Optimal Asymetric Encryption Padding in RSA

Luc Spachmann

Friedrich-Schiller-Universität Jena

17.12.2021

Probleme von RSA

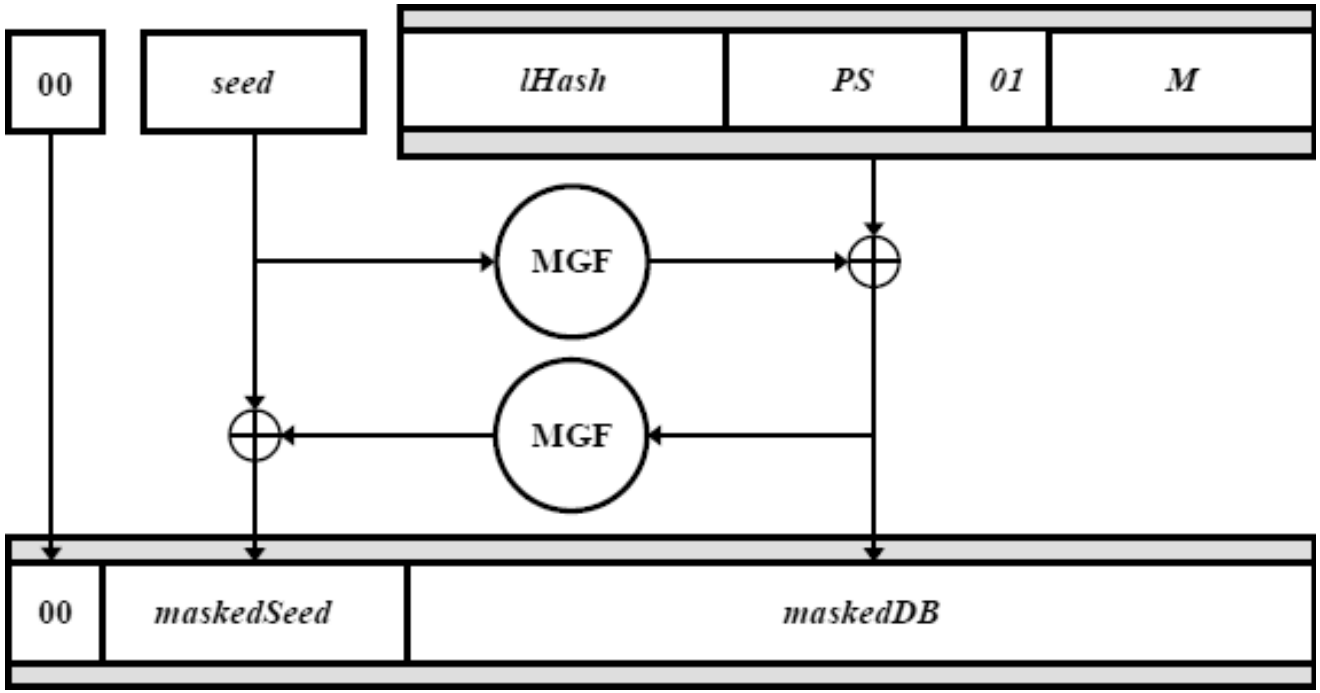
- RSA ist ohne Kenntnis von privatem Schlüssel deterministisch
- Einfache Nachrichten können erraten und überprüft werden
- Padding Protokolle verhindern dies
- Hier: Optimal Asymetric Encryption Padding (OAES)
- Benötigen dafür Hash Funktionen

Mask Generating Function MGF1

Require: ByteArray *seed*
Require: Output Länge *l*
Require: Hashfunktion *hash*

1: Sei T leeres ByteArray
2: $Counter = 0$
3: **repeat**
4: $\quad Counter+ = 1$
5: $\quad$ Sei C Counter als 4 Byte Array aufgefasst
6: $\quad T = T||hash(seed||C)$
7: **until** $|T| > l$
8: **return** Erste l Bytes von T

OAEP Transformation



OAEP Tranformation

- Benötigt Hashfunktion, MGF, RSA-Modul n und Nachricht m als Input
- Optionaler Parameter l, hier ein leeres ByteArray
- Nachricht m darf nur beschränkte Größe haben
- $len(m) \leq len(n) - 2 \cdot len(hash) - 2$
- Angaben in Byte
- $lHash = hash(l)$
- $PS = 00..0$, sodass $len(n) = len(m) + len(PS) - 2 \cdot len(hash) - 2$
- $seed$ zufälliges ByteArray der Länge $len(h)$
- Resultierende Ausgabe wird dann mit RSA verschlüsselt

Aufgaben

- Implementiert die OAEP Transformation in Ver- und Entschlüsselung
- Implementiert MGF1
- Benutzt dafür vordefinierte Hashfunktionen (Bspw. SHA-1)
- Python: Hashlib, Java/Kotlin: java.security.MessageDigest
- C#: System.Security.Cryptography.HashAlgorithm, etc