g)^{s \cdot q_{\text{parent}(z)}(\text{index}(z))}\right)^{\Delta_{i,S_x'}(0)} \quad \text{(by constr.)} \\ &= \prod_{z \in S_x} e(g,g)^{s \cdot q_x(i) \cdot \Delta_{i,S_x'}(0)} \\ &= e(g,g)^{s \cdot q_x(0)} \quad \text{(using polynomial interpolation)} \end{aligned}$$

and return the result.

Now that we have defined our function DecryptNode, the decryption algorithm simply calls the function on the root of the tree. We observe that DecryptNode$(E, D, r) = e(g, g)^{ys} = Y^s$ if and only if the ciphertext satisfies the tree. Since, $E' = MY^s$ the decryption algorithm simply divides out $Y^s$ and recovers the message $M$.

## 5. Audit Log Application

An important application of KP-ABE deals with secure forensic analysis: One of the most important needs for electronic forensic analysis is an "audit log" containing a detailed account of all activity on the system or network to be protected. Such audit logs, however, raise significant security concerns: a comprehensive audit log would become a prized target for enemy capture. Merely encrypting the audit log is not sufficient, since then any party who needs to legitimately access the audit log contents (for instance a forensic analyst) would require the secret key – thereby giving this single analyst access to essentially all secret information on the network. Such problematic security issues arise in nearly every secure system, and particularly in large-scale networked systems such as the Global Information Grid, where diverse secret, top secret, and highly classified information will need to appear intermingled in distributed audit logs.

Our KP-ABE system provides an attractive solution to the audit log problem. Audit log entries could be annotated with attributes such as, for instance, the name of the user, the date and time of the user action, and the type of data modified or accessed by the user action. Then, a forensic

analyst charged with some investigation would be issued a secret key associated with a particular "access structure" – which would correspond to the key allowing for a particular kind of encrypted search; such a key, for example, would only open audit log records whose attributes satisfied the condition that "the user name is Bob, OR (the date is between October 4, 2005 and October 7, 2005 AND the data accessed pertained to naval operations off the coast of North Korea)". Our system would provide the guarantee that even if multiple rogue analysts collude to try to extract unauthorized information from the audit log, they will fail.

A more concrete example audit-log application of our ABE system would be to the ArmyCERT program, which uses net flow logs [15]. Basically, an entry is created for every flow (e.g. TCP connection), indexed by seven attributes: source IP address, destination IP address, L3 protocol type, source port, destination port, To S byte (DSCP), and input logical interface (if Index). These aspects of every flow are in the clear, and the payload can be encrypted using our ABE system with these fields as attributes. Note that in our scheme, we would need to assume that the attributes associated with audit log entries would be available to all analysts.7 This may present a problem in highly secret environments where even attributes themselves would need to be kept hidden from analysts. We leave the problem of constructing KP-ABE systems where attributes associated with ciphertexts remain secret as an important open problem.

## 6. Conclusion

In this paper, we analyzed a novel yet important issue of fine-grained data access control for distributed storage in WSNs. To address the problem, we proposed a scheme called FDAC in which each sensor node is assigned a set of attributes, and each user is assigned an access structure which designates the access capability of the user. The sensor data is encrypted under the attributes such that only the users with the intended access structure are able to decrypt. As the access structure is extremely expressive, we are able to control data access precisely, and thus achieve fine-grained access control. Moreover, FDAC is able to provide security assurance such as resilience to user colluding and sensor compromising attacks as well as user revocability. Our experiment shows that the system overload in FDAC is reasonable in practical scenarios. An interesting future work of FDAC may be on its efficient implementation on WSNs with low-end sensor nodes.

## 7. References

[1] A. Sahai and B. Waters. Fuzzy Identity Based Encryption. In Advances in Cryptology – Eurocrypt, volume 3494 of LNCS, pages 457–473. Springer, 2005.

[2] Jeremy Horwitz and Ben Lynn. Toward hierarchical identity-based encryption. In Lars R. Knudsen, editor, EUROCRYPT, volume 2332 of Lecture Notes in Computer Science, pages 466–481. Springer, 2002.

[3] Craig Gentry and Alice Silverberg. Hierarchical id-based cryptography. In ASIACRYPT, pages 548–566, 2002.

[4] Michel Abdalla, Dario Catalano, Alexander W. Dent, John Malone-Lee, Gregory Neven, and Nigel P. Smart. Identity-based encryption gone wild. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, ICALP (2), volume 4052 of Lecture Notes in Computer Science, pages 300–311. Springer, 2006.

[5] Myong H. Kang, Joon S. Park, and Judith N. Froscher. Access control mechanisms for inter-organizational workflow. In SACMAT '01: Proceedings of the sixth ACM symposium on Access control models and technologies, pages 66–74, New York, NY, USA, 2001. ACM Press.

[6] Rita Gavriloaie, Wolfgang Nejdl, Daniel Olmedilla, Kent E. Seamons, and Marianne Winslett. No registration needed: How to use declarative policies and negotiation to access sensitive resources on the semantic web. In ESWS, pages 342–356, 2004.

[7] A. Shamir. How to share a secret. Commun. ACM, 22(11):612–613, 1979.

[8] D. Boneh, G.D. Crescenzo, R. Ostrovsky, and G. Persiano. Public-Key Encryption with Keyword Search. In Advances in Cryptology – Eurocrypt, volume 3027 of LNCS, pages 506–522. Springer, 2004.

[9] D. Boneh and M. Franklin. Identity Based Encryption from the Weil Pairing. In Advances in Cryptology –

 

CRYPTO, volume 2139 of LNCS, pages 213–229. Springer, 2001.

[10] M. Ito, A. Saito, and T. Nishizeki. Secret Sharing Scheme Realizing General Access Structure. In IEEE Globecom. IEEE, 1987.

[11] E. F. Brickell. Some ideal secret sharing schemes. Journal of Combinatorial Mathematics and Combinatorial Computing, 6:105–113, 1989.

[12] D. Boneh and M. Franklin. Identity Based Encryption from the Weil Pairing. In Advances in Cryptology – CRYPTO, volume 2139 of LNCS, pages 213–229. Springer, 2001.

[13] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In IMA Int. Conf., pages 360–363, 2001.

[14] Y. Dodis, N. Fazio, A. Lysyanskaya, and D.F. Yao. ID-Based Encryption for Complex Hierarchies with Applications to Forward Security and Broadcast Encryption. In ACM conference on Computer and Communications Security (ACM CCS), pages 354–363, 2004.

[15] Cisco Networks. http://netflow.cesnet.cz/nnetflow.php. [5] S. Gandham, Y. Zhang, and Q. Huang, "Distributed Time-Optimal Scheduling for Convergecast in Wireless Sensor Networks," Computer Networks, vol. 52, no. 3, pp. 610-629, 2008.

## 8. About the Authors

Komali Pradeep Kumar is currently pursuing his 5 Years Integrated M.Tech (IT) in Computer Science and Systems Engineering at Andhra University College of Engineering, Visakhapatnam. His area of interests includes Networks, Security.

CH. Satyananda Reddy is currently working as a Sr.Assistant Professor in Computer Science and Systems Engineering department, Andhra University College of Engineering, Visakhapatnam. He is registered for Ph.D. (CSSE) in Andhra University. His research interests include Software Engineering, DBMS, and Operating Systems.