Introduction to Attribute Based Searchable Encryption

Dalia Khader

daliakhader@googlemail.com

Abstract. An Attribute Based Searchable Encryption Scheme (ABSE) is a public key encryption with keyword search (PEKS) where each user owns a set of attributes, and the senders decide on a policy. The policy is a function of these attributes expressed as a predicate and determines, among the users of the system, who is eligible to decrypt and search the ciphertext. Only members who own sufficient attributes to satisfy that policy can send the server a valid search query. In our work we introduce the concept of a secure ABSE by defining the functionalities and the relevant security notions.

Keywords: PEKS, Attribute Based Encryption, Public Key Cryptography.

1 Introduction

Searchable encryption (SE) is an encryption scheme that supports keyword based retrieval of documents. The main challenge of SE is to allow third parties to search the ciphertexts without giving them decrypting capabilities. This has been an active research area for more than a decade. Song et al. [5] proposed the first scheme that enables searchability in symmetric encryption while Boneh et al. [2] introduced a scheme for public key encryption with keyword search (PEKS).

Searchable encryption schemes assume that the user sending the search query owns the decryption key and that the sender has to know the identity of the user querying the data in order to encrypt using the corresponding encryption key. This raises the question, what if the encrypted data is shared between several receivers and is kept in a remote shared storage that is not trusted for confidentiality?

Attribute-Based Encryption (ABE) [4] addresses this problem. An ABE is a scheme in which each user is identified by a set of attributes, and some function of those attributes, the policy, is used to decide on decryption capabilities. The two types of ABE schemes are: key-policy and ciphertext-policy [3, 1].

This paper defines a new primitive attribute based searchable encryption (ABSE). In ABSE senders decide on a policy that determines user's eligibility not only for decrypting but also for searching the data. Unlike existing proposals in the literature [6], ours is based on a hybrid system of key and cipher policy which gives more flexibility, a strong security, and allows for multi-authorities.

2 Formal Definition of ABSE

To define an ABSE we introduce the five entities involved: a central authority $\mathcal{TTP}$ who sets up the system, a server $\mathcal{S}$ where all encrypted data is uploaded to. An encryptor

B. De Decker and A. Zúquete (Eds.): CMS 2014, LNCS 8735, pp. 131–135, 2014.

$\mathcal{E}$ who uploads the data and sets the policy. The querier $\mathcal{Q}$ who wants to search the server and download documents. Many attribute authorities $\mathcal{AT}$ each responsible of a set of attributes and that give out private keys to users owning these attributes.

Definition 1. *An Attribute Based Searchable Encryption Scheme consists of the following probabilistic polynomial time algorithms:* $\mathbf{ABSE} := \big(TSetup,\ AddUser,\ ASetup,\ AttG,\ PrdG,\ PrdVK,\ PrdQT,\ ABSE,\ TrpG,\ TEST\big)$

$\mathbf{TSetup(k)} \rightarrow \mathbf{(PP, UMK)}$: Run by $\mathcal{TTP}$ to set up the system. Takes a security parameter k and outputs public parameters PP and a user master key UMK which is kept secret.
$\mathbf{AddUser(PP, UMK)} \rightarrow \mathbf{(RK_i, SK_i)}$: Run by $\mathcal{TTP}$ every time a user registers with the system. It outputs a registration key RK_i that will be used to register with attribute authorities and servers. It outputs SK_i that is secret to the user and will be used in creating trapdoors.
$\mathbf{ASetup(PP)} \rightarrow \mathbf{(AMK_j, APK_j)}$: Run by $\mathcal{AT}$ to set up the attribute authority. It outputs an attribute master key AMK_j which is secret to $\mathcal{AT}$ and is used to create attribute private keys when users register. It also outputs an attribute public key APK_j which is used in building the policies and is public to all.
$\mathbf{AttG(RK_i, AMK_j)} \rightarrow \mathbf{ASK_{i,j}}$: Run by $\mathcal{AT}$ to register a user i, and outputs an attribute private key $ASK_{i,j}$ that will be used in proving possession of attribute j.
$\mathbf{PrdG(\Psi, \mathcal{AP})} \rightarrow \mathbf{(ST_\Psi, IT_\Psi)}$: Given a predicate Ψ and a list of attribute public keys $\mathcal{AP} = \{APK_j\}_{j=1}^{m}$, the algorithm generates a searching token ST_Ψ that will be used in creating trapdoors and an indexing token IT_Ψ used for creating searchable ciphertext.
$\mathbf{PrdVK(\Psi)} \rightarrow \mathbf{VT_\Psi}$: Run by $\mathcal{S}$. For each predicate in the system the server creates a verification token VT_Ψ that is kept secret to the server.
$\mathbf{PrdQT(\Psi, VT_\Psi, RK_i, \mathcal{AP})} \rightarrow \mathbf{QT_{i,\Psi}}$: Run by the $\mathcal{S}$. Given a predicate verification token VT_Ψ and a registration key RK_i, the server outputs a query token $QT_{i,\Psi}$ that allows the user i to search for keywords encrypted under the predicate Ψ.
$\mathbf{ABSE(W, \Psi, IT_\Psi)} \rightarrow \mathbf{E_{\Psi,W}}$: Run by $\mathcal{E}$. For a keyword W and under token IT_Ψ create a searchable ciphertext $E_{\Psi,W}$.
$\mathbf{TrpG(W, \Psi, QT_{i,\Psi}, ST_\Psi, SK_i, \mathcal{AS}_i)} \rightarrow \mathbf{T_{\Psi,W}}$: Run by $\mathcal{Q}$. Given a keyword, a query token, a searching token, a user secret key and a set of user private attribute keys $\mathcal{AS}_i = \{ASK_{i,j}\}_{j=1}^{m}$, output a trapdoor $T_{\Psi,W}$.
$\mathbf{TEST(E_{\Psi,W}, T_{\Psi,W}, VT_\Psi, RK_i)} \rightarrow \mathbf{\{0, 1\}}$: Run by the $\mathcal{S}$. Given a searchable ciphertext, a trapdoor, a verification token and a registration key output 1 if the user satisfies the predicate and if the keyword is found, otherwise output 0.

On the Security of ABSE The security notions of an ABSE are: correctness, security against Attribute Based Chosen Keyword Attack (ACKA) and security against Attributes Forgeability Attacks (AFA). We need three game models to define these notions (See Figures 1(a), 1(b), 1(c)) where the adversary is given access to certain oracles and a trace of the responses is recorded. Both are explained below.

CUL	Corrupted Users	HUL	Honest Users	CRK	Corrupted RK_i
CRK	Corrupted RK_i	HA	Honest $\mathcal{AT}$	CA	Corrupted $\mathcal{AT}$
$CASK$	Revealed $ASK_{i,j}$	$TrapL$	Queried trapdoors	$HASK$	Non-revealed $ASK_{i,j}$
$PredL$	List of (ST_Ψ, IT_Ψ)	$RQTL$	Revealed $QT_{i,\Psi}$	QTL	Non-revealed $QT_{i,\Psi}$
VTL	Non-revealed VT_Ψ	$RVTL$	Revealed VT_Ψ		

AddUsr : Adds user i to the system by running $AddUser$, and adding (RK_i, SK_i) to HUL.
UsrCpt : Corrupts user i by revealing (RK_i, SK_i) and adding them to CUL and CRK.
RKCpt : Partially corrupts user i by revealing registration key RK_i and adding it to CRK.
AddAtt : Adds an honest attribute authority j to the system by running $ASetup$, computing (APK_j, AMK_j) and publishing APK_j.
AMKCpt : Corrupts attribute authority j by revealing AMK_j and adding to CA.
AddASK : Runs $AttG$ to compute $ASK_{i,j}$, and adds it to $HASK$.
ASKCpt : Corrupts an attribute private key by revealing $ASK_{i,j}$ and adding it to $CASK$.
TrapO : The challenger generates a trapdoor for certain keyword W using a querying token $QT_{i,\Psi}$ and searchable token ST_Ψ on behalf of user i with set of attributes $\mathcal{AS}_i$. The list $TrapL$ is updated with all the information used as input and as output to the algorithm $TrpG$.
AddPred : Generates a searchable token and an indexing token (ST_Ψ, IT_Ψ) for predicate Ψ by running $PrdG$ and then updates the list $PredL$.
AddVT : Runs $PrdVK$ to obtain a verification token VT_Ψ and updates VTL.
RevealVT : Corrupting the verification token VT_Ψ by revealing it and $RVTL$ is updated.
AddQT : Generates a querying token $QT_{i,\Psi}$ by running $PrdQT$, then QTL is updated.
RevealQT : The querying token $QT_{i,\Psi}$ of user i and predicate Ψ is revealed to the adversary and the list $RQTL$ is updated.
$\mathbf{Ch_b}$: Challenges the adversary to guess whether a trapdoor T_b ($b \in \{0,1\}$) was generated for keyword W_0 or W_1. The adversary chooses the predicate, the set of attributes and the user he would like to be challenged upon.

Correctness of ABSE. This property demands that if a searchable encryption $E_{\Psi,W}$ was produced correctly, i.e. using valid IT_Ψ and if a trapdoor $T_{\Psi,W}$ was introduced correctly using valid $QT_{i,\Psi}$, ST_Ψ, SK_i, $\mathcal{AS}_i$, then the $TEST$ algorithm should return 1 if the predicate is satisfied $\Psi(\mathcal{AS}_i) = 1$ and the keywords match $W = W'$, otherwise the $TEST$ algorithm should return 0. Figure 1(a) explains the details. Formally, the ABSE is said to be correct if for a security parameter k and all polynomial time adversaries $\mathcal{A}$ the following advantage is negligible: $\mathrm{Adv}_{\mathcal{A}}^{\mathrm{corr}}(k) = |Pr[\mathsf{Exp}_{\mathcal{A}}^{corr}(k) = 1]|$

Attribute Based Chosen Keyword Attacks. We define security for an ABSE in the sense of semantic–security. The aim is to ensure that an encryption $ABSE$ does not reveal any information about keyword W except to a $\mathcal{Q}$ who satisfies the policy and can create trapdoors. We define the security against an active attackers $\mathcal{A}$ whose given access to a set of oracles shown in Figure 1(b). Let the advantage of winning the game be defined as follows: $\mathrm{Adv}_{\mathcal{A}}^{\mathrm{ACKA}}(k) = |Pr[\mathsf{Exp}_{\mathcal{A}}^{ACKA_0}(k) = 1] - Pr[\mathsf{Exp}_{\mathcal{A}}^{ACKA_1}(k) = 1]|$

An ABSE scheme is said to be secure against an ACKA if for a given security parameter k and all polynomial time adversary $\mathcal{A}$ the advantage $\mathrm{Adv}_{\mathcal{A}}^{\mathrm{ACKA}}(k)$ is negligible.

Attribute Forgeability Attack. This security notion captures forgeability of trapdoors where the adversary can produce a trapdoor without having the sufficient attribute set that satisfies the predicate Ψ. The adversary is given access to the oracles described in Figure 1(c). The challenge is to produce a pair of searchable encryption $E^*_{\Psi'}$ and trapdoor $T^*_{\Psi'}$ under predicate Ψ' such that the $TEST(E^*_{\Psi'}, T^*_{\Psi'}, VT_{\Psi'}, RK'_i)$ outputs 1 for a given RK'_i. The definition includes coalition of attributes. Formally, an ABSE scheme is said to be secure against an AFA if for a security parameter k and all polynomial time adversaries $\mathcal{A}$ the following advantage is negligible: $\mathrm{Adv}_{\mathcal{A}}^{\mathrm{AFA}}(k) = |Pr[\mathsf{Exp}_{\mathcal{A}}^{AFA}(k) = 1]|$

3 Conclusion

We define a new ABSE scheme and the security notions required. A working construction and security proofs are provided in a full version of this paper.

Experiment $\mathsf{Exp}_{\mathcal{A}}^{\mathrm{corr}}(k)$:

- $(PP, UMK) \leftarrow TSetup(k)$
- $HUL, HA, PredL, HASK, QTL, VTL = \phi$
- $(SK_i, RK_i, \mathcal{AS}_i, QT_{i,\Psi}, ST_\Psi, IT_\Psi) \leftarrow \mathcal{A}\big(PP : AddUsr(.), AddAtt(.), AddASK(.,.), AddPred(.,.), AddQT(.,.), AddVT(.)\big)$
- If $[(SK_i, RK_i) \notin HUL] \vee [\exists j \in \Psi$ s.t. $(AMK_j, APK_j) \notin HA] \vee [\exists j \in \mathcal{AS}_i$ s.t. $(ASK_{i,j}) \notin HASK] \vee [VT_\Psi \notin VTL]$: Return 0
- $ABSE(W, IT_\Psi) \rightarrow E_{\Psi,W}$; $TrpG(W', QT_{i,\Psi}, ST_\Psi, SK_i, \mathcal{AS}_i) \rightarrow T_{\Psi,W}$
- If $\big[[\Psi(\mathcal{AS}_i) \neq 1] \vee [W \neq W']\big] \wedge [TEST(E_{\Psi,W}, T_{\Psi,W}, VT_\Psi, RK_i) = 0]$: Return 1
- If $\big[[\Psi(\mathcal{AS}_i) = 1] \wedge [W = W']\big] \wedge [TEST(E_{\Psi,W}, T_{\Psi,W}, VT_\Psi, RK_i) = 1]$: Return 1
- Else Return 0

(a) Correctness Game Model

Experiment $\mathsf{Exp}_{\mathcal{A}}^{\mathrm{ACKA}}(k)$:

- $(PP, UMK) \leftarrow TSetup(k)$
- $CUL, HUL, CRK, HA, CA, CASK, HASK, TrapL, PredL, QTL, RQTL, RVTL, VTL = \phi$
- $\grave{b} \leftarrow \mathcal{A}\big(PP : UsrCpt(.), RKCpt(.), AMKCpt(.), ASKCpt(.,.), RevealQT(.,.), AddVT(.), RevealVT(.), TrapO(.,.,.,.,.), AddUsr(.), AddAtt(.), AddPred(.,.), AddQT(.,.), AddASK(.,.), Ch(.,.,.,.,.)\big)$
- Return $\grave{b}$

(b) Security against ACKA

Experiment $\mathsf{Exp}_{\mathcal{A}}^{\mathrm{AFA}}(k)$:

- $(PP, UMK) \leftarrow TSetup(k)$
- $CUL, HUL, CRK, HA, CA, CASK, HASK, TrapL, PredL, QTL, RQTL, RVTL, VTL = \phi$
- $(T^*_{\Psi'}, E^*_{\Psi'}, \Psi', RK'_i) \leftarrow \mathcal{A}\big(PP : UsrCpt(.), RKCpt(.), AMKCpt(.), ASKCpt(.,.), RevealQT(.,.), AddVT(.), RevealVT(.), TrapO(.,.,.,.,.), AddUsr(.), AddAtt(.), AddPred(.,.), AddQT(.,.), AddASK(.,.)\big)$
- If $\big[TEST(E^*_{\Psi'}, T^*_{\Psi'}, VT_{\Psi'}, RK'_i) = 0\big] \vee \big[\Psi' \subseteq CA\big] \vee \big[T^*_{\Psi'} \in TrapL\big] \vee \big[[VT_{\Psi'} \in RVTL] \wedge [\forall j \in \Psi', j \in CASK \cup CA]\big] \vee \big[[VT_{\Psi'} \notin RVTL] \wedge [\exists i$ s.t $\forall j \in \mathcal{AS}_i, j \in CASK_i \cup CA$ and $RK'_i = RK_i]\big]$: Return 0.
- Else Return 1

(c) Security against AFA

References

1. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: IEEE SSP 2007, pp. 321–334 (2007)
2. Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public key encryption with keyword search. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 506–522. Springer, Heidelberg (2004)
3. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: ACM CCS 2006, pp. 89–98 (2006)
4. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
5. Song, D.X., Wagner, D., Perrig, A.: Practical techniques for searches on encrypted data. In: SSP 2000. IEEE (2000)
6. Zheng, Q., Xu, S., Ateniese, G.: Vabks: Verifiable attribute-based keyword search over outsourced encrypted data. IACR ePrint