PS E-Commerce and Mobile Business - Kryptographie und Mobile Kommunikation

19. Februar 2002

Blockchiffremodi

Gruppenmitglieder

Kathrin Meisl, Karl Frank, Christian Koidl

Inhaltsverzeichnis

Kapitel 1

Einleitung

Bei einer Blockchiffre wird der Klartext in Blöcke der Länge n geteilt, die durch einen Algorithmus chiffriert werden. Die entstandenen Chiffreblöcke haben ebenfalls die Länge n. Häufig wird n auf 64 gesetzt, da so acht ASCII Zeichen einen Block ausfüllen.

Für den Data Encryption Standard (DES) gibt es vier spezifizierte Standardbetriebsmodi:

- Electronic Code Book (ECB) = Elektronisches Codebuch
- Cipher Block Chaining (CBC) - Blockchiffre mit Blockverkettung
- Cipher Feedback (CFB) - Schlüsseltextrückführung
- Output Feedback (OFB) - Ergebnisrückführung

Es existieren noch wesentlich mehr Modi, wovon ebenfalls viele genormt sind, doch die sind in der Regel nur Abwandlungen dieser vier. Im folgenden werden die Standardbetriebsmodi beschrieben.

Die Standardbetriebsarten unterscheiden sich sehr stark in ihren Eigenschaften und in ihren Anwendungsbereichen, z.B. sind ECB und CBC blockorientiert, dagegen aber CFB und OFB zeichenorientiert. Ursprünglich für DES spezifiziert sind diese Modi mittlerweile für den Einsatz aller Blockchiffreverfahren von zentraler Bedeutung.

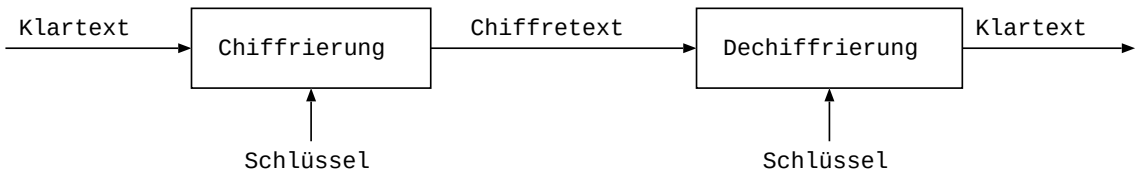
Anforderungen an das Chiffrierverfahren:
Die Hauptanforderung ist die Sicherheit des Verfahrens, dazu gehört, dass Muster im Klartext verborgen bleiben und damit auch Häufigkeitsanalysen schwierig sind. Manipulationen des Chiffretextes zur Veränderung des Klartextes sollten erkennbar sein. Es gibt noch weitere wichtige Punkte, die zu beachten sind z.B. die Effizienz, wie schnell ist der Algorithmus und sind Vorausberechnungen möglich. Auch die Ausfallsicherheit spielt dabei eine Rolle. Wie stark können sich Fehler im Chiffretext ausdehnen (Fehlerfortpflanzung), wenn Übertragungsfehler auftreten.

Kapitel 2

Electronic Code Book (ECB)

ECB ist der einfachste Chiffremodus. Es wird für jeden Klartextblock ein Chriffreblock gesetzt, unabhängig von den anderen. Man kann sich dies wie eine Tabelle vorstellen, auf der einen Seite stehen alle möglichen Klartextblöcke, auf der anderen alle möglichen Chiffreblöcke (wobei von diesen keiner doppelt vorkommen darf, da sonst keine eindeutige Zuordnung mehr möglich ist). Diese Tabelle wird Codebook genannt, daher der Name Electronic Codebook Modus (ECB). In den meisten Fällen wird der Modus nicht über eine Tabelle realisiert, da diese zu speicherintensiv wäre, sondern durch irgendeine Funktion. Durch diese Einfachheit ergeben sich einige Vorteile wie: Die Chiffrierung muss nicht linear erfolgen, d.h. es muss nicht von vorn angefangen werden und dann wird jeweils der nächste Block verschlüsselt und wieder in der Datenbank abgelegt.
Durch diese Eigenschaft ist die Verarbeitung auch parallelisierbar, d.h. es können mehrere Verschlüsselungsprozessoren gleichzeitig verschlüsseln, dasselbe gilt auch für die Entschlüsselung. Dadurch ergeben sich auch Nachteile. Da für jeden Klartextblock immer der gleiche Chiffreblock erzeugt wird, ist es möglich, dass ein Kryptoanalytiker ein Codebuch zusammenstellen kann, wenn er nur genügend Klartext und Chiffretext besitzt. Auch ohne Klartext ist es möglich, durch den Chiffretext Rückschlüsse auf die Nachricht zu ziehen, da der ECB-Modus sehr anfällig für Häufigkeitsanalysen ist. Die Gefahr des unerlaubten Entschlüsselns ist am Anfang und am Ende der Nachricht am größten, da in diesen Bereichen Informationen wie Absender, Empfänger, Datum usw. oder regelmäßige Strukturen und Schlüsselworte, die oft im Text auftauchen, enthalten sind.

Hat es der Kryptoanalytiker geschafft, einen bestimmten Chiffreblock zu entschlüsseln, kann er diesen immer wieder entschlüsseln, zumindest solange, wie der Schlüssel nicht geändert wird. Bei der Übertragung entstandene Bitfehler bewirken das fehlerhafte Entschlüsseln des Blockes, indem der Fehler aufgetreten ist, es hat aber keine Auswirkungen auf andere Blöcke. Fatal dagegen wirkt sich der Synchronisationsverlust aus. Wird nur ein Bit hinzugefügt oder herausgenommen, wird dieser Block fehlerhaft entschlüsselt und alle nachfolgenden ebenfalls. Es ist von großem Vorteil einen Synchronisationsrahmen einzführen, der die Synchronisation wieder herstellen kann. Des weiteren lassen sich Nachrichten nicht so einfach in bestimmte Blockgrößen pressen, da der letzte Block meist nicht komplett belegt ist. Da aber vollständige Blöcke nötig sind, wird aufgefüllt. Dieser Vorgang wird als Padding bezeichnet. Beim Padding wird ins letzte Byte des Blocks die Anzahl der Zeichen im Block geschrieben und die anderen Bytes mit Zufallszahlen aufgefüllt. Damit sind aber der Klartext und Chiffretext nicht mehr gleich lang, was bei bestimmten Systemen ein Problem darstellt.



Kapitel 3

Kryptoanalyse

Block Replay oder Cut and Paste sind Methoden den Klartext zu manipulieren, ohne den Schlüssel zu kennen. Was für Kryptoanalytiker den großen Vorteil hat, dass keine rechenstarken und teuren Computer verwendet werden müssen. Dies ist im ECB-Modus sehr einfach, da der Chiffre Block nur von seinem Klartext-Block abhängig ist. Ein Beispiel:
Uns ist gelungen die Datenleitung von Bank A zu Bank B abzuhorchen bzw. Daten einspeisen zu können und wir wissen, dass die Daten im ECB-Modus verschlüsselt werden. Außerdem haben wir Kenntnis von dem Standard der Übertragung, auf den sich die beiden Banken geeinigt haben.

Wir eröffnen bei Bank B ein Konto, und überweisen 100,-EUR von Bank A auf das neue Konto. Das wiederholen wir später noch einmal. Vorher haben wir unseren Rechner angewiesen, alle Daten die von Bank A zu Bank B übertragen werden, abzuspeichern. Jetzt durchsuchen wir die Daten nach zwei gleichen Überweisungen. Sollte mehr als ein Suchergebnis auftreten, überweisen wir noch einmal 100,-EUR und suchen nach drei gleichen Überweisungen, bis wir nur noch ein Suchergebnis erhallten. Nun können wir sicher sein, dass es sich um unsere verschlüsselten 100,-EUR Überweisungen handelt. Diese schicken wir so oft wie möglich zu Bank B, wo uns immer 100,-EUR gutgeschrieben werden. Das fällt erst bei einem Abgleich der Daten der beiden Banken auf, welcher wahrscheinlich am Abend stattfindet. Selbst wenn die Geldinstitute einen Zeit- und Datumsstempel einführen, können wir die Blöcke, die unseren Namen und Kontonummer enthalten herausnehmen und diese in alle Überweisungen von Bank A zu B einfügen. So dass jegliches Geld, das von A nach B transferiert wird, auf unserem Konto gutgeschrieben wird. Der Vorteil ist, dass beim Datenabgleich noch alles in Ordnung ist. Es fällt erst auf, wenn jemand sein Geld vermisst.

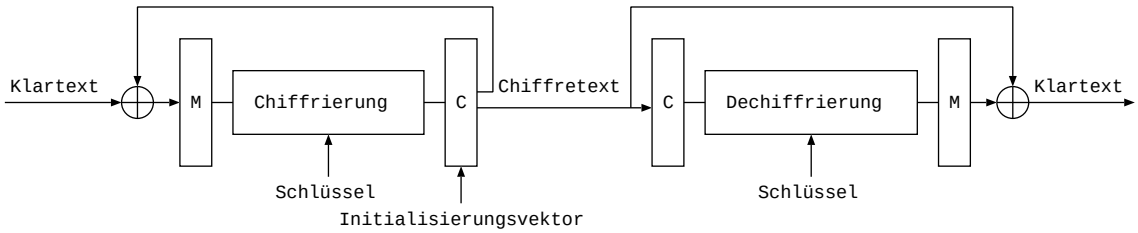
Kapitel 4

Cipher Block Chaining (CBC)

Beim CBC-Modus fließt der Chiffretext des vorherigen Klartextblockes in die Verschlüsselung des aktuellen Klartextblockes mit ein. Der Chiffretext vom vorherigen Klartextblock wird mit dem aktuellen Klartextblock XOR verknüpft und dann verschlüsselt. So hängt die Verschlüsselung des aktuellen Blocks von allen vorangegangenen Blöcken ab.

Die Entschlüsselung ist analog, der Chiffreblock wird entschlüsselt und mit dem vorherigen Chiffreblock XOR verknüpft. Es werden aber immer noch zwei identische Nachrichten aus einem identischen Chiffretext erzeugt. Es wird der gleiche Chiffretext produziert, bis ein Unterschied im Klartext vorhanden ist. Um das zu verhindern, wurde ein Initalisierungsvektor (IV) eingeführt. Er wird vor den ersten Klartextblock gesetzt und so wird jede Nachricht eindeutig. Der IV muss selbst keine Bedeutung haben, er kann aus Zufallsdaten oder einem Zeitstempel bestehen. Zwei identische Nachrichten erzeugen jetzt einen unterschiedlichen Chiffretext, damit wird ein Block-Replay-Angriff unmöglich.
Da der CBC blockorientiert arbeitet, ist es nur möglich vollständige Blöcke zu verschlüsseln, das kann aber in manchen Situationen sehr störend sein. Umgangen wird das Problem, wenn nur weniger wichtige Informationen am Ende stehen oder der Abschluss bekannt ist. Wird bei der Übertragung ein Bit umgedreht, durch einen Angreifer oder durch eine schlechte Übertragungsleitung, ist die Entschlüsselung des ganzen Blockes fehlerhaft und ein Bit des nächsten Blockes, an dem der Bitfehler sich befand. Danach erholt sich der CBC-Modus wieder und alle folgenden Blöcke werden fehlerfrei entschlüsselt. Anders ist es, wenn ein Bit herausgenommen oder hinzugefügt wird, da so die Synchronisation verloren geht. Dadurch werden der aktuelle Block und alle nachfolgenden falsch entschüsselt. Es ist daher empfehlenswert einen Synchronisationsrahmen einzuführen.

Der CBC hat eine gewisse Weiterentwicklung durch verschiedene Varianten erfahren. Dazu zählt der CBC-MAC. In ihm ist der message authentication code integriert, was eine eindeutige Zuordnung der Nachricht zu einem Absender möglich macht. Des weiteren gibt es den Propagating CBC. Er ist ähnlich wie der CBC aufgebaut, nur dass er eine Integritätsprüfung enthalten kann, um die Unverfälschtheit der Nachricht zu überprüfen. Leider kann der Modus die unangenehme Eigenschaft haben, dass eine unbegrenzte Fehlerfortpflanzung mögich ist.

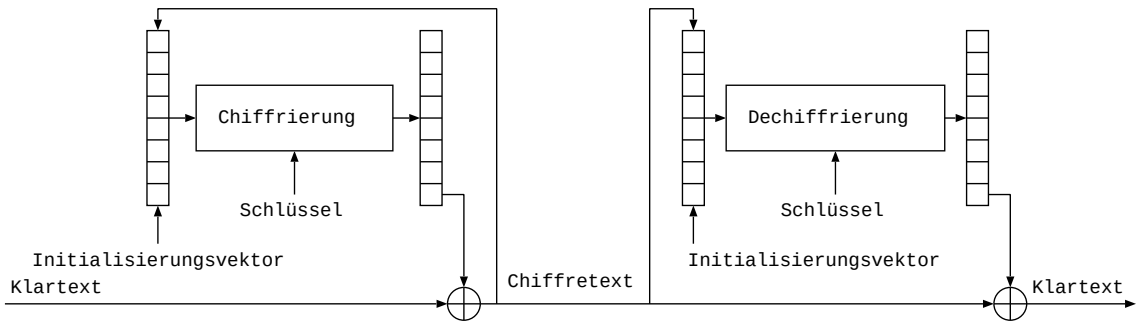


Kapitel 5

Cipher Feedback (CFB)

Der Cipher Feedback Modus, zu deutsch Schlüsselrückführung, ist im Gegensatz zu den vorherigen Modi zeichenorientiert. Das kann eine große Bedeutung bei Terminals haben, da Daten nicht erst übertragen werden wenn der Block komplett ist, sondern wenn jeder Tastendruck sofort übermittelt wird. Somit ist auch das Auffüllen nicht mehr notwendig. Bei diesem Modus wird nicht die Nachricht selbst verschlüsselt, sondern der Inhalt eines Schieberegisters. Das folgende Beispiel zeigt einen 8 Bit CFB, dabei werden einem Zeichen 8 Bit und einer Blockgröße 64 Bit zugeordnet (Block- und Zeichengrößen können aber auch anders gewählt werden). Der Inhalt des Schieberegisters, welcher einen Block großist, wird mit Hilfe des Schlüssels verschlüsselt. Das entstandene Byte wird nun übermittelt und in das Schieberegister geschoben, wobei an der anderen Seite ein Byte herausfällt.

Die Entschlüsselung läuft umgekehrt ab. Interessant dabei ist, dass diese den Verschlüsselungsalgorithmus benutzt. Das heißt, dass für Ver- und Entschlüsselung der selbe Algorithmus verwendet werden kann,was bei ECB und CBC nicht der Fall ist. Das erhaltene Chiffrebyte wird mit dem linken Byte des verschlüsselten Schieberegisters XOR verknüpft. Es entsteht das Klartextbyte. Nun muss noch das empfangene Chiffrebyte in das Schieberegister geschoben werden. Die Entschlüsselung ist also abhängig von den vorherigen 8 Byte, d.h., hat das Schieberegister der Entschlüsselung den selben Inhalt wie das der Verschlüsselung, so kann korrekt entschlüsselt werden, vorausgesetzt beide haben den selben Schlüssel benutzt.
Dadurch hat der CFB - Modus die Eigenschaft selbstsynchronisierend zu sein. Bei einem Verlust der Synchronisation dauert es eine Blocklänge bis das Schieberegister der Entschlüsselung sich wieder mit korrekten Daten gefüllt hat. Dann ist die Synchronisation wieder hergestellt.
Ähnlich verhält es sich bei Bit-Fehlern, die im Chiffretext auftauchen. Wird ein Bit umgedreht, so wird das aktuelle Byte fehlerhaft entschlüsselt. Nun kommt das Chiffrebyte in das Schieberegister und macht acht weitere Zeichen unbrauchbar, erst dann erholt sich das System wieder und entschlüsselt korrekt.
Die Bedingung, dass beide Schieberegister den selben Inhalt haben müssen, macht einen Initialisierungsvektor notwendig. Er ist hier nur ein Dummy, der für die Initialisierung benötigt wird, um das Schieberegister zu füllen.
Durch die selbstsynchronisierende Eigenschaft des CFB-Modus ist Block-Replay wieder möglich. Man speichert einen Chiffretext und speist diesen später wieder in das System ein. Am Anfang (eine Schieberegisterlänge) wird zwar nur Unsinn entschlüsselt, aber dann hat sich der Empfänger synchronisiert und entschlüsselt den eingespeisten Chiffretext.

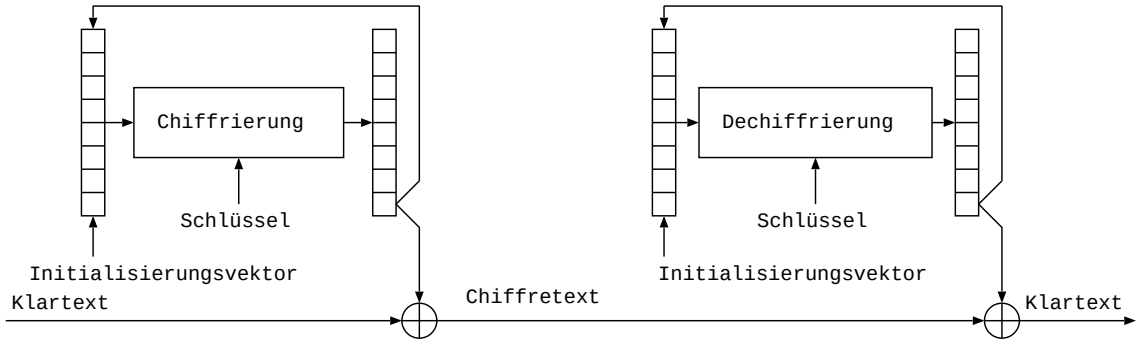


Kapitel 6

Output Feedack (OFB)

Dieser Modus ist ebenfalls zeichenorientiert wie der CFB. Auch die Vorgehensweise der Verschlüsselung ist ähnlich, beide arbeiten mit einem Schieberegister. Doch beim OFB wird nicht das Chiffrebyte in das Schieberegister geschoben, sondern das linke Byte des verschlüsselten Schieberegisters. Dieses Byte wird dann noch durch XOR mit dem Klartext verknüpft. Das so entstandene Chiffrebyte wird übermittelt. Bei der Entschlüsselung wird genauso verfahren, wobei auch hier der Verschlüsselungsalgorithmus benutzt wird.

Dadurch, dass vom verschlüsselten Schieberegister das linke Byte in das Schieberegister geschoben wird, ergibt sich die Möglichkeit, diese aufwendigen Berechnungen des Verschlüsselns schon vor dem Vorhandensein des Klartextes durchzuführen. Ist der Klartext dann vorhanden, wird er nur noch durch XOR mit den Berechnungen des Algorithmus verknüpft.
Auch der OFB-Modus wird initialisiert und benötigt dazu einen IV um das Schieberegister zu füllen.
Eine weitere Eigenschaft des Modus ist es, dass Bitfehler im Chiffretext auch Bitfehler im Klartext sind. Es gibt also keine Fehlerfortpflanzung, welche bei Übertragungen von Analogdaten, wie akustischen und visuellen Informationen, toleriert werden können. Der OFB ist nicht selbstsynchronisierend, deshalb ist es von Vorteil, einen Synchronisationsrahmen einzuführen, denn bei Verlust der Synchronisation wird der nachfolgende Chiffretext fehlerhaft entschlüsselt. Auch der OFB Modus hat einige Entwicklungsvarianten erfahren. Eine davon ist der OFB/Counter-Modus. Er macht es möglich wahlfrei auf jeden Block zuzugreifen, welches ein großer Vorteil bei Datenbanken ist.



Kapitel 7

Zusammenfassung

Es gibt keinen Verschlüsselungsmodus, der in allen Bereichen überlegen ist. Es gibt nur den optimalsten Modus für eine bestimmte Anwendung. Wenn man eine Datenbank verschlüsseln will, die minder wichtige Daten enthält, dann ist der ECB-Modus die beste Wahl. Handelt es sich um wichtige Daten, empfielt sich der OFB-Counter-Modus. Sind die Nachrichten kurz und von zufälligem Inhalt, bietet sich ebenfalls der ECB-Modus an, denn dort kommen seine Nachteile nicht zum Tragen. Der CBC ist gut geeignet für das Verschlüsseln von Daten, da hier ein blockorientiertes Verfahren ausreicht. Bei Zeichenströmen zwischen Terminal und Host eignet sich der CFB am besten, da er zeichenorienitert ist und auch selbstynchronisierend. Wenn es sich um synchrone Hochgeschwindigkeitsverbindungen dreht, kommt der OFB-Modus zum Einsatz, da hier keine Fehlerfortpflanzung existiert und Vorausberechnungen möglich sind. Genau so bietet sich der Modus bei fehleranfälligen Umgebungen an, da die Fehler dort nicht vergrößert werden können.