# Auswahlkriterien für elliptische Kurven in der Industrie

Martin Wittiger · Daimler AG · 30. Juni 2020

## Zusammenfassung

Klassische Public-Key-Kryptografie benötigt immer längere Schlüssel, um sicher zu sein. Diese verursachen einen deutlich höheren Ressourcenbedarf, sodass verbreitet Softwareprojekte angestoßen werden, die klassische Verfahren durch Elliptische-Kurven-Kryptografie ersetzen, welche mit kürzeren Schlüsseln auskommt. Reengineering- und Maintenance-Projekte müssen aus zahlreichen standardisierten Kurven eine geeignete auswählen. Bernstein und Lange schlagen Kriterien vor, um diese Auswahl zu treffen. Dieser Beitrag bewertet die Relevanz dieser Kriterien aus industrieller Sicht und empfiehlt konkrete Kurven.

## 1 Einleitung

Public-Key-Kryptografie (auch asymmetrische Kryptografie genannt) bildet das Rückgrat wichtiger Securitysysteme. Dazu gehören Public-Key-Infrastrukturen, die signierte Zertifikate ausstellen, die Identitäten an Schlüssel binden. Transport Layer Security (TLS) und Protokolle die darauf aufbauen – wie das im World Wide Web verbreitete HTTPS – nutzen solche Zertifikate.

Zwei konventionelle Verfahren sind verbreitet:

- RSA, entwickelt von Ron Rivest, Adi Shamir und Leonard Adleman, basierend auf der Faktorisierung großer Zahlen
- ElGamal/DSS DSA, entwickelt von Taher ElGamal, maßgeblich erforscht von Claus-Peter Schnorr, basierend auf dem diskreten Logarithmus

Beide Verfahren werden heute mit Schlüssellängen ab 2048 Bit verwendet. Schlüssel der Länge 1024 Bit sind unsicher. Bald werden auch Schlüssellängen von 2048 Bit als unsicher gelten[1]. Längere Schlüssellängen, etwa 4096 Bit, benötigen erheblich mehr Rechenleistung. Das beeinträchtigt Loadbalancer mit TLS-Offloading und Webserver.

## 2 Reengineering

Die Automobilindustrie setzt in eingebetteten Systemen aus Fahrzeugen vermehrt Signaturverfahren ein: Fahrzeuge authentifizieren sich bei Plug & Charge gegenüber Ladesäulen, Car2X-Nachrichten werden verifiziert und Softwarestände, -konfiguration sowie Onboard-Kommunikation wird mit Zertifikaten geschützt. In der Praxis sind für eingebettete Systeme oft bereits Schlüssel der Länge 2048 Bit zu groß.

Eine Lösung verspricht Elliptische-Kurven-Kryptografie. Dort gelten Schlüssellängen von 256 Bit derzeit als sicher. In absehbarer Zukunft reichen Schlüssellängen von 448 Bit aus [1].

Neben Softwareentwicklungsprojekten, die in neu entwickelter Software Public-Key-Kryptografie einsetzen sind verbreitet Reengineering- und Maintenance-Projekte nötig, die die konventionellen Algorithmen RSA und DSA ersetzen. Für Signaturen bieten sich zwei neue Verfahren an, die beide auf dem diskreten Logarithmus auf elliptischen Kurven basieren: der Elliptic Curve Digital Signature Algorithm (ECDSA) und der Edwards-curve Digital Signature Algorithm (EdDSA). Die Projekte entscheiden sich teils schnell Elliptische-Kurven-Kryptografie einzusetzen:

- Sowohl Schlüssel als auch Signaturen sind kürzer als die konventioneller Algorithmen (Anpassung von Felderlängen ist nötig, Kürzen is jedoch einfacher als Verlängern)
- Keine zusätzlichen kryptographischen Primitiven werden benötigt. Teils ist bei EdDSA ein zuvor benötigter Zufallszahlengenerator nun überflüssig.

## 3 Die Bernstein-Lange-Kriterien

Es bleibt die Frage nach der richtigen Kurve: Projekte müssen sich für eine von zahlreichen vorgeschlagenen und standardisierten elliptischen Kurven entscheiden. Mit der Auswahl einer Kurve geht die Entscheidung für ein Signaturverfahren einher: Alle Kurven eignen sich jeweils für eines der beiden Verfahren. Bernstein und Lange bieten mit SafeCurves[2] eine wissenschaftliche Entscheidungsgrundlage. Sie bewerten 20 Kurven anhand von 11 Kriterien. Als Hilfestellung für Software-Projekte bewertet dieser Beitrag die Relevanz der Kriterien für den industriellen Einsatz. Er erkennt zwei der Kriterien als besonders wichtig und empfiehlt den Einsatz der Kurven Curve25519 und Ed448-Goldilocks.

Die unter dem Überbegriff ‚Curve Parameters' gelisteten Kriterien **Field**, **Equation** und **Base** stellen einfache Prüfbedingungen an die Kurvengleichung, den endlichen Körper, über dem diese ausgewertet wird, und den gewählten Basispunkt. Die Ersteller der Standards erledigen diese sinnvolle und notwendige Prüfung lange vor dem Einsatz der Kurven in realen Systemen. Alle untersuchten Kurven erfüllen daher diese Kriterien. Bei der Evaluation für den industriellen Einsatz sind sie irrelevant.

Das Kriterium **Rho** prüft, ob die Kurve sicher gegenüber der einfachen Rho-Attacke ist. Dazu muss die Ordnung des Basispunktes groß sein. Alle Kurven haben Basispunkte mit nahezu maximal möglicher Ordnung und werden nicht beanstandet. Das Kriterium

| Name | Kurzbeschreibung | Rel. |
|---|---|---|
| Field | Größe des Körpers ist prim | — |
| Equation | Kurvengleichung sinnvoll | — |
| Base | Geeigneter Basispunkt gewählt | — |
| Rho | Rho-Attacke unpraktikabel | * |
| Transfer | Resistenz gegen MOV-Attacke | * |
| Disc | Betrag von D groß | * |
| Rigid | Versteckte Vorteile unmöglich | ** |
| Ladder | Schnell, simple, seitenkanalfrei | *** |
| Twist | Skalare Multiplikation sicher | ** |
| Complete | Keine Sonderfälle in Algebra | *** |
| Ind | Ununterscheidbares Encoding | * |

Tabelle 1: Industrielle Relevanz der Kriterien

hat geringe Relevanz für die industrielle Anwendung.

Durch multiplikative oder additive Transfers werden Attacken auf Kurven ermöglicht (MOV- und Smart-ASS-Attacke). Bernstein und Lange gestehen ein, das ihr **Transfers**-Kriterium ‚overkill' ist. Dennoch erfüllen es fast alle Kurven. Die industrielle Relevanz ist gering.

Zu dem Kriterium **Disc** schreiben die Autoren, dass es „keine Anhaltspunkte für ernste Probleme" bei Nichterfüllung gibt. In dem Kontext dieses Beitrages hat es geringe Relevanz.

Seit bekannt wurde, dass die NSA Kryptografiestandards so manipuliert hat, dass sie daraus Vorteile erlangen kann[1], fordern Kryptologen regelmäßig in Standards keinen oder wenig Raum für Beliebigkeit zu erlauben. Konstanten in Kryptografiestandards sollen Nothing-up-my-sleeve-Zahlen sein. Sie sollen so gewählt sein, dass sie erkennbar frei von schändlichen Einflüssen sind. Das Kriterium **Rigid** stellt dies sicher.

Zwar darf die Gefahr der Manipulation durch Geheimdienste nicht als reine Verschwörungstheorie abgetan werden, jedoch ist deren Abwehr normalerweise nicht Aufgabe eines Security-Architekten in der Industrie. Das Kriterium hat mittlere Relevanz.

Die Laufzeit einer kryptografischen Operation, etwa der Signaturerstellung, darf nicht vom privaten Schlüssel abhängen, sonst sind meist sogenannte Seitenkanal-Angriffe möglich[2]. Aus dieser Anforderung ergibt sich, dass die Basisoperation der skalaren Multiplikation eine konstante Laufzeit haben muss. Das Kriterium **Ladder** besagt, dass eine laufzeitkonstante Implementierung weder wesentlich langsamer noch wesentlich komplizierter sein darf, als andere Implementierungen. Die Regel verhindert Security-Kompromisse.

Das Kriterium ist von hoher Relevanz im industriellen Einsatz. Die seitenkanalfreie Implementierung ist schwierig. Wenn sie durch die Wahl der Kurve vereinfacht wird und dadurch die Wahrscheinlichkeit von Fehlern sinkt, gewinnt die Gesamtlösung erheblich an Sicherheit. Eingebettete Systemen in Fahrzeugen setzen ungewöhnliche Hardware ein. Dadurch wird die Implementierung schwieriger und jede Möglichkeit zur Vereinfachung noch wertvoller.

Mit dem Kriterium **Twist** wird die Anfälligkeit der skalaren Multiplikation auf verschiedene Attacken geprüft. Von den untersuchten Kurven haben 15 keine ernsthaften Probleme damit. Selbst Kurven, die es verletzen, verlieren ihre Sicherheit nicht gänzlich. Das Kriterium ist in der industrielle Anwendung wichtig.

Bernstein und Lange fordern mit dem Kriterium **Completeness** eine vollständige Punktadditionsformel *ohne Ausnahmen* für die Kurven ein. Bei der Implementierung sind Ausnahmen schwer zu behandeln. Zufällige Tests decken Fehler, die dabei gemacht werden, meist nicht auf und das Erzeugen passender Testvektoren ist eine schwierige algebraische Aufgabe. Durch Implementierungsfehler verlieren Verfahren ihre Sicherheit teils vollständig. Gerade auch im Reengineering ist das Kriterium von hoher Relevanz.

Das Kriterium **Ind** fordert einfache Algorithmen für das Codieren von Kurvenpunkten ein, die von Zufallszahlen ununterscheidbare Ergebnisse liefern. Solche Algorithmen ermöglichen ‚censorship circumvention' (die Umgehung von Internetzensur mittels technischer Maßnahmen) und Steganografie. In der industriellen Anwendung hat beides geringe Relevanz.

## 4 Empfehlung und Fazit

Von den durch SafeCurves bewerteten Kurven erfüllen 11 die zwei relevantesten Kriterien Ladder und Completeness. Alle diese Kurven erfüllen auch die weniger relevanten Kriterien. Nur zwei werden verbreitet unterstützt: Curve25519 und Ed448-Goldilocks. Dieser Beitrag empfiehlt eine dieser beiden Kurven zu verwenden. Damit geht die Empfehlung einher, beim Reengineering EdDSA und nicht ECDSA einzusetzen.

Curve25519 bietet ein ausreichendes Sicherheitsniveau von 128 Bit. Anwendungen, deren Anforderungen an den Ressourcenverbrauch es erlauben, können die deutlich größere Kurve Ed448-Goldilocks mit einem Sicherheitsniveau von 224 Bit einsetzen.

Leider lässt sich eine hohe Marktverfügbarkeit einiger Kurven beobachten, von denen SafeCurves explizit abrät. Die Vorteile der in diesem Beitrag empfohlenen Kurven sind nicht rein akademisch, sondern gerade in der Praxis relevant. Es liegt im Interesse von Industriekonzernen, die Verwendung dieser Kurven einzufordern und zu unterstützen.

[1] Siehe dazu den Artikel von Zetter[3] und den verlinkten Vortrag von Shumow und Ferguson.

[2] Gegebenenfalls muss sogar die elektromagnetische Abstrahlung und der Energieverbrauch vom privaten Schlüssel unabhängig sein.

## Literatur


[1] Kryptographische Verfahren: Empfehlungen und Schlüssellängen (TR-02102-1). Technische Richtlinie, Version 2019-01, Bundesamt für Sicherheit in der Informationstechnik.

[2] Daniel J. Bernstein und Tanja Lange. SafeCurves: choosing safe curves for elliptic-curve cryptography. `https://safecurves.cr.yp.to`, abgerufen am 12. Februar 2020.

[3] Kim Zetter. How a crypto backdoor pitted the tech world against the NSA. `https://www.wired.com/2013/09/nsa-backdoor`, abgerufen am 12. Februar 2020.