AES im Kurzüberblick

Da der Data Encryption Standard aufgrund seiner geringen Schlüsselgröße und der stark angestiegenen Rechenleistungen als zu schwach eingestuft worden war, musste ein neuer Verschlüsselungsstandard her. Diesbezüglich hat die US Regierung im Jahre 1997 ein Wettbewerb abgehalten zu dem sich viele Arbeitsgruppen angemeldet hatten und Ihre Lösungen präsentierten.

Die Algorithmen wurden auf folgende Kriterien hin untersucht:

* Sicherheit
* Implementierbarkeit
* Ressourcenverbrauch / Speicherbedarf
* Geschwindigkeit

Als Gewinner wurde eine leicht veränderte Version des Rijndael, nach den Entwicklern Joan Daemen und Vincent Rijmen benannt, ausgewählt.

Der Algorithmus sei mit den heute zur Verfügung stehenden Mitteln nicht zu knacken und erlaube dank seines einfachen Aufbaus eine sehr schnelle Implementierung in unterschiedlichen Programmiersprachen. Dank der Einfachheit sei es auch sehr effizient und biete eine gute bis sehr gute Performance auf verschiedenen Plattformen, wie einem 32Bit Prozessor, 8Bit Mikrocontroller und auch eine Implementierung in Hardware. Weiterhin benötigt der Algorithmus sehr wenig Ressourcen und ist damit ideal für die Anwendung in Umgebungen mit beschränkten Ressourcen, wie z.B. Chipkarten.

Der Rijndael ist ein Blockchiffre, d.h. der Algorithmus arbeitet mit einer fest vorgegebenen Bitgruppe arbeitet, einem sogenannten Block. Die gewöhnliche Größe eines Eingabeblocks beträgt 128Bit welcher in einen dazugehörigen Ausgabeblock gleicher Größe verschlüsselt wird. Dazu wird ein Schlüssel gebraucht. Bei Rijndael ist es theoretisch möglich einen Schlüssel beliebiger Größe zu nehmen, der AES jedoch nimmt nur die Schlüsselgrößen 128Bit, 192Bit und 256Bit an.

Rijndael verwendet zur Verschlüsselung eine umkehrbare 8x8Bit Matrix, genannt Substitution Box (kurz S-Box) und Berechnungen über einem Galoisekörper der Form GF[28]. Diese Berechnungen erinnern an Fehlererkennende und –korrigierende Codes.

Um einen Block zu verschlüsseln, werden verschiedene Operationen nacheinander auf den Block angewendet, nämlich Substitutionen (S-Box) und Permutationen (P-Box). AES beruht damit auf ein Substitutions- und Permutations- Netzwerk (SPN). Der AES führt die genannten Operationen iterativ auf je einen Block aus. Ein Block muss sequenziell verarbeitet werden, aber es können mehrere Blöcke gleichzeitig verarbeitet werden, da diese unabhängig voneinander sind. Dieser Effekt ist ideal für die Nutzung des Algorithmuses auf einer GPU, da diese sehr viele Operationen Parallel verarbeiten kann. Es können damit viele Blöcke Parallel verschlüsselt werden und dies führt zu einem theoretisch enormen Speedup.

Wie schon oben erwähnt ist eine Rijndael ein Blockchiffre, wobei die Blocklänge und die Schlüssellänge unabhängig voneinander variieren können. Es können die Werte 128 (min), 160, 192, 224 oder 256 Bits gewählt werden. Bei AES jedoch sind die Blockgrößen auf 128 Bit beschränkt, die Schlüsselgröße kann aber die Werte 128, 192 oder 256 Bit annehmen.

Jeder Block wird im Rijndael in eine zweidimensionale Tabelle mit vier Zeilen geschrieben, wobei jede Zelle die Größe ein Byte hat. Die Anzahl der Spalten ist bei AES konstant und beträgt 4 (128Bit)

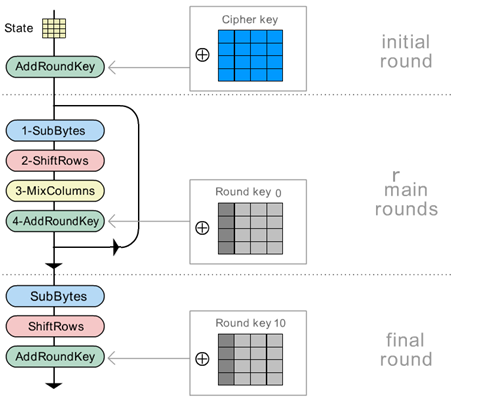


Diese Blöcke werden nacheinander mathematischen Transformationen unterzogen. Das Spezielle an AES / Rijndael ist, dass der Schlüssel (Key) nicht komplett für jeden Block einzeln verwendet wird sondern nur Teile des erweiterten Originalschlüssels. Die Anzahl der Runden (r) hängt direkt von der Schlüssellänge (k) und Blockgröße (b) ab. Da der AES eine einheitliche Blockgröße von 128 Bits hat ist die Anzahl der Runden nur von der Schlüssellänge abhängig.

Die Verschlüsselung kann vereinfacht folgendermaßen dargestellt werden (die einzelnen Schritte werden im folgenden näher beschrieben und erläutert):

* Schlüsselexpansion
* Vorrunde
  + KeyAddition (Rundenschlüssel[0])
* Verschlüsselungsrunden (wiederhole solange Runde < R)
  + Substitution()
  + ShiftRow()
  + MixColumn()
  + KeyAddition(Rundenschlüssel[runde])
* Schlussrunde
  + Substitution()
  + ShiftRow()
  + KeyAddition(Rundenschlüssel[R])

(Die Schlussrunde zählt auch als Runde, also R = Anzahl Verschlüsselungsrunden + 1 Schlussrunde)



Bei der Entschlüsselung von Daten wird genau rückwärts vorgegangen. Die Daten werden zunächst wieder in zweidimensionale Tabellen gelesen und die Rundenschlüssel generiert. Allerdings wird nun mit der Schlussrunde angefangen und alle Funktionen in jeder Runde in der umgekehrten Reihenfolge aufgerufen. Durch die vielen XOR-Verknüpfungen unterscheiden sich die meisten Funktionen zum Entschlüsseln nicht von denen zum Verschlüsseln. Jedoch muss eine andere S-Box genutzt werden (die sich aus der originalen S-Box berechnen lässt) und die Zeilenverschiebungen erfolgen in die andere Richtung.