Computersystemsicherheit – Übungsblatt Nr. 4 – Lösung

Marc Fischlin, Jacqueline Brendel, Christian Janson
TU Darmstadt, 07 Dezember 2018

Gruppenübung. Die Übungsaufgaben in diesem Bereich sind Gegenstand der Übungen in der Woche vom 10.12.2018 – 14.12.2018.

Aufgabe 1 (Verständnisaufgaben). In dieser Aufgabe prüfen wir unser Verständnis über den Inhalt der Vorlesung.

a) Was ist das Schutzziel beim Einsatz von Prüfsummen?

Lösung.
Integrität

b) Geben Sie die grundlegenden Unterschiede zwischen einer Hashfunktion und einem MAC an. Welche Schutzziele werden durch MACs verfolgt?

Lösung.
Hashfunktion sind in der Regel bekannt und können von jedem genutzt werden. Es wird kein zusätzliches Geheimnis benötigt, um einen validen Hashwert zu ereugen.

Im Gegensatz ist bei einem MAC ein zusätzlicher geheimer Schlüssel nötig. Dieser Schlüssel garantiert, dass nur Entitäten mit richtigem Schlüssel einen validen MAC erzeugen können.
Schutzziele des MAC: Integrität und Authentizität.

c) Warum ist ein MAC nicht als digitale Unterschrift geeignet?

Lösung.
Zur Überprüfung des MACs benötigt man Kenntnis über denselben Schlüssel, der auch zur Berechnung benutzt wurde. Daher kann jeder, der einen MAC überprüfen kann, diesen auch berechnen (symmetrische Kryptographie). Das macht es unmöglich, gegenüber einem Dritten, zu beweisen von wem die Nachricht ursprünglich stammt (kein non-repudiation).
Im Gegensatz dazu werden digitale Signaturen mit Hilfe eines nur dem Absender bekannten Schlüssels erstellt und mit Hilfe eines öffentlichen Schlüssels überprüft (asymmetrische Kryptographie).

d) Beschreiben Sie die Begriffe Authentisierung (Authentication) und Autorisierung (Authorization).

Lösung.
Authentisierung ist der Prozess eine angegebenen Identität einer Entität mit Credentials (Benuzername, Passwort oder ähnliches) zu prüfen. Die Entität kann in Form einer Person, eines Computers, Gerätes oder eine Gruppe von Netzwerkcomputern sein.
Die **Autorisierung** repräsentiert das Recht für einen bestimmten Benutzer eine bestimmte Aktion (lesen, schreiben, erstellen, löschen, und starten) auf ein bestimmtes Objekt durchzuführen.

e) Nennen Sie Probleme, die im Zusammenhang mit dem Speichern von Passwörtern in einer Datenbank entstehen können. (Nehmen Sie dazu an, dass der Angreifer Zugriff auf die Datenbank hat, welche die Nutzerkennungen mit zugehörigen Passwörtern enthält.)

Lösung.

- Passwörter werden im Klartext gespeichert. Daher hat ein Angreifer Zugriff auf alle Passwörter.
- Passwörter werden verschlüsselt gespeichert. Falls ein Angreifer die Verschlüsselung brechen kann bzw. in den Besitz des secret key gelangt ist der Zugriff auf alle Passwörter im Klartext möglich.
- Passwörter werden mit einer unsicheren kryptographischen Hashfunktion gehasht. Hier könnte der Angreifer die schlechte Hashfunktion brechen und damit Zugriff auf alle Passwörter erhalten.
- ...

f) Welche Minimalanforderungen für die Verschlüsselung von sensiblen Daten sollten eingehalten werden?

Lösung.

- Angemessene und sichere Standardverfahren sollten verwendet werden.
- Die Schlüssellänge ist angemessen.
- Alle Passwörter werden unter Verwendung einer starken und standardisierten Hashfunktion gehasht unter der Verwendung eines angemessenen Salts.
- Schlüssel und Passwörter sind vor unautorisierten Zugriffen geschützt.
- Daten-Backups sind verschlüsselt und werden getrennt von den Schlüssel-Backups verwaltet.
- ...

g) Was versteht man unter einem Salt?

Lösung.
Eine zufällig generierte Zeichenfolge, die in geeigneter Form mit dem Passwort verbunden wird, um so bei einem Hashvorgang den resultierenden Hashwert so zu verändern, dass zwei gleiche Klartexte (meist Passwörter) nicht auf den gleichen Hashwert abgebildet werden. Jeder Datensatz (oder auch Instanz in anderen Kontexten) sollte einen zufällig generierten Salt enthalten.

h) Geben Sie an, welche Einträge Sie in einer Datenbank speichern müssten, um einen Benutzernamen und Passwort mit Salt zu verwalten.

Lösung.
userid, salt, hash(salt || password)

i) Welche Art von Angriffen wird durch ein Salt erschwert?

Lösung.
Beim Speichern eines Hashwert, der keinen Salt enthält, wird ein gleiches Passwort immer auf den gleichen Wert abgebildet und kann damit verglichen werden. Haben 100 Personen das gleiche Passwort gewählt, erkennt man dies auf den ersten Blick. Beim Speichern eines Hashwerts mit Salt, sieht jeder Hashwert anders aus und dadurch ist die direkte Vergleichbarkeit nicht gegeben. Ein Angreifer kann keine vorberechneten Hashwerte für häufige Passwörter verwenden (Rainbow Tables), sondern muss alle Hashwerte mit dem genutzten Salt neu berechnen.

Aufgabe 2 (MAC). Im Folgenden betrachten wir MACs.

a) Erläutern Sie die allgemeine Funktionsweise eines MACs.

Lösung.
Der Sender möchte eine Nachricht an den Empfänger senden. Dazu erzeugt er mithilfe einer Funktion und einem geheimen Schlüssel einen MAC, den er zusätzlich zur Nachricht an den Empfänger sendet. Dieser erstellt mit dem gleichen Schlüssel und der empfangenen Nachricht den MAC und überprüft, ob dieser mit dem empfangenen MAC übereinstimmt. Ist dies der Fall, ist die Nachricht authentisch und die Integrität gesichert.

b) Aus welchem Grund kann man keine reine, öffentliche Hashfunktion (z.B. SHA-2) als MAC verwenden?

Lösung.
Da eine Hashfunktion allgemein bekannt ist, könnte ein Angreifer die Nachricht manipulieren und einen gültigen MAC dazu erstellen. Das Verfahren des MACs ist allgemein bekannt, allerdings wird für die konkrete Berechnung ein geheimer Schlüssel benötigt. Die Berechnung eines MACs ist also nur von den Schlüsselbesitzern durchführbar. Dadurch kann kein Angreifer einen gültigen MAC für eine manipulierte Nachricht erstellen und Integrität ist gewährleistet.

c) Sie empfangen neben der Nachricht $m = 15$ den MAC Tag $t = 7$. Die MAC-Funktion ist definiert als $x \mapsto (x^k \cdot (k \bmod 5)) \bmod 11$, wobei k der geheime Schlüssel ist. Überprüfen Sie, ob die Nachricht korrekt versandt wurde und verwenden Sie dazu $k = 8$.

Lösung.
Es lässt sich nachrechnen, dass $15^8 \cdot (8 \bmod 5)) \bmod 11 \ = 5$.
Da t nicht mit dem Ergebnis übereinstimmt, wurde entweder die Nachricht oder der MAC nicht korrekt übertragen.

Aufgabe 3 (Prüfsummen). Sie haben in der Vorlesung kurz Prüfsummen kennengelernt, welche wir in dieser Aufgabe etwas vertiefen. Prüfsummen können dazu verwendet werden um Fehler bei der digitalen Datenübertragung zu erkennen.

a) Erläutern Sie den Unterschied zwischen MACs und Prüfsummen. Wieso können MACs nicht zur Authentifizierung verwendet werden? Warum helfen Prüfsummen nicht dabei, Man-in-the-middle-Angriffe zu verhindern?

Lösung.
MACs werden dazu verwendet, um die Integrität und die Herkunft der Nachricht zu überprüfen. Dabei wird der MAC mithilfe eines geheimen, symmetrische Schlüssels vom Sender erstellt, zusätzlich zur Nachricht übertragen und vom Empfänger mithilfe des gleichen Schlüssels überprüft. Da ein MAC mit dem gleichen Schlüssel überprüft wird, kann dieser nicht zur Authentifizierung verwendet werden, da jeder, der einen MAC überprüfen kann, diesen auch erstellen kann.
Eine **Prüfsumme** dient zur Erkennung von Fehlern bei der digitalen Übertragung. Mithilfe einer öffentlich bekannten Funktion wird eine Prüfsumme berechnet, die an die Nachricht angehängt wird. Im Gegensatz zum MAC kann jeder überprüfen, ob die Nachricht korrekt übertragen wurde, ebenso kann jeder zu einer beliebigen Nachricht eine entsprechende Prüfsumme berechnen. Aus diesem Grund sind Man-in-the-middle-Angriffe problemlos möglich.

b) Alice sendet die Nachricht $m = 10010101$ an Bob. Um sicher zu gehen, dass Bob mögliche Fehler erkennt, verwendet Alice die Paritätsprüfsumme, die sie an die Nachricht anhängt. Berechnen Sie das Paritätsbit b für die Nachricht m.

Lösung.
$b = b_0 \oplus b_1 \oplus b_2 \oplus b_3 \oplus b_4 \oplus b_5 \oplus b_6 \oplus b_7 = 1 \oplus 0 \oplus 0 \oplus 1 \oplus 0 \oplus 1 \oplus 0 \oplus 1 = 0$

c) Bei der Übertragung der Nachricht m geht leider etwas schief und es kommt die Nachricht $m^* = 10100101$ bei Bob an. Kann Bob erkennen, dass die Nachricht kaputtgegangen ist? Begründen Sie.

Lösung.
$b = 1 \oplus 0 \oplus 1 \oplus 0 \oplus 0 \oplus 1 \oplus 0 \oplus 1 = 0$
Nein, er kann es nicht erkennen, da zwei Bits gekippt sind und das Paritätsbit ausschließlich 1-Bit-Fehler erkennt.

d) Alice möchte nun doch statt der Paritätssumme den "Cyclic Redundancy Check" verwenden, um auch größere Bit-Fehler zu erkennen. Dazu verwendet Sie das folgende CRC-Polynom: $x^3 + x^2 + 1$.
Berechnen Sie die Nachricht, die an Bob gesendet wird. Überprüfen Sie anschließend, dass die übertragene Nachricht korrekt übertragen wurde.

Lösung.
Zur Berechnung des CRCs wird die Nachricht um zusätzliche Nullen ergänzt. Die Anzahl der Nullen entspricht dem Grad des CRC-Polynoms (in diesem Beispiel also 3). Anschließend wird die Nachricht (inkl. der zusätzlichen Nullen) durch das CRC-Polynom geteilt:

10010101000 : 1101 = 11110010
1101
1000
1101
1011
1101
1100
1101
1100
1101
010

Der Rest, der bei der Division herauskommt, wird an die ursprüngliche Nachricht angehängt. m^* lautet also: 10010101010.
Um zu überprüfen, ob die Übertragung korrekt war muss Bob diese Nachricht nun durch das CRC-Polynom teilen. Falls bei der Division kein Rest übrig bleibt bedeutet dies, dass die Übertragung korrekt war.

10010101010 : 1101 = 11110010
1101
1000
1101
1011
1101
1100
1101
1101
1101
00

e) Alice möchte eine alternative Prüfsummenmethode wählen, dazu verwendet sie die sogenannte Blockprüfsumme. Bei dieser werden für Nachrichten sowohl Zeilen- als auch Spaltenweise jeweils das Paritätsbit bestimmt. Vervollständigen Sie die Tabelle.

1	0	0	0	
0	1	0	1	
0	1	1	1	1
0	0	0	1	
1				

Lösung.
Die ausgefüllte Tabelle sieht folgendermaßen aus:

1	0	0	0	1
0	1	0	1	0
0	1	1	1	1
0	0	0	1	1
1	0	1	1	

f) Bob erhält bei der Übertragung eine fehlerhafte Nachricht. Markieren Sie in der Tabelle die fehlerhafte Stelle im Nachrichtenblock und korrigieren sie diese.

1	1	1	0	1
0	1	0	0	0
1	1	0	1	1
0	1	0	1	0
0	1	1	0	

Lösung.
Das fehlerhafte Bit ist in der Tabelle rot hervorgehoben.

1	1	1	0	1
0	1	0	0	0
1	1	0	1	1
0	1	0	1	0
0	1	1	0	

Richtig wäre folgende Tabelle und entsprechende Nachricht:

1	1	1	0	1
0	0	0	0	0
1	1	0	1	1
0	1	0	1	0
0	1	1	0	

Aufgabe 4 (Passwörter). Klassische Passwörter bestehen aus einer Abfolge von Textzeichen und werden über die Tastatur eingegeben.

a) Die Universität entscheidet sich UNICaN durch das revolutionäre System UNICaN2.0 zu ersetzen. Dieses erlaubt die Nutzung von Passwörtern mit folgenden Eigenschaften:

- erlaubt sind die Zeichen a-z, A-Z, 0-9 und 20 verschiedene Sonderzeichen;
- Passwörter müssen mindestens 6 Zeichen lang sein;
- Passwörter dürfen maximal 8 Zeichen lang sein.

Bestimmen Sie die Größe des Passwortraumes P.

Lösung.
$P = 82^6 + 82^7 + 82^8 = 2069373412383168$.

b) Alice folgt den Vorgaben zur Erstellung eines Passworts im UNICaN2.0 System aus Aufgabenteil a).
Eve versucht das Passwort zu brechen und kann 10000 Passwörter pro Sekunde testen. Wie lange braucht sie längstens um Alices Passwort zu brechen?

Lösung.
Eve ist spätestens nach $\frac{2069373412383168}{10000}$ $s \approx 6561,9$ Jahre fertig.

c) Nach einiger Zeit wird entschieden, die Passwort Policy für UNICaN2.0 folgenermaßen zu ändern:

- erlaubt sind nur die Zeichen a-z (nur kleine Buchstaben);
- Passwörter müssen mindestens 12 Zeichen lang sein;
- Passwörter dürfen maximal 16 Zeichen lang sein.

Bestimmen Sie die Größe des Passwortraumes P.

Lösung.
$P = 26^{12} + 26^{13} + 26^{14} + 26^{15} + 26^{16} = 45353088798247762554880$.

d) Eve versucht nun, Bobs Passwort zu brechen und kann nun aber 1000000 Passwörter pro Sekunde testen. Wie lange braucht sie längstens um Bobs Passwort zu brechen?

Lösung.
$\frac{45353088798247762554880}{1000000}$ $s \approx 1438137011,61$ Jahre

e) Nehmen Sie an, dass Passwörter in einer Datenbank gehasht gespeichert werden unter der Hashfunktion MD5(passwort).
Bewerten Sie diese Vorgehensweise und schlagen Sie mögliche Verbesserungen vor.

Lösung.
Passwörter sollten nicht einfach gehasht in der Datenbank gespeichert werden. Ein Angreifer, der sich Zugang zu der Datenbank verschafft hat, könnte nämlich einen großen Teil der Passwörter mit Rainbow-Tables knacken. Man sollte stattdessen die Passwörter mit einem Salt S versehen und als $H(S||pw)$ in der Datenbank speichern. Des Weiteren sollte die Hashfunktion MD5 nicht mehr genutzt werden sondern bessere Hashfunktionen wie SHA-2 oder SHA-3. Noch besser wären Funktionen wie bcrypt oder Argon2, welche durch hohe Iterationszahlen die Hashfunktion verlangsamen und somit bruteforce-Angriffe erschweren.

Hausübung. Dieser Bereich ist dazu gedacht das Gelernte weiter zu vertiefen. Dazu werden je nach Themen weitere Übungsaufgaben, ergänzende Beweise oder ähnliche Aufgaben gestellt. Die Aufgaben sind freiwillig, können aber, bei erfolgreicher Bearbeitung, zu Bonuspunkten in der Klausur führen. Die Abgabe dieser Übungen erfolgt über **moodle** und kann in Gruppen mit bis zu vier Studenten (aus Ihrer **eigenen** Übungsgruppe) eingereicht werden. Abgaben werden nur als **.pdf-Dateien** akzeptiert. Denken Sie bitte daran, dass Ihre Lösungen nachvollziehbar und entsprechend ausführlich dargestellt werden sollen.

Für Gruppenabgaben ist folgendes zu beachten: Sie müssen in der Abgabe (.pdf-Datei) deutlich und eindeutig kennzeichnen mit welchen Gruppenpartnern die Aufgaben gelöst wurden.

Der Fachbereich Informatik misst der Einhaltung der Grundregeln der wissenschaftlichen Ethik großen Wert bei. Zu diesen gehört auch die strikte Verfolgung von Plagiarismus. Falls dieser Fall eintritt, behalten wir uns das Recht vor für diese Abgabe den jeweiligen Gruppen keine Punkte gutzuschreiben.

Bitte reichen Sie Ihre Abgabe bis **spätestens Freitag 21.12.2018 um 11:40 Uhr** ein. Verspätete Abgaben können **nicht** berücksichtigt werden.

Hausübung 1 (Prüfsummen (2 Punkte)). Der Weihnachtsmann empfängt die folgende Nachricht $m^* = 10111101110$. Kann der Weihnachtsmann hier einen Übertragungsfehler entdecken (unter Verwendung des CRC-Polynoms $x^3 + x^2 + 1$)?

Hausübung 2 (MAC (1 + 2 Punkte)). Weihnachten rückt immer näher und die Elfen des Weihnachtsmanns sind schon wieder fleißig am Arbeiten. Die Digitalisierung ist mittlerweile auch am Nordpol angekommen. Deswegen werden die Aufträge des Weihnachtsmanns nun elektronisch an die Elfen weitergeleitet.

a) Der Weihnachtsmann einigt sich mit den Elfen auf den Schlüssel $k = (a, b, p) = (1, 3, 5)$ und die parametrisierte Hashfunktion H:

$$H(a, b, m, p) = (a \cdot m + b) \bmod p$$

Die Elfen schicken dem Weihnachtsmann die Nachricht $m = 213$ und den MAC-Tag $t = 1$. Helfen Sie dem Weihnachtsmann und überprüfen Sie, ob diese Nachricht gültig ist.

b) Der Weihnachtsmann und die Elfen haben gehört, dass die Hashfunktion SHA3-256 sich gut für MACs eignen soll, da diese nicht anfällig für sogenannte Length-extension Attacks ist und daher kann man den Tag leicht als $H(k||m)$ berechnen (wobei $||$ bedeutet die Konkatenation beider Strings).
Die Elfen nutzen als Schlüssel das Wort NORDPOL, welche in ASCII (hexadezimal) codiert ist. Berechnen Sie nun den MAC Tag der Nachricht MUETZE und kodieren Sie die Ausgabe in hexadezimal.

Hausübung 3 (Passwörter (0,5+0,5+0,5+0,5+(2+1) Punkte)). Erinnern Sie sich an Aufgabe 4 aus der Gruppenübung.

a) Stellen Sie eine allgemeine Formel zur Berechnung des Passwortraumes P anhand des Alphabets A und der Länge des Passwortes l auf. Nehmen Sie zunächst an, dass nur Passworte einer Länge auftreten können[1]. Geben Sie eine zweite Formel an, die verschiedene Längen (in Form eines Längenbereichs) berücksichtigt.

b) Gegeben sei ein Textpasswortverfahren, welches nur die Zeichen a-z, A-Z, 0-9 und die Sonderzeichen ä, ö, ü, Ä, Ö, Ü zulässt. Andere Zeichen können nicht gewählt werden. Alle Passworte müssen aus genau 5 Zeichen bestehen. Wie groß ist der (theoretische) Passwortraum dieses Verfahrens[2]? Geben Sie die Größe sowohl dezimal als auch in bits an.

c) Der Grinch möchte Zugriff auf das E-Mail-Konto des Weihnachtsmanns erlangen. Wie lange braucht er maximal um das Passwort des Weihnachtsmanns zu erraten, wenn dessen E-Mail-Provider Passwörter verlangt, die mindestens 10 Zeichen und maximal 63 Zeichen der Form a-z,A-Z, 0-9, ä, ö, ü, Ä, Ö, Ü, ß sowie 15 verschiedene Sonderzeichen erlaubt, wenn Rudolf 2000 Passwörter pro Sekunde testen kann?

d) Der Weihnachtsmann erinnert sich, dass er in der Vorlesung *Passwörter* gelernt hat, dass Aneinanderreihungen von mehreren Wörtern als Passwort sehr sicher und leicht zu merken sind. Er entscheidet sich daher dazu, fünf zufällige Wörter aus dem etwa 300.000 Worte großen Wortschatz der deutschen Sprache aneinanderzureihen. Der Grinch hat sich in der Zwischenzeit Zugang zu einem Supercomputer organisiert, der 1.000.000.000 Passwörter pro Sekunde ausprobieren kann. Wie lange benötigt er, um das Passwort vom Weihnachtsmann mittels Brute-Force zu knacken? Gehen Sie dafür davon aus, dass der Weihnachtsmann zufällige Worte (nicht wiederholende) der Länge 5 gewählt hat von welchen es 20.000 Worte gibt.

e) Rudolf das Rentier wurde beauftragt ein Passwortsystem aufzusetzen, dass auch langfristig gegen den Grinch sicher ist. Um ein Passwort durch ausprobieren mit Sicherheit knacken zu können, soll dieser 42 Jahre lang alle Kombinationen ausprobieren. Aus vorangehenden Partnerschaften weiß Rudolf, dass der Grinch nur noch 1337 Passwörter pro Stunde testen können.

   (i) Wie groß muss der Passwortraum P mindestens sein?

   (ii) Schlagen Sie eine Passwortpolicy vor, die diese Bedingung erfüllt und berechnen Sie für Ihre Policy Passwortraum und Maximaldauer (in Jahren), die zum Durchprobieren benötigt wird.

[1] Dies ist nicht jenseits des Vorstellbaren und wird z.B. von einigen Banken verwendet.
[2] Dies ist eine reale extisterende Password Policy einer Bank.