InfoDesign GmbH

Development for Customers

Benutzerhandbuch IM4Crypt

Stand: 1.2.0 7/01/2022

Allgemeine Automationsverfahren

IM4Crypt 1.2.0

Gedruckt: Juli 2022

Herausgeber

InfoDesign GmbH

Großes Feld 23
25421 Pinneberg

Tel +49 4101 - 69 31 - 54
Fax +49 4101 - 69 31 - 56

Mail mail@infodesign.de

Support
Tel +49 4101 595 8991
Mail support@infodesign.de

www.infodesign.de

Inhaltsverzeichnis

1 Einleitung

Das IM4Crypt ist ein Java-Programm zur Verschlüsselung von Dateien. Es kann als Befehlszeilentool verwendet werden, ermöglicht jedoch auch eine API-Integration. Diese Anleitung beschreibt wie das IM4Crypt über eine Kommandozeilenschnittstelle benutzt wird.

Unter dem Begriff *Verschlüsselung* versteht man Verfahren, die Daten mittels eines Schlüssels in eine nicht lesbare Form umwandeln. Die verschlüsselte Daten lassen sich ohne Kenntnis eines bestimmten Schlüssels nicht mehr (oder nur unter sehr großem Aufwand) zurückwandeln. Dieser Vorgang, also die Transformation der Daten wieder in ihre ursprüngliche Form, ist die *Entschlüsselung*. Verschlüsselungsverfahren können in zwei Kategorien, die sich in der Anzahl der Schlüssel unterscheiden, unterteilt werden: symmetrische und asymmetrische Verschlüsselung. Neben diesen Methoden gibt es auch das hybride Verschlüsslungsverfahren, welches sich aus dem symmetrischen und asymmetrischen Verfahren zusammensetzt.

Bei symmetrischer Verschlüsselung wird derselbe geheime Schlüssel sowohl für die Verschlüsselung als auch für die Entschlüsselung einer Datei verwendet. Ein symmetrischer Schlüssel ist folglich in zwei Richtungen zu verwenden und kann sowohl dupliziert als auch von mehreren Personen verwendet werden.

Anderseits wird bei asymmetrischer Verschlüsselungsverfahren nicht nur ein einziger geheimer Schlüssel verwendet, sondern stattdessen ein Schlüsselpaar. Ein Schlüsselpaar besteht aus einem öffentlichen Schlüssel, der zum Verschlüsseln einer Datei dient und kann frei mit anderen geteilt werden, und aus einem privaten Schlüssel, der zur Entschlüsselung benötigt wird und sollte geheim gehalten werden. Jedes Schlüsselpaar ist einzigartig und hängt über einen mathematischen Algorithmus eng zusammen. Daten, die mit dem öffentlichen Schlüssel verschlüsselt werden, können mit dem privaten Schlüssel desselben Schlüsselpaars wieder entschlüsselt werden.

Wie oben erwähnt, kommen symmetrische und asymmetrische Verschlüsselungsverfahren in Kombination zum Einsatz, spricht man von Hybridverschlüsselung. Hierbei werden die eigentlichen Daten mit einem symmetrischen Schlüssel verschlüsselt, welcher asymmetrisch verschlüsselt bzw. eingehüllt wird.

Attributbasierte Verschlüsselung engl. Attribute-Based Encryption (ABE) ist eine relativ neue Art von asymmetrischer Verschlüsselung, die eine sehr flexible Zugriffsregulierung mittels der Verschlüsselung selbst ermöglicht. Das heißt, dass die Daten nicht für jeden Benutzer einzeln verschlüsselt werden, sondern nur einmalig, und zwar unter Zugriffsrichtlinien, engl. access policies. Eine Zugriffsrichtlinie ist im Wesentlichen eine Regel, die aus Attributen erstellt wird, die gibt an, wer berechtigt ist, welche Daten zu entschlüsseln. Bei diesem Konzept werden jedem Nutzer Attribute zugewiesen, die seine Eigenschaften beschreiben. Entsprechend seinen Eigenschaften wird für jeden Nutzer ein privater Schlüssel erzeugt bzw. jeder geheime Schlüssel ist mit den Attributen des Schlüsselbesitzers versehen. Die wichtigste Anforderung an das Attibutbasiertes Verschlüsselungsverfahren besteht darin, dass die Entschlüsselung von Daten nur für solche Benutzer möglich ist, deren Attributsmengen bzw. private Schlüssel die Verschlüsselungspolicies erfüllen.

Vereinfacht gesagt, nur wenn die Attribute des privaten Schlüssels mit den Attributen, die in der Veschlüsselungspolicy enthalten sind, übereinstimmen, kann der Benutzer die verschlüsselte Daten komplett entschlüsseln. Eine detailliertere Beschreibung der Bestandteille der Attributbasierte Verschlüsselung wird im Appendix bereitgestellt.

2 IM4Crypt

Die Hauptmerkmale des Programms im Überblick:

- Bietet zwei Veschlüsselungsverfahren an: symmetrische Verschlüsselung und hybride Verschlüsselung.
- Erwartet Befehle (und deren Optionen) über die Kommandozeile und erzeugt Ausgabedateien. Die Funktionalität und Benennung aller Argumente stehen in direktem Zusammenhang mit den entsprechenden Vorgänge der einzelnen Verschlüsselungsverfahren.
- Implementiert hybride Verschlüsselung basierend auf attributbasierter Verschlüsselung als asymmetrisches Verfahren und AES (Advanced Encryption Standard) als symmetrisches Verfahren.
- Unterstützt Schlüsselwiederherstellung (nur bei Hybridverschlüsselung).

2.1 Programmaufruf

Das IM4Crypt Programm wird als ausführbare JAR-Datei unter dem Namen „infodesign.im4Crypt.jar“ geliefert. Voraussetzung für das Ausführen von JAR-Dateien ist, dass die Software Java Runtime Environment installiert sein muss: https://www.java.com/de/download/

Die grundlegende Syntax zum Ausführen einer ausführbaren JAR-Datei mit Argumenten ist:

java -jar <JAR-Dateiname> [Args…]

Der Aufruf des Programms in der Kommandozeile erfolgt über die ausführbare JAR-Datei folgenderweise:

1. Starten Sie die Kommandozeile und navigieren zum Verzeichnis im dem die infodesign.im4crypt.jar Datei liegt.
2. Geben Sie ein:

   java -jar infodesign.im4crypt.jar [Argumente der ausgewählten Operation]

2.2 Befehle und Anwendung

Die Befehle des IM4Crypts haben die folgende Syntax:

Befehl --Option1 <Wert1> --Option2 <Wert2>……--OptionN <WertN>

Durch verschiedenen Optionen (beziehungsweise Argumente) kann man jeden Befehl vollständig definieren.

Diese Optionen werden mit dem Zeichen „--“ eingeleitet, gefolgt von dem Namen und abschließend dem Wert der jeweiligen Option; das ist die ausführliche Form. Außerdem kennt IM4Crypt die abgekürzte Form einer Option, wobei eine Option mit dem Zeichen „-“ und einem kurzen Teil ihres Namens (in der Regel, mit dem ersten Buchstaben) angegeben werden. Befehle und dazugehörigen Optionen und Werten werden durch Leerzeichen voneinander getrennt aber es dürfen keine Leerzeichen zwischen dem Zeichen „--“ und dem Namen der Option sein. Die spitzen Klammern werden bei der eigentlichen Befehlseingabe nicht mit eingegeben. Mit Ausnahme eines Falles (siehe Hybridverschlüsselung 12) setup Befehl) erlaubt IM4Crypt eine beliebige Reihenfolge der Argumente.

2.2.1 Symmetrische Verschlüsselung

Befehle: keygen, encrypt, decrypt

keygen: Erzeugt den symmetrischen Schlüssel. Dieser Schüssel wird sowohl für die Ver-als auch für die Entschlüsselung verwendet.

Syntax:

keygen --type symmetric --outfile <symmetricKeyFilePath>

keygen -t symmetric -out <symmetricKeyFilePath>

Optionen:

--type, -t: die Art des ausgewählten Verschlüsselungsverfahren, in diesem Fall, "*symmetric".*

--outfile, -out: der absolute Pfad der zu erstellenden symmetrischen Schlüsseldatei.

encrypt: Verschlüsselt eine Datei mithilfe des symmetrischen Schlüssels.

Syntax:

encrypt --type symmetric --infile <inputFilePath> --outfile <encryptedFilePath> --key <symmetricKeyFilePath>

encrypt -t symmetric -in <inputFilePath> -out <encryptedFilePath> -k <symmetricKeyFilePath>

Optionen:

--type, -t: die Art des ausgewählten Verschlüsselungsverfahren, in diesem Fall, "*symmetric".*

--infile, -in: der absolute Pfad der zu verschlüsselnden Datei.

--outfile, -out: der absolute Pfad der verschlüsselten Ausgabedatei.

--key, -k: der absolute Pfad der symmetrischen Schüsseldatei.

decrypt: Entschlüsselt die verschlüsselte Datei mithilfe des symmetrischen Schlüssels.

Syntax:

decrypt --type symmetric --infile <encryptedFilePath> --outfile <decryptedFilePath> --key <symmetricKeyFilePath>

decrypt -t symmetric -in <encryptedFilePath> -out <decryptedFilePath> -k <symmetricKeyFilePath>

Optionen:

-- *type*, -t: die Art des ausgewählten Verschlüsselungsverfahren, in diesem Fall, "*symmetric".*

--infile, -in: der absolute Pfad der zu entschlüsselnden Datei.

--outfile, -out: der absolute Pfad der entschlüsselten Ausgabedatei.

--key, -k: der absolute Pfad der symmetrischen Schüsseldatei.

Anwendungsbeispiel:

Zur symmetrischen Verschlüsselung einer Datei wird ein symmetrischer Schlüssel benötigt. Definieren Sie einen Zielpfad für die Datei, in dem der symmetrische Schlüssel gespeichert wird, und führen Sie dann den Befehl keygen wie folgt aus:

keygen -t symmetric -out symmetricKeyFile

Nach dem Ausführen von keygen wird der generierte symmetrische Schlüssel in der Datei "*symmetricKeyFile*" gespeichert, die sich im Ordner *"C:/Users/username/IM4Crypt"* befindet. Der nächste Schritt besteht darin, die beabsichtigte Datei zu verschlüsseln. Dazu muss der Befehl encrypt mit allen erforderlichen Parametern ausgeführt werden; wählen Sie zuerst einen Zielpfad für die generierte verschlüsselte Datei aus und dann verwenden Sie den im vorherigen Schritt generierten symmetrischen Schlüssel. Der encrypt Befehl sieht dann so aus:

encrypt -t symmetric -in dataFile.txt -out encryptedFile -k symmetricKeyFile

Für die symmetrische Entschlüsselung der verschlüsselte Datei *"encryptedFile"* sollte zuerst ein Zielpfad für die entschlüsselte Datei (als Wert für den Parameter -out) ausgewählt werden. Anschließend wird der decrypt Befehl unter Verwendung des gleichen symmetrischen Schlüssels wie folgt ausgeführt:

decrypt -t symmetric -in encryptedFile -out decryptedFile -k symmetricKeyFile

2.2.2 Hybridverschlüsselung

Befehle: setup, encrypt, decrypt, keygen

setup: Generiert die Parameter, die zum Ausführen anderer Vorgänge erforderlich sind.

Im Einzelnen, der setup Befehl erzeugt:

a. Die öffentliche Parameter ("public parameters"), die bei der Hybridverschlüsselung verwendet werden.

b. Die Hauptparameter ("master parameters"), die bei der Generierung von privaten Schlüsseln verwendet werden.

Syntax:

setup --public <publicParamsFile> --master <masterParams>

setup -p <publicParamsFile> -m <masterParamsFile>

Optionen:

--public, -p: der absolute Zielpfad der Datei, in dem die öffentlichen Parameter gespeichert werden.

--master,-m: der absolute Zielpfad der Datei, in dem die Hauptparameter gespeichert werden.

Anwendungsbeispiel:

Führen Sie den setup Befehl wie folgt aus:

setup -p "publicParamsFile, masterParamsFile"

Nach dem Ausführen des setup-Befehls, werden die Dateien *"publicParamsFile"* und *"masterParamsFile"*, die die entsprechenden Parameter enthalten, erstellt.

encrypt: Verschlüsselt eine Datei mit der öffentlichen Parameter unter eine Zugriffsrichtlinie.

Syntax:

encrypt --type hybrid --infile <plaintextFile> --outfile <encryptedFile> --public <publicParamsFile> --accessPolicy <access policy>

encrypt -t hybrid -in <plaintextFile> -out <encryptedFile> -p <publicParamsFile> -a <access policy

Optionen:

--type, -t: die Art des ausgewählten Verschlüsselungsverfahren, in diesem Fall, "*hybrid".*

--infile, -in: der absolute Pfad der zu verschlüsselnden Datei.

--outfile, -out: der absolute Pfad der verschlüsselten Ausgabedatei.

--public, -p: der absolute Pfad der öffentlichen Parameterdatei.

--accessPolicy, -a: die Zugriffsrichtlinie, die zur Verschlüsselung eingesetzt wird.

Eine Zugriffsrichtlinie ist ein boolescher Ausdruck, der aus Attributen und Operatoren besteht (AND, OR werden als Operatoren unterstützt), z.B. "university AND student". Das erwartete Format für eine Zugriffsrichtlinie ist eine Zeichenkette in einer Postfixnotation. Die Postfixnotation für die obenangegebene Zugriffslinie "university AND student" lautet "university student AND".

Anwendungsbeispiel:

encrypt -t hybrid -in dataFile.txt -out hybridEncryptedFile -p publicParamsFile -c "university student and"
Nachdem man den obigen Befehl ausgeführt hat, wird die Datei "*dataFile.txt*" unter Verwendung der in der "*publicParamsFile"* Datei gespeicherten öffentlichen Parameter und unter die Zugriffsrichtlinie "*university student and"* verschlüsselt. Dies führt zu der verschlüsselte Datei "*hybridEncryptedFile*" im Ordner *"C:/Users/username/IM4Crypt"*.

keygen: Erzeugt den Privatschlüssel.

Syntax:

keygen -- type hybrid --outfile <privateKeyFilePath> --masters <masterParamsFile> --attributes <attributeSet>

keygen -t hybrid -out <privateKeyFilePath> -m <masterParamsFile> -a <attributeSet>

Optionen:

--type, -t: die Art des ausgewählten Verschlüsselungsverfahren, in diesem Fall, "*hybrid".*

--outfile, -out: der absolute Zielpfad der Datei, in dem die für die Rekonstruktion des privaten Schlüssels erforderlichen Werte gespeichert werden.

--key, -k: der absolute Pfad der zu erstellenden privaten Schlüsseldatei.

--master, -m: der absolute Pfad der Hauptparameterdatei.

--attributes, -a: das Attributset, das zur Erstellung des Privatschlüssels eingesetzt wird.

Erwartetes Format für das Attributset: Attribute die als Komma getrennte Zeichenkette angegeben werden.
z.B. "university, student": dieses Attributset führ zu einem Privatschlüssel, der den Attributen {university, student} entspricht.

Anwendungsbeispiel:

keygen -m hybrid -k privKeyFile -p masterParamsFile -c "university, student" -out storedValuesFile

Nach dem Ausführen des Befehls keygen wird der private Schlüsseldatei, der den Attributen "university, student" entspricht, generiert. Die Datei „storedValuesFile“ enthält jetzt die Werte, die für die Rekonstruktion des privaten Schlüssels benötigt werden.

decrypt: Entschlüsselt die verschlüsselte Datei mithilfe des Privatschlüssels.

Syntax:

decrypt -- mode hybrid --infile <encryptedFile> --outfile <decryptedFile> --key <privKeyFile>

decrypt -t hybrid --in <encryptedFile> -out <decryptedFile> -k <privKeyFile>

Optionen:

--type, -t: die Art des ausgewählten Verschlüsselungsverfahren, in diesem Fall, "*hybrid".*

--infile, -in: der absolute Pfad der zu entschlüsselnden Datei.

--outfile, -out: der absolute Pfad der entschlüsselten Ausgabedatei.

--key, -k: der absolute Pfad der privaten Schüsseldatei

Anwendungsbeispiel:

decrypt -t hybrid -in hybridEncryptedFile -out hybridDecryptedFile -k privKeyFile

Nach dem Ausführen des obigen Befehls wird die verschlüsselte Datei *"hybridEncryptedFile"* (mit Hilfe des entsprechenden Privatschlüssels) entschlüsselt. Die entschlüsselte Datei unter dem Namen *"hybridDecryptedFile"* befindet sich im Ordner *"C:/Users/username/IM4Crypt*".

2.2.3 Hilfe zu Befehlen

help: Gibt einen Überblick über alle verwendeten Befehle und Optionen des IM4Crypts sowie deren Anwendung.

Syntax:

help <command>

Optionen:

<command> Name des Befehles für den die Hilfe angezeigt werden soll. Ohne die Option werden alle Befehle angezeigt

2.2.4 Exit-Codes

Die folgende Tabelle bietet eine Übersicht über die Exit-Codes. bzw. Fehlermeldungen des IM4Crypts, die Ausgabemeldung wenn sie auftreten und deren Bedeutung.

Exit-Code	Ausgabe-Nachricht	Bedeutung
0	Operation terminated successfully	Kein Fehler. Der Vorgang wurde erfolgreich ausgeführt.
1	Invalid format for access policy	Ungültiges Format der Zugriffsrichtlinie. Das erwartete Format für eine Zugriffsrichtlinie ist ein boolescher Ausdruck als Zeichenkette in einer Postfixnotation (siehe Hybridverschlüsselung [12], encrypt Befehl).
2	Access policy is empty	Die Zugriffsrichtlinie ist leer. Geben Sie eine gültig Zugriffsrichtlinie ein.
3	The file used is not an ABE encrypted file	Die Eingabedatei ist keine ABE-verschlüsselte Datei.
4	Invalid format for attibute set	Ungültiges Format der Attributsmenge. Attribute sollen als Komma getrennte Zeichenketten angegeben werden (siehe Hybridverschlüsselung [12], keygen Befehl).
5	An I/O error occured during crypto-operation	Bei einer Krypto-Operation ist ein I/O Fehler aufgetreten. Bei der Enschlüsselung überprüfen Sie ob der passende Privatschlüssel verwendet wird.
6	Invalid command line argument	Ungültiges Kommandozeileargument. Überprüfen Sie die Syntax und Schreibweise des Befehls für den gewünschten Vorgang.
7	File not found	Die Datei wurde nicht gefunden.Überprüfen Sie die Schreibweise oder den Dateipfad.
8	File already exists	Die Datei ist bereits vorhanden. Dieser Fehler kann auftreten, wenn der Name der zu erstellenden Datei bereits benutzt wurde.
9	I/O error during serialization of ABE data	I/O Fehler bei der Serialisierung bzw. Speicherung von ABE Daten.
10	I/O error during deserialization of ABE data	I/O Fehler bei der Deserialisierung von ABE Daten. Dieser Fehler konnte auftreten, wenn die falche Parameterdatei bei einen Vorgang verwendet wird, z.B. statt der öffentlichen Parameter, die Hauptparameter bei der Veschlüsselung benutzt werden.

3 Appendix

3.1 Attributbasierte Veschlüsselung

Wie in der Einleitung erwähnt, liefert Attributbasierte Verschlüsselung eine flexibel Lösungsmöglichkeit zur Realisierung komplexer Zugriffsrichtlinien mit Kryptografie. Dieses asymmetrisches kryptographisches Verfahren erlaubt Daten mit Zugriffsrichtlinien über Attributten zu verschlüsseln. Attributsmengen und Zuggriffsrichtlinien sind der Grundkomponenten der Attributbasierten Veschlusselung.

Die Attribute beschreiben die Eigenschaften eines Nutzers. Solche Attribute können sowohl persönliche Eigenschaften als auch berufliche Merkmale sein. Entsprechend seinen Eigenschaften wird für jeden Nutzer ein privater Schlüssel erzeugt, dem eine Reihe von beschreibenden Attributen zugeordnet wird.

Durch Zugriffsrichtilinien können Berechtigungen eines Nutzers in die verschlüsselte Daten durchgesetzt werden. Eine Zugriffsrichtlinie bereitstellt Zugriffsregeln basierend auf Attributsmengen. Sie wird als boolescher Ausdruck formuliert, der aus Attributen und Operatoren besteht, z.B. (Attibute_1 AND Attribute_2) OR Attribute_3.

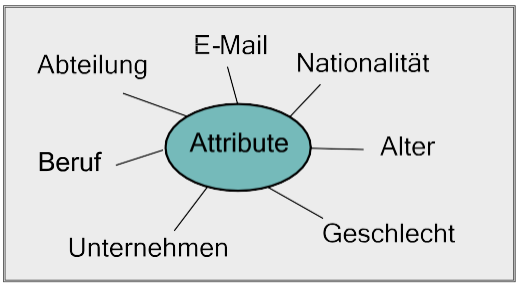

Figure 15: Attribute

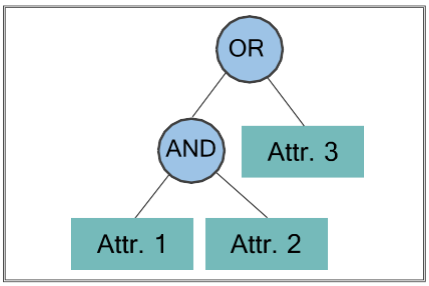

Figure 16: Zugriffsrichtlinie als boolescher Ausdruck

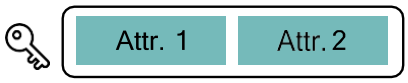

Figure 17: Privatschlüssel mit zwei Attributen

Privatschlüssel könnten als analog zu gewöhnlichen Schlüsseln, die man im täglichen Leben verwendet, betrachten werden. In ähnlicher Weise könnte man sich eine Zugriffsrichtlinie als Türschloss vorstellen. Verschiedene Schlüssel, die verschiedenen Personen gehören, können dieselbe Tür öffnen. Die Tür öffnet sich, solange der Schlüssel in das Türschloss passt.

Wie in Abbildung 18 dargestellt ist, wirkt die Zugriffsrichtlinie *"(InfoDesign GmbH OR InfoDesign Consulting) AND Software Entwickler"* als Türschloss und festlegt, wer die Tür öffnen darf. Laut der angegebenen Zugriffsrichtlinie, einer der Schlüssel der in das Türschloss passt, ist derjenige eines Softwareentwicklers der in der Firma "InfoDesign GmbH" arbeitet. Ein Angestellter der Firma "InfoDesign Consulting" in der Position als "Softwareentwickler" ist auch berechtigt die Tür zu öffnen.

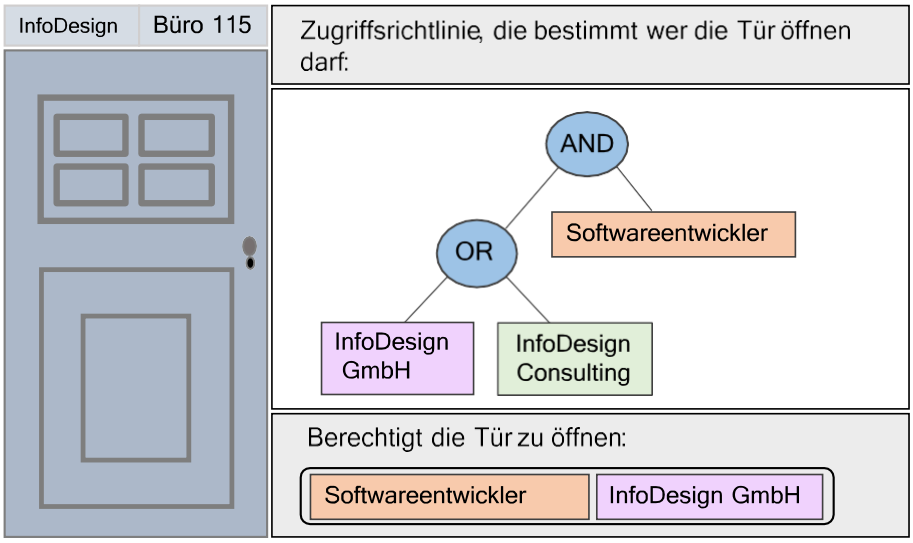

Figure 18: Bespiel zur Zufriffsrichtlinie

Weitere wichtige Bestandteile eines attributbasierten Verschlüsselungsverfahrens sind die *Hauptparameter* und die *öffentliche Parameter*. Die öffentliche Parameter werden für die Verschlüsselung von Daten verwendet aber sollten nicht mit dem Konzept eines öffentlichen Schlüssels in einem traditionellen asymmetrischen Verschlüsselungsverfahren verwechselt werden. Die Hauptparameter sind an der Erzeugung der privaten Schlüssel beteiligt.

Ein typisches attributbasiertes Veschlüsselungsverfahren besteht aus vier grundlegenden Funktionen bzw. Algorithmen: *Set-Up*, *Keygen*, *Encrypt*, *Decrypt*. Der Set-Up Algorithmus generiert die obengenannte Hauptparameter und öffentliche Parameter. Die Keygen Funktion erzeugt einen privaten Schlüssel mit Attributen; hierfür werden die Hauptparameter benötigt. Der Encrypt Algorithmus wird, wie der Name vermuten lässt, zum Verschlüsseln von Daten verwendet. Daten werden mit den öffentlichen Parametern under eine Zugriffsrichtline verschlüsselt. Für die Entschlüsselung benötigt der Decrypt Algorithmus die verschlüsselten Daten und einen privaten Schlüssel.

Abbildung 19 stellt schematisch die Funktionsweise eines attributbasierten Systems dar. Um eine Datei zu verschlüsseln, muss der Nutzer zuerst eine Zugriffsrichtlinie angeben, die legt fest, wer die verschlüsselten Daten abrufen darf bzw. welche Attribute der Schlüssel des Nutzers besitzen muss um die Daten entschlüsseln zu können. Die Datei wird unter diese Zugriffsrichtlinie mithilfe der öffentlichen Parameter verschlüsselt. Wer auf diese Datei zugreifen möchte, kann diese nur entschlüsseln, wenn sein privater Schlüssel, die geforderten Attribute besitzt, d.h., die Attribute seines Schlüssels die in der verschlüsselte Datei definierte Zugriffsrichtlinie erfüllen. Die Zugriffsrichtlinie wird erfüllt, wenn die Attribute des entsprechenden privaten Schlüssels mit den Attributen des booleschen Ausdrucks übereinstimmen. Das Beispiel in Abbildungen 21, 22 gibt Ausschluss darüber wann eine Zugriffsrichtlinie erfüllt ist. Attributbasierte Veschlüsselung ermöglicht die gemeinsame Nutzung eines einzelnen Dateninhalts für mehrere Benutzergruppen mit denselben Zugriffsrechten.

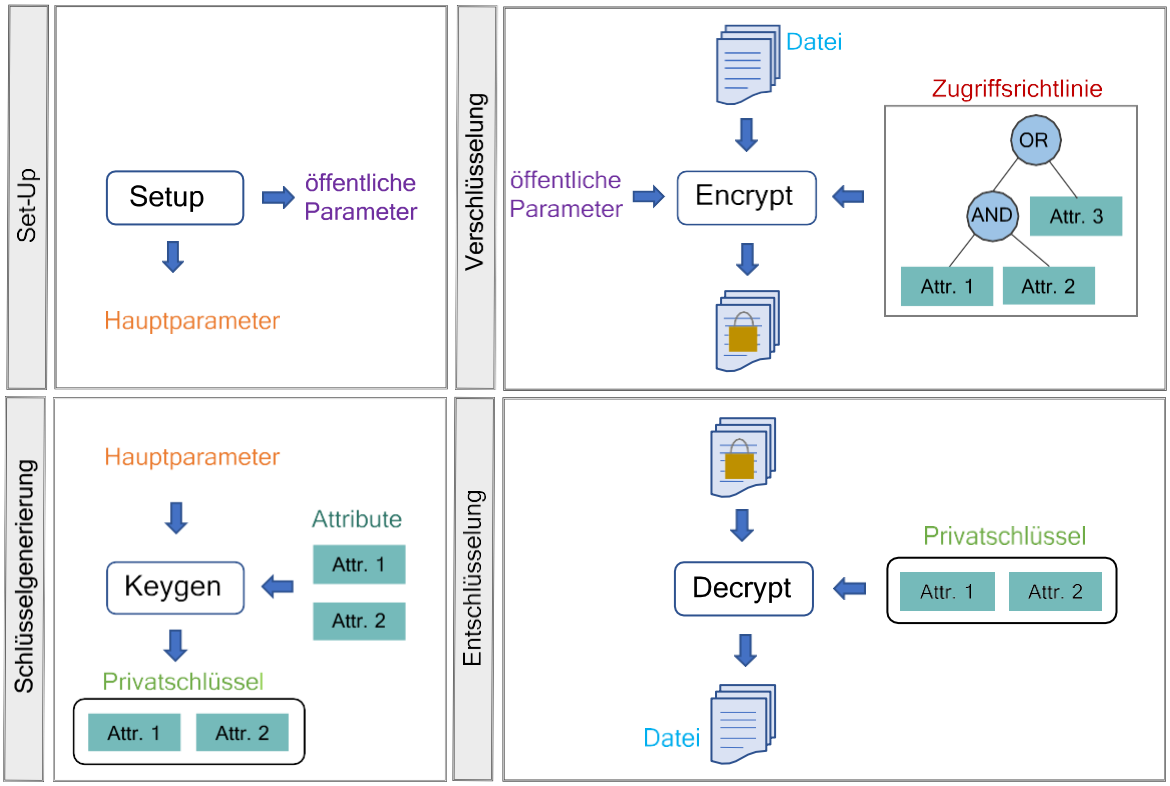

Figure 19: Ein attributbasiertes Veschlüsselungsystem

Stellen Sie sich vor dass eine Datei unter die Zugriffsrichtlinie *"(IT-Abteilung AND Softwareentwickler) OR Manager"* [Abb. 20] verschlüsselt wird. Diese Zugriffsrichtlinie beschreibt dass entweder der Manager der Firma oder ein Softwareentwickler, der in der IT-Abteilung arbeitet, auf die Datei zugreifen soll. Wie in Abbildung 21 angezeigt, erfüllen die Attribute des Privatschlüssels 1 die Zugriffsrichtlinie, da das Attibut "Manager" läuft das Oder-Gatter der Zugriffsrichtlinie durch. Das bedeutet dass der Besitzer des Privatschlüssels 1 die Datei entschlüsseln kann. Das Gleiche gilt für den zweiten Privatschlüssel [Abb. 22]; die Attributen <IT-Abteilung, Softwareentwickler> laufen das Und-Gatter des booleschen Ausdrücks durch und damit wird die Zugriffsrichtlinie erfüllt.

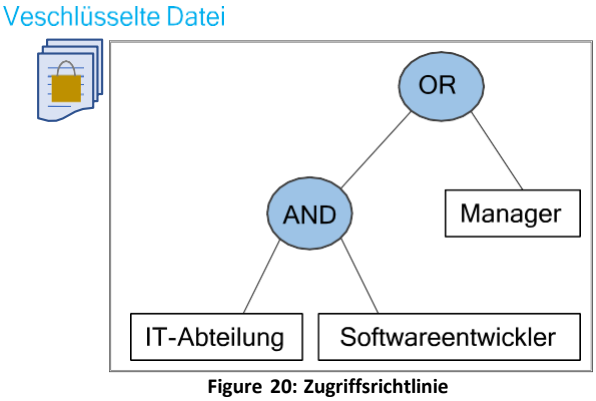

Figure 20: Zugriffsrichtlinie "(IT-Abteilung AND Softwareentwickler) OR Manager"

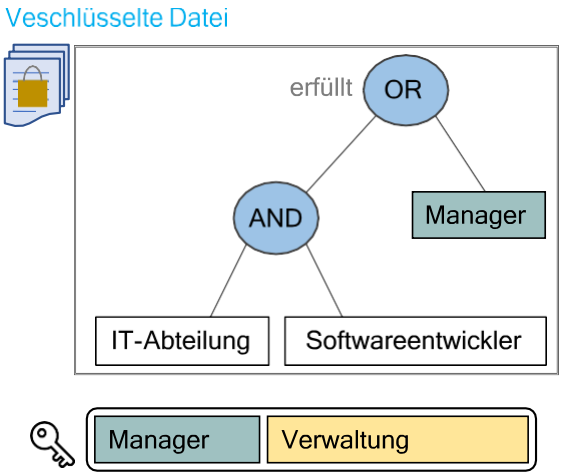

Figure 21: Die Attribute des Privatschlüssels 1 erfüllen die Zugriffsrichtlinie

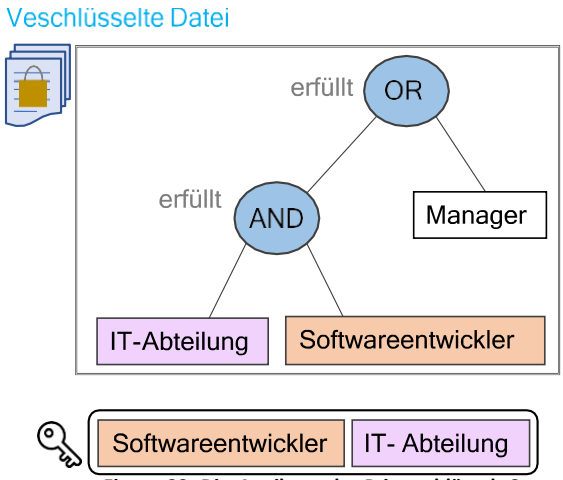

Figure 22: Die Attribute des Privatschlüssels 2 erfüllen die Zugriffsrichtlinie

An dieser Stelle ist es wichtig zu beachten dass private Schlüssel nicht kombiniert werden können. Nutzer können nicht ihre indivinduelle private Schlüssel zusammenlegen um eine Datei zu entschlüsseln. Dies ist eine Eigenschaft der attritbutbasierten Veschlüsselung, die als Kollusionsresistenz bezeichnet wird. Das Beispiel in Abbildung 23 verdeutlicht dieses Prinzip; der erste Nutzer besitzt einen privaten Schlüssel mit dem Attibut <Softwareentwickler> während der zweite einen privaten Schlüssel mit dem Attribut <IT-Abteilung>. Keiner von ihnen ist in der Lage die Datei zu entschlüsseln, da diese Attribute, wenn getrennt genommen werden, das Und-Gatter bzw. den booleschen Ausdrück der Zugriffsrichtlinie nicht erfüllen.

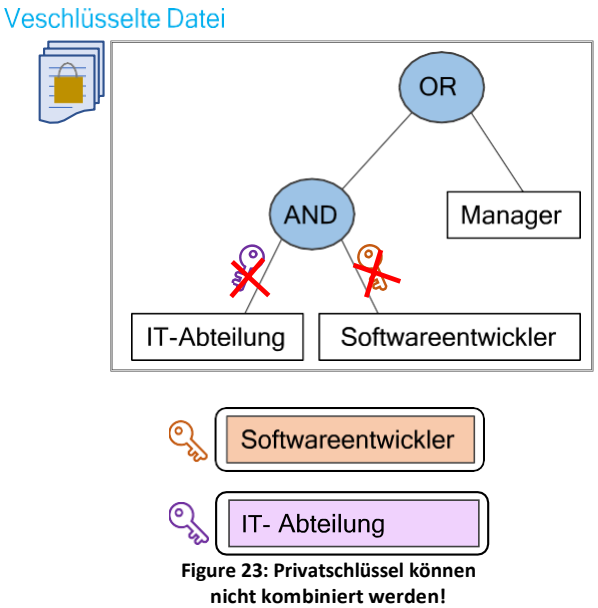

Figure 23: Privatschlüssel können nicht kombiniert werden!