

# Einsatz und Bedeutung Elliptischer Kurven für die elektronische Signatur

[Elisabeth.Oswald@iaik.at](mailto:Elisabeth.Oswald@iaik.at)



# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>3</b>
<b>2</b>	<b>Einsatzmöglichkeiten Elliptischer Kurven in der Public Key Kryptographie</b>	<b>4</b>
2.1	Algorithmen . . . . .	4
2.1.1	Schlüsselaustausch- bzw. Schlüsseinigungsverfahren . . . . .	4
2.1.1.1	Schlüsseltransportverfahren . . . . .	4
2.1.1.2	Schlüsseinigungsverfahren . . . . .	5
2.1.1.3	Diffie-Hellman Schlüsselaustausch . . . . .	5
2.1.1.4	ECMQV . . . . .	5
2.1.2	Elektronische Signatur . . . . .	6
2.1.3	Identifikations- und Authentifikationsschemata . . . . .	7
2.2	Standards . . . . .	7
2.2.1	TLS/SSL . . . . .	8
2.2.2	PKCS . . . . .	8
2.2.3	SECG . . . . .	9
2.2.4	WAP/WTLS . . . . .	9
2.2.5	ATM . . . . .	9
2.2.6	S/MIME . . . . .	10
2.3	Internet-Zertifikate X.509v3 . . . . .	10
2.3.1	Zertifikatshierarchien . . . . .	13
2.3.2	Widerruf von Zertifikaten . . . . .	13
2.4	Praktischer Einsatz von ECDSA in Zertifikaten . . . . .	14
2.4.1	Ein einfaches ECDSA Root-Zertifikat . . . . .	14
2.4.2	Ein mit ECDSA signierter RSA-Schlüssel . . . . .	16

<b>3</b>	<b>Derzeitiger Einsatz Elliptischer Kurven in Security Produkten</b>	<b>18</b>
3.1	Sichtweise der Hardware- bzw. Softwarehersteller . . . . .	18
3.1.1	Chipkartenhersteller . . . . .	18
3.1.2	Security Software Provider . . . . .	19
3.1.2.1	Crypto API's . . . . .	19
3.1.2.2	Security Software . . . . .	20
3.1.2.3	Standard Software . . . . .	20
3.2	Sichtweise der Zertifizierungsdiensteanbieter . . . . .	21
3.2.1	Österreich . . . . .	21
3.2.2	Deutschland . . . . .	21
3.2.3	EU . . . . .	21
3.2.4	International . . . . .	21
<b>4</b>	<b>Zusammenfassung</b>	<b>22</b>
<b>A</b>	<b>Technische Details</b>	<b>23</b>
A.1	Grundlagen von ECC . . . . .	23
A.2	Vergleich mit RSA, DSA . . . . .	25
A.3	Ist ECC patentiert? . . . . .	27
A.4	ASN1-Dump des Testzertifikates . . . . .	27
<b>B</b>	<b>ECC-Brainpool</b>	<b>31</b>
	<b>Bibliographie</b>	<b>32</b>

# Kapitel 1

## Einleitung

Kryptosysteme basierend auf elliptischen Kurven (kurz ECC <sup>1</sup>) haben seit ihrer Einführung durch Miller und Kobitz, Interesse auf sich gezogen. Der grösste Vorteil von ECC ist die wesentlich kürzere Schlüssellänge im Vergleich zu Kryptosystemen die auf endlichen Körpern operieren. Die Konsequenzen die sich daraus ergeben sind einfachere Implementierung der Arithmetik, geringere Bandbreite und geringere Speicheranforderungen. Diese Eigenschaften sind insbesondere bei Applikationen auf Smartcards gefordert, aber auch die drahtlose Datenübertragung (WTLS, WAP) und der boomende Handheld Markt erfordern von Applikationen die gerade genannten Eigenschaften.

Ein wichtiger Aspekt bei der praktischen Einsetzbarkeit neuer kryptografischer Techniken ist einerseits die Interoperabilität verschiedener Implementierungen, und andererseits die Verfügbarkeit von patentfreien Algorithmen. Um die Interoperabilität zu gewährleisten wird momentan an einer Fülle von Standards gearbeitet. Wichtige Institutionen wie ANSI, ISO oder IEEE haben bereits fertige Standards für ECC erarbeitet. Ein spezieller Aspekt dieses Dokuments ist die Einschätzung von ECC im Bereich der digitalen Signatur, da die digitale Signatur Bestandteil vieler Protokolle ist. Fazit ist, dass ECC in vielen Standards bereits inkludiert ist, und das weitere Standardisierungsverfahren laufen. Auch die Verfügbarkeit von ECC Implementierungen, sowohl in Hard- als auch in Software ist gegeben. Alle renommierten Firmen aus der Security-Branche bieten solche Implementierungen an, und wollen ihre Produktpalette ausweiten. Das Zugpferd ist dabei die Firma Certicom, deren Lizenzen von fast allen Securitysoftware-, und Securityhardwareanbietern verwendet werden.

Kapitel 2 gibt einen Überblick über den aktuellen Stand der Standardisierung relevanter Protokolle wie S/MIME oder SSL, und zur Standardisierung relevanter kryptografischer Verfahren (ECDSA, ECDH). In Kapitel 3 wird die Sichtweise von Hard- und Softwarehersteller und Zertifizierungsdiensteanbieter dargestellt. Dabei wird das Augenmerk auf die Verfügbarkeit von konkreten ECC Implementierungen gelegt. Technische Details werden in Anhang A behandelt. In der Zusammenfassung wird über den aktuellen Stand von ECC resümiert, und Ausblicke für die zukünftige Entwicklung gegeben.

---

<sup>1</sup>ECC bildet den Überbegriff für diverse auf elliptischen Kurven basierender Kryptosysteme, und wird deswegen in den nachfolgenden Abschnitten immer stellvertretend dafür verwendet.

## Kapitel 2

# Einsatzmöglichkeiten Elliptischer Kurven in der Public Key Kryptographie

Dieser Abschnitt gibt einen Überblick über die Einsatzmöglichkeiten elliptischer Kurven in kryptografischen Protokollen. Dazu werden zunächst die wichtigsten Algorithmen (ECDSA, ECDH, ...) beschrieben und in Zusammenhang mit den derzeit verfügbaren „Kernstandards“ gebracht. Zusätzlich werden andere meist noch im Entwurfsstadium befindliche Standards aufgelistet. Schlußendlich wird auch noch der Zusammenhang von Zertifikaten und Elliptischen Kurven beleuchtet.

### 2.1 Algorithmen

#### 2.1.1 Schlüsselaustausch- bzw. Schlüsseleinigungsverfahren

Wollen mehrere Parteien miteinander auf sicherem Weg kommunizieren, und dabei ein symmetrisches Verschlüsselungsverfahren verwenden, so muss ein gemeinsamer, geheimer (Sitzungs)-Schlüssel festgelegt werden. Schlüsselfestlegungsverfahren unterscheidet man danach, ob der Sitzungsschlüssel nur von einem, oder von mehreren Kommunikationspartnern generiert wird.

##### 2.1.1.1 Schlüsseltransportverfahren

Ein Schlüsseltransportverfahren ist ein Schlüsselfestlegungsverfahren bei dem der Sitzungsschlüssel von nur einem einzigen Kommunikationsteilnehmer generiert wird. Im ANSI X9.63 Working Draft [2] werden zwei konkrete Schemata spezifiziert.

Das ist zum einen das **1-Pass Transport Schemata**, in dem die zu transportierenden Schlüsseldaten (Schlüssel, Schlüssellänge), mit dem öffentlichen Schlüssel des Empfängers verschlüsselt werden. Als Kryptosystem kann dabei eines der in [2] spezifizierten ECC Verfahren dienen. Das **3-Pass Transport Schemata** verwendet zusätzlich noch ein Signaturverfahren um eine Schlüsselauthentifizierung für den mit dem 1-Pass Transport Schemata übertragenen Schlüssel zu gewährleisten.

### 2.1.1.2 Schlüsseleinigungsverfahren

Der festzustellende Sitzungsschlüssel wird hier von allen Kommunikationspartnern gemeinsam erzeugt. Der Beitrag eines Kommunikationspartners darf dabei für die anderen Kommunikationspartner nicht vorhersehbar sein. Die in [2] spezifizierten Verfahren, wie verschiedene **Unified Model Schemata** (Ephemeral, Static, 1-Pass, Full Unified, oder auch Full, bzw. Combined Unified Model with Key Confirmation Scheme), **DH Schemata**, **Station-to-Station Schemata** oder **MQV Schemata**) beruhen prinzipiell auf dem Diffie-Hellman Schlüsselaustausch, und/oder auf dem MQV-Verfahren.

### 2.1.1.3 Diffie-Hellman Schlüsselaustausch

Das *Diffie-Hellman Schlüsselaustauschverfahren*, wurde 1976 erstmals vorgestellt [6]. Die dem Kryptosystem zugrundegelegte algebraische Struktur ist die endliche abelsche Gruppe  $\mathbb{F}_q^*$ . Das Prinzip beruht dabei auf der **Diffie-Hellman Annahme**, dass es praktisch unmöglich ist, aus der Kenntnis von  $g^a$  und  $g^b$ , den Wert von  $g^{ab}$  zu berechnen. Diese Idee lässt sich analog auf die Struktur der Punkte auf einer elliptischen Kurve übertragen. Angenommen Alice und Bob wollen einen gemeinsamen Schlüssel erzeugen. Dazu wählen sie sich (öffentlich) einen endlichen Körper  $\mathbb{F}_q$  und darüber eine elliptische Kurve  $E$ . Weiters wählen sie sich einen öffentlichen Punkt  $P \in E$  mit Ordnung  $n$  als „Basispunkt“ aus. Alice berechnet eine Zufallszahl  $k_A$  und sendet  $k_AP$  an Bob. Analog generiert Bob eine Zufallszahl  $k_B$  und sendet  $k_BP$  an Alice. Ihr gemeinsamer geheimer Schlüssel ist damit  $k_Ak_BP$ . In Abbildung 2.1 ist das Protokoll zusammengefasst.

Vorbereitung : Öffentliche Wahl von $\mathbb{F}_q$ , $E$ und $P$ .		
Schritt	Alice	Bob
1.	Generiere $k_A \in \{2, 3, \dots, n-2\}$	Generiere $k_B \in \{2, 3, \dots, n-2\}$
2.	Sende $k_AP$ zu Bob	Sende $k_BP$ zu Alice
3.	Berechne $Q = k_A(k_BP)$	Berechne $Q = k_B(k_AP)$
Der gemeinsame geheime Schlüssel ist $Q$ .		

Abbildung 2.1: Diffie-Hellman Schlüsselaustausch (ECDH)

### 2.1.1.4 ECMQV

Das *Menezes-Qu-Vanstone* (MQV) - Verfahren leitet einen gemeinsamen, geheimen Parameter aus zwei privaten und zwei öffentlichen Schlüsseln ab. Dabei wird der Kofaktor  $h = \#E(F_q)/n$  der elliptischen Kurve verwendet. Ein Schlüsselpaar ist dabei statisch  $((d_{1,A}, Q_{1,A})$  bzw.  $(d_{1,B}, Q_{1,B}))$ , und das andere Schlüsselpaar wird regelmässig neu berechnet  $((d_{2,A}, Q_{2,A})$  bzw.  $(d_{2,B}, Q_{2,B}))$ . In Abbildung 2.2 ist das Protokoll skizziert.

Schritt	Alice	Bob
1.	Sende $Q_{2,A}$ an Bob	Sende $Q_{2,B}$ an Alice
2.	Berechne $s_A = d_{2,A} + \bar{Q}_{2,A}d_{1,A} \bmod n$	Berechne $s_B = d_{2,B} + \bar{Q}_{2,B}d_{1,B} \bmod n$
3.	Berechne $K = hs_A s_B$	Berechne $K = hs_A s_B$
Der gemeinsame geheime Schlüssel ist $K$ .		

Abbildung 2.2: Menezes-Qu-Vanstone Schlüsseleinigung (ECMQV)

### 2.1.2 Elektronische Signatur

Der auf elliptischen Kurven basierende Signaturalgorithmus ECDSA wurde 1998 als ISO (International Standards Organization) Standard (ISO 14888-3), 1999 als ANSI (American National Standards Institute) Standard (ANSI X9.62), und 2000 als IEEE (Institute of Electrical and Electronics Engineers) Standard (IEEE P1363) und FIPS (Federal Information Processing Standards) Standard (FIPS 186-2) akzeptiert.

Digitale Signaturen gelten als elektronisches Pendant zur handgeschriebenen Unterschrift. Eine digitale Signatur ist eine Zahl (bzw. ein Zahlenpaar) das von einem Geheimnis des Signierers und der zu signierenden Nachricht abhängig ist. Die Gültigkeit einer digitalen Signatur muss von allen Parteien überprüfbar sein. Weiters muss mit einer digitalen Signatur die Datenintegrität und die Datenauthentizität sichergestellt werden können. Digitale Signaturschemen finden auch in zahlreichen kryptografischen Protokollen, wie z. B. authentifizierter Schlüsseltransfer (ANSI X9.63 [2], ISO/IEC 11770-3 [12]) oder authentifizierte Schlüsseleinigung (ISO/IEC 11770-3 [12]), ihre Anwendung. Abbildung 2.3 bzw. 2.4 skizziert den ECDSA. Alice signiert hierbei eine Nachricht  $M$  mit den laut [1] gewählten Kurvenparametern  $(q, a, b, G, n)$  und ihrem geheimen Schlüssel  $d$ . Bob verifiziert die Gültigkeit der Signatur mit Alice öffentlichen Schlüssel  $Q = dG$ .

1.	Wähle Zufallszahl $k : 1 \leq k \leq n - 1$ .
2.	Berechne $kG = (x_1, y_1)$ und $r = x_1 \bmod n$ . Ist $r = 0$ dann wiederhole 1.
3.	Berechne $k^{-1} \bmod n$ .
4.	Berechne $e = \text{SHA-1}(M)$ .
5.	Berechne $s = k^{-1}(e + dr) \bmod n$ . Ist $s = 0$ dann wiederhole 1.
6.	Alice Signatur für die Nachricht $M$ ist das Zahlenpaar $(r, s)$ .

Abbildung 2.3: ECDSA Signatur Generierung

Ein anderes bekanntes Signaturschemata ist die *Nyberg-Rueppel* Signatur. Der grosse Vorteil dieses Verfahrens ist die Möglichkeit die Nachricht aus der Signatur extrahieren zu können. Ein Nyberg-Rueppel Primitiv ist im P1363A spezifiziert, nicht aber in Ansi X9.62.  $r^3$ -Security Engineering (<http://www.r3.ch>) besitzt ein Patent auf dieses Schemata, und die Firma Certicom behauptet in einem Schreiben an den IEEE P1363 Chair, die Exklusivrechte an diesem Patent für Nordamerika zu haben.

1.	Verifiziere das $r, s$ im Intervall $[1, n - 1]$ sind.
2.	Berechne $e = \text{SHA-1}(M)$ .
3.	Berechne $w = s^{-1} \bmod n$ .
4.	Berechne $u_1 = ew \bmod n$ und $u_2 = rw \bmod n$ .
5.	Berechne $X = u_1G + u_2Q$ . Ist $X = \mathcal{O}$ dann ist die Signatur ungültig, sonst berechne $v = x_1 \bmod n$ wobei $X = (x_1, y_1)$ ist.
6.	Akzeptiere die Gültigkeit der Signatur wenn $v = r$ gilt.

Abbildung 2.4: ECDSA Signatur Verifikation

### 2.1.3 Identifikations- und Authentifikationsschemata

Identifikationsprotokolle erlauben einem Kommunikationspartner (dem *Verifizierer*) die Identität eines anderen Kommunikationspartners (dem *Beweiser*) sicherzustellen. Dazu stellt der Verifizierer (Bob) dem Beweiser (Alice) eine Aufgabe (challenge). Diese Aufgabe kann Alice nur dann lösen wenn sie ein Geheimnis kennt. Die Lösung (response) wird von Bob verifiziert. Er erkennt die Identität von Alice an, wenn die Lösung korrekt ist. Dieses Konzept kann mittels eines Signaturverfahrens (zum Beispiel ECDSA) realisiert werden. Will sich Alice bei Bob identifizieren, bekommt sie von Bob eine Zufallszahl (Challenge) und signiert sie. Sie schickt die Signatur zurück an Bob (Response). Bob kann durch die Verifikation der Signatur die Identität von Alice überprüfen. Bei diesem Verfahren ist es wichtig, dass Alice öffentlicher Schlüssel vor Manipulationen geschützt ist. Das heißt er muss zertifiziert sein.

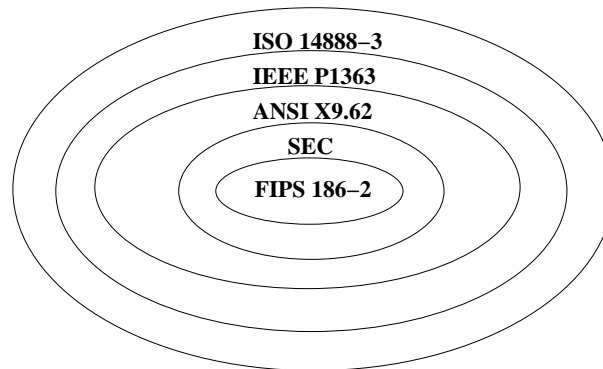
## 2.2 Standards

Die wichtigsten Standards, also ANSI X9.62, IEEE P1363, FIPS 186-2 und ISO/IEC 1488-3 sind bereits erwähnt worden. Zusätzlich wurde mit dem ISO/IEC 15946 noch ein weiterer, mehrteiliger Standard publiziert. Die Spezifikationen des Standards umfassen sowohl Schlüsselaustausch bzw. Schlüsseleinigung, als auch Signaturschemata. Die nachfolgende Tabelle gibt einen Überblick über die verschiedenen Standards und über einige der darin spezifizierten Algorithmen.

Standard	inkludierte Schemata	Entwicklungsstadium
ANSI X9.62	ECDSA	Fertiggestellt
ANSI X9.63	ECIES, ECDH, ECMQV	Entwurf
FIPS 186-2	ECDSA	Fertiggestellt
IEEE 1363	ECDSA, ECDH, ECMQV	Fertiggestellt
IEEE P1363A	ECIES	Entwurf
IPSEC	ECDSA, ECDH	Entwurf
ISO 14888-3	ECDSA	Fertiggestellt
ISO 15946	ECDSA, ECDH, ECMQV	Fertiggestellt

Tabelle 2.1: Standards und Schematas

Die Beziehung der Standards untereinander ist in Abbildung 2.2 veranschaulicht.



### 2.2.1 TLS/SSL

*Transport Layer Security* (kurz TLS), soll prinzipiell eine sichere und vertrauliche Kommunikation zwischen Applikationen auf der Transport Layer Ebene ermöglichen. TLS gliedert sich dabei in zwei Ebenen, dem TLS Record und dem TLS Handshake. Während TLS Record für die sichere Datenübertragung zuständig ist, realisiert TLS Handshake dabei die Authentifizierung und den Schlüsselaustausch zwischen Client und Server. Die aktuelle TLS Version 1.0 [18] ist ein direkter Nachfolger vom *Secure Sockets Layer* (kurz SSL) 3.0 Protokoll. Obwohl die Unterschiede zwischen TLS und SSL relativ gering sind ist Interoperabilität nicht a priori gegeben (TLS muss auf SSL zurückschalten). Die wesentlichen Unterschiede (im jetzigen RFC) sind die Spezifikation eines anderen MAC Algorithmus und die Einbeziehung zusätzlicher kryptografischer Algorithmen (eben elliptischer Kurven). Die Firma Certicom bietet in ihrer aktuellen Implementierung von TLS/SSL die Verwendung elliptischer Kurven bereits an.

### 2.2.2 PKCS

Public Key Cryptography Standards (PKCS) sind von RSA Laboratories initiierte Standards um die Verbreitung von Public-Key Kryptografie zu fördern. Viele der de facto Standards (PKIX, SSL, S/MIME, ANSI X9.x) sind aus PKCS Dokumenten entstanden. Der noch in der Entwicklung befindliche PKCS 13 formuliert den Elliptic Curve Cryptography Standard. Beinhaltet werden sollen dabei die verschiedenen Aspekte von ECC wie

- Schlüsselgenerierung und Validierung,
- Digitale Signatur,
- Verschlüsselung,
- Schlüsseleinigung
- ASN.1 Beschreibung der Parameter, Schlüssel und Schemataspezifikationen,
- Sicherheitsaspekte.

PKCS 13 soll vor allem die Integration von ECC in Applikationen, die auf anderen PKCS Dokumenten basieren, ermöglichen.

### 2.2.3 SECG

Die *Standards for Efficient Cryptography Group* (SECG) ist ein Zusammenschluss von Firmen die im Bereich der Kryptografie und Informationssicherheit arbeiten. Ein Hauptziel dieser Gruppe ist verstärkt auf die Interoperabilität der verschiedenen Kryptografieprodukte einzugehen um die Entwicklung im Bereich des E-Commerce zu beschleunigen. Dazu werden eben die *Standards for Efficient Cryptography* (SEC) entwickelt. Im SEC 1 wird die Implementierung der ECC Basisalgorithmen (analog zu den bereits existierenden Standards) beschrieben. Dies inkludiert

- Parametergenerierung,
- Schlüsselgenerierung,
- Signatur (ECDSA),
- Verschlüsselung (ECIES<sup>1</sup>)
- Schlüsseleinigung.

Im SEC 2 werden Parameter für ECC empfohlen. Dabei wird auf auf bestehende Standards und in Entwicklung befindliche Standards rücksicht genommen. Für die im Dokument empfohlenen Parameter werden auch ASN.1 Bezeichner festgelegt.

### 2.2.4 WAP/WTLS

Das *Wireless Application Protocol* (WAP) (aktueller Stand 18. Februar 2000) stellt Sicherheitsmechanismen für die drahtlose Kommunikation zur Verfügung. ECC ist dabei im WAP Security Layer, über das WTLS (Wireless Transport Layer Security) Protokoll eingebunden. WTLS ermöglicht eine sichere End-zu-End Verbindung, d. h. es gibt Mechanismen zur Verschlüsselung, zur Wahrung der Datenintegrität und der Authentifikation.

### 2.2.5 ATM

In der *ATM Security Specification Version 1.0* werden ECC Verfahren spezifiziert. Dazu gehören ECDSA-like, ECDSA-like Asymmetric Authentication und das Elliptic Curve Key Agreement Scheme Diffie-Hellman Analogue. Diese Schemata lehnen sich stark an die Spezifikationen laut ANSI X9.62 oder IEEE 1363 an. Sie unterscheiden sich nur in einem kleinen Detail bei der Signaturerzeugung. Die ATM Spezifikation verlangt bei der Signaturerzeugung die Invertierung des privaten Schlüssels statt der Invertierung der Zufallszahl. Das hat den Vorteil, dass diese Inverse vorausberechnet werden kann, und so die Signaturerzeugung beschleunigt wird. Tabelle 2.2 stellt kurz die ECDSA Signatur und die ECDSA-like Signatur gegenüber:  $M$  ist hierbei die zu signierende Nachricht,  $e$  der Hashwert davon,  $x_1$  ist die  $x$ -Koordinate von  $kG$ , wobei  $G$  der gewählte Basispunkt auf der Kurve ist.  $n$  ist der bekannte Modul.

---

<sup>1</sup>ECIES ist ein Verschlüsselungsschema das nach einem Vorschlag von Abdullah, Bellare und Rogaway entworfen wurde. Es ist ein Schema das ECDH, symmetrische Verschlüsselung und ein MAC Verfahren kombiniert und ist auch in P1363 und ANSI X9.63 spezifiziert.

ECDSA-like	ECDSA
$r = x_1 \bmod n$	$r = x_1 \bmod n$
$s = (kr - e)d^{-1}$	$s = k^{-1}(e + dr)$

Tabelle 2.2: ECDSA vs. ECDSA-like

### 2.2.6 S/MIME

Secure/Multipurpose Internet Mail Extensions (S/MIME) stellt Sicherheitsmechanismen für den elektronischen Nachrichtenverkehr zur Verfügung. Dabei ist S/MIME aber nicht auf E-Mails begrenzt, sondern kann mit allen MIME kompatiblen Protokollen (z. B. http) verwendet werden. Eine S/MIME formatierte Nachricht ist prinzipiell eine Kombination vom MIME mit dem CMS Format. Die Cryptographic Message Syntax (CMS) definiert die Syntax wie zum Beispiel eine signierte Nachricht auszusehen hat. CMS erlaubt dabei eine mehrfache Kapselung von Nachrichten, das heißt eine bereits verschlüsselte Nachricht kann z. B. auch noch signiert werden. CMS erlaubt auch zusätzliche Attribute zur Nachricht (z. B. Datum, Uhrzeit). CMS selbst ist algorithmenunabhängig. Der Internet Draft *Elliptic Curve S/MIME* [16] regelt die Einbindung von ECC in den CMS. Dabei werden Ergänzungen zu folgenden CMS Inhaltstypen definiert :

- „SignedData“, unterstützt nun ECDSA laut [1].
- „EnvelopedData“, unterstützt Schlüsseleinigungsverfahren laut [2].
- „AuthenticatedData“, unterstützt Schlüsseleinigungsverfahren mit Authentifikation laut [2].

## 2.3 Internet-Zertifikate X.509v3

Um digitalen Signaturen vertrauen zu können, muss die korrekte Zuordnung von öffentlichen und privaten Schlüsseln sichergestellt werden. Dazu werden Zertifikate verwendet. Zertifikate sind Datenstrukturen, die einem Subjekt (z. B. einer Person, einem privaten Schlüssel) einen öffentlichen Schlüssel zuordnen. Diese Zuordnung erfolgt durch eine Certification Authority (CA), die sicherstellt dass die Zuordnung korrekt ist, und dann den öffentlichen Schlüssel signiert.

X.509 ist das am weitesten verbreitete Zertifikatsformat und wird bei Applikationen wie SSL oder S/MIME verwendet. Abbildung 2.5 zeigt den prinzipiellen Aufbau eines X.509 Zertifikates. Die erste X.509 Version entstand 1988 als ITU-T Definition. X.509v2 hatte als zusätzliche Felder „Issuer“ und „Subject Identifier“. Die aktuelle Version X.509v3 beinhaltet zusätzlich das Feld „Extensions“. Obwohl einige Implementierungen manchmal noch die Versionen 1 und 2 unterstützen, geht der Trend eindeutig in Richtung X.509v3. Die im Zertifikat enthaltenen Daten sind in der ASN.1 Syntax formuliert. X.509 erörtert dabei aber prinzipiell keine Algorithmen, sondern gibt nur einen „Platzhalter“ oder Object Identifier (OID) für unterstützte Algorithmen und Parameter vor, und überlässt sogenannten Profilen, wie PKIX

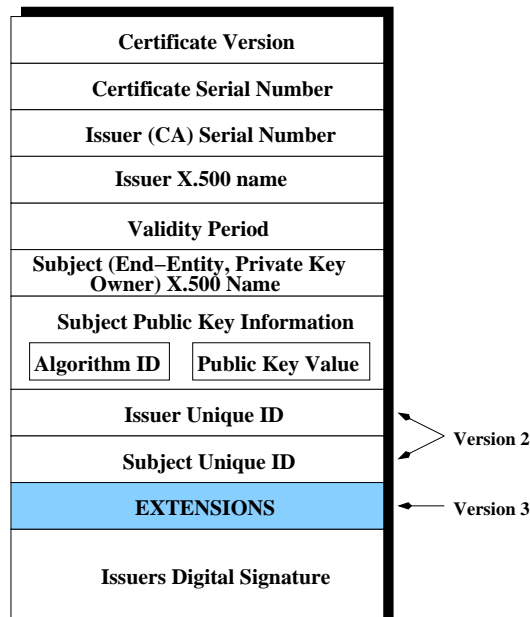


Abbildung 2.5: X.509 Zertifikatsstruktur

und X9.55, die genauere Spezifikation. Um elliptische Kurven zu unterstützen, muss ein Syntax für ECC Schlüssel und ihre Verwendung definiert werden. Konkrete Auswirkungen hat eine solche Definition auf folgende Felder des Zertifikates :

- **Certificate Signature Format.** Im Zertifikat muss eindeutig der von der CA verwendete Public Key Algorithmus identifizierbar sein. Das betrifft sowohl das Feld das den Namen des Algorithmus angibt, als auch das Format der Signatur.
- **Public Key Format.** Ein X.509 Zertifikat enthält explizit den Public Key. Deswegen muss dessen Format eindeutig spezifiziert sein. In der ASN.1 Syntax werden zum Public Key auch noch dessen Parameter gespeichert.
- **Key Usage.** Hier kann der Verwendungszweck des Public Keys angegeben werden. Das könnte z. B. Verschlüsselung, oder Signaturerstellung sein.

Im aktuellen Internetdraft [3] sind OIDs für ECDSA und ECDH spezifiziert. CAs die mit dieser Spezifikation konform gehen wollen müssen diese OID erkennen. Zu diesen Spezifikationen gibt es folgende Anmerkungen:

1. Die Benutzung desselben Schlüssels für Signatur und Verschlüsselung ist erlaubt, wird aber nicht empfohlen.
2. Kurvenparameter können sowohl explizit im Zertifikat angegeben werden, als auch durch eine Referenz auf eine „namentlich bekannte Kurve“ definiert werden.
3. Als Key Usage sind für ECDSA erlaubt :
  - digitalSignature,

- nonRepudiation,
- keyCertSign,
- cRLSign,

4. Als KeyUsage sind für ECDH erlaubt :

- keyAgreement,
- encipherOnly,
- decipherOnly,

wobei encipherOnly und decipherOnly exklusiv sind, d. h. ein ECDH Schlüssel kann nur entweder encipherOnly oder decipherOnly sein. Abbildung 2.6 zeigt ein Zertifikat, das einen ECDSA Schlüssel zertifiziert.

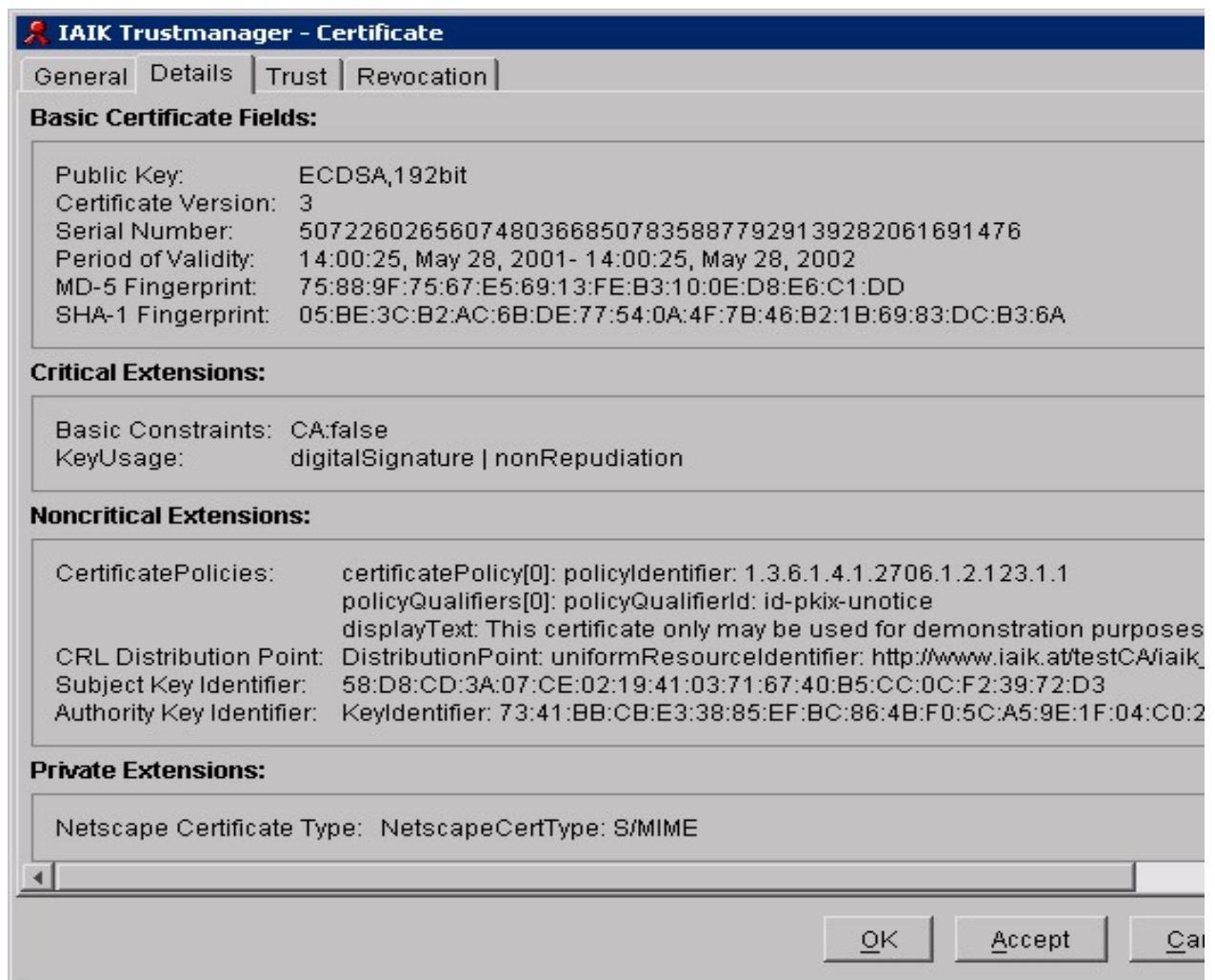


Abbildung 2.6: ECDSA Zertifikat im IAIK-Trustmanager

### 2.3.1 Zertifikatshierarchien

Zur Überprüfung eines öffentlichen Schlüssels dient das entsprechende Zertifikat. Der öffentliche Schlüssel des Zertifikats kann wiederum digital signiert sein, usw. Auf diese Weise ergeben sich ganze Zertifikatshierarchien. Die CA an der Spitze wird immer als „Root CA“ bezeichnet. Welche Auswirkungen hat der Einsatz von bsp. ECDSA auf die Zertifikatsverifikation, wenn man davon ausgeht, dass alle Teilnehmer die in [1] spezifizierten OIDs verstehen? Zumeist vertraut man der direkt übergeordneten CA. Verwendet diese ECDSA zur Signatur, so müssen alle Benutzer die dieser CA zugeordnet sind ECDSA verstehen um die Signatur überprüfen zu können. Muss eine ganze Zertifikatskette abgearbeitet werden und verwendet eine der darin vorkommenden CAs ECDSA („gemischte Hierarchie“), so muss ebenfalls der Benutzer der die gesamte Kette überprüfen will, ECDSA implementiert haben.

### 2.3.2 Widerruf von Zertifikaten

Wenn ein Zertifikat kompromitiert wurde, ist es notwendig, es auch vor Ablauf seiner definierten Gültigkeitsdauer widerrufen zu können. X.509 definiert eine Methode zum Widerruf von Zertifikaten, die sogenannte *Certificate revocation list* (CRL). Eine CRL ist eine mit einem Zeitstempel und einer Signatur versehene Liste, die die Seriennummern aller widerrufenen Zertifikate enthält.

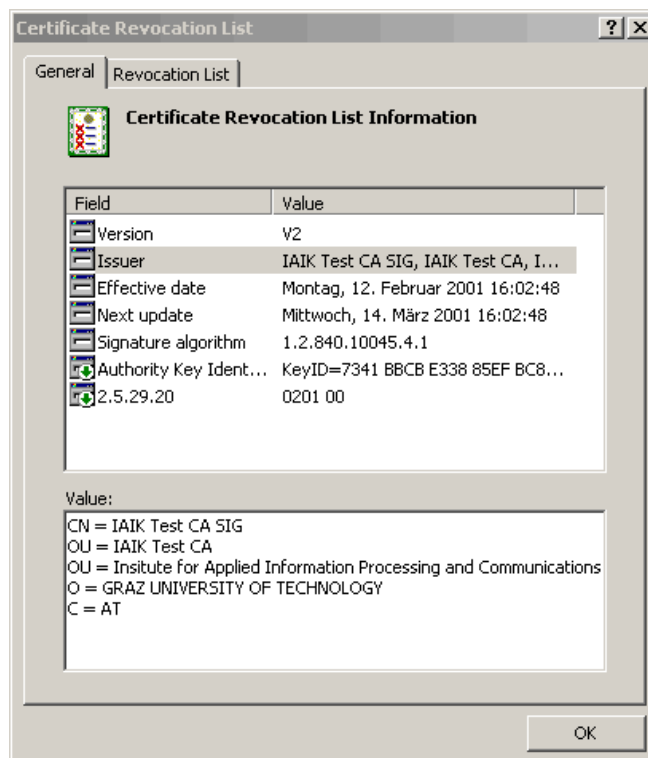


Abbildung 2.7: ECDSA signierte CRL

Diese digitale Signatur kann natürlich ebenfalls mit dem ECDSA Algorithmus berechnet werden. Abbildung 2.7 zeigt eine mit ECDSA signierte CRL. Erkennen kann man dies durch den verwendeten OID (1.2.840.10045.4.1). Die CRL wird jeweils von der CA für die von ihr ausgestellten Zertifikate bereitgestellt, und ist in einem öffentlichen Verzeichnis zugänglich. Die CA ist dafür verantwortlich aktuelle CRLs bereitzustellen. Wird die digitale Signatur der CRL mit ECDSA berechnet, so müssen klarerweise alle Benutzer in dieser Hierarchieebene ECDSA ebenfalls implementiert haben, um die CRL verifizieren zu können. Das Problem bei CRLs ist sie aktuell zu halten. Das *Online Certificate Status Protocol* (OCSP) ermöglicht es Applikationen eine Statusabfrage zu Zertifikaten zu machen. Ein OCSP Client setzt dazu eine entsprechende Anfrage an einen OCSP Server (Responder) und hebt die Gültigkeit eines Zertifikats auf je nachdem wie die Antwort des Responders ausfällt. Die Antwort des Responders ist digital signiert. Wenn dies eine ECDSA Signatur ist, müssen wiederum die Benutzer in der Zertifikatshierarchie ebenfalls ECDSA implementiert haben um die Signatur überprüfen zu können.

## 2.4 Praktischer Einsatz von ECDSA in Zertifikaten

In diesem Abschnitt wird die Verwendung des Signaturalgorithmus ECDSA anhand einiger konkreter Beispiele illustriert. Die verwendeten Zertifikate wurden mit der ECC Implementierung der IAIK-JCE erstellt. Die IAIK Java Cryptographic Extension (kurz IAIK-JCE) ist eine Menge von APIs und Implementierungen kryptographischer Funktionen, die sowohl symmetrische als auch asymmetrische Algorithmen enthält. Die ECC Implementierung ist noch in der Entwicklung und umfaßt deswegen noch nicht alle möglichen Algorithmen. Die Kernfunktionalität des ECDSA ist jedoch implementiert.

### 2.4.1 Ein einfaches ECDSA Root-Zertifikat

Das erste hier vorgestellte Zertifikat ist ein Root-Zertifikat (also ein selbstsigniertes Zertifikat) mit der definierten *keyUsage : Digital Signature, Certificate Signing, Off-line CRL Signing und CRL Signing*. Als ASN.1 Bezeichner für den ECDSA-Algorithmus wurde der in ANSI X.62 definierte OID 1.2.840.10045.4.1 verwendet. Signiert wurde ebenfalls ein ECDSA Public-Key (OID 1.2.840.10045.2.1) unter Verwendung des Hashalgorithmus SHA-1.



Abbildung 2.8: ECC Root Zertifikat

Wie zu sehen ist, wird ECDSA von den *Microsoft Crypto Shell Extensions* noch nicht unterstützt. Deswegen ist die Zertifikatsinformationsbox auch nicht in der Lage, ein ECDSA Zertifikat zu verifizieren. Aus diesem Grund erscheint in der Zertifikatsinformationsbox (siehe Abbildung 2.8) die Mitteilung, daß die Integrität des Zertifikats nicht garantiert werden kann. Die Schlussfolgerung daraus, daß das Zertifikat gefälscht wäre ist aber, zumindest in diesem Fall, falsch. Ein „Beweis“ dafür, daß Microsoft ECDSA schlichtweg noch nicht versteht ist auch die Tatsache, daß der OID im Feld „Signature algorithm“ nicht durch seinen Namen angegeben wird, sondern über seine numerische Repräsentation im Objektbezeichnerbaum. Trotzdem kann man sich in der Detailansicht (siehe Abbildung 2.9) das Zertifikat genauer ansehen. Wie in der Grafik zu erkennen ist, wird als OID 1.2.840.10045.4.1 verwendet. Dieser, und die anderen ECDSA spezifischen OIDs sind in den schon vorgestellten Kernstandards (ANSI, IEEE, ISO) spezifiziert. Der zweite, im Zertifikat verwendete und ECDSA spezifische OID ist der *ecPublicKey* OID 1.2.840.10045.2.1. Dieser öffentliche Schlüssel beinhaltet den

OID des zugrunde liegenden endlichen Körpers, die Parameter  $a$  und  $b$  der Kurve sowie den Basispunkt und dessen Ordnung. Im Appendix A.4 ist der komplette ASN1-Dump des ECC Testzertifikates beigefügt.

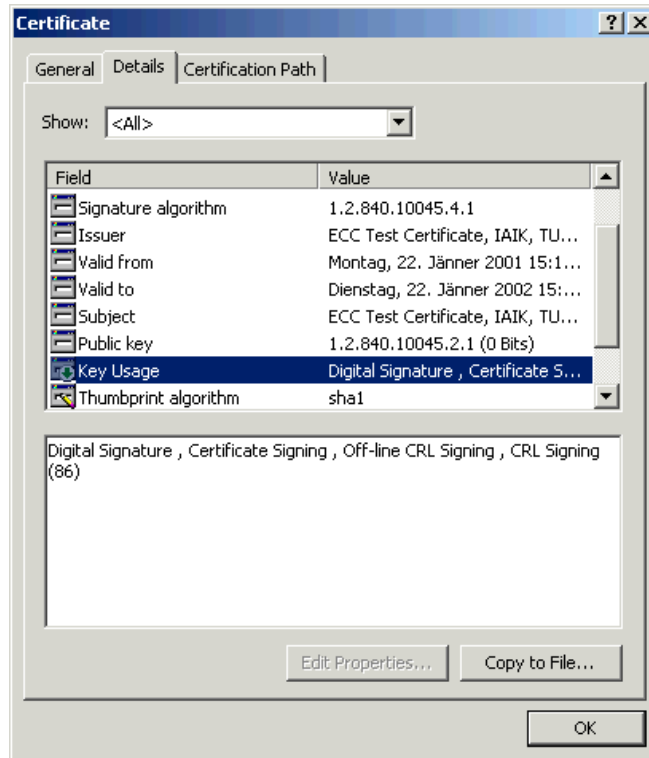


Abbildung 2.9: ECC Root Zertifikat - Detailansicht

Vergleicht man dazu das in Abbildung (2.10) dargestellte ECDSA-Zertifikat eines RSA Public Keys, sieht man das Microsoft zwar nicht den Signaturalgorithmus aber immerhin den Public-Key Algorithmus (eben RSA) erkennt (und deswegen auch den Namen anstatt des OID ausgibt).

### 2.4.2 Ein mit ECDSA signierter RSA-Schlüssel

Natürlich kann ein RSA-Schlüssel auch mit ECDSA signiert werden (siehe dazu ebenfalls Abbildung 2.10). In diesem Fall wäre beispielsweise Microsoft Outlook (Microsofts Email-Client) in der Lage eine mit dem RSA-Schlüssel signierte Email zu verifizieren, nicht jedoch die Zertifikatskette selbst (solange Microsoft ECDSA nicht unterstützt).

## 2.4. PRAKTISCHER EINSATZ VON ECDSA IN ZERTIFIKATEN

---

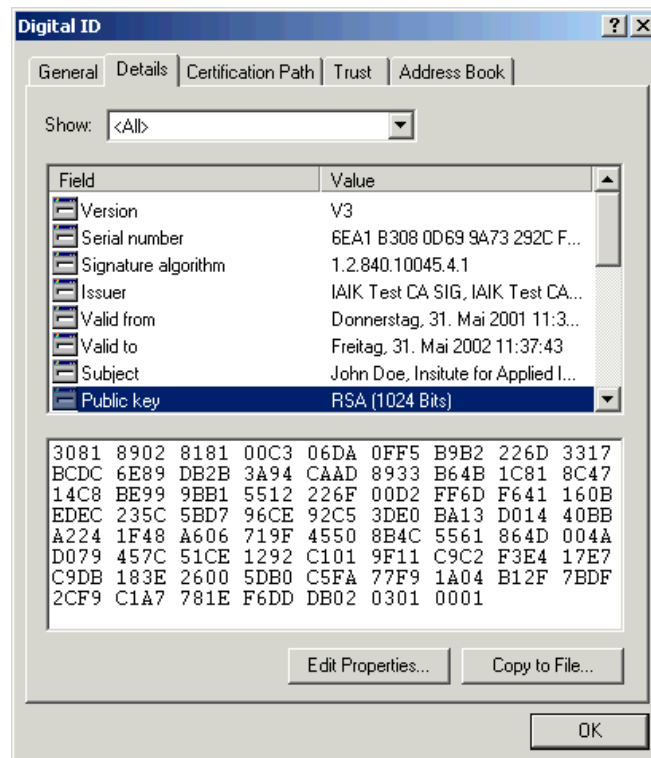


Abbildung 2.10: ECDSA signierter RSA Public Key

## Kapitel 3

# Derzeitiger Einsatz Elliptischer Kurven in Security Produkten

Die aktuelle aber auch zukünftige Bedeutung eines Kryptosystems kann immer auch daran gemessen werden, ob und wie es in gängigen Hard- und Softwareprodukten eingesetzt wird. In den folgenden Abschnitten wird ein Überblick über aktuelle Securityprodukte gegeben, die bereits elliptische Kurven einsetzen. Ein weiterer Aspekt der behandelt wird, ist die Sichtweise der Zertifizierungsdiensteanbieter zum Einsatz elliptischer Kurven.

### 3.1 Sichtweise der Hardware- bzw. Softwarehersteller

#### 3.1.1 Chipkartenhersteller

Smartcards sind aus dem alltäglichen Gebrauch nicht mehr wegzudenken. Dementsprechend wichtig ist die Einsetzbarkeit eines Kryptosystems auf ihnen. ECDSA und ECDH sind aufgrund der verwendeten, viel kürzeren Schlüssel prädestiniert für diesen Einsatz. Einen zusätzlicher Vorteil bieten EC Implementierungen auf  $F_{2^m}$ , da diese ohne Kryptokoprozessor realisiert werden können, und so die Kosten pro Karte deutlich gesenkt werden können.<sup>1</sup>

Viele der renommierten Smartcardhersteller bieten bereits fertige Security-Produkte, die elliptische Kurven unterstützen, an. Unter anderem sind dies : Orga in Kooperation mit Cryptovision (<http://www.orga.com>, <http://www.cryptovision.com>), On Track Innovations (OTI, <http://www.oti.co.il>) und Schlumberger (<http://www.slb.com>). Dabei werden auf den Smartcards Kryptokoprozessoren eingesetzt, oder es kommen spezielle Mikroprozessoren zum Einsatz.

Von folgenden Firmen gibt es konkrete Informationen zu ICs (für Smartcards aber auch andere Applikationen) die elliptische Kurven unterstützen:

**Siemens.** In [9] wird der *Pluto-IC* genannt. Dieser ist ein Verschlüsselungs-IC der auf besonders hohe Verschlüsselungsraten (2 Gbit/sec) optimiert ist. Verwendet werden dabei elliptische

---

<sup>1</sup>Die folgende Auflistung von Herstellern für Smartcard ICs bzw. Smartcards erhebt keinerlei Anspruch auf Vollständigkeit. Auch die Reihung der Firmen ist willkürlich.

Kurven über  $F_p$ . Anwendung findet der Pluto-IC bereits im *ELCRODAT 6-2* Verschlüsselungsgerät das zur Sicherung von ISDN Verbindungen dient. Praktisch eingesetzt wird ELCRODAT 6-2 im Informationsverbund Bonn-Berlin (IVBB), welcher das staatliche, deutsche ISDN-Netzwerk ist.

**Infineon.** Infineon bietet unter dem Namen *SLE66xxP* eine Familie von ICs an, die Public Key Verfahren mit Moduloarithmetik, unterstützen. Konkret werden EC Verfahren mit einer Schlüssellänge bis zu 256 Bit realisiert (siehe [9], und <http://www.infineon.de>).

Da sowohl Siemens als auch Infineon, elliptische Kurven auf  $F_p$  verwenden, sind die Implementierungen frei von Patenten <sup>2</sup>.

**Motorola.** Die Produkte von Motorola (*MPC180x*) sind zur Beschleunigung von intensiven Rechenoperationen, die bei Applikationen wie SSL, WTLS oder WAP vorkommen, entwickelt. Unterstützt wird auch die Signaturgenerierung mittels ECDSA mit Schlüssellängen von 55 bis 511 Bits. Motorola ist gemeinsam mit Certicom Inhaber diverser Patente und kann deswegen Kurven auf beliebigen Grundkörpern beschleunigen. Motorola bietet auch Smartcard ICs mit ECC-Unterstützung an.

**Philips.** Der Philips *SmartXA* kann mittels dem Philips 32-bit *FameX* Kryptokoprozessor modulare Arithmetik beschleunigen. Damit können EC Verfahren mit Schlüssellängen bis 239 Bit gerechnet werden.

#### 3.1.2 Security Software Provider

Die Einstellung der Softwarehersteller ist ebenfalls relevant für die Verbreitung von ECC. Daraus kann durchaus das „Vertrauen“ in die Technologie ableitet werden. Die verschiedenen Softwareprodukte lassen sich grob in verschiedene Sparten kategorisieren. Das sind zum einen Krypto API's bzw. Krypto Toolboxen, und zum anderen Applikationen. API's stellen immer kryptografische Grundfunktionen zur Verfügung, und werden in Applikationen eingebunden. Deswegen ist die Verfügbarkeit eines Kryptosystems in diesen APIs besonders wichtig. Bei Applikationen die ECC verwenden kann man prinzipiell unterscheiden ob das Applikation aus dem Security-Bereich sind (also z. B. Implementierungen diverser in Kapitel 1 aufgeführter Protokolle) oder ob man von Standardsoftware (bsp. Office-Paketen, Webbrowser, etc.) spricht.

##### 3.1.2.1 Crypto API's

Im Bereich der Crypto API's die elliptische Kurven unterstützen gibt es bereits sehr viele Produkte. Dazu gehören natürlich die *Security Builder* Libraries der Firma Certicom (<http://www.certicom.com>). Aber auch die Firma RSA hat in ihrer Toolkit *BSAFE CRYPTO-C* (<http://www.rsasecurity.com/products/bsafe>) ECC Implementierungen in C inkludiert. Genauso bietet die Firma Cryptomathic ([http://www.cryptomathic.dk/index\\_net.html](http://www.cryptomathic.dk/index_net.html)) in ihrer *PrimInk Ansi-C* Toolbox ECC Unterstützung an. Ein weiteres in C geschriebenes

---

<sup>2</sup>Weitere Bemerkungen zur Patentrechtlichen Situation finden sich in Anhang A

Toolkit ist von der Firma SECUDE (<http://www.secude.com>) erhältlich. In der C++Bibliothek (cv act library) bilden ECC einen wesentlichen Bestandteil. Eine Java-Implementierung ist in der IAIK-JCE (<http://jcewww.iaik.at>,<http://www.st-labs.at>) zu finden. Auch eine grosse Anzahl von frei verfügbaren Toolboxes bieten ECC Implementierungen an. Der prominentesten Vertreter ist dabei sicher *Crypto++* (<http://www.eskimo.com/~weidai/cryptlib.html>).

#### 3.1.2.2 Security Software

Wiederum ist die Firma Certicom der führende Anbieter von verschiedenen Security-Lösungen. Dazu gehören PKI (Public Key Infrastructure) Lösungen, aber auch VPN (Virtual Private Network) Lösungen. Hewlett-Packard bietet unter dem Name *VirtualVault* (<http://www.hp.com/security/products/virtualvault/>) einen sicheren Web Server an. Dabei kommt ein Verschlüsselungsmodul der Firma Rainbow Technologies zum Einsatz das ECC unterstützt. Baltimore Technologies (<http://www.baltimore.com>) bietet diverse Securityprodukte die elliptische Kurven unterstützen an. Unter dem Namen *UniCert* wird eine Palette von Software und um Zertifizierungsdienste angeboten. Die dabei inkludierte CA unterstützt beispielsweise ECC. Entrust (<http://www.entrust.com>) unterstützt ganz generell ab Version 5.0 ihrer Produktlinie ECDSA, mit der Einschränkung dass CAs noch nicht mit ECDSA Schlüssel ihre Zertifikate ausstellen können. Dieses Feature soll aber laut Entrust in der nächsten Version eingefügt werden. Die Produktpalette der Firma Cryptovision beinhaltet ausschliesslich Tools die auch ECC unterstützen.

#### 3.1.2.3 Standard Software

Die Integration von ECC in Standardsoftware ist noch in den Anfängen, jedoch haben bereits viele der namhaften Firmen, wie z. B. Oracle, Sybase oder Hewlett Packard, Certicom Produkte lizenziert. Eine detaillierte Liste der Lizenznehmer findet sich auf der Certicom Homepage unter dem Abschnitt „Partners & Customers“.

**EMail Produkte.** Qualcomm (<http://www.qualcomm.com>) hat Certicom's SSL und ECC Implementierungen lizenziert und setzt diese Technologie in ihren Emailprodukten und Webbrowsern ein. Das erste Produkt das hierbei auf den Markt kam war die *Eudora Internet Suite[tm]* (*EIS[tm]*) 2.1 Software für die *Palm OS* Plattform. Aus der Dokumentation zur Software ist allerdings nicht ersichtlich ob ECC auch tatsächlich verwendet wird.

**Microsoft.** Microsofts Crypto API lizenziert die Technologie der RSA Company. Dementsprechend ist in naher Zukunft keine Einführung von ECC in Produkte der Firma Microsoft zu erwarten.

### 3.2 Sichtweise der Zertifizierungsdiensteanbieter

#### 3.2.1 Österreich

Die *Generali Office Service und Consulting AG* (<http://www.generali.co.at/security>) verwendet die Technologie von Entrust, die ECC unterstützt. ECC wird im Moment aber nicht verwendet. Es gibt jedoch Überlegungen, ECC in Zukunft, vor allem wegen der besseren Performance, zu verwenden. *A-Sign* verwendet ECC nicht.

*Innosys* wird gemeinsam mit dem TÜV Österreich ein Trustcenter einrichten (Start : erstes Quartal 2001). Es sollen auch Verfahren basierend auf elliptischen Kurven zum Einsatz kommen. Man verwendet dabei ein Produkt der Partnerfirma SECUDE das ECC unterstützt.

#### 3.2.2 Deutschland

Die Deutsche Telekom AG (*T-Telesec*, <http://www.telekom.de/t-telesec>) verwendet momentan nur den RSA Algorithmus. Der Einsatz Elliptischer Kurven wird anvisiert, man sieht momentan aber noch keinen Markt dafür. Die nächste Version des Chipkartensystems TCOS 3.0 wird die Funktionalität implementiert haben.

#### 3.2.3 EU

Die Hauptprodukte von *Globalsign* (<http://www.globalsign.com>) sind TLS (SSL) Zertifikate für Emails und Webserver. Die Firma ist aber auch im Bereich WTLS stark engagiert. Für WTLS kann Globalsign sich vorstellen ECC zu unterstützen.

#### 3.2.4 International

Auf der Certicom Homepage kann man sowohl *Baltimore* als auch *Xcert International* unter den Geschäftspartnern finden. Baltimore verwendet Certicom Technologie in seiner *UniCert Certification Authority Software*. *Xcerts Sentry CA* hat die Technologie ebenfalls inkludiert. *Entrust* (<http://www.entrust.com>) unterstützt momentan Endanwender ECDSA Schlüssel. Die Unterstützung für CA ECDSA Schlüssel ist geplant.

## Kapitel 4

# Zusammenfassung

Die Entwicklung in der Informationstechnologie macht es notwendig, Sicherheitsmechanismen in verschiedener Art und Weise zu inkludieren. Insbesondere ist das im Bereich des E-Commerce bzw. E-Business notwendig. Da der Trend in der Informationstechnologie in Richtung mobile Geräte wie Mobiltelefone und Handheldgeräte geht, müssen die eingesetzten kryptografischen Verfahren, wie Verschlüsselung, Digitale Signatur und Digitale Zertifikate, unter diesen speziellen Bedingungen, also eingeschränkter Speicherplatz, eingeschränkte Prozessorleistung, usw., effizient funktionieren. Kryptosysteme basierend auf elliptischen Kurven, sind besonders gut dafür geeignet. Zahlreiche Standards stehen dabei bereits zur Verfügung. Auch die Akzeptanz der Hard- und Softwarehersteller von ECC ist gegeben. Softwareseitig beherrscht die Firma Certicom sowohl den Markt, als hat sie auch das grösste technologische Know-how. Ihre Technologien werden von sehr vielen anderen Firmen lizenziert oder gekauft. Auf der Seite der Hardwarehersteller ist man ebenfalls zu ECC positiv eingestellt. Es gibt zahlreiche Chiphersteller die schon seit längerer Zeit entsprechende Produkte am Markt haben. Diese Produkte finden auch auf diversen Chipkarten ihre Anwendung. Aus der Sichtweise der Zertifizierungsdiensteanbieter (abgesehen von Baltimore und Xcert) gibt es momentan noch keinen Markt für ECC. Überlegungen ECC in Zukunft einzusetzen gibt es jedoch schon.

Die allgemeine Stimmung zu ECC ist eher positiv. Man erwartet sich vor allem in Zukunft den verstärkten Einsatz dieser Technologie. Sowohl Industrie als auch Wissenschaft, hat Vertrauen in ECC. Dementsprechend kann man sich erwarten, das die Bedeutung von ECC wesentlich zunehmen wird.

# Anhang A

## Technische Details

### A.1 Grundlagen von ECC

Elliptische Kurven kann man durch die sogenannte (kurze) *Weierstrassgleichung* beschreiben. Ist die Charakteristik<sup>1</sup> des Körpers  $K$  ungleich 2 oder 3 dann hat diese die Form  $y^2 = x^3 + ax + b$ , mit  $a, b, \in K$ . Verlangt man zusätzlich noch dass diese Gleichung keine mehrfachen Nullstellen besitzt, also dass  $-(4a^3 + 27b^2)$  ungleich null ist, dann hat man die Definition einer elliptischen Kurve über einem Körper dessen Charakteristik weder 2 noch 3 ist (man stelle sich der Einfachheit halber den Körper der reellen Zahlen, oder den Körper der rationalen Zahlen vor). Die Punkte auf dieser Kurve, bilden gemeinsam mit dem sogenannten Punkt

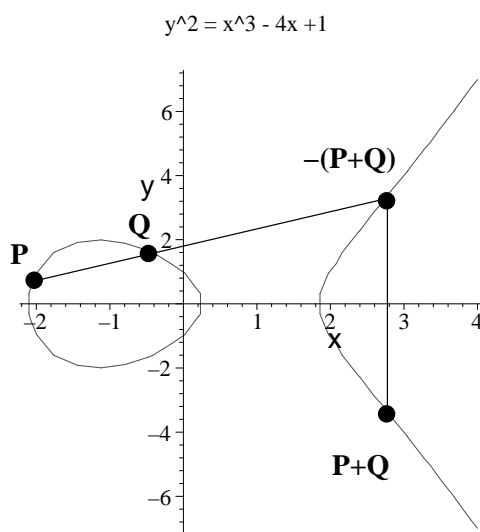


Abbildung A.1: Punktaddition auf einer elliptischen Kurve

im Unendlichen  $\mathcal{O}$ , den man sich unendlich weit entfernt in Richtung der  $y$ -Achse denken

<sup>1</sup>Bezeichnet man die Charakteristik eines Körpers mit  $k$ , dann gilt für alle Elemente  $a \in K : ak = 0$ .

kann, eine abelsche Gruppe. Der Punkt im Unendlichen ist das neutrale Element bezüglich der Addition, d. h. für einen Punkt  $P$  und sein Inverses  $-P$  gilt  $P + (-P) = 0$ .

**Arithmetik.** Die Regeln zum Addieren zweier Punkte lassen sich am besten grafisch erklären. In Abbildung A.1 werden zwei voneinander verschiedenen Kurvenpunkte  $P$  und  $Q$  addiert. Dazu schneidet man ihre Verbindungsgerade mit der elliptischen Kurve. Der daraus resultierende Punkt ist, aus mathematischen Gründen, der Punkt  $-(P + Q)$ . Um zur eigentlichen Summe der beiden Punkte zu kommen muss noch die Inverse berechnet werden. Diese ist, da man sich das neutrale Element in Richtung y-Achse denken muss, genau der an der x-Achse gespiegelte Punkt  $P + Q$ . Koblitz [15] fasst diese Regeln folgendermassen zusammen :

*Die Summe dreier auf einer Geraden liegenden Kurvenpunkten ist null ( $\mathcal{O}$ ).*

In der Kryptografie arbeitet man mit algebraischen Strukturen die nur endlich viele Elemente enthalten. Das macht viele, in den reellen Zahlen harmlose Probleme, schwierig. Elliptische Kurven auf endlichen Körpern werden wiederum durch Weierstrassgleichungen beschrieben. Man hat hier verschiedene, von der Charakteristik des Körpers abhängige, Gleichungen. Für Charakteristik ungleich 2 oder 3 gilt wieder obige Gleichung. Für Charakteristik gleich 2 verwendet man die Gleichung  $y^2 + xy = x^3 + ax^2 + b$ . Die nachfolgenden Tabellen beinhalten eine Aufstellung der Kurvengleichungen, der Punktadditions- und Punktverdoppelungsformel, sowie der Inversen.

	$y^2 = x^3 + ax + b$ , mit $a, b \in \mathbb{F}_p$	$y^2 + xy = x^3 + ax^2 + b$ , mit $a, b \in \mathbb{F}_{2^m}$
Inversion	$-P = (x_1, -y_1)$	$-P = (x_1, x_1 + y_1)$
P+Q	$x_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2$ $y_3 = -y_1 + \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x_1 - x_2)$	$x_3 = \left(\frac{y_2 + y_1}{x_2 + x_1}\right)^2 + \left(\frac{y_2 + y_1}{x_2 + x_1}\right) + x_1 + x_2 + a$ $y_3 = \left(\frac{y_2 + y_1}{x_2 + x_1}\right)(x_1 + x_3) + x_3 + y_1$
2P	$x_3 = \left(\frac{3x_1^2 + a}{2y_1}\right)^2 - 2x_1$ $y_3 = \left(\frac{3x_1^2 + a}{2y_1}\right)(x_1 - x_3) - y_1$	$x_3 = x_1^2 + \frac{b}{x_1^2}$ $y_3 = x_1^2 + \left(x_1 + \frac{y_1}{x_1}\right)x_3 + x_3$

Tabelle A.1: Formelsammlung

Für kryptografische Anwendungen ist es wichtig, dass die verwendete algebraische Struktur groß genug ist. D. h. die Anzahl der Punkte auf einer elliptischen Kurve ist eine wichtige Grösse. Der Satz von Hasse gibt eine Abschätzung für diese Anzahl. Es gilt nämlich, wenn man mit  $\#E(\mathbb{F}_q)$  die Anzahl der Punkte der elliptischen Kurve  $E$  über einem Körper  $\mathbb{F}_q$  bezeichnet, dass,

$$q + 1 - 2\sqrt{q} \leq \#E(\mathbb{F}_q) \leq q + 1 + 2\sqrt{q}$$

ist. In Worten ausgedrückt heisst das, dass die Anzahl der Punkte auf einer elliptischen Kurve über einem Grundkörper in etwa gleich gross ist, wie die Anzahl der Elemente des Grundkörpers selbst.

**Einbettung einer Nachricht.** Die zu verschlüsselnde Nachricht  $m$  ist immer eine ganze Zahl, die man zunächst mit einem Kurvenpunkt assoziieren muss. Dazu wählt man eine nicht zu kleine Zahl  $k$  (die  $q$  nicht teilt), sodaß  $0 \leq m \leq M_k$ , mit  $M_k = \lfloor \frac{q}{k} \rfloor$ . Man setzt jetzt  $x = mk + i$  und versucht (für verschiedene, mögliche  $i$ 's) die Kurvengleichung zu lösen (also ein  $y$  zu finden, sodaß der Punkt  $P = (x, y)$  auf der elliptischen Kurve  $E$  liegt). Die Wahrscheinlichkeit, dass man auf diese Weise keinen Kurvenpunkt findet ist  $2^{-k}$ . Für ein genügend grosses  $k$  ist die Fehlerwahrscheinlichkeit also sehr gering.

**Verschlüsselungsverfahren.** Öffentlich bekannt sind hier die elliptische Kurve  $E$ , der endliche Körper  $F_q$ , und der Basispunkt  $B \in E$ . Jeder Benutzer wählt sich ein zufälliges, geheimes  $a$ , berechnet sich  $aB$  und veröffentlicht  $aB$  als seinen öffentlichen Schlüssel. Um Bob eine verschlüsselte Nachricht  $M$  zu schicken, wählt Alice eine Zufallszahl  $k$  und schickt das Punktepaar  $(kB, M + k(a_B B))$  an Bob (mit  $a_B B$  ist Bobs öffentlicher Schlüssel gemeint). Um die Nachricht zu entschlüsseln, multipliziert Bob den ersten Teil der Nachricht mit seinem Geheimnis  $a_B$  und subtrahiert dies vom zweiten Teil der Nachricht:  $M + k(a_B B) - a_B kB = M$ .

**Ordnung des Basispunktes.** Wie schon erwähnt, ist es wichtig, eine elliptische Kurve mit möglichst vielen Punkten zur Verfügung zu haben. Um alle diese Punkte auch tatsächlich beschreiben zu können bedient man sich des Basispunktes. Die Vielfachen des Basispunktes sind ja auch Punkte auf der elliptischen Kurve. Ist es möglich alle Punkte auf der elliptischen Kurve als Vielfache eines Basispunktes anzuschreiben, dann heisst dieser ein *Generator* der Kurve. Die *Ordnung* eines Punktes ist die Anzahl von verschiedenen Kurvenpunkten die man mit dem Punkt erzeugen kann. Für ECC ist es wichtig, dass die Ordnung des Basispunktes gross ist, und zusätzlich, dass die Ordnung des Basispunktes nicht durch kleine Primfaktoren teilbar ist. Die zweite Forderung garantiert die „Schwierigkeit“ des ECDLP.

## A.2 Vergleich mit RSA, DSA

Die Hauptgesichtspunkte bei einem Vergleich verschiedener Kryptosysteme liegen im Bereich Sicherheit und Effizienz. Die theoretische Sicherheit der heute üblichen Kryptosysteme ist nicht beweisbar, da die Sicherheit der zugrundeliegenden Probleme nicht beweisbar ist. Man kann allerdings die Laufzeiten der Algorithmen die zur Lösung dieser Probleme verwendet werden, betrachten. Bei diesen Algorithmen muss man prinzipiell zwischen den sogenannten „special-purpose“ und den „general-purpose“ Algorithmen unterscheiden. Special-purpose Algorithmen lösen spezielle Instanzen eines Problems. Z. B. im Fall das die Zahl  $p - 1$  kleine Primfaktoren hat, ist das Diskrete Logarithmusproblem in  $\mathbb{F}_p$  leicht lösbar. Diese Spezialfälle sind aber allgemein bekannt, und können so vermieden werden. Daher muss man nur noch die general-purpose Algorithmen betrachten.

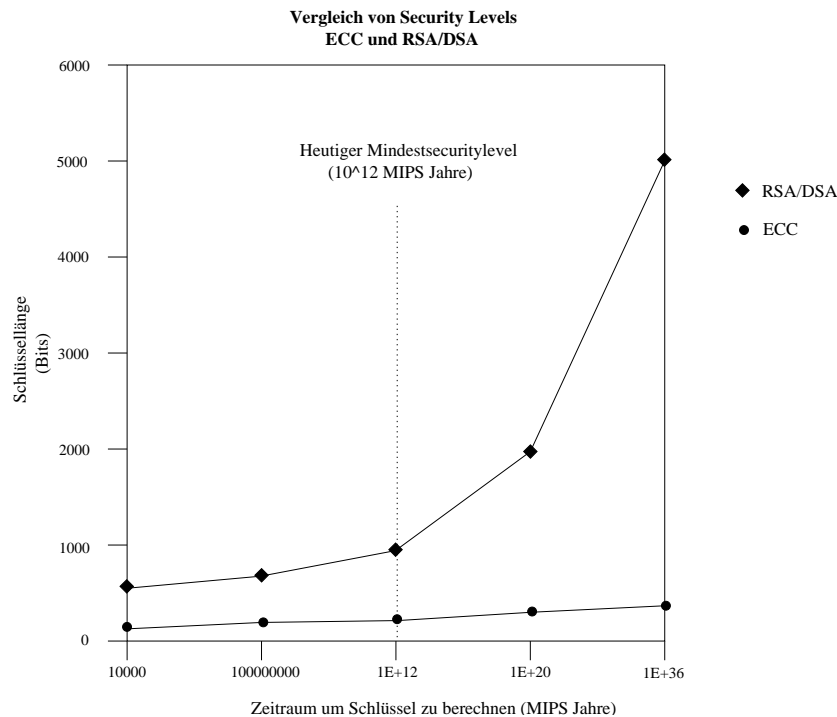


Abbildung A.2: Vergleich der Security Levels

**Sicherheit.** Sowohl für das Faktorisierungsproblem (IFP) als auch für das Diskrete Logarithmusproblem (DLP) gibt es subexponentielle Algorithmen zur Lösung. Der beste bekannte Algorithmus zur Lösung des ECDLP ist allerdings in seiner Laufzeit voll exponentiell. Damit ist das ECDLP im Vergleich zu DLP und IFP schwerer. Abbildung A.2 ([4]) vergleicht die Laufzeiten zum Brechen von ECC und RSA/DSA für verschiedene Modullängen. Aus der Abbildung sieht man, dass um dem heutigen Securitylevel zu entsprechen, ein RSA Modul mit Länge 1024, und ein 160 Bit Modul für ECC notwendig ist. Der Unterschied in der Schlüssellänge, der heute schon enorm ist, wird sich aber noch verstärken, wie auch der Grafik entnommen werden kann.

**Effizienz.** Die Effizienz eines Kryptosystems hängt vom Berechnungsaufwand, von der Schlüssellänge und von der benötigten Bandbreite ab. Für ECC gibt es aufgrund der verschiedenen Grundkörper, und der dadurch induzierten Möglichkeiten der Implementierung der Körperarithmetik, sehr viele verschiedene Implementierungsmöglichkeiten. Besonders Implementierungen in  $F_{2^m}$  können wesentlich effizienter als gewöhnliche Moduloarithmetik arbeiten. Laut [9] arbeitet die ECC Implementierung auf dem für RSA optimierten Infineon IC SLE66CxxP schneller als RSA. Bei der Schlüssellänge ist, wie schon im Paragraph zur Sicherheit bemerkt, ECC wesentlich kürzer. Bei der Bandbreite muss man zwischen Verschlüsselung und Signatur unterscheiden. Bei der Verschlüsselung sind sowohl RSA, als auch ECC in etwa gleich einzustufen, mit der Ausnahme bei kurzen Nachrichten. Bei der Signaturlänge gibt es Unterschiede. Tabelle A.2 und Tabelle A.3([4]) fassen die Fakten zusammen :

	Signaturlänge (Bits)
RSA	1024
DSA	320
ECC	320

Tabelle A.2: Signaturlängenvergleich

	Länge der verschlüsselten Nachricht (Bits)
RSA	1024
ElGamal	2048
ECC	321

Tabelle A.3: Länge einer verschlüsselten 100-Bit Nachricht

**Implementierungsangriffen.** In den letzten Jahren haben sogenannte *Side-Channel* oder Implementierungsangriffen für Aufregung gesorgt. Angriffen solcher Art nutzen Informationen die bei einem Verschlüsselungsvorgang anfallen, wie Stromaufnahme oder Zeitdauer, aus. ECC lässt sich auf vielerlei Art und Weise gegen diese Art von Angriffen schützen. Siehe [5], [8], für weitere Informationen dazu.

### A.3 Ist ECC patentiert?

Kryptosysteme basierend auf elliptischen Kurven sind prinzipiell patentfrei. Es gibt allerdings einige Algorithmen und Techniken zur effizienten Implementierung, auf die Patentansprüche (in Nordamerika) erhoben werden.

**Apple** Computer hat beispielsweise ein Patent auf effiziente Implementierung von ECC über Körpern mit ungerader Charakteristik, bzw.  $\mathbb{F}_p$  wobei  $p$  in etwa einer Potenz von 2 ist. **Certicom** hat ein Patent auf eine effiziente Methode der Multiplikation zweier Körperelemente in Normalbasisdarstellung. Auch **Cylink** hat ein solches Patent angemeldet. Certicom hat zwei schiebende Patentverfahren. Eines beinhaltet die MQV Schlüsseleinigung, das andere bezieht sich auf die sog. Punktkompression (ein Verfahren zur effizienten Darstellung eines EC Punktes).

Die Patentsituation in Europa ist ähnlich. Es gibt einige (schwebende) Patente zu effizienten Implementierungen, aber keine zu ECC selbst.

### A.4 ASN1-Dump des Testzertifikates

```

0 30 551: SEQUENCE {
4 30 476: SEQUENCE {
8 A0 3: [0] {
10 02 1: INTEGER 2

```

#### A.4. ASN1-DUMP DES TESTZERTIFIKATES

---

```

      :      }
13 02   1:    INTEGER 1
16 30  11:    SEQUENCE {
18 06   7:      OBJECT IDENTIFIER '1 2 840 10045 4 1'
27 05   0:      NULL
      :      }
29 30  77:    SEQUENCE {
31 31  11:      SET {
33 30   9:        SEQUENCE {
35 06   3:          OBJECT IDENTIFIER countryName (2 5 4 6)
40 13   2:          PrintableString 'AT'
      :          }
      :        }
44 31  16:    SET {
46 30  14:      SEQUENCE {
48 06   3:        OBJECT IDENTIFIER organizationName (2 5 4 10)
53 13   7:        PrintableString 'TU Graz'
      :        }
      :      }
62 31  13:    SET {
64 30  11:      SEQUENCE {
66 06   3:        OBJECT IDENTIFIER organizationalUnitName (2 5 4 11)
71 13   4:        PrintableString 'IAIK'
      :        }
      :      }
77 31  29:    SET {
79 30  27:      SEQUENCE {
81 06   3:        OBJECT IDENTIFIER commonName (2 5 4 3)
86 13  20:        PrintableString 'ECC Test Certificate'
      :        }
      :      }
      :    }
108 30 30:    SEQUENCE {
110 17 13:      UTCTime '010122131411Z'
125 17 13:      UTCTime '020122131411Z'
      :      }
140 30 77:    SEQUENCE {
142 31 11:      SET {
144 30   9:        SEQUENCE {
146 06   3:          OBJECT IDENTIFIER countryName (2 5 4 6)
151 13   2:          PrintableString 'AT'
      :          }
      :        }
155 31 16:    SET {
157 30 14:      SEQUENCE {
159 06   3:        OBJECT IDENTIFIER organizationName (2 5 4 10)
164 13   7:        PrintableString 'TU Graz'

```

#### A.4. ASN1-DUMP DES TESTZERTIFIKATES

---

```

      :      }
      :      }
173 31 13:    SET {
175 30 11:    SEQUENCE {
177 06  3:    OBJECT IDENTIFIER organizationalUnitName (2 5 4 11)
182 13  4:    PrintableString 'IAIK'
      :      }
      :      }
188 31 29:    SET {
190 30 27:    SEQUENCE {
192 06  3:    OBJECT IDENTIFIER commonName (2 5 4 3)
197 13 20:    PrintableString 'ECC Test Certificate'
      :      }
      :      }
      :      }
219 30 245:  SEQUENCE {
222 30 188:  SEQUENCE {
225 06  7:    OBJECT IDENTIFIER ecPublicKey (1 2 840 10045 2 1)
234 30 176:  SEQUENCE {
237 02  1:    INTEGER 1
240 30 36:    SEQUENCE {
242 06  7:    OBJECT IDENTIFIER prime-field (1 2 840 10045 1 1)
251 02 25:    INTEGER
      :          00 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
      :          FE FF FF FF FF FF FF FF FF FF
      :          }
278 30 52:    SEQUENCE {
280 04 24:    OCTET STRING
      :          FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FE
      :          FF FF FF FF FF FF FF FC
306 04 24:    OCTET STRING
      :          64 21 05 19 E5 9C 80 E7 0F A7 E9 AB 72 24 30 49
      :          FE B8 DE EC C1 46 B9 B1
      :          }
332 04 49:    OCTET STRING
      :          04 18 8D A8 0E B0 30 90 F6 7C BF 20 EB 43 A1 88
      :          00 F4 FF 0A FD 82 FF 10 12 07 19 2B 95 FF C8 DA
      :          78 63 10 11 ED 6B 24 CD D5 73 F9 77 A1 1E 79 48
      :          11
383 02 25:    INTEGER
      :          00 FF FF FF FF FF FF FF FF FF FF FF FF 99 DE F8
      :          36 14 6B C9 B1 B4 D2 28 31
410 02  1:    INTEGER 1
      :      }
      :      }
413 03 52:    BIT STRING 0 unused bits
      :          04 31 04 FD CE E4 F8 9C EE B5 D0 8E EA D5 79 B4

```

#### A.4. ASN1-DUMP DES TESTZERTIFIKATES

---

```
      :      C2 68 C9 EF B0 2C 14 FD 19 2C 6C 43 52 CA AF 41
      :      72 EF F8 21 E5 8F 02 46 4F 0F AF 9B AF 0F 4D B3
      :      0B F7 87
      :      }
467 A3 15:    [3] {
469 30 13:    SEQUENCE {
471 30 11:    SEQUENCE {
473 06  3:    OBJECT IDENTIFIER keyUsage (2 5 29 15)
478 04  4:    OCTET STRING
      :          03 02 01 86
      :          }
      :        }
      :      }
      :    }
484 30 11:    SEQUENCE {
486 06  7:    OBJECT IDENTIFIER '1 2 840 10045 4 1'
495 05  0:    NULL
      :    }
497 03 56:    BIT STRING 0 unused bits
      :      30 35 02 18 23 F7 0B D5 1D 94 1B 30 21 7B 19 01
      :      9F 08 D2 CD BE 70 BE 40 3B 08 09 02 02 19 00 E5
      :      0F 63 89 E8 B8 56 75 DD 2E D7 38 38 D0 FF A6 77
      :      9E 03 F0 D2 85 C1 C8
      :    }
```

0 warnings, 0 errors.

## Anhang B

# ECC-Brainpool

Der ECC-Brainpool ist eine im Frühjahr 2000 gegründete Arbeitsgruppe, in der sich Firmen Institutionen engagieren, die im Bereich ECC tätig sind. Die Arbeitsgruppe wurde durch die Initiative der Firma *cv cryptovision* ins Leben gerufen und umfaßt mittlerweile rund 20 Mitglieder. Aktive Mitglieder aus der Wirtschaft sind :

- Bundesamt für Sicherheit in der Informationstechnik (BSI),
- atsec, cryptovision, Flexsecure, Gemplus, Giesecke & Devrient, Rohde & Schwarz, Secunet AG, SRC, Siemens AG, T-Systems, und seit kurzem auch Orga,
- Infineon Technologies und Philips Semiconductors.

Aus dem universitären Bereich engagieren sich :

- Institut für Experimentelle Mathematik der Universität Essen (IEM), die Technische Universität Darmstadt, die Universität Bonn, die Universität Kassel sowie die Universität Siegen,
- seit kurzem auch EUROBITS, und das IAIK (in seiner Funktion als Mitglied des A-SIT).

Prinzipiell steht der Brainpool allen weiteren Firmen und Institutionen offen, die sich mit der Kryptographie auf Basis elliptischer Kurven beschäftigen. In der auf der Website des ECC-Brainpool <http://www.ecc-brainpool.org> erhältlichen Presseinformation wird ein weiteres Argument für den Einsatz elliptischer Kurven genannt : „Der ECC-Brainpool betonte, dass in dem immer bedeutender werdenden Bereich der Verschlüsselungstechnologien eine Beschränkung auf nur ein Verfahren sicherheits-strategisch bedenklich sei und ein schwer kalkulierbares volkswirtschaftliches Risiko darstelle. Daher sei es sinnvoll und notwendig, neben dem etablierten RSA-Verfahren moderne Verschlüsselungsverfahren auf Basis elliptischer Kurven (Elliptic Curve Cryptography, ECC) einzusetzen.“ Die Aktivitäten des Brainpools bestehen aus dem Bereitstellen aktueller Informationen zum Thema ECC auf seiner Website, dem Organisieren regelmässiger Treffen der Mitglieder zum Informationsaustausch sowie aus dem Ausrichten von Workshops. Der erste solche Workshop fand am 12.12.2001 zum Thema „Side-Channel-Attacks auf Kryptoalgorithmen“ statt.

# Literaturverzeichnis

- [1] ANSI X9.62, *Public Key Cryptography for the Financial Services Industry : The Elliptic Curve Digital Signatur Algorithm (ECDSA)*, 1999
- [2] ANSI X9.63, *Public Key Cryptography for the Financial Services Industry : Elliptic Curve Key Agreement and Key Transport Protocols*, working draft, 1999
- [3] L. Bassham, R. Housley, W. Polk, *Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and CRL Profile*, IETF working draft, 2000
- [4] Certicom Whitepaper, *Current Public Key Cryptosystems*, 2000
- [5] J.-S. Coron, *Resistance against differential power analysis for elliptic curve cryptosystems*, Workshop on Cryptographic Hardware and Embedded Systems, Lecture Notes in Computer Science, vol. 1717, Springer, 1999, pp. 292-301
- [6] W. Diffie, M. E. Hellman. *New directions in cryptography*. IEEE Trans. Inform. Theory, IT-22,p:644-654, November 1976.
- [7] G. H. Hardy, E. M. Wright. *An Introduction to the Theory of Numbers*. 5. Aufl. Oxford University Press, Oxford, 1979.
- [8] M. A. Hasan, *Power Analysis Attacks and Algorithmic Approaches to Their Countermeasures for Koblitz Cryptosystems*, to appear in Proceedings of Workshop on Cryptographic Hardware and Embedded Systems 2000
- [9] E. Hess, *Aspects of Public Key Cryptosystems in Practice*, The 4th Workshop on Elliptic Curve Cryptography (ECC 2000), <http://www.exp-math.uni-essen.de/~galbra/eccslides/eccslides.html>
- [10] IEEE P1363, *Standard Specifications for Public-Key Cryptography*, Draft, 1999.<http://grouper.ieee.org/groups/1363/index.html>
- [11] ISO/IEC 9798-3, *Information Technology-Security Techniques-Entity Authentication Mechanisms-Part 3: Entity authentication Using a Public-Key Algorithm*, first edition, 1993
- [12] ISO/IEC 11770-3, *Information Technology-Security Techniques-Key Management-Part 3: Mechanisms Using Asymmetric Techniques*, 1999
- [13] K. Ireland, R. Rosen. *A Classical Introduction to Modern Number Theory*. 2. Aufl., Bd. 84 aus Graduate Texts in Mathematics. Springer-Verlag, Berlin, 1992.

- [14] N. Koblitz. *A Course in Number Theory and Cryptography*. 2. Aufl., Bd. 114 aus Graduate Texts in Mathematics. Springer-Verlag, Berlin, 1994.
- [15] N. Koblitz. *Algebraic Aspects of Cryptography*. 2. Aufl., Bd. 3 aus Algorithms and Computation in Mathematics. Springer-Verlag, 1999.
- [16] P. Lambert, *Elliptic Curve S/MIME*, Internet Draft, <http://www.ietf.org>
- [17] A. Menezes, P. van Oorschot, S. Vanstone. *Handbook of Applied Cryptography*. CRC-Press, Boca Raton, 1997.
- [18] RFC 2246, *The TLS Protocol Version 1.0*, T. Dierks, C. Allen