LDAP und die Ablage von PGP–Schlüsseln im X.500–Directory

– Der Versuch einer Standardisierung –

Dr. Kurt Spanier
Zentrum für Datenverarbeitung, Universität Tübingen
kurt.spanier@zdv.uni–tuebingen.de

18. Februar 1999

Zusammenfassung

Die Entwicklung des X.500–Directory–Zugriffsprotokolls LDAP (Lightweight–Directory–Access–Protocol) hin zur weitverbreitetsten Methode der Verzeichnisdienstabfrage wird dargestellt. Nach einer kurzen Behandlung des X.500–Datenmodells zur Speicherung von beliebigen Informationen werden die gegenwärtigen Bemühungen und Fortschritte bei der Definition eines Speicherschemas für öffentliche PGP–Schlüssel (Pretty Good Privacy) angesprochen. Aufgrund der eher stockenden internationalen Standardisierungsbemühungen wird ein Übergangsmodell vorgestellt, das ohne größeren Aufwand umzusetzen ist. Die notwendigen Schritte für den Einsatz des Modells im DFN–geförderten Projekt AMBIX (Aufnahme von Mail–Benutzern in das X.500–Verzeichnis) werden aufgezeigt.

1 Das Internet – wie komme ich an einen (PGP–)Schlüssel?

Das weltweite Internet, das Netz der Computernetze, erlebt seit einigen Jahren einen Boom, der immer mehr Benutzer, aber auch Daten zur Folge hat. Dieser Anstieg von Angebot und Nachfrage beschert mannigfaltige Sicherheitsprobleme, von denen nur einige genannt sein sollen: es muß sicher sein, daß Benutzer auch diejenigen sind, für die sie sich ausgeben (Authentification); ein (Ver–)fälschen von Daten muß unterbunden werden, zumindest jedoch erkannt werden können (Verification); ein einmal erfolgter Datenaustausch darf nicht geleugnet werden, z.B. bei sog. On–Line–Geschäften (Non–Repudiation); sensible Daten, aber auch deren Absender und/oder Empfänger, müssen gegen das „Mitlesen“ durch Unberechtigte geschützt werden (Encryption).

All diese Probleme haben dazu geführt, daß Verschlüsselungsverfahren auch für den Internet–„Normalbenutzer“ immer mehr in Gebrauch kamen, wobei die asymmetrischen Verfahren eine prominente Stellung einnahmen. Doch Public–Key–Verfahren, wie das weitverbreitete PGP (Pretty Good Privacy; ftp://ftp.cert.dfn.de/pub/pgp/pgpi/), haben den erheblichen Nachteil, daß die öffentlichen Schlüssel jederzeit auch öffentlich verfügbar sein müssen, damit sie bei den oben genannten Problemen helfen können; ein Austausch zwischen wenigen Beteiligten mag zwar kurzfristig helfen, führt aber global gesehen nicht weiter.

Daher wurde gerade bei PGP von Beginn an auch ein PGP–Key–Server eingesetzt, der unter einer mehr oder minder bekannten Adresse öffentliche Schlüssel anbot. Nur, ein einzelner Server kommt sicherlich schnell an seine Leistungsgrenzen, wenn er massiv genutzt wird. Das gilt sowohl für die Anzahl der gespeicherten Schlüssel als auch für die Häufigkeit der Anfragen. Es muß daher nach Alternativen gesucht werden, öffentliche Schlüssel verteilt und für jeden zugänglich vorzuhalten.

Hier bietet sich das internationale X.500–Directory an, das bereits seit Jahren eine wachsende Infrastruktur von beteiligten Servern und auf diesen eine wachsende Anzahl von Benutzereinträgen aufweist. In Deutschland sind derzeit ca. 100.000 Personen auf 35 X.500–Servern eingetragen, wobei das DFN–Projekt AMBIX (http://www.directory.dfn.de/ambix/) mit ca. 35.000 Benutzern wohl den größten Bestand aufweist; hier soll der Datenbestand durch die Aufnahme weiterer Benutzergruppen in näherer Zukunft ausgebaut werden.

Die Ablage von PGP–Schlüsseln in diesem X.500–Verzeichnis hat neben der Verbreitung den Vorteil, daß über den Eigentümer auch weitere Informationen, wie Telefonnummer, Arbeitsgebiet, Verweis auf eine eigene WWW–HomePage u.a. zur Verfügung stehen; man bekommt nicht nur die reinen Schlüsseldaten. Ein großer Nachteil dieser Lösung ist aber, daß zur Zeit noch keine internationale Übereinkunft darüber besteht, wie PGP–Schlüssel im X.500–Verzeichnis abgelegt werden sollen. Es fehlt ein entsprechender Standard. Dies zu ändern wird von einer Reihe von Leuten angestrebt, die sich in jüngster Zeit mit dem Versuch einer Standardisierung befassen. Im Folgenden soll auf das X.500–Verzeichnis, insbesondere auf das LDAP–Protokoll, und die Ablage von PGP–Schlüsseln in diesem Verzeichnis eingegangen werden.

2 Die Geschichte mit LDAP

2.1 Zugriff auf das X.500–Verzeichnis: DAP und LDAP(v2)

Das X.500–Directory ist das letzte verbliebene Anwendungs–Protokoll des Open–System–Interconnection–Standards (OSI), das in der Internet–beherrschten Netzlandschaft noch von Bedeutung ist. Protokolle wie FTAM oder X.400–Mailing wurden spätestens in der zweiten Hälfte der 90er–Jahre aus dem Kommunikationsmarkt weitestgehend verdrängt. Dieses Überleben von X.500 ist zu einem nicht unerheblichen

Teil auf den Übergang vom OSI–definierten Directory–Access–Protocol (DAP) auf das Lightweight–Directory–Access–Protocol (LDAP) zurückzuführen.

Das LDAP–Protokoll wurde 1993[1] (und nochmals 1995[2]) zunächst als reine Ergänzung zum herkömmlichen DAP von einer Gruppe um Tim Howes, University of Michigan, eingeführt, um die Notwendigkeit für Directory–Clienten zu umgehen, auf dem eigenen Rechner einen OSI–Protokoll–Stack aufzusetzen. Dieses Protokoll war von Beginn an TCP/IP–basiert und sah den Einsatz eines sog. LDAP–Daemonen auf dem Rechner vor, der auch den X.500–DSA, den Directory System Agent bereitstellte. Dieser Daemon war das Bindeglied zwischen der LDAP–Anfrage des Clienten und der eigentlichen X.500–Anfrage beim DSA, die weiterhin über das DAP–Protokoll erfolgte. Damit mußte der umfangreiche OSI–Protokoll–Stack nur auf den Server–Maschinen eingesetzt werden, wohingegen die X.500–Clienten über TCP/IP arbeiteten.

Zugleich mit der Definition dieses Lightweight–Protokolls für den X.500–Zugriff wurde auch ein Application–Programming–Interface für die Programmiersprache C als RFC (Request for Comment) standardisiert[3]. Dieses API machte es möglich, neue oder bestehende Anwendungsprogramme in einfacher Weise „directory–aware“ zu machen, ihnen also die Möglichkeit zu geben, Informationen aus dem X.500–Directory zu holen bzw. die Daten dort zu verwalten.

2.2 Ersatz von Directory–System–Agents (DSAs): Standalone LDAP–Server

Kurz nach Einführung des LDAP–Protokolls wurde in der wichtigsten Implementation, der der University of Michigan (http://www.umich.edu/~dirsvcs/ldap/ldap.html), unter Leitung von Tim Howes auch der Directory–Server gegen etwas „Leichtgewichtigeres“ ausgetauscht, den Standalone LDAP Server (slapd in der Michigan–Implementation). Im Gegensatz zum bis dahin dominierenden ISODE–DSA baute dieser Slapd von Beginn an auf ein Platten–basiertes Speichersystem auf Grundlage von Datenbank–Libraries wie GDBM bzw. der Berkeley DB auf, wobei Datenzugriffe durch Indizes beschleunigt wurden. Auf diese Weise wurde aus einem Hauptspeicher–fressenden DSA–Monstrum ein schlanker X.500–Server, der nur noch über TCP/IP mit Clienten kommunizierte.

Heutzutage sind eine ganze Reihe solcher Standalone–Daemonen im Einsatz, die in der Hauptsache Abkömmlinge der Michigan–Implementation sind. Nach dem Weggang von Tim Howes zu Netscape wurde der frei verfügbare Michigan–Slapd nicht weiterentwickelt. Dafür baute Tim den kommerziellen Netscape Directory–Server, und Mark Wahl von Critical Angle Inc. entwickelte einen kommerziellen Server, der ausgehend vom Michigan–Server viele Bug–Fixes und Erweiterungen enthielt. Seit Mitte 1998 arbeitet

[1] Yeong, W., Howes, T., Kille, S. (1993): X.500 Lightweight Directory Access Protocol. RFC 1487

[2] Yeong, W., Howes, T., Kille, S. (1995): Lightweight Directory Access Protocol. RFC 1777

[3] Howes, T., Smith, M. (1995): The LDAP Application Program Interface. RFC 1823

das OpenLDAP–Projekt an einer Implementation, die den Michigan–Code unter dem Dach der „Open–Source–Definition“ weiterentwickelt (http://www.openldap.org/).

2.3 Weiterentwicklung von LDAP: LDAPv3

Die erste Version von LDAP, LDAPv2, fand weltweit bei Entwicklern so viel Anklang, daß 1997 von der IETF unter Vorsitz von Tim Howes ein neuer Standard, LDAPv3, in Angriff genommen wurde[4]. Dieser Standard wurde im Dezember 1997 in einer Reihe von ersten RFCs veröffentlicht. Neben seiner Kompatibilität zu den eher spartanisch ausgelegten Operationen von LDAPv2 (BIND, UNBIND, ADD, DELETE, SEARCH, MODIFY und MODRDN) und dem grundsätzlichen Festhalten am X.500–Datenmodell enthält dieser Standard eine ganze Menge von Erweiterungen, die es erlauben, neue Funktionalitäten in einfacher Weise zu implementieren und Datentypen im X.500–Verzeichnis selbst zu veröffentlichen. So können Server und Clienten sich zu Beginn eines Datenaustausches über zusätzliche Funktionalität einigen und bei fast jeder Operation modifizierende Optionen mitgeben. Der Server kann es z.B. auf Anfrage des Clienten übernehmen, Ergebnisse sortiert und/oder päckchenweise an den Clienten abzugeben oder auch definierte Teilausschnitte von Ergebnislisten zu schicken, so daß der Client nicht zunächst das ganze Ergebnis annehmen muß, bevor er dem Benutzer (Mensch oder Maschine) das Ergebnis einer Anfrage z.B. in einer Scroll–Box darbietet. Ein weiteres Feature von LDAPv3 ist, daß sich Client und Server auch über einzuhaltende Sicherheitsmechanismen (Authentifizierung des Nutzers oder Sicherung der Datenübertragung) einigen können, auch wenn hierfür im Protokoll u.a. wegen bestehender Exportbeschränkungen von Crypto–Software noch kein Default–Verfahren vorgesehen ist und die IETF daher zunächst noch formell vor dem Einsatz des Protokolls für die Änderung von Daten warnt.

Da LDAP auch in der Version v3 primär als Zugangs–Protokoll zum X.500–Directory definiert wurde, bleiben zwei wichtige Funktionalitäten, die im originären, auch den Server–Bereich abdeckenden X.500–Standard mittlerweilen enthalten sind, für die Implementierung von Standalone–LDAP–Servern mehr oder minder außen vor: die Frage der nutzer–initiierten Datenlieferung von beliebigen Servern und die Replikation von (Teil–)Daten zwischen Slapd–Servern. Solange im Hintergrund tatsächlich auf X.500–DSAs zugegriffen wird, wird beides von den DSAs geregelt, die sowohl nicht–vorliegende Daten von einem anderen DSA holen als auch ganze Datenbereiche in regelmäßigen Abständen austauschen können. Bei Standalone–LDAP–Servern ist der Zugriff auf nicht–vorliegende Daten durch ein sog. Referral, ähnlich dem HTTP–Referral geregelt, so daß es Aufgabe des Clienten ist, durch nochmalige Anfrage bei einem anderen Server fehlende Daten zu besorgen. Replikation wird bei Slapd–Daemonen durch sog. Replika–Dateien ermöglicht, in denen alle Änderungen am Datenbestand festgehalten und nachträglich durch ein Replikations–Programm an andere Server weiterverteilt werden. Durch diese Replikation lassen sich durchaus komplexe Master–Slave–Beziehungen aufbauen,

[4]Wahl, M., Howes, T., Kille, S. (1997): Lightweight Directory Access Protocoll (v3). RFC 2251

die allerdings immer nur in einer Richtung funktionieren; das Vorhalten von Misch–Datenbeständen mit Original– und Kopie–Daten auf ein und demselben Server, wie das mit OSI–konformen X.500–DSAs funktioniert, ist dagegen nicht möglich.

2.4 Die Daten in der Box: das X.500–Datenmodell

Durch das Festhalten am Datenmodell des X.500–Standards zeigt auch LDAP dessen Flexibilität. Daten sind hierarchisch in Einträgen abgelegt, deren absolute Namen (der Directory–Name oder DN) global eindeutig sind. Der relative Name eines Eintrags (RDN) unterhalb eines gegebenen Eintrags muß dagegen nur dort eindeutig sein, während gleiche RDNs in anderen Bereichen des Directory–Baums (DIT) erlaubt sind. Über die Zuordnung zu einer oder mehreren Objektklassen wird festgelegt, welche Attribute (die Felder im klassischen Datenbank–Sinn) im Eintrag erlaubt sind. Attribute können grundsätzlich, bis auf wenige Ausnahmen, mehr als einen Wert enthalten und man unterscheidet zwischen solchen Attributen, die obligatorisch, und solchen, die optional sind. Durch die Festlegung einer Attribut–Syntax kann ein Server Werte interpretieren und dadurch z.B. bei Eintragung oder Suche entsprechende Informationen nutzen.

Sowohl Objektklassen als auch Attribute können lokal neu definiert werden, ohne daß dies die Implementierung eines X.500–Servers berührt. Alle einem Server bekannten Definitionen können entsprechend den neueren Standardversionen (X.500 und LDAP) im X.500–Verzeichnis selbst abgelegt werden und sind damit sowohl dem Server als auch einem abfragenden Clienten verfügbar; bei älteren Programmen werden diese Definitionen noch beim Start aus Konfigurationsdateien gelesen. So ergibt sich eine hohe Flexibilität, die es erlaubt, nahezu jede Art von Daten im X.500–Verzeichnis abzulegen.

3 PGP–Schlüssel im Verzeichnis

3.1 Ein erster Versuch von Roland Hedberg und das Tübinger Modell

Anfang 1996 schrieb Roland Hedberg, damals Umea University, Schweden, einen ersten Draft für die IETF–ASID–Working Group, in dem er Objektklassen und Attribute für die Ablage von PGP–Schlüsseln in Personeneinträgen des X.500–Verzeichnisses vorschlug[5]. Als Objektklasse führte er „pGPKeyObject" mit fünf optionalen PGP–Attributen ein: „pGPKey" – eine ASCII–armored Version des Schlüssels –, „pGPKeyRev" – ein Feld für Key–Revocation –, „pGPKeyID" „pGPUserID" – die KeyID und UserID des Schlüssels – und „pGPKeyURL" – ein URL (Universal Ressource Locator), der auf eine externe

[5] Hedberg, R. (1996): Definition of X.500 Attribute Types and a Object Class to hold public PGP keys. Internet-Draft der IETF ASID Working Group (draft–ietf–asid–pgp–02.txt)

Ablage des Schlüssels verweisen konnte. Die Auswahl der Attribute war auf den damaligen Stand von PGP ausgerichtet. Das wichtigste Attribut war der Schlüssel, KeyID und UserID wurden als suchbare Informationen hinzugefügt.

Die Einführung von „pGPKeyURL“ löste eine heftige Diskussion aus, da hiermit ein nicht näher spezifizierter Verweis auf Information außerhalb des X.500–Verzeichnisses eingeführt wurde. Da die Erweiterung des URL–Formats auf andere Zugriffsprotokolle noch nicht diskutiert worden war, insbesondere die Definition des LDAP–URL, wurde befürchtet, daß LDAP–Clienten bzw. –Server um Protokolle wie HTTP oder FTP erweitert werden müßten. Dies stieß auf heftige Ablehnung, insbesondere auch bei denen, die nicht LDAP einsetzen wollten. Darüberhinaus wurde das Problem der Daten–Konsistenz angesprochen, das bei Verbindung so unterschiedlicher Zugriffsprotokolle als sehr hoch eingeschätzt wurde.

Außer „pGPKeyRev“ und „pGPKeyURL“ waren die Attribute als single–valued definiert, konnten also nur jeweils einen Wert enthalten. Diese Einschränkung zeigte auch schon den gravierenden Nachteil der vorgeschlagenen Lösung: es war von der Definition her nicht möglich, mehr als einen Schlüssel für eine gegebene Person zu speichern. Zum damaligen Zeitpunkt war das sicherlich vertretbar, da man noch nicht daran dachte, daß eine Person mehr als einen All–Round–Schlüssel haben könnte.

Zeitgleich und unabhängig von Hedberg wurde an der Universität Tübingen ebenfalls über die Speicherung von PGP–Schlüsseln im X.500–Verzeichnis nachgedacht. Hierbei kam die Objektklasse „pGPSecurityObject“ heraus mit den Attributen „pGPPublicKey“ „pGPKeyID“ und „pGPUserID“ Auch diese Lösung hatte den Nachteil, daß nur jeweils ein Schlüssel gespeichert werden konnte, doch wurde bereits hierfür der Zugriff auf die Attribute mit entsprechender Formatierung in einen LDAP–Finger–Daemonen und das Tübinger Web–X.500–Gateway TWEB (ftp://ftp–x500.uni–tuebingen.de/tweb/) eingebaut und ist seither nutzbar.

3.2 Der aktuelle Versuch einer internationalen Standardisierung

Ende 1998 fand sich eine Gruppe von Betreibern des deutschen DFN–X.500–Verzeichnisses, um den erneuten Versuch einer internationalen Standardisierung zu unternehmen. Die Diskussion wurde angestoßen durch Frank Brückner von der Firma Siemens, der sich bei Dante, speziell Karl–Peter Gietz, einem ehemaligen Mitarbeiter des DFN–Projekts AMBIX, nach dem Stand der Diskussion im europäischen Rahmen erkundigte. Da es nichts dergleichen gab, trat Peter Gietz auf das Projekt AMBIX zu, da er von drängenden Benutzeranfragen bezüglich Speicherung von PGP–Schlüsseln im AMBIX–Datenbestand wußte. Nach der Übereinkunft, daß die Diskussion eher im internationalen Rahmen anzusiedeln sei, wurde die Mail–Verteilerliste pgp–directory@dante.net initiiert, der sich neben den deutschen Teilnehmern auch eine Reihe internationaler Experten anschlossen, u.a. Roland Hedberg und David Chadwick von der Salford University, England (für ein Web–Archiv der Liste siehe http://www.dante.net/np/listarchives/pgp).

Ausgehend von einem aus Geheimhaltungsgründen nur grob bekanntgegebenen Modell der NAI (Network Associates, Inc., Kalifornien, USA) wurde von den Mitarbeitern des Projekts AMBIX und dem Betreiber der CA (Certification Authority) an der Universität Stuttgart, Oliver Göbel, ein Vorschlag entwickelt, wie PGP–Schlüssel zusammen mit einer Reihe von zusätzlichen Attributen, die in den vorherigen Ansätzen noch nicht berücksichtigt wurden, im X.500–Verzeichnis abgelegt werden sollten. Hierbei war ein Ziel, die vorherige Beschränkung auf nur einen Schlüssel pro Personeneintrag zu umgehen. Es wurden daher Objektklassen und Attribute definiert, die nicht als Ergänzung in bestehende Verzeichniseinträge aufgenommen, sondern für eigenständige PGP–Schlüssel–Einträge genutzt werden sollten; für Personen–Einträge wurde ein URL–Attribut definiert, das im Gegensatz zum Hedberg'schen pGPKeyURL als sog. LDAP–URL der Form „ldap://ldap–server:port/PGP–Eintrag“[6] auf einen Eintrag im X.500–Verzeichnis selbst verweist, so daß LDAP–Clienten die referenzierten Daten mit dem LDAP–Protokoll abfragen können.

Da bei der Entwicklung des Vorschlags ein aktiver CA–Betreiber beteiligt war, wurden eine Reihe von Attributen aufgenommen, die aus der CA–Policy heraus als sinnvoll erschienen. So sind neben dem Speichern schlüssel–relevanter Informationen – wie der Schlüssel selbst, seine KeyID, UserID, KeyLength und Fingerprint – und der Aufnahme von neueren PGP–Informationen – wie PGP–Version, Schlüssel–Algorithmus, Creation– und ExpirationDate – auch CA–relevante Attribute vorgesehen: KeyUsage, CA–Policy, CreationMode, Verweis auf zertifizierende Schlüssel u.a. Gerade mit dem letztgenannten Attribut ist es möglich, nicht nur den gesuchten Schlüssel selbst, sondern auch eine Hierarchie von zertifizierenden PGP–Schlüsseln zu erhalten, um darüber so etwas wie eine „Chain of Trust“ aufzubauen.

Dieser Vorschlag wurde Anfang Dezember 1998 über die Dante–Liste bekanntgemacht und zur Diskussion gestellt. Die Reaktion beschränkte sich neben einigen editorialen Vorschlägen zunächst nur auf die Frage, ob zur Schlüssel–Ablage ein oder zwei Einträge genutzt werden sollten. Der Vorschlag sah hier je einen Eintrag für den Schlüssel und die unmittelbar schlüssel–relevanten Attribute und einen weiteren für die übrigen Attribute vor. Insbesondere die Tatsache, daß zu einem Schlüssel mehrere UserIDs, das vorgeschlagene namensgebende Attribut der zweiten Eintragsart, vorhanden sein können, löste diese Diskussion aus. Eine Übereinkunft wurde noch nicht erzielt.

Wegen der zu diesem Zeitpunkt im vollen Gange befindlichen Vorbereitungen des Treffens der Open–PGP–Arbeitsgruppe der IETF in Orlando, war zum einen die Beteiligung gerade amerikanischer Experten eher spärlich, zum anderen konnte der Vorschlag nicht mehr auf die Tagesordnung der PGP–Gruppe gesetzt werden. Roland Hedberg konnte jedoch die Dante–Diskussionsliste bekanntmachen, so daß in Zukunft eine breitere Diskussion zu erhoffen ist. Daneben gelang es Roland, Ed Reed von Novell und Jeff Hodges von der Stanford University für das Thema zu interessieren, so daß beim nächsten Treffen der OpenPGP–Arbeitsgruppe im März 1999 möglicherweise ein Internet–Draft als

[6] Howes, T., Smith, M. (1997): The LDAP URL Format. RFC 2255

„individual submission“ eingereicht wird. In wie weit dieser Draft dem Vorschlag der Tübinger/Stuttgarter Gruppe entspricht, bleibt abzuwarten.

3.3 Eine experimentelle Implementation des NAI–Schemas

Aufgrund des fehlenden internationalen Standards und der gleichzeitigen Notwendigkeit zur Ablage von PGP–Schlüsseln im X.500–Verzeichnis wurde Anfang 1999 von Alex Iliynsky vom Red Rat's Club, Moskau, eine Implementation bekanntgemacht, die auf dem Schema der NAI beruht, soweit dieses bekannt ist (http://rednest.rosinter.ru/pgp_ldap_server.htm). Bei dieser Implementation werden Einträge, die ausschließlich PGP–Schlüssel in der ASCII–armored Form in einem Attribut „pGPKey“ enthalten, in einem separaten sog. Keyspace im Verzeichnis abgelegt. Unterhalb eines solchen Schlüssel–Eintrags wird ein Eintrag für die bei der Schlüsselerzeugung vorgegebenen UserID und andere schlüssel–relevante Informationen eingehängt. In diesem UserID–Eintrag sind Informationen enthalten wie „pGPKeyID“ „pGPKeyType“ (RSA oder DH/DSS), „pGPKeySize“ (Länge des Schlüssels), „pGPCreationTime“ und „pGPExpirationTime“ Nachträglich hinzugefügte UserIDs werden allerdings nicht in dem UserID–Eintrag hinterlegt, sondern ausschließlich im PGP–Schlüssel angefügt. Hierdurch sind diese alternativen Werte nicht suchbar, ein Nachteil, der in dieser Form in der NAI–Implementierung nicht enthalten ist.

Dieser LDAP–Keyserver, der auf dem Code des OpenLDAP–Projektes beruht, ist so angepaßt, daß er über die LDAP–Schnittstelle von PGP 5.x abgefragt werden kann. Suchen nach User– oder KeyID liefern den Schlüssel zurück, der direkt in einen PGP–Schlüsselring übernommen werden kann.

Wegen der Auftrennung der relevanten Attribute auf zwei Einträge mußte der LDAP–Server gepatched werden, da die Suche in Attributen des untergeordneten Eintrags stattfindet, während als Suchergebnis ein Attribut aus dem übergeordneten Eintrag zurückgeliefert wird. Diese Art der Implementation ist in diesem Fall jedoch wenig sinnvoll, da ebensogut alle Attribute des UserID–Eintrags auch in den PGP–Schlüsseleintrag selbst verlagert werden könnten, wodurch die Notwendigkeit zur Änderung des Server–Codes entfiele. Ein weiterer Nachteil der Implementation ist, daß so, wie der Server im Binär–Format für Windows–NT verbreitet wird, ein Zeitstempel in den beiden Einträgen bei Änderungen nicht fortgeschrieben wird, sondern immer den Wert der Eintrags–Erzeugung beibehält, wodurch nachträgliche Manipulationen an den Einträgen nicht nachvollziehbar sind.

3.4 Eine „Übergangslösung“ für das DFN–Projekt AMBIX

Der oben beschriebe Server von Alex Iliynsky ist für das DFN–Projekt AMBIX in dieser Form nicht geeignet. Neben der fehlenden Suchbarkeit von Alternativ–UserIDs und der Manipulationsmöglichkeit wegen inkorrekter Zeitstempel hat die Implementation darüberhinaus den Nachteil, daß die Quellen derzeit nicht zur Verfügung stehen. OpenLDAP hat große Vorbehalte, Änderungen am LDAP–Server vorzunehmen, die nicht auf einem internationalen Standard beruhen und deren lizenz–rechtliche Fragen noch unklar sind. Dieses ist sowohl aus Sicht der Software–Pflege als auch deren Akzeptanz und Einsetzbarkeit durchaus verständlich.

Das Projekt AMBIX hat daher in jüngster Zeit ein weiteres Konzept entwickelt, das unter Berücksichtigung der zurückliegenden internationalen Diskussion Objektklassen und Attribute, aber auch Mechanismen zur Einbindung in die bestehende AMBIX–Systemumgebung definiert. Oberste Maxime dieses als „schnelle Übergangslösung“ anzusehenden Konzepts war, daß PGP–Schlüssel und –Attribute in bestehende X.500–Server aufzunehmen seien, und daß Abfragen von diesen Servern ohne Software–Änderung beantwortet werden müßten. Möglich ist dies durch die Anwendung lokaler Schema–Erweiterungen, die dem Server über Konfigurationsdateien bekannt gemacht werden. Darüberhinaus sollten auch Clienten ohne größere Änderungen die Schlüssel recherchieren können, wobei insbesondere LDAP–fähige PGP–Versionen (z.B. 5.x) und das im AMBIX–Projekt eingesetzte Web–X.500–Gateway TWEB berücksichtigt wurden.

3.4.1 Definition von Objektklassen und Attributen

Im Gegensatz zu der genannten NAI–Implementation und dem ersten Vorschlag der Arbeitsgruppe aus Stuttgart und Tübingen wird in der AMBIX–Übergangslösung nur ein Eintrag für alle Schlüssel–Attribute angelegt. Diese PGP–Schlüsseleinträge, sowohl Benutzer– als auch CA–Schlüssel, werden in einem eigenen PGP–Keyspace abgelegt, wobei dieser X.500–Bereich auch von einem eigenständigen Server verwaltet werden kann. In Benutzer–Einträgen werden in Form von LDAP–URLs (s.o.) Verweise auf die jeweiligen Benutzerschlüssel eingetragen, die von Client–Software (z.B. TWEB) zum Abfragen der Schlüssel genutzt werden können.

Um bei der Anwendung des Konzepts möglichst flexibel zu sein, wurden mehrere Objektklassen definiert, die obligatorische und optionale Attribute zur Speicherung der Informationen zu Verfügung stellen. Hier ist zunächst die Objektklasse „pGPKeyContainer“ zu nennen, die alle für den PGP–Schlüssel selbst relevanten Attribute definiert. In dieser Objektklasse sind obligatorische Attribute wie „pGPKey“ für den ASCII–armored Schlüssel, „pGPKeyEntryName“ als Namensattribut für den Eintrag, „pGPKeyID“ „pGPUserID“ „pGPKeyLength“ „pGPAlgorithm“ u.a. enthalten. Als optionale Attribute stehen „pGPVersion“, „cn“, „mail“ oder „description“ zur Verfügung. Eine vollständige Definition dieser Objektklasse ist aus Tabelle 1 zu entnehmen.

Neben der Speicherung von Informationen über den PGP–Schlüssel selbst sieht das vom Projekt AMBIX entwickelte Konzept auch vor, daß in Benutzerschlüsseln Informationen über eine mögliche zertifizierende CA und in CA–Schlüsseln Informationen über die CA abgelegt werden. Zu diesem Zweck wurde die Objektklasse „cAInformationObject“ definiert, deren Attribute aus Tabelle 2 zu entnehmen sind. Durch Aufnahme von Attributen

Attribut	**Syntax**	**opt.**	**Beschreibung**
pGPKeyEntryName	CIS		Namensattribut; für User–Keys: KeyID_ServerID_Time CA–Keys: KeyID_ServerID_CA–ID
pGPKey	IA5		ASCII–armored PGP–Schlüssel
pGPKeyID	CIS		Schlüssel–ID
pGPUserID	CIS		User–ID
pGPFingerPrint	CIS		Fingerprint
pGPKeyLength	INT		Schlüssel–Länge in Bit
pGPAlgorithm	CIS		Schlüssel–Algorithmus (RSA, DH/DSS etc.)
pGPVersion	CIS	***	PGP–Version des Schlüssels
pGPCreationTime	UTC	***	Zeit der Schlüssel–Erzeugung
pGPExpirationTime	UTC	***	Zeit des Schlüsselablaufs
pGPKeyUsage	CIS	***	Verwendungszweck des Schlüssels nach X.509 (z.B. certificate signing, digital signature, data encryption)
pGPKeyRevoked	INT	***	Flag für Schlüsselrücknahme 0 = gültig, 1 = vom Benutzer zurückgezogen, 2 = CA–Schlüssel zurückgezogen, 3 = von Benutzer und CA zurückgezogen
cn	CIS	***	Name des Benutzers
sn	CIS	***	Nachname des Benutzers
mail	CIS	***	E–Mail–Adresse des Benutzers
description	CIS	***	Freitext–Beschreibung
pGPContainerVersion	CIS		Version der PGP–Objektklassen–Definition; für die vorliegende Definition 1.01

Tabelle 1: *Definition der Objektklasse „pGPKeyContainer“.*
*Die Objektklasse stellt PGP–Schlüssel–relevante Attribute mit den angegebenen Syntaxdefinitionen zur Verfügung (CIS = CaseIgnoreString, IA5 = IA5String [kann Zeilenumbrüche enthalten], INT = Integer, UTC = Universal Time Code); Attribute, die mit '***' in der Spalte 'opt.' markiert sind, sind optional*

wie „cACertKeyLink“ oder „cAExternalCertDN“ kann mit einem geeigneten Clienten gezielt Information über einen zertifizierenden Schlüssel abgefragt werden. Auf diese Weise kann eine komplette „Chain of Trust“ erzeugt werden, vorausgesetzt, daß alle Schlüssel auf dem Server selbst oder einem bekannten externen LDAP–Server abgelegt sind.

Im Fall von „cAExternalCertDN“ wird der im Attribut gespeicherte Directory–Name (DN) der CA durch Informationen aus einem im Server vorhandenen Eintrag „cn=pGPServerInfo“ ergänzt (siehe unten), so daß der Client auf den Directory–Eintrag der CA zugreifen kann. Ist in diesem Eintrag ein Verweis auf den Schlüssel der CA in Form eines LDAP–URL vorhanden, können durch einen weiteren Zugriff auf das Verzeichnis dieser Schlüssel und ggf. nachfolgende zertifizierende Schlüssel erhalten werden.

Attribut	**Syntax**	**Typ**	**Beschreibung**
cACertKeyLink	DN	U/C	DN des zertifizierenden Schlüssels auf demselben Server
cAExternalCertDN	DN	U/C	DN einer CA, deren Schlüssel **nicht** auf dem Server selbst hinterlegt ist
cAExternalCertURL	URL	U/C	(LDAP–)URL einer CA, deren Schlüssel **nicht** auf dem Server selbst hinterlegt ist und von der **kein** DN bekannt ist (z.B. Verweis auf HTTP–Seite mit CA–Policy)
cAPolicy	URL	C	Verweis auf eine HTTP–Seite, auf der die Policy einer CA beschrieben ist
cADN	DN	C	DN einer CA
keyOwner	CIS	U/C	Beschreibung des Schlüssel–Eigentümers (USER, CA, CA–ROOT, RA)
keyCreationMode	CIS	U/C	Beschreibung der Schlüssel–Erzeugung (OWN = auf Nutzerhardware, AUTO = auf CA–Hardware durch Benutzer, CA = von der CA erzeugt)
withPGPRevCert	INT	U/C	Flag für Hinterlegung eines Revocation–Certificates (0 = nein, 1 = ja)

Tabelle 2: *Definition der Objektklasse „cAInformationObject“.*
Die Objektklasse stellt Attribute zur Verfügung, mit denen Informationen über einen zertifizierenden Schlüssel oder eine CA in einem PGP–Schlüssel–Eintrag abgelegt werden können. Alle Attribute sind optional. In der Spalte „Typ“ ist der Typ des jeweiligen PGP–Schlüssel–Eintrags genannt, in dem das Attribut sinnvoll ist (C = CA–Schlüssel, U = User–Schlüssel). Syntaxangaben: DN = Directory–Name, URL = Universal Ressource–Locator; weitere Angaben siehe Tabelle 1

Die LDAP–Schnittstelle von PGP 5.x liest in einem ersten Zugriff auf einen Server zunächst einen Eintrag mit Namen „cn=pGPServerInfo“, um aus diesem die beiden Attribute „baseKeySpaceDN“ und „basePendingDN“ zu erhalten. Diese beiden Attribute mit Syntax DN geben für nachfolgende Schlüssel–Abfragen die Suchbasis für PGP–Schlüssel im Server an. Um neben diesen beiden Attributen weitere Informationen für den Betrieb eines LDAP–basierten PGP–Keyservers bereitstellen zu können, wurde im Konzept des Projekts AMBIX eine dritte Objektklasse mit Namen „pGPServerInfo“ definiert. Diese Objektklasse enthält auch Attribute für die Versionsnummern der unterstützten „pGPKeyContainer“–Definition(en), eine Server–Kennung und Angaben von CAs, für die im Server Schlüssel abgelegt sind bzw. (optional) für die ein externer Server bekannt ist. Die Definition dieser Objektklasse ist in Tabelle 3 enthalten.

Attribut	**Syntax**	**opt.**	**Beschreibung**
cn	CIS		Name des Eintrags (cn=pGPServerInfo)
baseKeySpaceDN	DN		DN, unterhalb dessen die PGP–Schlüsseleinträge abgelegt sind
basePendingDN	DN	***	DN, unterhalb dessen in Bearbeitung befindliche Schlüssel abgelegt werden
pGPServerID	CIS		Freiwählbare Identifikation für den Server (Nutzung: Generieren der Namen von PGP–Einträgen)
pGPContainerVersion	CIS		(Liste von) Versionsnummer(n) der vom Server unterstützten „pGPKeyContainer“–Definition(en)
pGPCAList	DN	***	DN–Liste der CAs, für die Schlüssel im Server abgelegt sind
pGPCAExList	URL	***	(LDAP–)URL–Liste von CAs, für die **keine** Schlüssel im Server abgelegt sind, deren Key–Server jedoch bekannt ist

Tabelle 3: *Definition der Objektklasse „pGPServerInfo“.*
Die Objektklasse stellt Attribute für die Beschreibung eines LDAP–Keyservers zur Verfügung. Für die Beschreibung von „Syntax“ und „opt.“ siehe Tabellen 1 und 2

3.4.2 Ein Ablaufschema zur Nutzung der Definition

Mit den oben beschriebenen Definitionen von Objektklassen und Attributen können PGP–Schlüssel mit der Objektklasse „pGPKeyContainer“ in einfacher Weise in einem X.500–Server abgelegt werden. Hierzu ist die Einrichtung eines Directory–Bereichs als BaseKeySpace notwendig, unterhalb dessen die Schlüssel flach angeordnet eingehängt werden. Alle relevanten Informationen zum Server werden im Eintrag „cn=pGPServerInfo“

mit Objektklasse „pGPServerInfo“ abgelegt. Um die Namen von Benutzerschlüsseln unterhalb des BaseKeySpace eindeutig zu machen, wird der RDN eines PGP–Eintrags im Attribut „pGPKeyEntryName“ aus der KeyID des abzulegenden Schlüssels, der Kennung des Servers und der aktuellen Zeit (im UTC–Format) bei Erzeugung des Eintrags zusammengesetzt. Für CA–Schlüssel wird der Namensbestandteil „Zeit“ gegen eine aus dem DN der CA abgeleitete eindeutige Kennung ausgetauscht, so daß diese Einträge zeitunabhängige Namen bekommen. Liegt bei Lieferung nur der PGP–Schlüssel vor, müssen vor dessen Ablage im Verzeichnis Werte für alle obligatorischen Attribute der Objektklasse „pGPKeyContainer“ aus dem Schlüssel selbst mittels PGP entnommen werden. Bei Lieferung der Daten durch eine CA sollten dies aber seltene Fälle sein. Das Attribut „pGPContainerVersion“ wird für die vorliegende Definition immer auf „1.01“ gesetzt.

Für den Einsatz eines solchen Servers im Rahmen des Projekts AMBIX ist vorgesehen, daß nur von einer CA zertifizierte Schlüssel abgelegt werden. Durch Setzen der

Objektklasse „cAInformationObject“ werden hierbei auch CA–relevante Attribute im PGP–Schlüssel– Eintrag aufgenommen. Insbesondere ein Verweis auf den zertifizierenden CA–Schlüssel ist

wichtig, um eine „Chain of Trust“ aufbauen zu können. Natürlich ist hierfür die vorhergehende Ablage des CA–Schlüssels erforderlich, einschließlich der Ablage von Schlüsseln, die ihrerseits den Schlüssel zertifizieren. Als „oberster“ Schlüssel im Bereich des Deutschen Forschungsnetzes (DFN) ist als erster der Schlüssel der DFN–PCA aufzunehmen, von dem alle CA–Schlüssel zunächst zertifiziert werden. Es ist abzusehen, daß die Aufnahme solcher CA–Schlüssel eher selten ist, so daß sie „von Hand“ verarbeitet werden können, während für Routine–Lieferungen entsprechende Programme zu entwickeln sind.

Das Projekt AMBIX muß vor der ersten Lieferung von PGP–Schlüsseln durch CAs ein geeignetes Datenformat definieren, mit dem alle benötigten Informationen zur Verfügung gestellt werden können. Diese Definition steht jedoch zum gegenwärtigen Zeitpunkt noch aus und soll daher hier nicht weiter behandelt werden. In jedem Fall ist vorgesehen, daß die Daten mit einem dem Projekt bekannten CA–Schlüssel signiert und anschließend mit einem öffentlichen Schlüssel des Projekts AMBIX verschlüsselt werden. Auf diese Weise ist sichergestellt, daß alle in das Verzeichnis aufgenommenen Schlüssel authentisch sind.

Auch die Abbildung einer Cross–Zertifizierung von CA–Schlüsseln ist mit der gegebenen Definition möglich, da mit den Attributen „cACertKeyLink“, „cAExternalCertDN“ und „cAExternalCertURL“ alle Fälle von Verweisen auf cross–zertifizierende CAs bzw. deren Schlüssel abgedeckt sind. Sollte ein cross–zertifizierter Schlüssel an das Projekt AMBIX geliefert werden, muß die liefernde CA alle notwendigen Angaben über die beteiligte Cross–CA machen, so daß eine sichere Identifizierung und Erstellung eines entsprechenden Verweises auf diese CA möglich ist. Für den Fall einer nachträglichen Aufnahme der Cross–CA in den Datenbestand des Projekts AMBIX können im Rahmen der hier gegebenen Definitionen alle betroffenen Schlüssel–Einträge so geändert werden, daß sie anschließend Verweise auf den dann internen CA–Schlüssel enthalten.

Die vorgestellte Definition ist so flexibel, daß mit ihr auch andere Schlüsselarten entweder ohne Änderung oder durch geringfügige Erweiterungen abgelegt werden können. Hier sind vor allem SSL– und SMIME–Schlüssel zu nennen. Für Feldinhalte gleicher Bedeutung und Syntax können hierbei dieselben Attribute benutzt werden, die ggf. neue Alternativnamen erhalten. Neue Feldinhalte werden durch Erweiterung bestehender bzw. durch neue Objektklassen hinzugefügt. In diesem Fall erhält die Definition eine neue Versionsnummer, die von Clienten entsprechend erkannt und verarbeitet werden kann.

3.4.3 Implementation der Definition im Projekt AMBIX

Neben der Einbindung der hier vorgestellten Objektklassen und Attribute in die Schemadefinition der im Projekt AMBIX eingesetzten X.500–Server und der Definition eines geeigneten Datenformats zur Lieferung von PGP–Schlüsseln an das Projekt muß insbesondere die eingesetzte Software ergänzt bzw. geändert werden. Neue Programme sind notwendig, die Daten einer CA verarbeiten, überprüfen und in den X.500–Servern eintragen. Da die Aufnahme von zertifizierten PGP–Schlüsseln in einem eigenständigen Server und der Eintrag eines entsprechenden Verweises im bereits vorhandenen oder in Bearbeitung befindlichen Benutzereintrag datenschutzrechtlich nicht kritisch ist, können diese Schlüssel–Daten unmittelbar im frei zugänglichen X.500–Verzeichnis des Projekts veröffentlicht werden. Eine erneute zustimmende Einbeziehung des betroffenen Benutzers ist aus Sicht des Projekts AMBIX nicht notwendig.

Darüberhinaus ist das projektseitig eingesetzte Web–X.500–Gateway TWEB dahingehend zu ändern, daß es alle neuen Attribute in der erforderlichen Art und Weise darstellen kann. Die meisten Änderungen beziehen sich hierbei auf die Konfiguration der Software, da viele Darstellungstypen bereits bekannt und verfügbar sind. Lediglich bei der Verarbeitung von Verweisen auf CAs (interne oder externe) muß die Software dahingehend erweitert werden, daß sie für die Erstellung eines solchen Verweises Informationen aus dem fraglichen Eintrag selbst mit Informationen aus einem weiteren Eintrag („cn=pGPServerInfo“) verknüpft. Dieses sollte ohne Probleme zu implementieren sein; bereits zum gegenwärtigen Zeitpunkt ist oftmals die gezeigte Information aus verschiedenen X.500–Einträgen zusammengesetzt.

Die hier vorgestellten Definitionen zur Ablage von öffentlichen PGP–Schlüsseln sollten relativ zügig in den Regelbetrieb des Projekts AMBIX aufzunehmen sein. Sollte danach eine internationale Standardisierung zu konkreten Ergebnissen führen, wird die interne Ablage der Schlüssel durch das Projekt entsprechend umgestellt, wobei eine Änderung von Schnittstellen zur Außenwelt als eher unwahrscheinlich angesehen wird. Dies betrifft sowohl die Frage der Datenlieferung durch CAs als auch die Anpassung der TWEB–Software zur Darstellung der Daten im X.500–Verzeichnis.