Hochschule RheinMain
Studiengang Informatik - Technische Systeme
Prof. Dr. Marc Stöttinger

Probeklausur Security (LV4120)

Sommersemester 2023

Nachname: **Mustermeier**	Vorname: **Theo**
Matrikelnummer: **VOID**	
Datum: 25.07.2023	**Unterschrift:**

Sie erhalten eine geheftete Klausur. **Bitte lösen Sie die Heftung nicht**. Bitte tragen Sie zu Beginn der Bearbeitungszeit Ihren Namen und Ihre Matrikelnummer an den dafür vorgesehenen Stellen ein und unterschreiben Sie die Klausur. Die Klausur ist **nur mit Unterschrift** gültig. Die Klausur muss mit dem Verlassen des Raumes abgegeben werden.

Zum Bestehen der Klausur sind 45 Punkte (50%) notwendig

Im Falle nicht ausreichenden Platzes benutzen Sie bitte zusätzliche Blätter, die Sie mit Name und Matrikelnummer versehen. Machen Sie bitte eindeutig kenntlich, auf welche Aufgabe sich Ihre Antwort bezieht.

Dauer: 90 min (Klausur)
Hilfsmittel: eigene Formelsammlung von maximal einer doppelseitig handschriftlich beschriebenen DIN A4 Seite.

Punkte:

Aufgabe	Soll-Punkte	Ist-Punkte
1	20	
2	20	
3	20	
4	20	
5	10	
Gesamt	90	

Note:

Aufgabe 1: (20 Punkte)

a) Welche Klasse von symmetrischen Chiffren kann zur Erstellung von Message Authentication Codes (MAC) verwendet werden? (1P.)
Ein MAC kann unter Verwendung symmetrischer Blockchiffren mit geeigneten Betriebsmodi erstellt werden.

b) Erläutern Sie, warum Token, die mit Hilfe von Message Authentication Codes erzeugt werden, nicht das gleiche Schutzniveau bieten wie Signaturen, die asymmetrische Methoden verwenden. (2P.)
Das Schutzziel der Bindung mit einem MAC wird nicht erfüllt. Jeder, der den geheimen symmetrischen Schlüssel zur Verifizierung des MAC-Tokens besitzt, kann auch einen Token erzeugen. Die Verifikations- und Generierungsvorschriften für den MAC-basierten Token sind identisch. Bei einer Signaturmethode sind die Vorschriften für die Erzeugung und Überprüfung der Signatur unterschiedlich. Nur der private Schlüssel kann zur Erzeugung der Signatur verwendet werden und nur der öffentliche Schlüssel kann zur Überprüfung verwendet werden.

c) Warum ist es für einen sicheren Software-Aktualisierungsprozess besser, Software mit einer Signatur zu signieren und zu verifizieren als mit einem MAC? (2P.)
Für die Signatur kann nur der öffentliche Schlüssel zur Überprüfung verwendet werden. Es ist jedoch nicht möglich, mit diesem Schlüssel eine Signatur zu erzeugen. Anders verhält es sich bei einem MAC. Das gemeinsame Geheimnis kann zur Erzeugung und Überprüfung eines MAC verwendet werden. Wenn also alle Geräte denselben geheimen Schlüssel für die Verifizierung verwenden, besteht ein potenzielles Risiko, dass Angreifer vertrauenswürdige Software für alle Geräte mit demselben Schlüssel erzeugen können, wenn ein Gerät geknackt und der Schlüssel extrahiert wurde.

d) Schreiben Sie die Vorschrift zur Überprüfung einer DSA-Signatur in Pseudocode auf. (5P.)

Algorithm 1 Verifkation der Signatur (r, s)

Require: Öffentlicher Schlüssel K_{public}, Nachricht m, Signatur (r, s)
Ensure: Signatur valide oder nicht

$w \equiv s^{-1} \mod q$
$u_1 \equiv m \cdot w \mod q$
$u_2 \equiv r \cdot w \mod q$
$v \equiv (g^{u_1} \cdot K_{pb}^{u_2} \mod p) \mod q$
if $v == r$ **then**
  Signature ist valide
else
  Signature ist falsch
end if

e) Gegeben seien folgende Parameter für eine DSA-Signatur: $g = 2, p = 13, q = 11$. Mit diesen Parametern ist die Nachricht $m = 2$ signiert worden. Die Signaturparameter lauten $r = 5$ und $s = 8$. Prüfen Sie mithilfe des öffentlichen Schlüssels $K_{public} = 8$, ob die Signatur valide ist. (10P.)

$w \equiv s^{-1} \mod q \equiv 8^{-1} \mod 11 \equiv 6$
$u_1 \equiv m \cdot w \mod q \equiv 2 \cdot 7 \mod 11 \equiv 3$
$u_2 \equiv r \cdot w \mod q \equiv 5 \cdot 7 \mod 11 \equiv 2$
$v \equiv (g^{u_1} \cdot K_{public}^{u_2} \mod p) \mod q \equiv (2^3 \cdot 8^2 \mod 13) \mod 11 \equiv 5$

Damit ist $v = r$ und die Signatur ist valide.

Aufgabe 2: (20 Punkte)

a) Das folgende Szenario ist gegeben: Ein Client und ein Server wollen miteinander kommunizieren. Zuvor muss eine Authentifizierung über ein Challenge-Response-Verfahren mit einem RSA-Signaturverfahren durchgeführt werden. Zu diesem Zweck besitzt der Server einen privaten $K_{CR_{pr}}$ und einen zugehörigen öffentlichen Schlüssel $K_{CR_{pb}}$. Der Client verfügt ebenfalls über den öffentlichen Schlüssel des Servers $K_{CR_{pb}}$. Wenn eine Challenge Response auf der Grundlage dieser Schlüssel ausgeführt wird, wer authentifiziert sich gegenüber wem und handelt es sich um einen einseitigen oder zweiseitigen (mutual) Authentifizierungsprozess? (2P.)
Bei diesem Szenario ist nur eine einseitige Authentifizierung möglich. Der Server authentifiziert sich gegenüber dem Client.

b) Skizzieren Sie den Protokollablauf des Challenge-Response-Verfahrens unter Verwendung einer RSA-Signatur. Geben Sie an, welche Schlüssel der Client und der Server haben und aus welchen RSA-Parametern sie bestehen. (6P)

Client		**Server**
Besitzt: $K_{CR_{pb}} = (N, e)$		**Besitzt:**$K_{CR_{pb}} = (N, e)$ $K_{CR_{pr}} = d$
	$\xleftarrow{\textbf{Anfrage}}$	
Erzeugt Challenge: **Wähle zufälliges** C		
	$\xrightarrow{\textbf{sende } C}$	
		Berechne Signatur: $s \equiv C^d \mod N$
	$\xleftarrow{\textbf{sende } s}$	
Validiere Signatur: $C' \equiv s^e \mod N$ **Prüfe ob** $C' = C$		

c) Ein Client erhält einen öffentlichen Schlüssel, eine Signatur und eine Challenge mit den folgenden Parametern:

- $N = 55$
- $e = 7$
- $s = 17$
- $C = 8$

Prüfen Sie, ob die Signatur gültig ist. (12P)

$$
\begin{aligned}
C' &\equiv s^e \mod N \equiv 17^7 \mod 55 \\
&\equiv (17 \cdot 17 \mod 55) \cdot (17^5 \mod 55) \\
&\equiv (289 \mod 55) \cdot (17^5 \mod 55) \\
&\equiv 14 \cdot (17^5 \mod 55) \\
&\equiv 14 \cdot 14 \cdot 14 \cdot 17 \mod 55 \\
&\equiv (14 \cdot 14 \mod 55) \cdot (14 \cdot 17 \mod 55) \mod 55 \\
&\equiv (196 \mod 55) \cdot (238 \mod 55) \mod 55 \\
&\equiv 31 \cdot 18 \mod 55 \\
&\equiv 558 \mod 55 \\
&\equiv 8
\end{aligned}
$$

Die Signatur ist valide, da $C = C'$ ist.

Aufgabe 3: (20 Punkte)

a) Auf welcher Ebene werden VLAN abgesichert und welches Security Protokoll kann dafür einsetzt werden? (1P.)
VLANS werden auf der Schicht von Netzzugriff umgesetzt. MACSec kann genutzt werden, um die virtuellen Netzwerke mit authentischer Verschlüsselung zu sichern.

b) Was sind die unterstützten Betriebsarten von IPSec und wie unterscheiden sie sich? (2P.)
P-Sec verfügt über zwei Betriebsarten: Transportmodus und Tunnelmodus. Im Transportmodus wird nur die Payload authentisch verschlüsselt und der ursprüngliche IP-Header und Tailer bleiben erhalten. Im Tunnelmodus werden der ursprüngliche Header und Tailer ebenfalls authentisch verschlüsselt und ein neuer IP-Header und Tailer hinzugefügt.

c) Gehört die folgende Ciphersuite `TLS_DHE_RSA_WITH_AES_128_GCM_SHA256` zu TLS1.2 oder TLS1.3? Erläutern Sie, welche Algorithmen in dieser Ciphersuite für welche kryptographischen Sicherheitsoperationen verwendet werden, um eine sichere Kommunikation aufzubauen? (5P.)
Es handelt sich hierbei um einen Ciphersuite der Version TLS1.2. Folgende Sicherheitsoperationen werden mit folgenden Algorithmen in dieser Ciphersuite umgesetzt:

- **Schlüsselaustausch l: DHE - Diffie-Hellman mit Ephemera Schlüssel**
- **Authentifizierung: RSA - RSA Signatur**
- **Verschlüsselung der Transportdaten: AES_128_GCM - AES mit 128 bit Schlüssellänge im Betriebsmode Galois/Counter Mode**
- **Ableiten des Sessionschlüssels: SHA256 - SHA2 mit 256 Bit digest länge**

d) Skizzieren Sie den Ablauf des Handshake-Protokolls für TLS1.3 mit DHE, ECDSA und der TLS_AES_128_GCM_SHA256 Cipher Suite. Geben Sie an, welche Parameter dem Client und dem Server bekannt sind. (12P.)

Dem Client und Server sind folgende Parameter bekannt:

- **DHKE-Parameter:** p **und** g
- **ECC-Parameter:** $E(p_E, a_E, b_E, q_E, A_E)$
- **CS: Algorithmen der erlaubten Ciphersuites**

Client		**Server**
wähle ein $a \in \{2 \dots p-2\}$		
Berechne $A = g^a \mod p$		
	$\xrightarrow{\textbf{Client Hello: Nonce}_A\textbf{,}}$ $\textbf{CS, Version, } C$	
		wähle ein $b \in \{2 \dots p-2\}$
		Berechne $B = g^b \mod p$
		Wähle ein d
		Berechne $B_E = d \circ A_E$
		$K_{S_{pr}} = d$
		$K_{S_{pb}} = (p_E, a_E, b_E, q_E, A_E, B_E)$
		Wähle ein $k_E \in \{1 \dots q_E - 1\}$
		$R = k_E \circ A_E$
		r_x **X-Koordinate von** R
		Signatur öffentlicher Schlüssel B**:**
		$s(Hash(B) + d_E \cdot r_x) \circ k_E^{-1} \mod q_E$
		Berechne Sessionschlüssel:
		$K_{session} =$
		$Hash(\textbf{Nonce}_A \vert \textbf{Nonce}_B \vert A^b \mod p)$
	$\xleftarrow{\textbf{Server Hello: Nonce}_B\textbf{,}}$ $\textbf{Zertifikat, selected CS, } B,r_x,s$	
Prüfe Zertifikat		
Validiere Signatur:		
$w \equiv s^{-1} \mod q_E$		
$u_1 \equiv w \cdot hash(B) \mod q_E$		
$u_2 \equiv w \cdot r_x \mod q_E$		
$P = u_1 \circ A_E + u_2 \circ B_E$		
Prüfe ob $P_x = r_x$		
Berechne Sessionschlüssel:		
$K_{session} =$		
$Hash(\textbf{Nonce}_A \vert \textbf{Nonce}_B \vert B^a \mod p)$		
	$\xleftrightarrow{AES-GCM_{K_{Session}}(data)}$	

Aufgabe 4: (20 Punkte)

a) Erklären Sie den Begriff "Buffer Overflow". (1P.)
Der Begriff "Buffer Overflow"bezieht sich auf eine Sicherheitslücke, die auftritt, wenn ein Programm versucht, mehr Daten in einen Puffer (Buffer) zu schreiben, als dieser tatsächlich aufnehmen kann.

b) Welche Kriterien verwendet der Basic Score des Common Vulnerability Scoring Systems zur Bestimmung der Komplexität der Ausnutzung im Basic Score, zusätzlich zu den Aspekten Angriffskomplexität, erforderliche Rechte und Reichweite? (2P.)

- **Angriff möglich aus**
- **Interaktion des Opfers notwendig**

c) Welche Code-Analysemethode findet keine Laufzeitfehler im Code? (1P.)
Die statische Code-Analyse findet keine Fehler die zur Laufzeit entstehen.

d) Gegeben sei folgender Abschnitt eines C-Programms:

```
...
int main()
{
        unsigned short a = 42;
        unsigned int b;
        scanf("%u", &b);

        if(a == (short) b)
        {
                printf("a ist gleich b\n");
        }
        else
        {
                printf("a ist ungleich b\n");
        }
        return 0;
}
```

d1) Was für ein Implementierungsfehler liegt hier vor, der einen potentielle Schwachstelle sein könnte? (2P.)
Informationsverlust durch Interpretationsfehler durch explizites Typcasting.

d2) Für welche Werte von `b` wird fälschlicherweise die Aussage `a ist gleich b` getroffen? (2P.)
Für alle Werte größer als 0x0000 0000 0001 002A oder 65578.

d3) Wie kann dieses Problem in diesem Fall umgangen werden? (1P.)
Durch implizites Typcasting.

e) Gegeben sei folgendes kleines C-Programm zum Ersetzten von Zeichen in Zeichenketten inklusive dessen beispielhaften Aufruf in der Funktion `main()`:

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  # define MAX_SIZE 7
5
6  char * copy_input(char *user_supplied_string){
7      int i, dst_index;
8      char *dst_buf = (char*) malloc (4*sizeof(char) * MAX_SIZE);
9      if ( MAX_SIZE <= strlen(user_supplied_string) )
10     {
11         printf("user string too long, die evil hacker!");
12     }
13     dst_index = 0;
14     for ( i = 0; i < strlen(user_supplied_string); i++ )
15     {
16         if( '&' == user_supplied_string[i] )
17         {
18             dst_buf[dst_index++] = '&';
19             dst_buf[dst_index++] = 'a';
20             dst_buf[dst_index++] = 'm';
21             dst_buf[dst_index++] = 'p';
22             dst_buf[dst_index++] = ';';
23         }
24     else dst_buf[dst_index++] = user_supplied_string[i];
25     }
26 return dst_buf;
27 }
28
29 int main()
30 {
31     char test [7]="mundms";
32
33     copy_input(&test[0]);
34     return 0;
35 }
```

e1) Analysieren Sie den Code. Prüfen Sie, was in diesem Code die Ursache für einen BufferOverflow auf dem Stack ist. Geben Sie die Zeile an, welche den zu dem BufferOverflow führt und erläutern Sie wie es zu dem Overflow kommen kann. (5P.)

In Zeile 8 wird nicht ausreichend Platz im Puffer reserviert, falls die Zeichenkette ausschließlich aus &-Zeichen besteht. In der Schleife von Zeile 14 bis Zeile 23 werden pro &-Zeichen fünf neue Zeichen hinzugefügt, anstatt nur vier. Dadurch entsteht ein größerer Platzbedarf als ursprünglich reserviert wurde.

e2) Geben Sie ein Beispiel dafür, wie ein Buffer Overflow ausgelöst werden könnte, ohne die Anzahl der Zeichen in der Zeichenkette `test` zu erhöhen. (1P.)

```
1     char test [7]="&&&&&&";
2
3
4 ...
```

Aufgabe 5: (10 Punkte)

a) Warum ist Plattformintegrität für das Sicherheitskonzept von Programmen oder Anwendungen, die auf der Plattform laufen, wichtig? (1 P.)
Eine Anwendung oder ein Programm kann in der Regel nicht überprüfen, ob sie/es in einer integren Umgebung läuft oder kommuniziert.

b) Nennen Sie vier Sicherheitsmaßnahmen, die auf der Plattformebene eingesetzt werden können, um die Software auf der Plattform vor Angriffen oder Manipulationen zu schützen. (4P.)

- **Sicherer Wartungszugang oder sichere Diagnose**
- **Sicheres Software-Update oder sicheres Aktualisieren der Software**
- **Sicheres Booten der Software oder secure/authentic boot**
- **Vertrauenswürdige Ausführungsumgebung oder Trusted Execution Environment**

c) Wofür können Secure Boot und Measured Boot verwendet werden? (1P.)
Secure Boot und Measured Boot können verwendet werden, um festzustellen, ob die auszuführende Software manipuliert wurde oder authentisch ist.

d) Was ist der Unterschied zwischen Secure Boot und Authentic Boot. Erklären Sie genau, wie sich die beiden Methoden unterscheiden. (4P.)
Der Unterschied zwischen den beiden Methoden liegt in der Reaktion oder Aktion, wenn die Prüfung der zu startenden Software fehlschlägt. Bei Secure Boot wird die Software nicht ausgeführt, wenn die Prüfung fehlschlägt. Bei Authentic Boot wird die Software trotzdem ausgeführt und es werden andere Gegenmaßnahmen ergriffen, wie z. B. sichere Protokollierung, keine Freigabe von Schlüsselmaterial oder kein bzw. eingeschränkter Zugriff auf Ressourcen.