Die „Security Architecure for IP" zählt vier Themenbereiche auf: Confidentiality, Authentication, Integrity und Non-Repudiation. Charakterisieren Sie diese kurz. Erklären Sie weiters kurz die Basisbegriffe der Kryptologie: Cryptography, Cryptanalysis, Encryption, Decryption, Key, Key-Length, Algorithm (Secret-Key und Public-Key), Brute-Force-Attack, Passive Intruder, Active Intruder, Ciphertext-only-Attack, Known-Plaintext-Attack, Chosen-Plaintext-Attack, Replay Attack, Dictionary Attack. Geben Sie die prinzipiellen Formel zum Verschlüsseln und Entschlüsseln bzw. Signieren und Verifizieren für die beiden Hauptalgorithmen an. [100 Punkte]

Security Architecture for IP:

- **Confidentiality (Vertraulichkeit, Geheimhaltung)**

  ist die Eigenschaft der Kommunikation damit nur die beabsichtigten Empfänger wissen was gesendet wurde und unbeabsichtigte Teilnehmer / Partein nicht ermitteln können, was gesendet wurde. Confidentiality wird hauptsächlich durch Kryptographie erreicht.Damit man durch Security keinen ungewollten Informationsverlust hat.

- **Authentication**

  ist die Eigenschaft des Wissens das der angenommene Sender tatsächlich auch der wirkliche aktuelle Sender ist. Durch digitale Signaturen und Zertifikate wird nachgeprüft, wer die Daten über das Netzwerk sendet.

- **Integrity (checking)**

  ist die Eigenschaft der Sicherstellung, dass die Daten von der Quelle bis zum Ziel übermittelt wurden, ohne unentdeckt Verändert worden zu sein. Integritätscheck um gegen ungewollte Veränderung der Nachrichten zu schützen. Die Richtigkeit der Information wird überprüft.

- **Non-Repudiation (Nicht-Ableugbarkeit , Unleugbarkeit)**

  ist die Eigenschaft des Empfängers in der Lage zu sein zu prüfen ob der Sender von Daten die Daten auch gesendet hat, obwohl der Sender möglicherweise später bestreitet möchte diese Daten jemals versendet zu haben.
  Um sicherzustellen, dass eine Aktion nicht von der Person bestritten wird, von der sie durchgeführt wurde (Beweißbarkeit).

Basisbegriffe der Kryptologie:

- **Cryptography:**
  - erfindet Methoden zur Verschlüsselung
- **Cryptanalysis:**
  - versucht die Verschlüsselung zu brechen
- **Encryption:**
  - der Prozess eine Nachricht so zu verändern, dass der Originalinhalt nicht erkennbar ist. (verschlüsseln)

- **Decryption:**
  - der verschlüsselte Text wird wieder in den Klartext zurückgewandelt
- **Key:**
  - nur den teilnehmenden Personen bekannt, schwer erratbar. Ausreichende Länge erforderlich um Brute-Force-Attacken zu widerstehen
- **Key-Length:**
  - umso länger der Schlüssel ist, desto länger dauert es ihn zu knacken (wächst exponential). Aber auch Rechenaufwand zum Ver- und Entschlüsseln steigt.
- **Algorithm (Secret-Key und Public-Key):**
  - mathematische Funktion zur Verschlüsselung (muss gut verwahrt werden wenn die Verschlüsselung nur darauf basiert)
- **Brute-Force-Attack:**
  - alle möglichen Schlüssel werden ausprobiert um den richtigen Schlüssel zu finden
- **Passive Intruder:**
  - sie beschaffen sich Informationen über Dinge, die nicht für sie bestimmt sind (wie zB Passwörter, Kreditkartennummern) und verwenden sie um in Systeme einzubrechen und Information zu bekommen bzw. Schaden anzurichten.
- **Active Intruder:**
  - manipuliert die Nachrichten sofort > Integritätsüberprüfung, oder spielt sie noch mal ein (replay attack)
- **Ciphertext-only-Attack:**
  - der Kryptoanalytiker hat mehrere verschlüsselte Texte aber keinen Klartext und versucht per Brute-Force-Attack den Inhalt herauszufinden, dafür benötigt man genug verschlüsselten Text, bei modernen Verschlüsselungen funktioniert dies nicht.(in endlicher Zeit)
- **Known-Plaintext-Attack:**
  - Schlüssel wird aus dem verschlüsselten Text und dem passenden Klartext herausgesucht – nützlich wenn der Schlüssel bei späteren Verschlüsselungen genauso verwendet wird (> Enigma Maschine)
- **Chosen-Plaintext-Attack:**
  - nur bestimmte Teile des verschlüsselten Textes können verschlüsselt werden (frei wählbar), daraus wird dann der Schlüssel ermittelt.
- **Replay Attack:**
  - die Nachrichten werden nochmal versendet > Verwirrung für den Benutzer
- **Dictionary Attack:**
  - ein Wörterbuch wird verwendet um das Passwort herauszufinden

Formeln zum Verschlüsseln und Entschlüsseln

- **Secret Key:**
  - $C = f_E (K, M)$
  - $M = f_D (K, C)$
  - C (Ciphertext)
  - K (Key)
  - M (Plaintext)
  - E (Encryption)
  - D (Decryption)
  - f_E (K, M) (encryption function performed on M using K)
  - f_D (K, C) (decryption function performed on C using K)

- **Public Key:**
  - $C = f_E (P_B, M)$ -> Alice verschlüsselt mit Hilfe des Public-Keys von Bob.
  - $M = f_D (S_B, C)$ -> Bob entschlüsselt mit seinem eigenen Private-Key.
  - $C = f_E (P_A, M)$ -> Bob verschlüsselt mit Hilfe des Public-Keys von Alice.
  - $M = f_D (S_A, C)$ -> Alice entschlüsselt mit seinem eigenen Private-Key.
  - C (Ciphertext)
  - S (Private-Key)
  - P (Public-Key)
  - M (Plaintext)
  - E (Encryption)
  - D (Decryption)
  - f_E (K, M) (encryption function performed on M using K)
  - f_D (K, C) (decryption function performed on C using K)

Welche Gefahren bezüglich Network Security treten auf Layer 2 (Ethernet Technology) auf? Stichworte: Network Sniffing, MAC Flooding, MAC Address Spoofing. Beschreiben Sie diese kurz. Welche Gefahren bezüglich Network Security treten auf Layer 3 bzw. 4 (IP bzw. TCP/UDP Technologie) auf? Stichworte: IP Fragmentation Attack, Wege zum Umleiten von IP, IP Spoofing, ARP Spoofing, ICMP Attacks, Dos Attacks, TCP SYN Flooding, UDP Spoofing / Hijacking, UDP Storm, DNS Spoofing. Beschreiben Sie diese kurz. [100 Punkte]

- **Network Sniffing (passive attack)**

◊ enable „promiscuous mode" on your Ethernet Card and you will receive all traffic which appears at your card
◊ in a repeater environment you will see every Ethernet frame carried over the shared media
◊ in a bridged/ switched environment you will see only Ethernet frames destined to your MAC address and Broadcast/ Multicast frames
  - assumption: bridge has already learned all MAC addresses of the given LAN and hence flooding is not used any longer

- **MAC Flooding (active attack)**

◊ even in a bridged/ switched environment can get all traffic on the given LAN appearing on your card by performing by "MAC flooding"

- ◊ get your machine producing a huge number of MAC-frames every single frame carrying a different MAC address (so called bogus address)
- ◊ bridge/ switch table will overrun
- ◊ will cause the bridge/ switch to perform a "Flooding Decision" for every frame received
- ◊ hence to will see every frame on your LAN

• MAC Address Spoofing

- ◊ BIA address of your Ethernet adapter can be changed
- ◊ So you can impersonate another station
  - o Either send frames in the name of the impersonated
  - o Or receive frames destined for this station
- ◊ Will cause a problem if for example authentication or a trusted relationship is based on MAC address

• IP Fragmentation Attack

- ◊ ping of death
- ◊ maximum length of an IP packet = 65535
- ◊ send a fragmented IP ping with a resulting length greater than 65535 octets after reassembly
  - o the offset of the last segment is such that the total size of the reassembled datagram is bigger than the maximum allowed size
  - o kernel buffer overflow may cause a collapse of the OS
- ◊ Type: DoS

• IP spoofing

- ◊ impersonate another IP station by using the stations IP address as source address in own packets (so called faked address)
- ◊ either for DoS attack or to break into a system which has authentication based on IP address

• ARP spoofing

- ◊ impersonate another IP station by faking the stations MAC address with the own MAC address in a foreign ARP cache

• ICMP attacks

- ◊ manipulating ICMP redirect message
- ◊ attacker sends a spoofed ICMP redirect message that appears to come maybe from the hosts default gateway
- ◊ Type: redirection

• DoS (Denial of Service)

- – disturbing the service a machine which offers a service in the internet
- –

Wie ging die klassische Kryptography vor? Erklären Sie in diesem Zusammenhang

gängige Subsitutionsverfahren. Was versteht dabei man unter monoalphabetic? Erklären Sie in diesem Zusammenhang gängige Transpositionsverfahren. Wie können diese Verfahren geknackt werden. Welchen Ansatz wählte Vigenere (Stichwort polyaphabetic)? Was sind Homophone? Was versteht man unter One-Time-Pad? Wieso ist er die sicherste aber auch die unpraktikabelste Methode? Wie kommt man über S- und P Boxen zu den heutigen Product-Ciphers? [90 Punkte]

Klassische Kryptographie:

- **Substitutionsverfahren:**

Ein Verschlüsselungsverfahren, bei dem jeder Buchstabe der Mitteilung durch einen anderen Buchstaben (oder ein anderes Zeichen) ersetzt wird, bei dem die Buchstaben jedoch ihre Position in der Mitteilung behalten.

Caesar Verschlüsselung:

Der Klartext wird durch 2tes, um k Stellen verschobenes, Alphabet ersetzt. *z.B: k=3*

1	2	3	4	5	6	7	8	9	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	26
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w

Die Caesar Verschlüsselung ist sehr leicht zu knacken (Brute Force Attacke). Nur der Wert k muss herausgefunden werden (max 26 Versuche).
Diese Form der Verschlüsselung gehört zur

Monoalphabetischen Substitution:

jeder Buchstabe des Alphabets wird von einem anderen ersetzt, und zwar jedesmal mit dem selben – ein Buchstabe im Geheimtext repräsentiert einen einzigen Klartextbuchstaben (Geheimtextalphabet bleibt während der Verschlüsselung unverändert).

Generelles system:

Zufälliges Muster, Schlüssel ist die Tabelle:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
x	v	r	q	u	w	s	y	z	g	h	k	m	o	p	c	a	f	t	c	b	d	j	i	l	n

Jeder buchstabe wird durch IRGENDEINEN anderen ersetzt (scheint zufällig, nicht nur verschobenes Alphabet). Eine Brute Force Attacke würde hier sehr lange dauern, da es 26! Möglichkeiten (4*10^26) gibt.

Codewort System:

Kein zufälliges Muster sondern Tabelle ergibt sich aus Codewort und Schlüssel:
zB: Codewort = GEHEIMSCHRIFT, Schlüssel = der Buchstabe 'K'

Das Codewort wird ohne doppelt vorkommende Buchstaben erzeugt:

GEHEIMSCHRIFT = gehimscrft

Um das Geheimtextalphabet zu generieren, wird ab der Stelle 'K' das Codewort geschrieben. Das restliche Code Alphabet wird mit 'a' beginnend aufgefüllt, ohne den bereits im Codwort vorkommenden Buchstaben:

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
n o p q u v w x y z q e h i m s c r f t a b d j k l

Monoalphabetische Verschlüsselung knacken:

Statistische Daten über die Sprache können sehr hilfreich sein: manche Buchstaben und Buchstabenkombinationen kommen häufiger vor als andere. Zählt man die relative Häufigkeit jedes Buchstaben im codierten Text kann man auf die jeweiligen Klartext Buchstaben schließen. (häufigster Buchstabe, häufige Kombinationen/Trigramme,...)

Homophone Substitution:

Ein Verschlüsselungsverfahren, bei dem für jeden Klarbuchstaben mehrere Ersetzungsmöglichkeiten vorhanden sind. Entscheidend ist jedoch: Zwar mag es beispielsweise sechs mögliche Zeichen für den Buchstaben 'a' geben, doch diese Zeichen sind ausschließlich für diesen Buchstaben vorgesehen. Es handelt sich um eine Form der monoaphabetischen Verschlüsselung. Die Anzahl der Zeichen durch welche ein Buchstabe ersetz wird, ist abhängig von der Häufigkeit seines Vorkommens. Ziel ist die Verfälschung der statistischen Werte.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
09	78	48	13	45	25	39	65	83	51	84	22	58	71	95	29	35	40	76	49	61	89	28	21	52	66
12	92	81	41	79	23	50	68	88			27	59	91	94			42	86	69	63					
33			62	14		56	32	93			18		00				77	96	75	34					
47			01	16			70	15					05				80	17	85	60					
53			03	24			73	04					07				11	20	97						
67				44				26					54				19	30	08						
				46				37					72				36	43							
				55				58					90												
				64									99												
				74									38												
				82																					
				87																					
				98																					
				10																					
				31																					
				06																					
				57																					

6	2	2	5	17	2	3	5	8	1	1	3	2	10	2	1	1	7	7	6	4	1	1	1	1	1

% of frequency German

- **Transpositionsverfahren:**

Ein Verschlüsselungsverfahren, bei dem jeder Buchstabe innerhalb der Mitteilung seinen Platz wechselt, doch selbst unverändert bleibt.

Bsp: „Gartenzauntransposition"

Klartext: NAHT IHR EUCH WIEDER,SCHWANKENDE GESTALTEN

N H I R U H I D R C W N E D G S A T N
A T H E C W E E S H A K N E E T L E

--> NHIRUHIDRCWNEDGSATN ATHECWEESHAKNEETLE

Entschlüsselung durch Umkehrung des Verfahrens.

Bsp: „Skytale"

= ein Holzstab, um den ein Streifen Leder oder Pergament gewickelt wird. Der Sender schreibt die Nachricht der Länge des Stabes nach auf den Streifen und wickelt ihn dann ab --> sieht nach einer sinnlosen Aufreihung von Buchstaben aus. Der Empfänger kann die Nachricht mittels einer Skytale vom gleichen Durchmesser wieder lesbar machen.

Bsp: „Columnar Transposition"

Der Geheimtext wird durch einen Geheimwort erzeugt in dem, wie beim Codewort System, doppelt vorkommende Buchstaben nicht berücksichtigt werden. Dieser Schlüssel dient dazu, die Kolumnen zu nummerieren. Der Buchstabe des verbleibenden Ausdrucks, der am nächsten zum Alphabetstart liegt „nummeriert" die erste Kolummne . Der Klartext wird horizontal in die Kolummnen eingetragen. Liest man jede Spalte für sich, erhält man den Geheimtext.

	I	P	S	E	C	U	R
	3	4	6	2	1	7	5
cleartext →	I	W	A	N	T	T	O
	U	N	D	E	R	S	T
	A	N	D	A	L	L	P
	R	I	N	C	I	P	L
	E	S	O	F	I	N	T
	E	R	N	E	T	S	E
	C	U	R	I	T	Y	B
	E	F	O	R	E	I	W
	I	L	L	U	S	E	E
	B	A	N	K	I	N	G
					↑		
					ciphertext		

Columnar Transposition knacken:

Dieses Transpositionsverfahren kann erneut mit statistischen Daten über die Spracheigenschaften geknackt werden (Buchstabenvorkommen ist unverfälscht, nur die Anordnung ist anders). Man kann versuchen die Anzahl der Kolummnen zu erraten, und weiters, sie zu ordnen.

Vorgehensweise:
Wenn Wörter oder Phrasen aus dem Klartext erraten werden die länger als das Schlüsselwort

sind, könnte der Geheimtext entschlüsselt werden. (eventuell Wörter die man in Zusammenhang mit der Botschaft vermutet).

z.B.: wird das Wort „INTERNETSECURITY" erraten, müssten die Buchstaben in den Kolummnen bei einem Schlüssel der (geratenen) Länge 7 folgendermaßen angeordnet sein: IT, NS, TE, EC, RU, NR, EI, TT, SY:

1	**2**	**3**	**4**	**5**	**6**	**7**
E	Z	N	G	H	A	G
T	E	M	E	E	I	R
E	**N**	**I**	X	**R**	T	**T**
C	**S**	**T**	**N**	**U**	**E**	**E**
	Y	**T**	**R**		**I**	

Ist die Schlüssellänge erst herausgefunden, kann durch Untersuchen verschiedener Kolummnenkombinationen (auf erkennbare Fragmente des vermuteten Klartextes) die Reihenfolge bestimmt werden:

6	**4**	**3**	**1**	**7**	**2**	**5**
G	A	N	Z	G	E	H
E	I	M	E	R	T	E
X	T	**I**	**N**	**T**	**E**	**R**
N	**E**	**T**	**S**	**E**	**C**	**U**
R	**I**	**T**	**Y**			

- **Vigenère-Verschlüsselung:**

Eine um 1500 entwickelte ployalphabetische Verschlüsselung. Das Vigenère-Quadrat enthält 26 verschiedene Geheimtextalphabete, die gegeneinander caesar-verschoben sind. Mit einem Schlüsselwort wird festgelegt, welches Geheimtextalphabet für den jeweiligen Buchstaben des Klartextes verwendet wird.

Vigenere Coding Table

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a
2	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b
3	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c
4	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d
5	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e
6	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f
7	...																									
	...																									
22	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v
23	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w
24	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x
25	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y
26	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z

Beispielverschlüsselung nach dieser Tabelle:

Angenommenes Schlüsselwort: **BEWA**
Das Schlüsselwort markiert die zur Ver- und Entschlüsselung nötigen Zeilen: 1, 4, 22 und 26

verwendete Zeilen:	1	4	22	26	1	4	22	26	1	4	22	26	1	4	22	26
Klartext:	**T**	**O**	**P**	**S**	**E**	**C**	**R**	**E**	**T**	**A**	**N**	**D**	**C**	**O**	**N**	**F**

Zum Verschlüsseln des Textes, wird nun jeder Buchstabe in der darüber angegebenen Zeile nachgesehn und notiert:

T in der ersten Zeile ergibt 'u', O in der vierten Zeile ergibt 's', usw....

Geheimtext:	**u**	**s**	**l**	**s**	**f**	**g**	**n**	**e**	**u**	**d**	**j**	**d**	**d**	**s**	**j**	**f**

Vigenère Code knacken:

Im Geheimtext wird nach Wiederholungen gesucht, die enstehen wenn die selbe Buchstabenfolge im Klartext mit dem selben Teil des Schlüssels chiffriert wurde:

01	04	22	26	01	04	22	26	01	04	22	26	01	04	22	26	01	04
T	O	P	S	E	C	R	E	T	A	N	D	T	O	P	G	E	H
u	s	l	s	f	g	n	e	u	d	j	d	u	s	l	g	f	l

Nun wird versucht, den zum Lesen nötigen Schlüssel ausfindig zu machen:
Der Abstand von einer Wiederholung zur nächsten beträgt 12, d.h. der Schlüssel muss 12 ohne Rest teilen können.

Das heisst es bleiben 6 Möglichkeiten (1,2,3,4,6,12 – die Teiler von 12):

1. Der Schlüssel ist 1 lang und läuft 12 mal bis zur nächsten Wiederholung
2. Der Schlüssel ist 2 lang und läuft 6 mal bis zur nächsten Wiederholung
3. Der Schlüssel ist 3 lang und läuft 4 mal bis zur nächsten Wiederholung
4. Der Schlüssel ist 4 lang und läuft 3 mal bis zur nächsten Wiederholung
5. Der Schlüssel ist 6 lang und läuft 2 mal bis zur nächsten Wiederholung
6. Der Schlüssel ist 12 lang und läuft 1 mal bis zur nächsten Wiederholung

Die erste Möglichkeit kann ausgeschlossen werden, da es sich sonst „bloß" um eine monoalphabetische Verschlüsselung handelt.

Können noch weitere Wiederholungen im Geheimtext ausfindig gemacht werden, müssen die gemeinsamen Teiler der sich ergebenden Wdh-Abstände eruiert werden um die exakte Schlüssellänge herauszufinden.

Angenommen alle analysierten Wiederholungen deuten auf die Schlüssellänge 4 hin:
Durch Betrachtung des 1., 5., 9., 13.....n+4.... Buchstaben (alle mit dem ersten Zeichen des Codewortes verschlüsselt), und dem Einsatz der Häufigkeitsanalyse, kann das Geheimtextalphabet erschlossen werden. Das Ergebnis ist eine Häufigkeitsverteilung des Geheimtextalphabets, das ähnliche Merkmale wie das Klartextalphabet aufweist, nur um einige Stellen caesar-verschoben. Durch Vergleich des Ergebnisses mit statistischen Sprachwerten kann der erste Buchstabe des Codewortes ausfindig gemacht werden, und somit die erste Zeile des Vigenère-Quadrats. Es folgt das gleiche Prozedere für den 2. 3. und 4. Buchstaben....

- **One-Time-Pad:**

System:

Die Schlüssellänge entspricht der Klartextlänge.
Der aktuelle Buchstabe des Schlüsselwortes (zur Zeilenangabe im V-Quadrat) wird zufällig generiert (zufällige „sinnlose" Schlüsselwörter sind mittels Häufigkeitsanalyse nicht zu knacken). --> **sehr hohe Sicherheit**!
Die Schlüssel Tabelle wird One-Time-Pad genannt, da jeder Schlüssel (früher jeweils auf einer Seite eines Blocks notiert) nur einmal verwendet und danach vernichtet wird. Leider ist die Übermittlung des Schlüssels sehr unpraktisch.

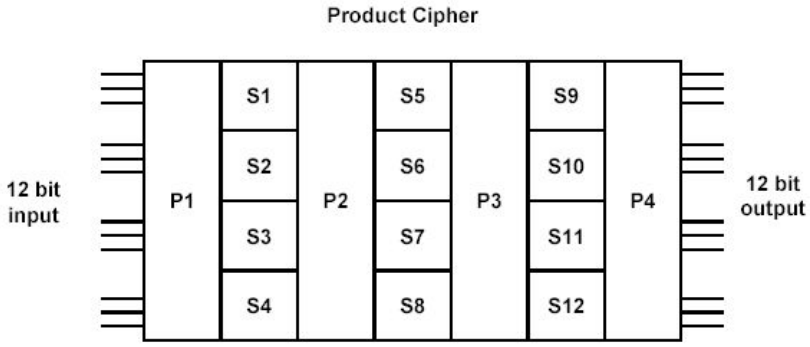
One Time Pad digital:

Es wird ein zufälliger Bit String als Schlüssel erzeugt (wiederum genauso lang wie der Bit String der Klartext Nachricht). Diese beiden Bit Strings wird nun Exclusive OR verknüpft. Die totale Zufälligkeit (KEINE Pseudozufallsfolgen!) des Geheimtextes garantiert absolute Sicherheit.

Unpraktisch: Schlüssel kann sich niemand merken --> muss aufgeschrieben werden und so an den Empfänger vermittelt werden. Ausserdem ist die Klartextlänge an die Schlüssellänge gebunden.

• Product Ciphers:

= Kombination von mehreren unsicheren Verschlüsselungsverfahren wie Substutution, Transposition,...in Form von P-Boxen (Permutationsboxen) und S-Boxen (Substitutionsboxen). Nach mehrmaligen Durchläufen ist ein Geheimtext entstanden, der resistent gegen Angriffe ist.

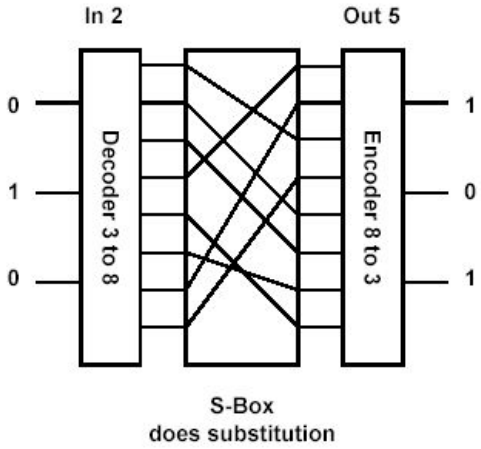


S-Boxen:

Eine S-Box ist eine Substitutionsoperation, bei der eine m-stelligen Binärzahl durch eine n-stellige Binärzahl ersetzt wird. Sie kann beispielsweise mit einer Tabelle implementiert werden, die 2^m Zeilen enthält. Jede 6 Bit Eingabe wird zu einer 4 Bit Ausgabe. Bit 1 und 6 wählen Substitutionsfunktionen (Zeilen) der Box aus. Bit 2-5 wählen Spalte der Box aus. Der Eintrag der Box wird mit 4 Bit kodiert.

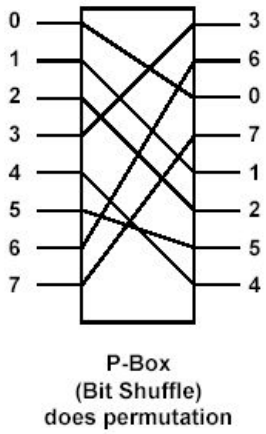
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0:	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7
1:	0	F	7	4	E	2	D	1	A	6	C	B	9	5	3	8
2:	4	1	E	8	D	6	2	B	F	C	9	7	3	A	5	0
3:	F	C	8	2	4	9	1	7	5	B	3	E	A	0	6	D

Substitutionsbox S1 des DES-Algorithmus >>



P-Boxen:

In der Permutations-Box werden die Bits nach einer festgelegten Permutationsordnung (Tabelle) umsortiert. Der so erhaltene Wert wird mit der linken Hälfte des Blocks XOR-verknüpft und stellt die neue rechte Hälfte des gesamten Blockes dar. Die neue linke Hälfte bildet die vorhergehende rechte Hälfte. Die Anzahl der Bits wird nicht verändert.



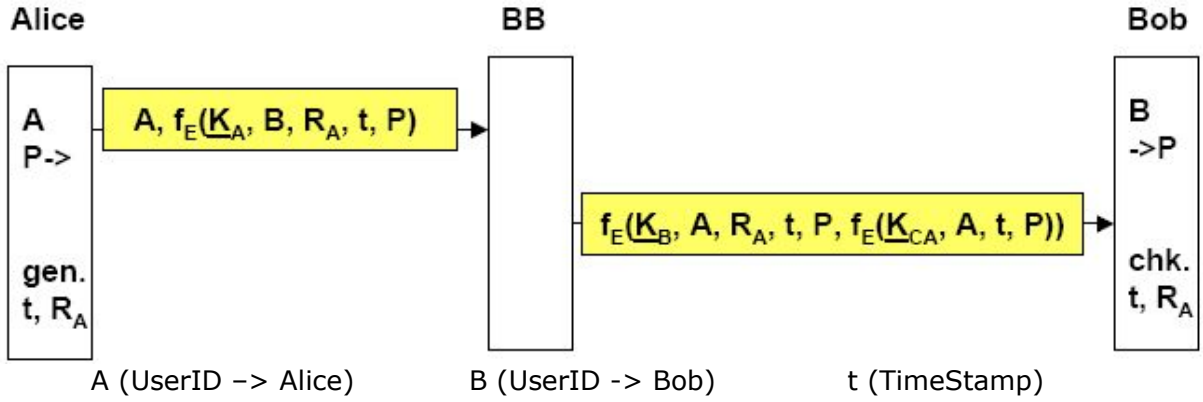
Was versteht man unter Digitaler Signatur prinzipiell? Wie könnte man das mittels

Secret-Key Technik machen? Warum macht man das nicht in der Realität? Wie kann man das mit Public-Key Technik alleine machen? Warum macht man das nicht in der Realität? Wie bildet man DS tatsächlich und warum (MD, Timestamp)? Welche Rolle spielt eine CA in diesem Zusammenhang? Was ist ein Zertifikat? Geben Sie ein Beispiel für die prinzipielle Vorgehensweise? [90 Punkte]

Digitaler Signatur prinzipiell:

- Stellt sicher, dass der Absender einer Nachricht auch tatsächlich der Verfasser ist, dass der Inhalt nicht nachträglich verändert wurde. Weiters kann der Absender dadurch nicht bestreiten diese Nachricht verschickt zu haben. Sie wird angebracht, in dem eine Prüfsumme der Nachricht mit dem Private-Key des Absenders verschlüsselt wird. Der Empfänger muss den Public Key des Absenders kennen, um die Signatur überprüfen zu können.-.

- **Secret-Key Technik:**



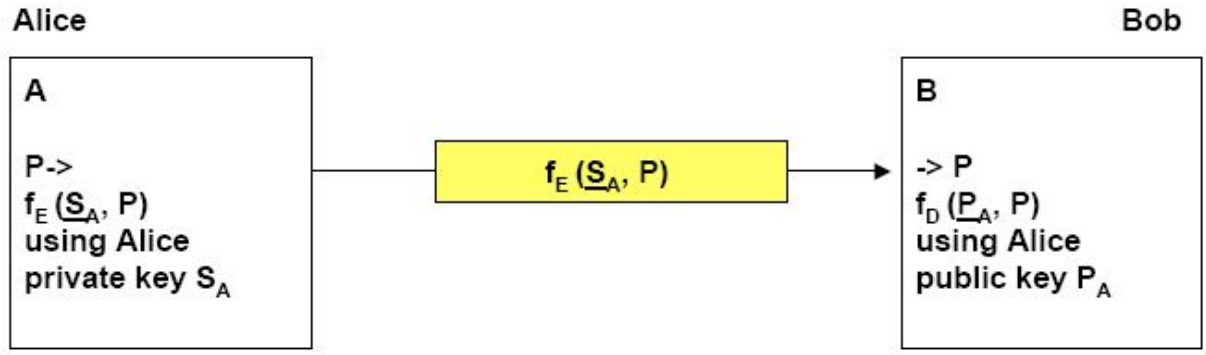
A (UserID –> Alice) B (UserID -> Bob) t (TimeStamp)

B_B als verlässlicher Dritter speichert einen Secret-Key/User (K_A für Alice und K_B für Bob). Alice sendet die Nachricht P verschlüsselt mit ihren geheimen Schlüssel K_A und signiert die Nachricht (A, t, P) mit geheimen Schlüssel K_{CA}, welcher später bei Gericht verwendet werden kann um sicherzustellen, dass Alice wirklich diese Nachricht gesendet hat. (wirklich getan durch Dekodierungstätigkeit von B_B selbst, auf Antrag des Gerichtes) B_B leitet die Nachricht verschlüsselt mit Bob's geheimem Schlüssel weiter.

Warum macht man das nicht in der Realität?

1. Big Brother Problem von BB (kann alles lesen), man muss jemanden wirklich vertrauen

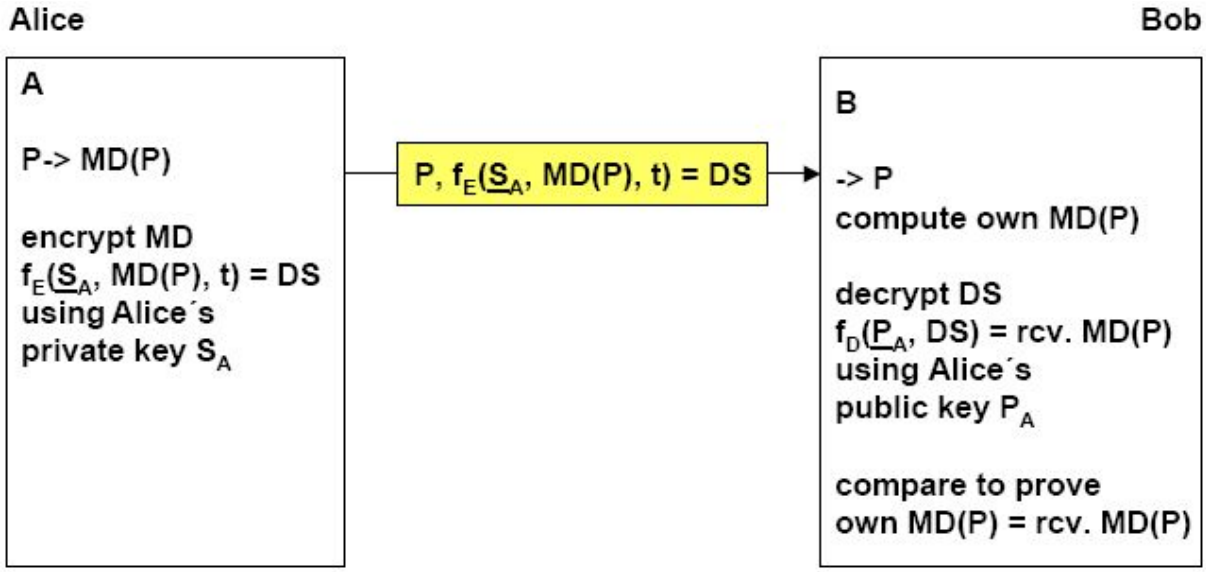
- **Public-Key Technik alleine**



Warum macht man das nicht in der Realität?

2. Nachricht von Alice zu Bob kann entschlüsselt werden von jedem der den Public Key von Alice hat! Performance vom RSA-Verfahren ist das zweite und wesentliche Problem(zu langsam für lange Nachrichten).

- **DS bilden, (MD, Timestamp)**



Digitale Signatur(DS) prüft die Authentifizierung, weil nur Alice ein solches Message Digest (MD) produziert haben kann. DS erlaubt auch die vollständige Überprüfung. Es ist unmöglich P zu verändern und einen gültigen MD zu kreieren, weil nur Alice den Private-Key weiß.Die Zeitstempel sind zum Verhindern von replay Angriff.

Alice schickt ihre Nachricht, einen mit ihrem Private-Key verschlüsselten Message Digest von ihrer Nachricht und einen ebenfalls verschlüsselten TimeStamp an Bob.
Bob erzeugt seinen eigenen Message Digest von der Nachricht welchen er mit dem von Alice erzeugten vergleicht. Diesen erhält er durch die Entschlüsselung mit dem Public-Key von Alice.

CA in diesem Zusammenhang

CA= Certification Authority

Um die Public-Key Signatur zu verwenden, um diese Probleme zu lösen:
◊ man benötigt eine glaubwürdige Autorität
◊ wo Schlüsseländerungen und die veränderten Daten gespeichert werden
◊ wo Public Key's hinterlegt und unterzeichnet werden können
◊ wo Public Key's wieder aufgehoben werden können
  - ähnlich Rücknahmeliste bei Kreditkarten

• Zertifikat

Ist eine Art "Ausweis". Es enthält Informationen über die Identität einer Person und Signaturprüfdaten, mit deren Hilfe Signaturen dieser Person zugeordnet werden können. Das Zertifikat ist durch die elektronische Signatur seines Ausstellers vor Veränderungen geschützt.

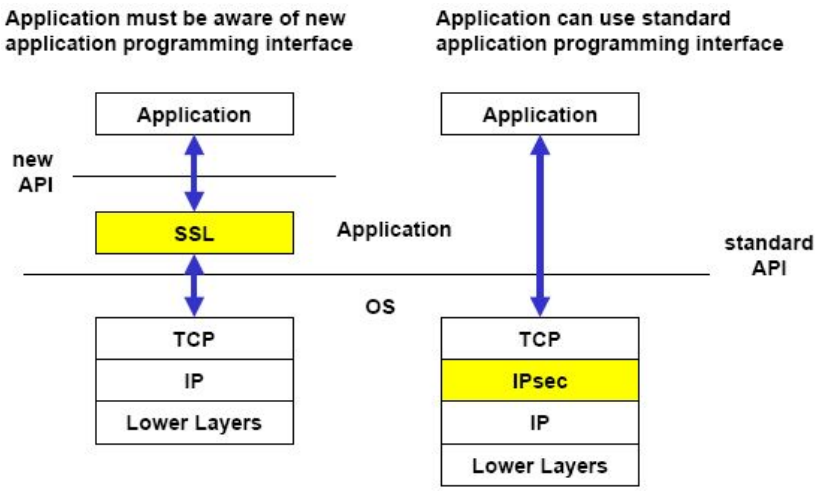
Beispiel für Vorgehensweise

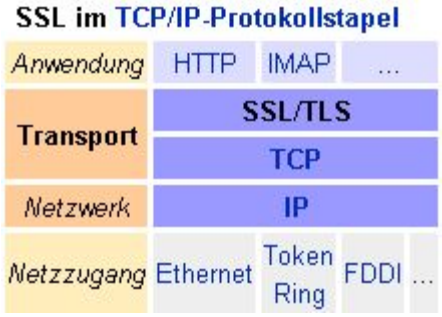
Digitale Zertifikate zwischen Alice und Bob
- Bob hat seine Public Key von CA signiert und hat ein Zertifikat von seinem Schlüssel
- P_{Cert} = Public Key von der Zertifizierungsauthorität CA muss manuell konfiguriert werden oder in die Anwendungssoftware im Endsystem vom Überprüfungssystem bei Alice implementiert werden
- Um zu überprüfen ob Bob den Public Key(unterzeichnet von CA) gesendet hat -> P_{Bob} + $f_E(_{SCert}$, $P_{Bob})$
  - f_E (S_{Cert} , P_{Bob}) is Digital Certificate of P_{Bob} = DC (P_{Bob})
- Alice überprüft
  - If f_D (P_{Cert} , DC (P_{Bob})) = received P_{Bob}
- Jetzt kann Alice einen verteilten symmetrischen Schlüssel K_{AB} Bob schicken
  - Indem sie seinen validierten empfangenen Public Key P_{Bob} verwendet
  - Nur Bob kann den verteilten symmetrischen Schlüssel K_{AB} entschlüsseln
- Verkehr zwischen Bob und Alice kann jetzt verschlüsselt werden, indem man den geteilten Schlüssel K_{AB} verwendet
- Diese Technik wird vom SSL Handshake Protocol verwendet
  - Secure Socket Layer
  - Layer between TCP and application
  - Included in WEB browser to perform encryption HTTP session
  - SSL erfunden und eingeführt von Netscape
  - RFC version of SSL called Transport Layer Security (TLS)

Wo ist SSL angesiedelt und was wird abgedeckt? In welchen Phasen läuft SSL prinzipiell ab? Wozu dient die Session- ID? Welche Arten der Session Key Generierung gibt es? Beschreiben Sie für die RSA Methode und die Ephemeral DH Methode im Detail [130 Punkte]

Wo ist SSL angesiedelt?

SSL liegt zwischen der Application Layer und TCP (Transportschicht):





Im OSI-Modell ist SSL oberhalb der Transportschicht (z.B. TCP) und unter Applikationsprotokollen wie HTTP oder SMTP angesiedelt.

• In welchen Phasen läuft SSL prinzipiell ab?

- **parameter negotiation between client and server**
  session key generation method, authentication method and encryption algorithms to be used for data transfer phase
- **mutual authentication of client and server**
  client authentication may be optional
- **session key building and activation of cipher suite**
  integrity key and encryption key

> Secure connection can then be used for the actual data transfer
– protected by session keys build during establishment

• Wozu dient die Session- ID?

– Session keys werden während dem Aufbau generiert und stellen sicher, dass der aktuelle Datentransfer sicher ist
– Client und Server tauschen mit diesem *session key* verschlüsselte Nachrichten aus und signalisieren damit ihre Kommunikationsbereitschaft.

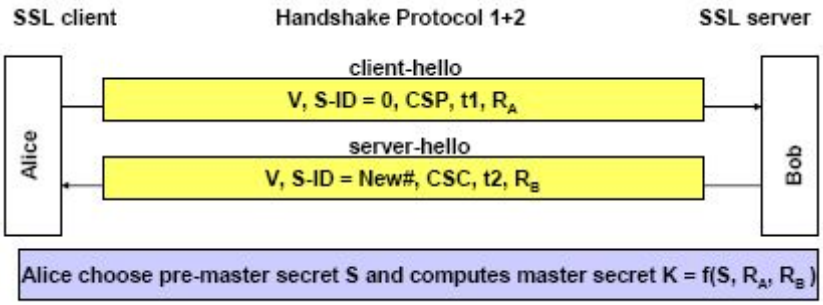
• Welche Arten der Session Key Generierung gibt es?

Four methods for session keys generation:

- **RSA**
  - shared secret S encrypted with public-key of partner
- **Fixed DH key exchange**
  - fixed public-DH value contained in DC (certificate)
  - session keys are based on the same base parameters
- **Ephemeral DH key exchange**
  - actual public-DH value signed with private-key of sender
    - best protection because every session will have a completely different set of generated keys
- **Anonymous DH key exchange**
  - basic DH key exchange without signatures and certificates
    - no protection against man-in-the-middle-attack

- **Die RSA Methode und die Ephemeral DH Methode im Detail**

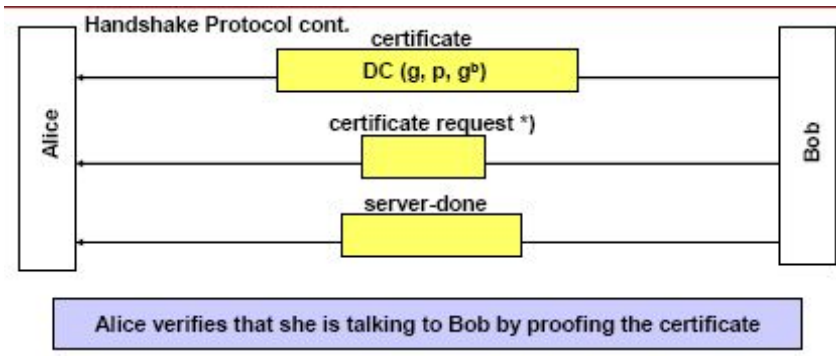
Phase 1: Protocol Procedures All Modes



V ... SSL version number
S-ID ... Session-Identifier (0 for new session, if unequal 0 then resuming a former established session -> phase 2 and 3 are not used
CSP ... cipher suite proposal of client
CSC ... cipher suite chosen by server
R_A, R_B ... random, number
t1, t2 ... timestamps

Beschreibung:
Alice wählt einen Premaster-Key S und berechnet daraus den Master-Key K = f(S, RA, RB)
Die Übertragung erfolgt hier noch unverschlüsselt
CSP ... Vorschlag über verfügbare Verfahren
CSC ... Server wählt eines aus CSP

Protocol Procedures Phase 2 - Fixed DH



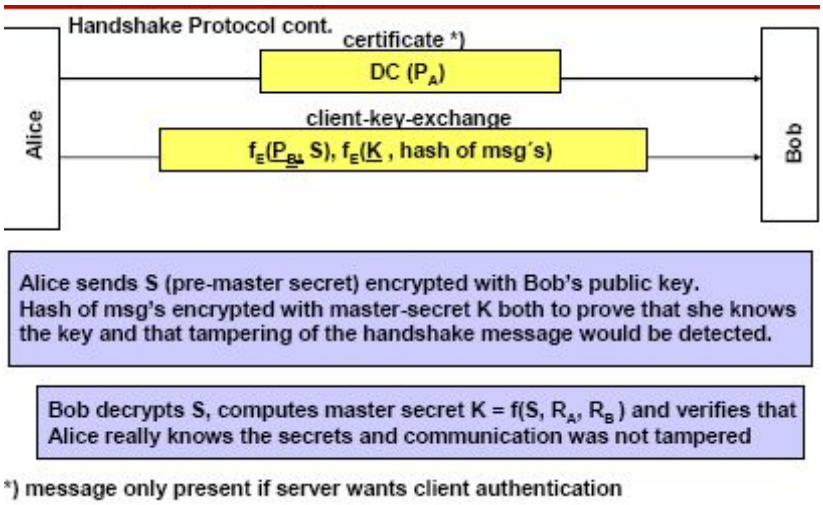
DC = g, p, g^b + $f_E(S_{Cert}$, g, p, g^b) ... Digital Certificate of Bob´s Public DH values g^b and g, p and chain of certificates to reach a well-known root-CA if necessary

*) message only present if server wants client authentication

*) Message erfolgt nur wenn der Server eine Authentifizierung des Client will

Beschreibung:
Bob schickt ein Zertifikat seines Public-Key, unterschrieben von einer Zertifizierungsstelle
$DC = P_B + f_E(S_{Cert} , P_B)$
Alice prüft anhand des Zertifikates ob sie mit Bob spricht

Protocol Procedures Phase 3 - RSA

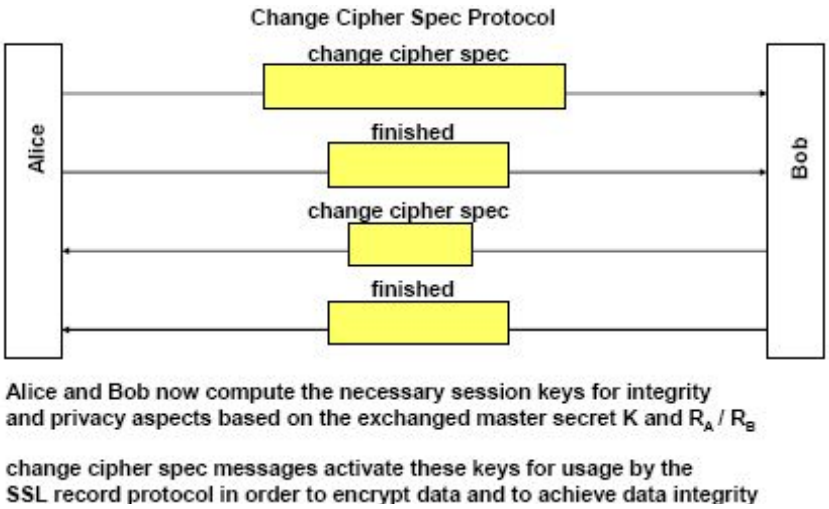


*) message only present if server wants client authentication

Beschreibung:
Alice schickt den Premaster-Key S verschlüsselt mit dem Public-Key von Bob
und einen Hash der Message verschlüsselt mit dem Master-Key K um zu beweisen, dass sie die Schlüssel kennt, und damit ein Verfälschen der Nachricht bemerkt würde.

Bob entschlüsselt S mit seinem Private-Key, berechnet den Master-Key K = f(S,RA,RB), überprüft ob Alice das Geheimnis weiß, und verifiziert ob die Nachricht nicht gefälscht wurde.

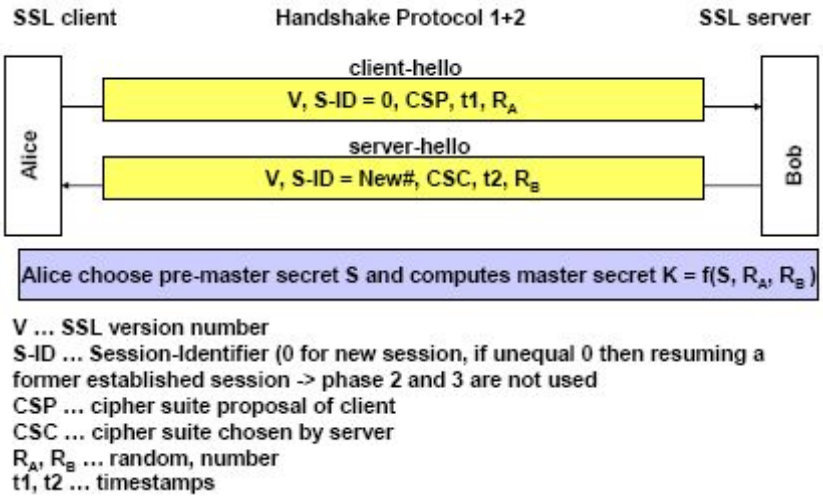
Protocol Procedures Phase 4 - All Modes



Beschreibung:
Alice und Bob berechnen basierend auf dem Master-Key K und RA,RB die Session Keys für Integrität und Geheimhaltung (Privacy)
Messages (change cipher spec messages) aktivieren die Schlüssel für das SSL Record Protokoll um Daten zu entschlüsseln und Integrität zu wahren.

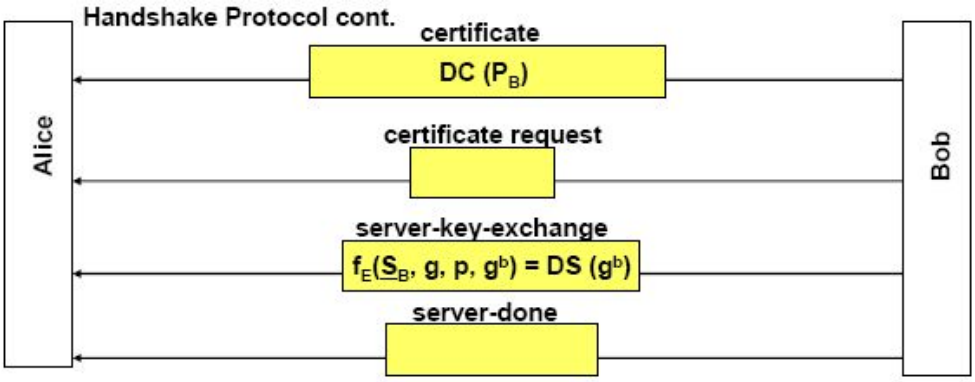
- ***Ephemeral DH Methode***

Phase 1: Protocol Procedures All Modes (s.o.)



Beschreibung:
Alice wählt einen Premaster-Key S und berechnet daraus den Master-Key K = f(S, RA, RB)
Die Übertragung erfolgt hier noch unverschlüsselt
CSP ... Vorschlag über verfügbare Verfahren
CSC ... Server wählt eines aus CSP

Empheral DH-Phase2 :



DC = P_B + $f_E(S_{Cert}, P_B)$ … Digital Certificate of Bob´s Public Key P_B and chain of certificates to reach a well-known root-CA (certification authority) if necessary

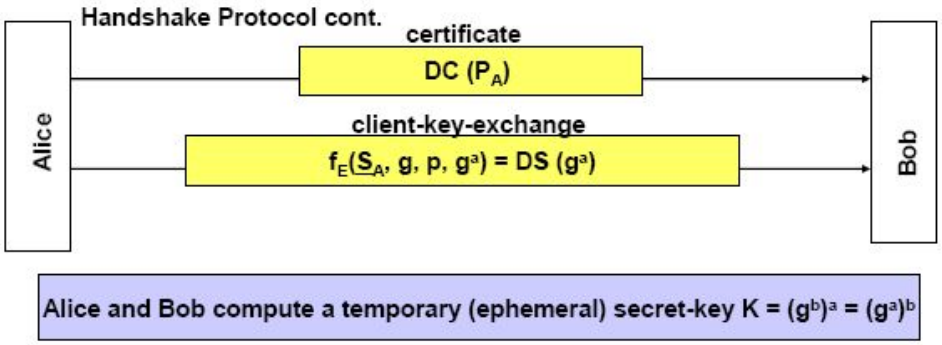
S_B = Bob´s private-key

DS = Digital Signature -> (g, p and Public-DH value g^b) signed with Bob´s private-key S_B

Beschreibung:

Bob schickt ein Zertifikat seines Public-Key, unterschrieben von einer Zertifizierungsstelle
DC = P_B + $f_E(S_{Cert}, P_B)$
und
eine digitale Signatur DS (public DH Werte), verschlüsselt mit seinem PublicPrivate-Key S_B

Empheral DH-Phase3 :



DC = P_A + $f_E(S_{Cert}, P_A)$ … Digital Certificate of Alice´s Public Key P_A and chain of certificates to reach a well-known root-CA (certification authority) if necessary

S_A = Alice´s private-key

DS = Digital Signature -> (g, p and Public-DH value g^b) signed with Bob´s private-key S_B

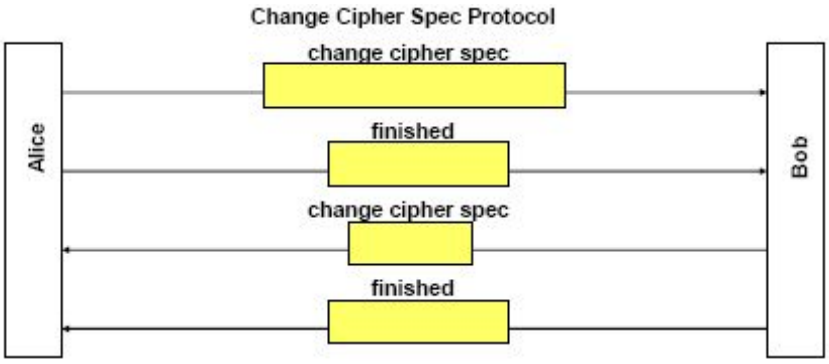
Beschreibung

Alice schickt ihrerseits ein Zertifikat ihres Public-Key, unterschrieben von einer

Zertifizierungsstelle
$DC = P_A + f_E(S_{Cert}, P_A)$
und
eine digitale Signatur DS (public DH Werte), verschlüsselt mit ihrem PublicPrivate-Key S_A

Alice und Bob berechnen temporäre Secret-Keys $K=(g^b)^a = (g^a)^b$

Protocol Procedures Phase 4 - All Modes



Alice and Bob now compute the necessary session keys for integrity and privacy aspects based on the exchanged master secret K and R_A / R_B

change cipher spec messages activate these keys for usage by the SSL record protocol in order to encrypt data and to achieve data integrity

Beschreibung:
Alice und Bob berechnen basierend auf dem Master-Key K und RA,RB die Session Keys für Integrität und Geheimhaltung (Privacy)
Messages (change cipher spec messages) aktivieren die Schlüssel für das SSL Record Protokoll um Daten zu entschlüsseln und Integrität zu wahren.

Erklären Sie zunächst das Grundprinzip der symmetrischen Verschlüsselung und gehen Sie auf die Methode DES näher ein. Was ist die Grundidee von DES? Was spielt sich prinzipiell ab (Rounds, Subkeys)? Wie erfolgt die Entschlüsselung? Was sind die Sicherheitsaspekte von DES? Was ist Triple-DES (3DES)? [70 Punkte]

Symmetrische Verschlüsselung:

Ein symmetrisches Kryptosystem ist ein Kryptosystem, das im Gegensatz zu einem asymmetrischen Kryptosystem **denselben Schlüssel zur Ver- und Entschlüsselung** einer Nachricht verwendet

DES:

= Data Encryption Standard
besteht aus einfachen Operationen, deshalb einfach in HW realisierbar

Bei DES handelt es sich um einen symmetrischen Algorithmus, das heißt zur Ver- und Entschlüsselung wird derselbe Schlüssel verwendet. DES funktioniert als Blockchiffre, das heißt jeder Block wird unter Verwendung des Schlüssels einzeln chiffriert, wobei die Daten in 16 Runden von Substitution und Permutation "verwürfelt" werden. Die Blockgröße beträgt 64 Bits, das heißt ein 64-Bit-Block Klartext wird in einen 64-Bit-Block Geheimtext transformiert. Auch der Schlüssel, der diese Transformation kontrolliert, besitzt 64 Bits. Jedoch stehen dem Benutzer von diesen 64 Bits nur 56 Bits zur Verfügung; die übrigen 8 Bits (jeweils ein Bit aus jedem Byte) werden zum Paritäts-Check (Erkennung fehlerhafter Information) benötigt. Die wirkliche Schlüssellänge beträgt daher nur 56 Bits. Die Entschlüsselung wird mit dem gleichen Algorithmus durchgeführt wobei die einzelnen Rundenschlüssel in umgekehrter Reihenfolge verwendet werden.

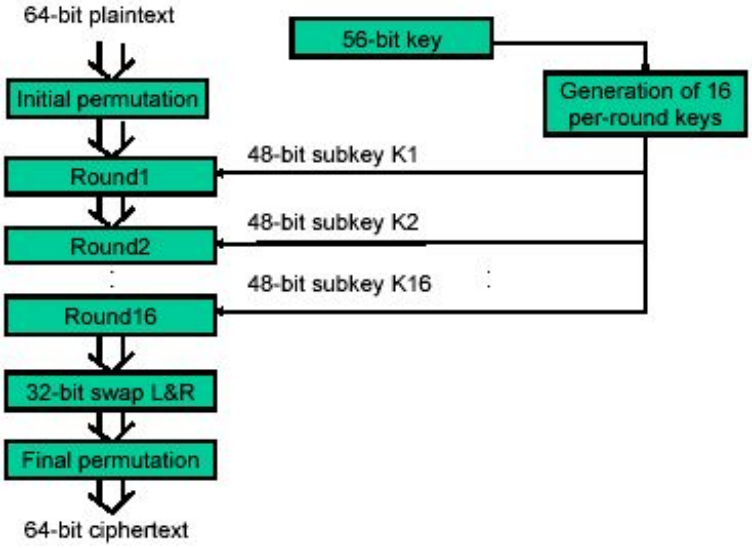
• Schritte:

Die Sicherheit des Verfahrens beruht auf einer Kombination von Permutationen und Substitutionen. Die Ver- bzw. Entschlüsselung eines 64-Bit-Blocks kann grob in die folgenden drei Schritte zerlegt werden:

1. Der Eingabeblock wird der Eingangspermutation unterworfen. Hierdurch wird die Reihenfolge der Bits verändert (Transposition). Das Ergebnis wird in zwei 32-Bit-Register geschrieben. Mit den 64 Bit des Schlüssels wird ebenso verfahren, nachdem die 56 relevanten Bits bestimmt worden sind, werden diese ebenfalls transponiert und in zwei 28-Bit-Register geschrieben.
2. In diesem Schritt werden 16 Mal die gleichen Rechenschritte wiederholt (Runden), wobei in jeder Runde der Schlüssel neu bestimmt wird:
    - Zunächst werden die Bits der "Schlüsselregister" zyklisch um ein bzw. zwei Bit verschoben und 48 der 56 Bit als Rundenschlüssel bestimmt.
    - Das "rechte" Datenregister (R-Block) wird mittels einer Expansion von 32 auf 48 Bit vergrößert.
    - Daten- und Schlüsselblock werden durch logisches XOR miteinander kombiniert.
    - Das Resultat wird in 6 Bit große Abschnitte aufgeteilt und durch acht S-Boxen (Substitutionsboxen) gesandt, die hieraus wiederum 32 neue Bits erzeugen.
    - Zum Schluss werden der Block nochmals einer Transposition unterzogen.

Der so erzeugte 32-Bit-Datenblock wird mittels XOR mit den Daten des "linken" Datenregisters verknüpft. Das Ergebnis dieser Operationen bildet den neuen Inhalt des rechte Registers, wobei der alte R-Block zuvor ins linke Register geschoben wird.

3. Nach der 16. Runde werden das linke und rechte Register wieder zu einem 64-Bit-Block zusammengefügt, bevor die zur Eingangspermutation inverse Ausgangspermutation angewendet wird. (Da Eingangs- und Ausgangspermutation zu einander invers sind und nur zu Beginn und am Ende durchgeführt werden, haben sie auf die kryptografische Sicherheit des Verfahrens keinen Einfluss.)



- **Triple-DES:**

Mit Triple-DES wurde der Versuch unternommen, die Sicherheit des Verfahrens weit genug zu erhöhen, um den Zeitraum bis zur Verfügbarkeit eines besseren Verfahrens zu überbrücken. Triple-DES ist nichts weiter, als dass eine dreimalige Verschlüsselung mit DES vorgenommen wird, dabei werden zwei unterschiedliche Schlüssel eingesetzt.

Was wird mit den Betriebsarten DES-ECB und DES-CBC realisiert? Warum benötigt man diese bei Block Ciphers? Geben Sie eine Skizze des DES-CBC Modes für Verschlüsselung und Entschlüsselung an. Was ist ein IV im Zusammenhang mit DES-CBC? Was sind Stream Ciphers? Wozu werden Sie benötigt? Geben Sie in einer Skizze den CFB Mode für Verschlüsselung und Entschlüsselung an.

◊ Was wird mit den Betriebsarten DES[1]-ECB[2]und DES-CBC[3] realisiert?

Beide Betriebsarten werden verwendet, um Nachrichten zu verschlüsseln, die größer sind als 64 bit. Dabei wird die Nachricht in 64-bit-Blöcke zerteilt, wobei jeder einzelne dieser 64 bit langen Blöcke mittels DES verschlüsselt wird (ECB – Electronic Code Book-Mode). Bei CBC (Cipher-Block-Chaining) handelt es sich insofern um eine Erweiterung von ECB, als dass die einzelnen Blöcke zueinander in Beziehung gesetzt werden (zueinander in Abhängigkeit gestellt werden).

◊ Skizze des DES-CBC Modes für Verschlüsselung und Entschlüsselung

Bei CBC handelt es sich um eine Methode, um ein paar der Probleme, welche bei ECB auftreten zu vermeiden, indem man vermeidet, dass 2 identische Plaintext-Blöcke zu 2 identischen Ciphertext-Blöcken werden. – Die grundlegende Idee besteht darin, Zufallszahlen in ECB einzuführen. Das Problem dahinter: Wie bekomme ich dieselben Zahlen für die Entschlüsselung? – Die Lösung: Durch Hinzufügen eines Feedbacks zur verschlüsselten Nachricht:

- Bevor der aktuelle Plaintext-Block verschlüsselt wird, wird er mit dem

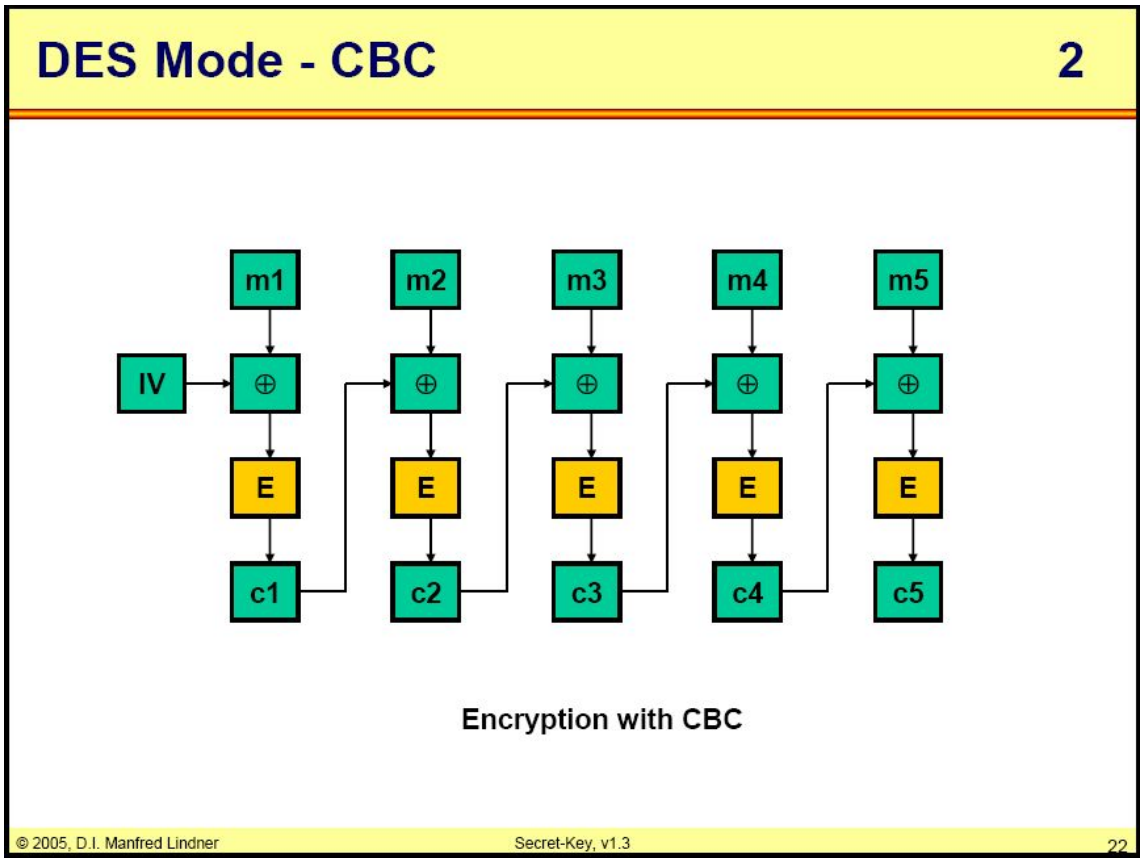
[1] Digital Encription Standard
[2] Electronic Codebook Mode
[3] Cipher Block Chaining

vorangegangenen Cipher-Block xor-verknüpft.

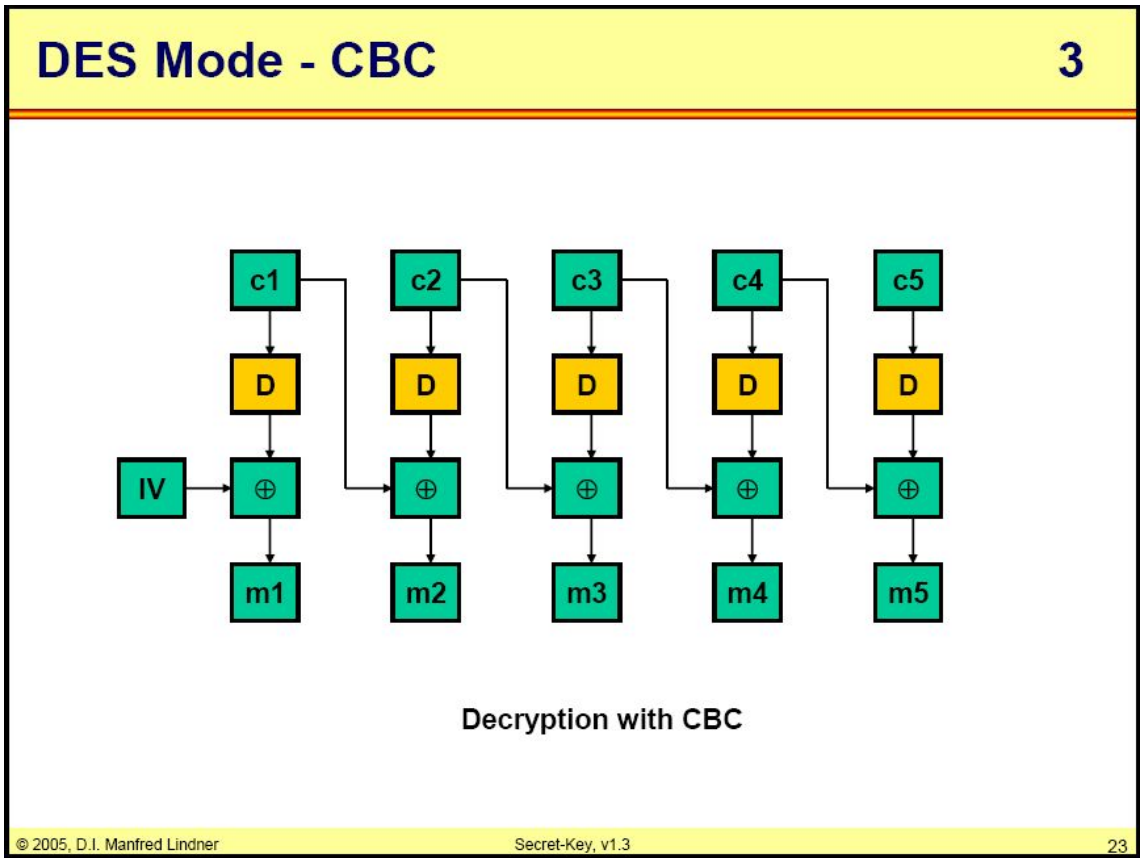
- Für den ersten Block wird ein IV (Initialisierungsvektor) verwendet (Zufallswert, um Block-Replays zu vermeiden)
- Der IV muss dem Empfänger gegeben werden, bevor dieser mit der Entschlüsselung startet (beginnen kann).

Zum IV ist folgendes zu sagen: Normalerweise wird der gerade aktuelle Plaintext-Block vor der Verschlüsselung mit dem vorangegangenen Cipher-Block xor-verknüpft (siehe dazu auch die Skizze!). Zu Beginn existiert jedoch noch kein vorangegangener Cipher-Block, weswegen anstelle eines solchen ein InitialisierungsVektor verwendet wird!

DES-CBC beim Verschlüsseln:



DES-CBC beim Entschlüsseln:



◊ **Was ist ein IV im Zusammenhang mit DES-CBC?**

Siehe c)!

◊ **Was sind Stream Ciphers? Wozu werden Sie benötigt?**

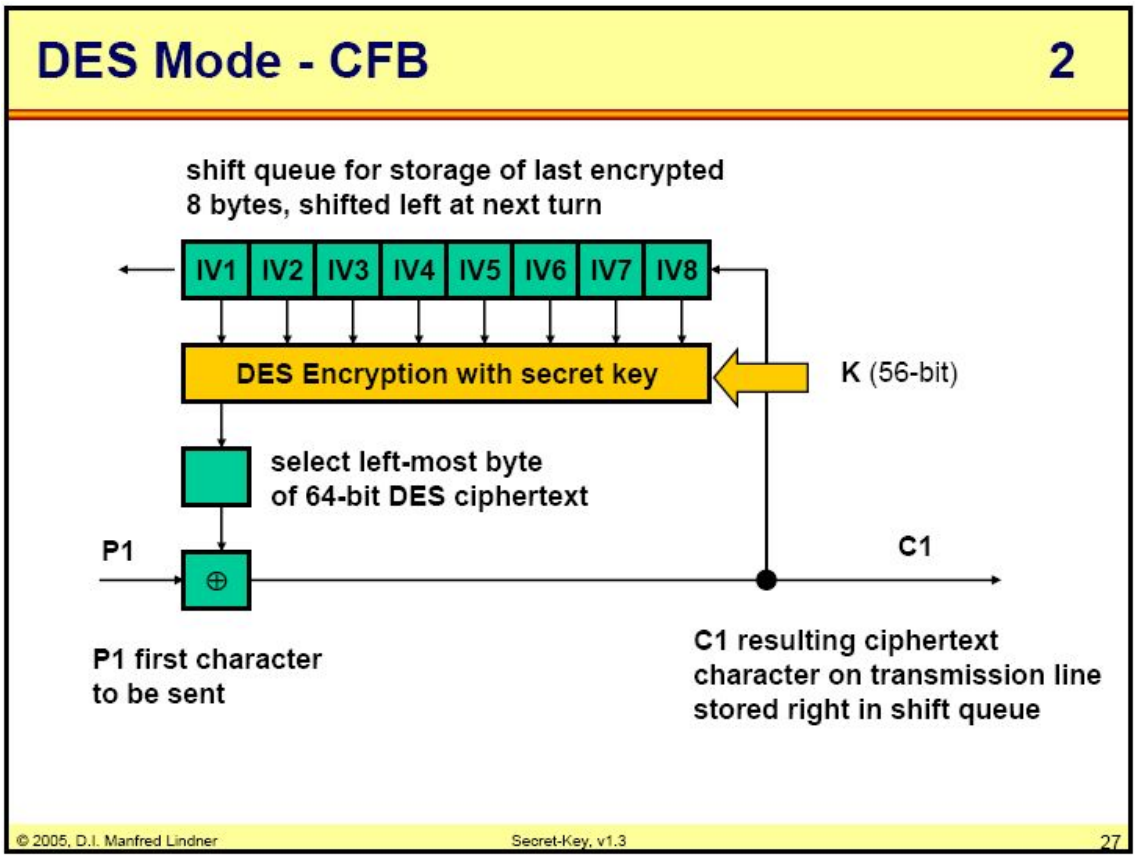
Stream Ciphers werden benötigt, um Nachrichten zu verschlüsseln, die kleiner sind als 64 bit. – Cipher Block Chaining hat den Nachteil, dass mit der Entschlüsselung erst nach Erhalt eines ganzen 64 bit-Block begonnen werden kann, was es ungeeignet für interaktive Terminals macht, wo in der Regel weniger als 8 Zeichen geschrieben werden und auf Antwort gewartet wird.

Stream Ciphers sind in der Lage, eine Byte-by-Byte-Verschlüsselung durchzuführen.

Details > Siehe die folgenden Skizzen für Ver- und Entschlüsselung!

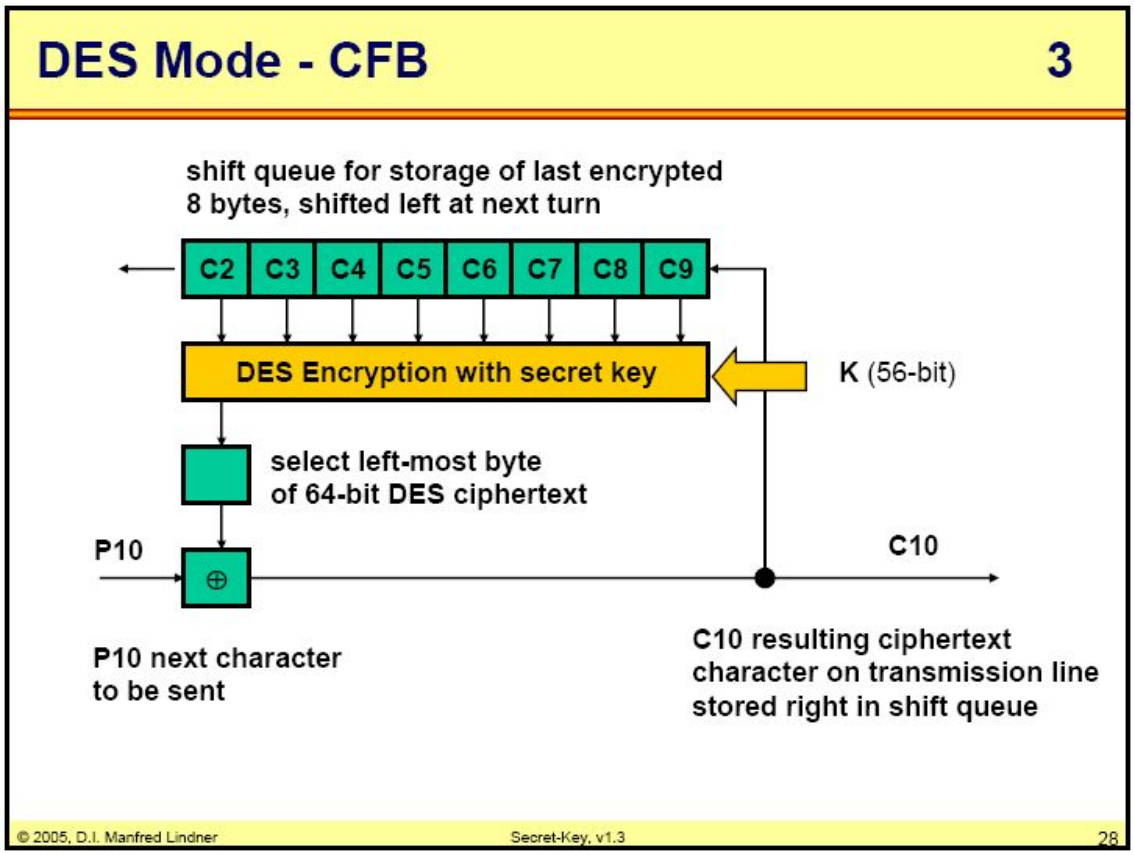
◊ **Skizze: CFB[4] Mode für Verschlüsselung und Entschlüsselung**

Verschlüsselung (für das 1. Plaintext-Zeichen – P1):

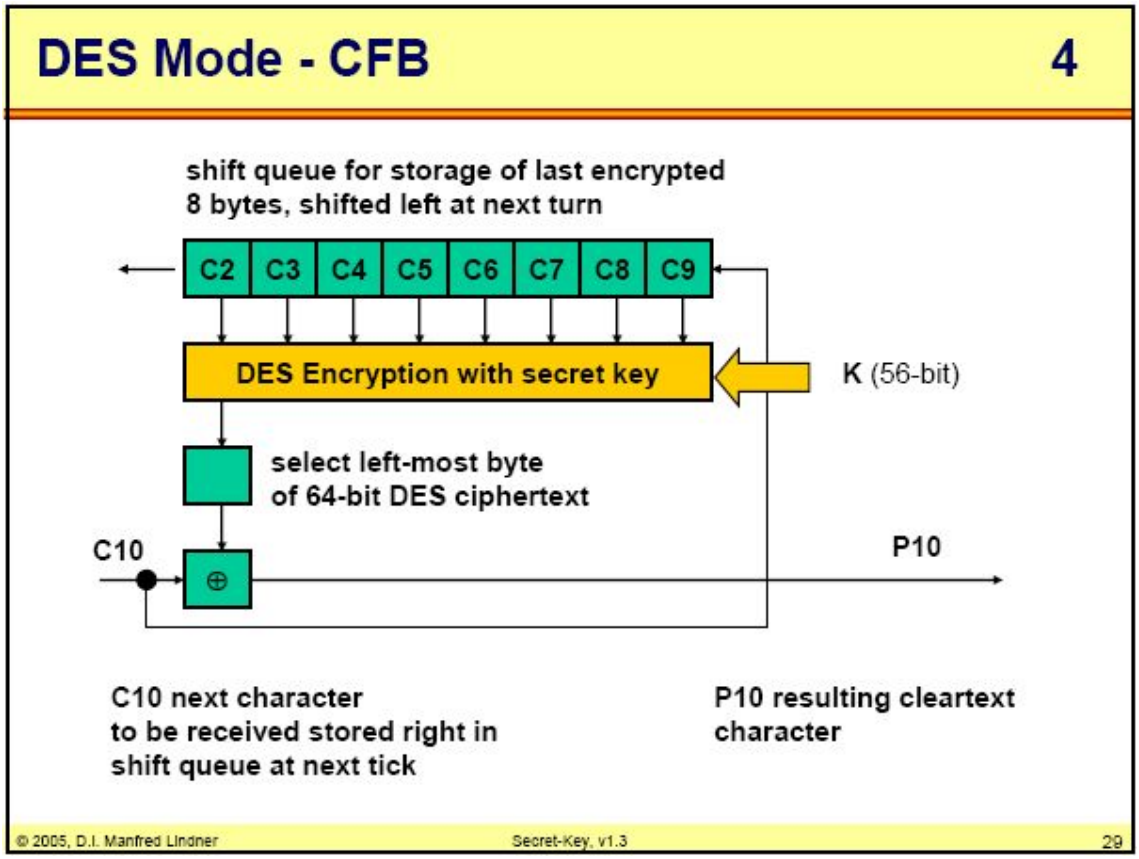


[4] CFB - Cipher Feedback Mode

Die Verschlüsselung des 10. Plaintext-Zeichnes (P10):



Die Entschlüsselung des 10. Ciphertext-Zeichens (C10):



Die Entschlüsselung verläuft auf ähnliche Weise:

- Bei C10 handelt es sich um das 10. Ciphertext-Zeichen, welches zuvor vom Sender über die Leitung geschickt wurde und jetzt beim Empfänger einlangt
- Bevor C10 entschlüsselt wird, gelangt eine Kopie davon zunächst in die Queue, erst dann erfolgt die Entschlüsselung, woraufhin aus C10 das Plaintext-Zeichen P10 wird (Auch hier wird wieder die gesamte Shift-Queue mittels DES Block-Cipher verschlüsselt wird, woraus ein 64-bit Ciphertext entsteht (orangener Balken) > die 8 ganz linken Bits dieses 64-bit Ciphertextes werden mit dem 1-Byte Ciphertext-Zeichen (C10) xor-verknüpft > Daraus resultiert wiederum das entsprechende Plaintext-Zeichen (also P10)

Erklären Sie das DH Verfahren im Detail. Wozu dient es? Welche prinzipielle Gefahr gibt es dabei? Schildern Sie diese. [70 Punkte]

Diffie Hellman

Ein wichtiger Fortschritt war die Veröffentlichung des Artikels „New Directions in Cryptography" (http://citeseer.nj.nec.com/340126.html) durch Whitfield Diffie und Martin Hellman. Dieser Artikel stellte eine radikal neue Methode der Schlüsselverteilung vor, welche als Public Key Kryptographie bekannt wurde. Dies löste eines der fundamentalen Probleme der Kryptographie, die Schlüsselverteilung.

Vor dieser Entdeckung waren die Schlüssel symmetrisch, und der Besitz eines Schlüssels erlaubte sowohl das Verschlüsseln als auch das Entschlüsseln einer Nachricht. Daher musste

der Schlüssel zwischen den Kommunikationspartner über einen sicheren Weg ausgetauscht werden, wie beispielsweise einem vertrauenswürdigen Kurier oder dem direkten Treffen der Kommunikationspartner. Diese Situation wurde schnell unüberschaubar, wenn die Anzahl der beteiligten Personen an stieg. Auch wurde ein jeweils neuer Schlüssel für jeden Kommunikationspartner benötigt, wenn die anderen Teilnehmer nicht in der Lage sein sollten die Nachrichten zu entschlüsseln. Dieses System wird als Private Key Kryptographie oder Shared Secret bezeichnet.

Der Diffie-Hellman-Schlüsselaustausch ist kein Verschlüsselungsverfahren, sondern beschreibt die Möglichkeit, Schlüssel sicher über unsichere Kanäle auszuhandeln. Hierbei handelt es sich um Schlüssel, wie sie in der Kryptographie verwendet werden.

- **Funktionsweise**

Zwei Kommunikationspartner wollen über ein unsicheres Medium, etwa eine Kabel- oder Funkleitung, verschlüsselt kommunizieren. Sie wollen ein symmetrisches Kryptosystem einsetzen. Dazu benötigen beide jedoch zunächst einen Schlüssel, den sie über den Diffie-Hellman-Schlüsselaustausch austauschen.

1. Die Kommunikationspartner einigen sich zunächst auf eine Primzahl *p* und eine Primitivwurzel $g \bmod p$ mit $2 \leq g \leq p-2$. Diese Parameter müssen nicht geheim bleiben, können also insbesondere auch über ein unsicheres Medium übertragen werden.
2. Beide Kommunikationspartner erzeugen jeweils eine geheim zu haltende Zufallszahl *a* bzw. *b* aus der Menge {0,...,*p* - 2}. *a* und *b* werden nicht übertragen, bleiben also dem jeweiligen Kommunikationspartner, aber auch potenziellen Lauschern unbekannt.
3. Die Kommunikationspartner berechnen $A = g^a \bmod p$ bzw. $B = g^b \bmod p$. Nun werden *A* und *B* über das unsichere Medium übertragen.
4. Die Kommunikationspartner berechnen nun $B^a \bmod p = (g^b \bmod p)^a \bmod p = (g^a \bmod p)^b \bmod p = A^b \bmod p = K$. Das Ergebnis K ist für beide Partner gleich und kann als Schlüssel für die weitere Kommunikation verwendet werden.

- **Sicherheit**

Die Sicherheit des Verfahrens basiert auf der Verwendung einer so genannten Einwegfunktion. Hierbei macht man sich die Tatsache zunutze, dass es zwar sehr einfach ist, eine Zahl zu potenzieren, dass es jedoch nur mit sehr großem Aufwand möglich ist, den diskreten Logarithmus einer Zahl zu berechnen. In der Praxis werden sehr große Primzahlen verwendet, die Sicherheit kann weiter erhöht werden, wenn *g* = (*p* - 1) / 2 ebenfalls eine Primzahl ist (g ist dann eine Sophie-Germain-Primzahl (http://de.wikipedia.org/wiki/Sophie-Germain-Primzahl)).

Potenzielle Lauscher erfahren zwar *p*, *g*, *A* und *B*. *a* und *b* bleiben hingegen unbekannt.

Das Diffie-Hellman-Problem, $K = g^{ab} \bmod p$ zu berechnen, ist lösbar, wenn man diskrete Logarithmen $\bmod p$ berechnen kann. Eine andere Lösungsmethode ist nicht bekannt.

Das Verfahren gilt als sicher gegen passives Abhören ("Eavesdropping"), da die im Klartext übertragenen Informationen nicht zur Konstruktion des Schlüssels K ausreichen. Wenn der Angreifer jedoch aktiv Datenpakete abfangen und verändert weiterschicken kann, besteht die Möglichkeit des Man-In-The-Middle-Angriffs: Der Angreifer M vereinbart mit Partner A den Schlüssel K_{AM} und mit Partner B den Schlüssel K_{MB}, wobei A und B davon ausgehen, der Austausch fände mit dem berechtigten Kommunikationspartner statt. Auch die folgende symmetrisch verschlüsselte Kommunikation wird über den Angreifer M umgeleitet. Daten von A

an B werden von A mit K_{AM} verschlüsselt und können von M gelesen werden, bevor sie mit K_{MB} verschlüsselt an B weitergeschickt werden (und umgekehrt). Dabei kann der Nachrichteninhalt sogar unbemerkt verändert werden. Diesem Problem kann durch elektronische Unterschrift und ähnliche Maßnahmen erfolgreich begegnet werden.

Was versteht man unter MAC? Schildern Sie die Methode DES-CBC-Residue in diesem Zusammenhang? Was ist Hashing (Message Digest) und warum wird Hashing anstelle von DES-CBC-Residue eingeführt? Was versteht man unter HMAC? [80 Punkte]

Message Authentication Code (oder Message Integrity Code – MIC)

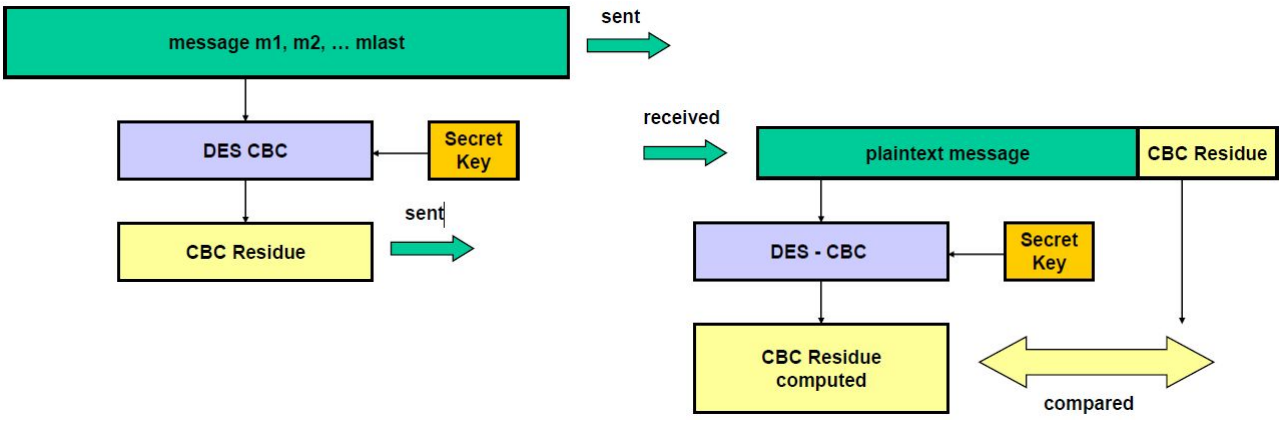
Ein Message Authentication Code (MAC) dient zur Sicherung der Integrität (Unverfälschtheit) einer Nachricht. Ein MAC wird erzeugt, indem die Nachricht mit einem geheimen, symmetrischen Schlüssel verknüpft wird und auf diese Verknüpfung eine Hashfunktion angewandt wird. Dann wird die Nachricht im Klartext zusammen mit dem MAC verschickt. Der Empfänger kann den MAC auf die gleiche Weise wie der Absender berechnen, da er ebenfalls Kenntnis vom geheimen Schlüssel hat. Stimmt der berechnete MAC mit dem mitgeschickten überein, wurde die Nachricht nicht verändert. Ein Angreifer kann die Nachricht zwar ändern, da er aber den geheimen Schlüssel nicht kennt, kann er keinen gültigen MAC für die veränderte Nachricht erstellen. Dadurch lässt sich eine nachträgliche Änderung erkennen.

MACs dienen nur sehr beschränkt zur Sicherung der Authentizität. Da mindestens zwei Personen den geheimen Schlüssel kennen, kann im nachhinein nicht festgestellt werden, von welchem die Nachricht stammte (im Gegensatz zu digitalen Signaturen).

Hashfunktionen, die zur Erzeugung von MACs benutzt werden sind z.B. MD5 und SHA-1.

- **Cipher Block Chaining Mode**

Mittels CBC kann man einen MAC erzeugen, genauer gesagt wird hier DES CBC im Block Mode angewandt.
Dieses Verfahren wird auf den Plaintext angewendet, dazu verwendet der Sender den eigenen Secret Key (welcher dem Empfänger auch bekannt ist). Dadurch wird dann ein Ciphertext für den letzten Plaintext Block, CBD Residue genannt, erzeugt. Bei diesem Ciphertext handelt es sich um einen 64Bit Ciphertext. Der Plaintext wird dann gemeinsam mit dem CBC Residue versendet. Der Empfänger kann dann mittels des erhalten Plaintextes und seines Secret Keys den CBC Residue Teil berechnen. Vergleicht er diesen mit dem erhalten CBC Code kann er bei nicht übereinstimmen der beiden erkennen, dass jmd. die Nachricht verändert haben muss.



- **Hash Funktion**

(Der wohl bekanntest Vertreter ist MD5.)

Bei einer Hash-Funktion geht es allgemein darum, eine lange Eingabe (zum Beispiel einen Text) in eine kurze Ausgabe (den Hash-Wert des Textes) zu verwandeln.

Das ist etwa dann sinnvoll, wenn man zwei große ähnliche Dateien vergleichen will: Anstatt die 25 Seiten eines Textes durchzusehen, ob auch wirklich jeder Buchstabe gleich ist, schauen wir auf die kurzen Hash-Werte der beiden Dokumente und sehen sofort, ob diese beiden gleich oder verschieden sind.

Beschreiben Sie die Chellange/Response Technik für Mutual Authentication mittels Secret-Key im Detail.

Chellange Response Technik:

A,B sind die User-ID's von Alice und Bob.

Die Zufallszahlen Ra und Rb (siehe nachfolgende Skizzen)werden aus einem großen Bereich gewählt. Es ist sehr unwahrscheinlich, dass ein Intruder (Trudy) R_B oder R_A, und die entsprechenden Antworten von einer früheren session(replay attack) gesehen hat.

Die

- **Mutual Authentication**

kann auf 2 Arten durchgeführt werden – entweder mit 5 oder mit 3 Messages.

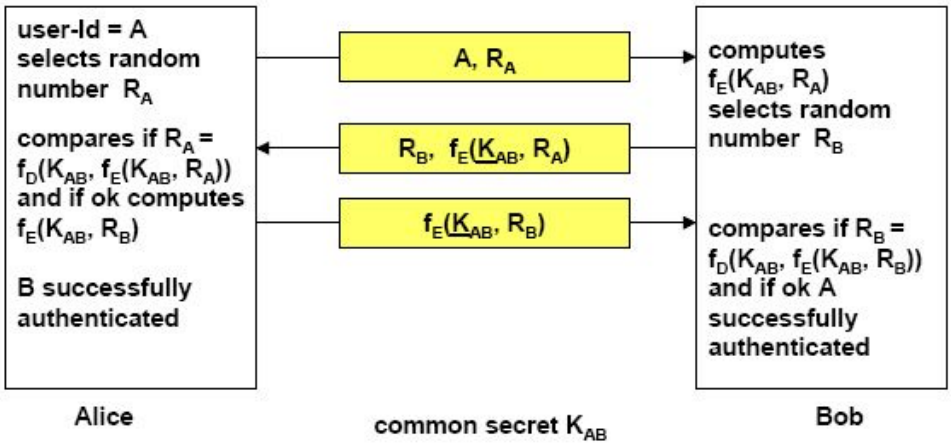
a) 3 Messages

Alice und Bob haben einen gemeinsamen secret key (K_{AB}).
Alice sendet eine Zufallsnummer (R_A) an Bob(heißt nonces=number used only once).

Dieser verschlüsselt die Zufallszahl mit K_{AB} und sendet das Ergebnis($f_E(K_{AB}, R_A)$) + einer von ihm generierten Zufallszahl (R_B) zurück an Alice.

Alice entschlüsselt nun das Ergebnis $f_E(K_{AB}, R_A)$ (=die verschlüsselte Zufallszahl), die sie von Bob erhalten hat mit K_{AB}.
Dann vergleicht sie das Ergebnis mit ihrer Zufallszahl und falls die beiden übereinstimmen, ist Bob als Bob authentifiziert.

Alice verschlüsselt nun die Zufallszahl von Bob (R_B) mit K_{AB} und schickt das Ergebnis $f_E(K_{AB}, R_B)$ an Bob.
Bob entschlüsselt nun wie Alice zuvor das Ergebnis mit K_{AB} und vergleicht sie mit seiner R_B. Sind sie identisch, ist Alice als Alice authentifiziert.



b) 5 Messages

Alice und Bob haben wieder einen gemeinsamen secret key (K_{AB}).
Alice sendet in einem ersten Schritt aber nur ihre Id (A) an Bob.

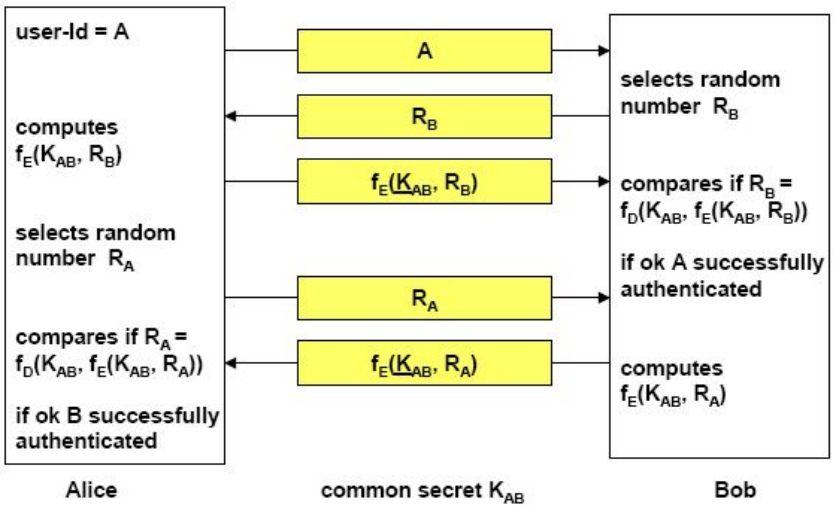
Danach sendet Bob eine von ihm generierte Zufallsnummer (R_B) an Alice zurück.

Alice verschlüsselt nun die Zufallszahl von Bob mit K_{AB} und sendet das Ergebnis $f_E(K_{AB}, R_A)$ an Bob.

Bob entschlüsselt nun $f_E(K_{AB}, R_A)$ mit dem K_{AB} und vergleicht ob das Ergebnis $f_D(K_{AB}, f_E(K_{AB}, R_B))$ gleich seiner Zufallszahl ist. Ist sie gleich, weiß Bob, dass Alice auch wirklich Alice und nicht z.B. Trudy ist.

Nun möchte aber auch Alice wissen, ob sie es wirklich mit Bob zu tun hat und sendet eine von ihr generierte Zufallszahl an Bob.

Bob verschlüsselt nun R_A mit K_{AB} und sendet das Ergebnis($f_E(K_{AB}, R_A)$) zurück an Alice.
Sie entschlüsselt das ganze wieder und vergleicht es mit R_A, sind die Zahlen gleich, sind beide authentifiziert.



Welches Problem entsteht beim abgekürzten Verfahren mit nur 3 Messages? Schildern Sie dieses?

- **Problem mit 3 Messages:**

Hier existiert eine Security Pitfall (Sicherheitsfalle), die als Reflection Attack bekannt ist:

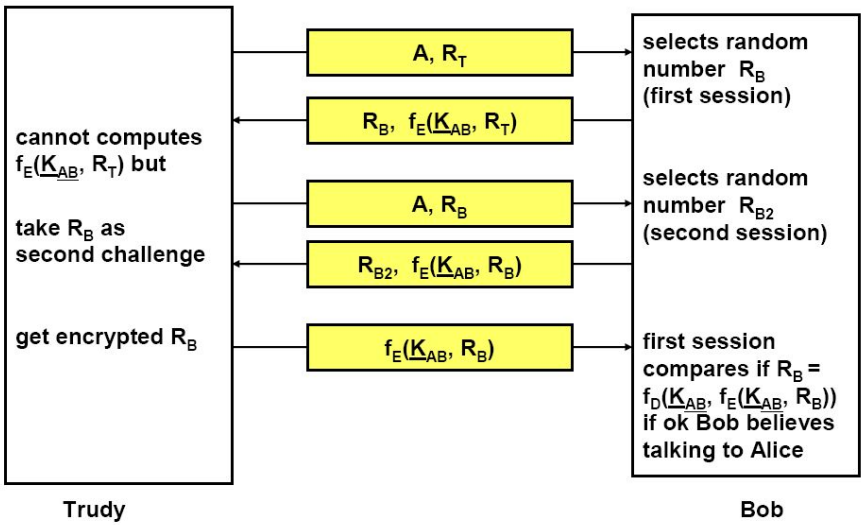
Ein Intruder (Trudy) schickt gibt sich als Alice aus und schickt die Id (A) + eine von Trudy generierte Zufallszahl R_T an Bob.

Bob öffnet durch diese Anfrage nun eine 1. session und schickt seine Zufallszahl R_B und die mit K_{AB} verschlüsselte Zufallszahl von Trudy) an Trudy zurück.

Diese schickt nun eine zweite SessionEröffnungsnachricht an Bob, wieder mit der Id von Alice, aber diesmal mit der Zufallszahl von Bob R_B.

Dadurch öffnet Bob eine 2. session und sendet eine 2. von ihm generierte Zufallszahl R_{B2} + die Verschlüsselung $f_E(K_{AB}, R_B)$ von seiner eigenen 1. Zufallszahl R_B an die vermeintliche Alice zurück.

Trudy erhält nun die verschlüsselte Zufallszahl $f_E(K_{AB}, R_B)$ und sendet sie postwendend an Bob zurück, der nun Trudy in seiner 1. session als Alice authentifiziert.
Die 2. session wird nie beendet.



Was kann man mit einem KDC bewerkstelligen? Schildern sie die Abläufe mittels Ticket

- **KDC**

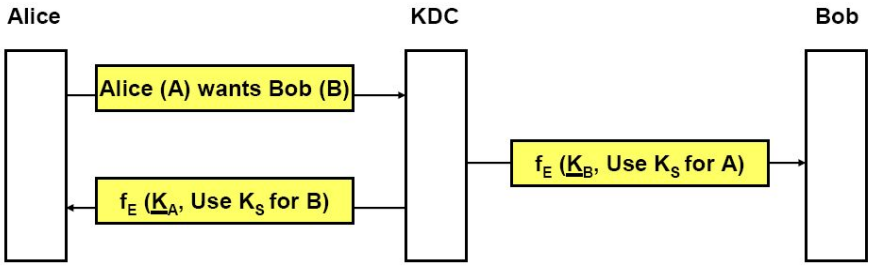
Eine Authentifizierung mittels Key Distribution Center (KDC):
Das Key Distribution Center ist eine 3rd party, der man vertrauen kann. Dort wird für jeden User ein Secret Key gespeichert (K_A für Alice und K_B Bob).

Authentication mit Hilfe eines KDC (Basic):
Alice möchte mit Bob kommunizieren und wendet sich mit A, B an das KDC.

Das KDC wählt nun einen Session Key K_S, den sie mit dem Secret Key von Bob K_B verschlüsselt; danach sendet das KDC das Ergebnis f_E (K_B, K_S +Id von Alice) an Bob.

Das KDC sendet danach auch den Session Key K_S, verschlüsselt mit dem Secret Key von Alice K_A, an Alice = f_E (K_A, K_S + Id von Bob).

Nun können Alice und Bob eine gegenseitige Authentifizierung mit dem erhaltenen Session Key K_S durchführen.



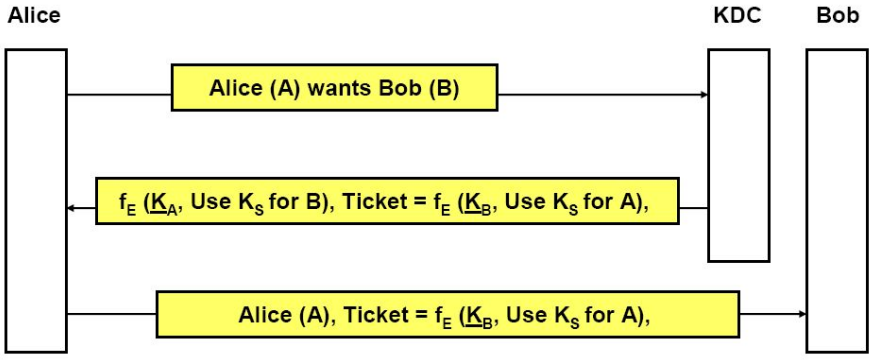
Authentication mit Hilfe eines KDC (Ticket):

Alice wendet sich wieder an das KDC, mit A und B.

Das KDC wählt auch hier einen Session Key K_S. Das KDC sendet nun 2 Dinge an Alice zurück:

- K_S mit dem Secret Key von Alice K_A verschlüsselt f_E (K_A, K_S für Bob)
- ein **Ticket** = enthält den Session Key und die Id von Alice mit dem Secret Key von Bob verschlüsselt

Alice sendet das erhaltene Ticket nun direkt an Bob. Alice und Bob können nun mit ihren Keys den Session Key bzw. Ticket entschlüsseln und eine gegenseitige Authentifizierung mit Hilfe des Session Keys durchführen.



KDC selects a session key K_S sent it encrypted with K_A to Alice together with a so called Ticket

Ticket itself contains session key K_S and user A both encrypted with K_B

Now Alice and Bob can perform mutual authentication with obtained K_S

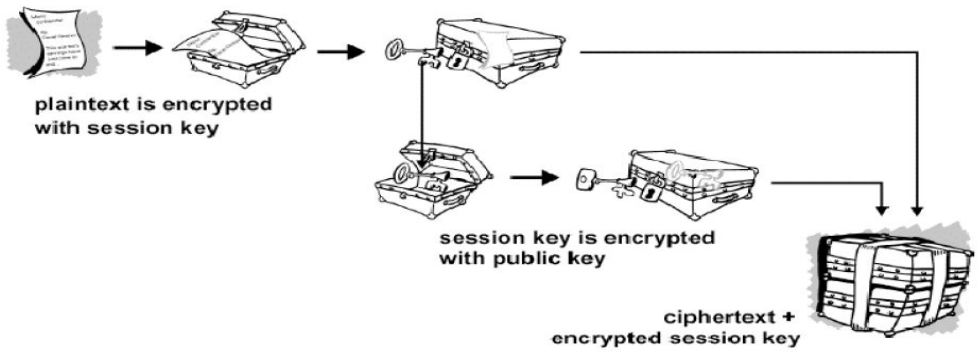
Beschreiben Sie das Grundprinzip von PGP. Wo und wie wird es eingesetzt. Beschreiben Sie im Detail, in welchen Schritten eine PGP-Nachricht beim Sender generiert wird bzw. beim Empfänger überprüft und entschlüsselt wird. Wie erfolgt die Verwaltung der Schlüsseln bei PGP? Beschreiben Sie die Abläufe und die Speicherplätze im Detail?

Was ist PGP (pretty good privacy):

Ein E-Mail-Security-Paket, das *Privacy, Authentification, Digital Signature* und *Kompression* enthält.

- **Grundprinzip:**

Plaintext wird mit dem Session-Key verschlüsselt > Session-Key wird mit Public-Key verschlüsselt > verschlüsselter Text und verschlüsselter Key werden gesendet.



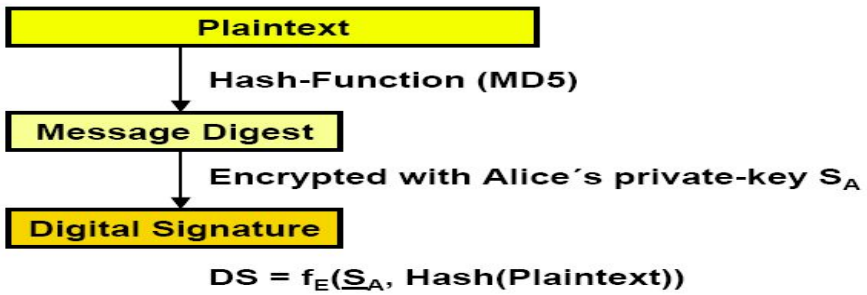
- **Wo und wie wird PGP eingesetzt:**

???? E-Mail-Security

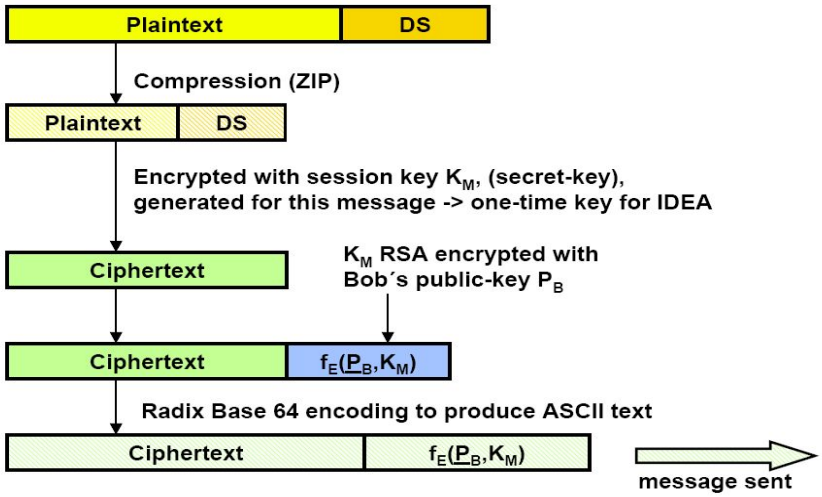
- **Generierung einer PGP Nachricht beim Sender:**

Authentication + Integrity

1. Generierung der digitalen Signatur:

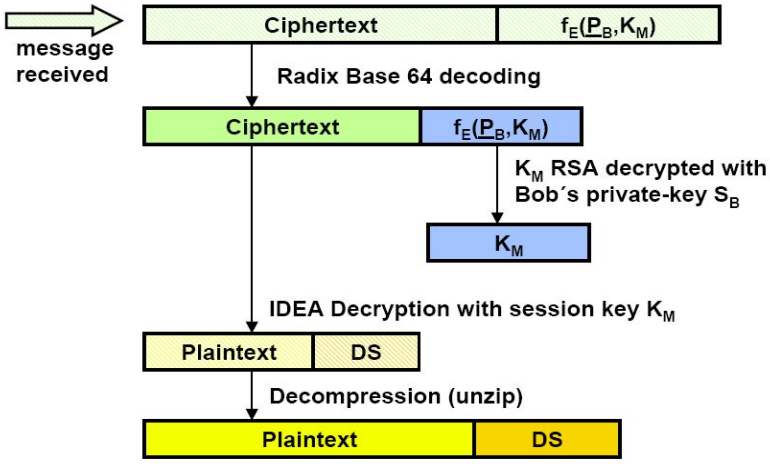


2. Privacy

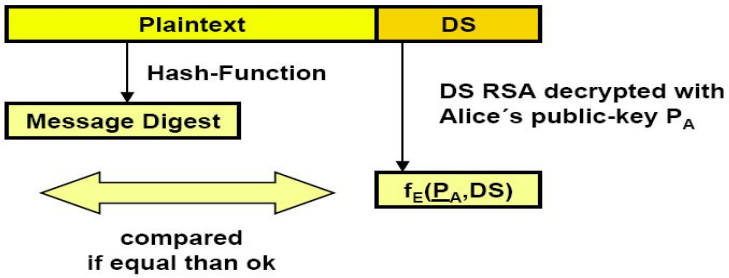


- **Empfang:**

1. Entschlüsselung:



2. Integrity check + identity:



- **Verwaltung der Schlüssel:**

◊ Nach der **Installation von PGP** auf Alice´s Computer wird ein RSA public/private Schlüsselpaar generiert.
Speicherung der Schlüssel:

- *Public-Key*:
  Wird in einer Datenstruktur („public-key ring) gespeichert (User-ID von der Alice + Key-ID [= least significant 64 bits des Public-Keys]).
- *Private-Key*:
  Wird im „Private-Key-Ring" zusammen mit der User-ID und der dazugehörigen User-ID in verschlüsselter Form gespeichert.
  - Alice wird um „pass-phrase" gefragt um Zugang zu ihrem private-key zu bekommen > nach der Verwendung wird dieser sofort aus dem Speicher des Computers (Alice) gelöscht.

- **Private Key Schutz:**
  Alice´s „pass-phrase": Wird verwendet um eine 128-bit MD5 Nachricht zu generieren welche als 128-bit IDEA Schlüssel verwendet werden kann
- **Private-Key:**
  Verschlüsselt mittels IDEA Algorithmus mit einem "pass-phrase" basierten Schlüssel, der im „private-key ring" gespeichert wird.
  „Pass-phrase" und IDEA Schlüssel werden nach Verwendung verworfen um die private-keys in Alice's Computer zu schützen
  Wann immer Alice eine Nachricht signieren möchte muss sie die "pass-phrase" bestimmen um den private-key mittels IDEA zu entschlüsseln
  - **Public Key Ring: =** Speicherplatz für public-keys
    - alle Partners die mit Alice kommunizieren möchten verwenden PGP
    - auch ihr eigener public-key ist dort gespeichert damit alle Partner Zugang zu diesem haben

• Abläufe im Detail:

- **Handhabung der Keys auf der Empfängerseite:**
  Bob´s Speicherplatz für die Privaten Schlüssel ist sein eigener „private-key ring"
  • Wenn man eine Nachricht erhält:
    - Bob muß seine "pass-phrase" zur Verfügung stellen um Zugang zu seinem private-key zu erhalten
    - Bob´s private-key wird zur Entschlüsselung des „IDEA onetime session key" verwendet
  • nach der IDEA Entschlüsselung
  Bob findet Alice's public-key in seinem "public-key ring" und prüft die Signatur der Nachricht

- **Public-Key Management:**
  • Ursprünglich dezentralisiert, User kontrollierter Zugang
  • manche nennen es Anarchie
  • gegen zentralisierte PKI Schemas wurden Vertrauensstufen eingeführt
  • jeder User kann entscheiden, welchen Schlüssel er vertraut
  • jeder User kann entscheiden welchen User er vertraut
    - Abstufungen: none, partial and complete
    - Public-Keys der anderen User können mit eigenen Private-Key signiert werden
  • Signierte public-keys (= Zertifikat) von vertrauenswürdigen Usern kann vertraut werden
  • heute: PGP Versionen sind interoperabel mit PKI Infrastruktur
  • CA and X.509

1. **Wie macht man einen Public-Key sicher?**
   - ein großes Problem ist der „man-in-the-middle" Angriff
   - Hier hilft:
     - Speicherung auf Floppy Disk oder CD-ROM
     - Erhalten bzw. Prüfen des Schlüssels via Telefon
   - Authentifizierung durch Spracherkennung und und dann den Schlüssel via Telefon ansagen oder einfach den Schlüssel via Mail schicken/erhalten.
   - Einen Fingerabdruck des erhaltenen Schlüssels erzeugen
   - den Partner anrufen, die Fingerprints vergleichen (nur dann ist der Schlüssel zertifiziert)
     - Einen signierten Schlüssel von einem vertrauten User erhalten
     - Einen Schlüssel von einem Server laden und den Fingerprint direkt mit dem dazugehörigen Partner kontrollieren
     - Einen Signierten Schlüssel von einem vertrauten Key-Server laden

Wo ist IPSec angesiedelt und was wird abgedeckt? Aus welchen Komponenten besteht IPsec prinzipiell? Welche Header spielen bei IPsec eine Rolle und wo sind diese angeordnet? Welche Modes gibt es? Aus welchen Phasen besteht IKE? Was wird in den einzelnen Phasen prinzipiell bewerkstelligt? Beschreiben Sie im Detail Phase 1 Aggressive Mode Preshared Secret-Key und Phase 2 Quick Mode. [120 Punkte]

Wo ist IPSec angesiedelt und was wird abgedeckt?

IPSEC liegt zwischen Schicht 3 und 4, sprich zwischen IP und TCP angesiedelt. Ffür die Verwendung von IPSEC muss somit nur das Betriebssystem angepasst werden muss und nicht wie bei SSL & SSH die Applikation.

IPSEC deckt damit die Bereiche **Authentication** (= ich kenne Identität des Senders), **Integrität** (=Nachricht wurde nicht verändert) sowie **Vertraulichkeit oder Privatsphäre** (=niemand außer der autorisierte Empfänger kann die Nachricht lesen) ab. Diese wird durch HMAC, Digitale Signaturen, Verschlüsselung sowie Schlüsselverteilungstechniken erreicht.

IPSEC deckt jedoch NICHT den Bereich **non-repudation** (=Sender kann Nachrichten nicht nachträglich abstreiten).

• Aus welchen Komponenten besteht IPsec prinzipiell?

- Sicherheits Associations
- Sicherheitsprotokollen (AH, ESP, Secret Key Algorithmen)
- Management von Sicherheitsverbänden und Schlüsseln (manuell oder automatisch durch ISAKMP)
- Algorithmen für Vertraulichkeit und Verschlüsselung

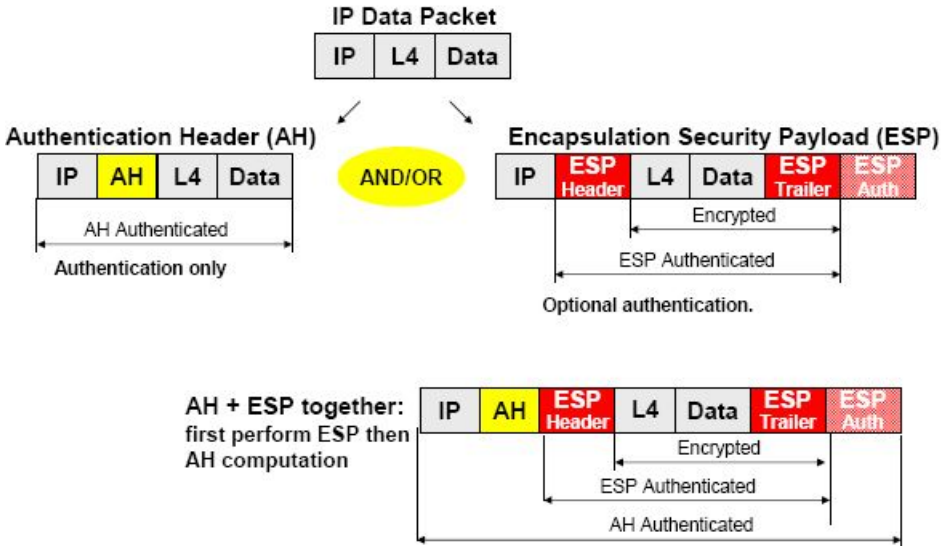
• Welche Header spielen bei IPsec eine Rolle und wo sind diese angeordnet?

- Authentication Header (AH)
- Encapsulation Security Payload (ESP)

Der AH dient der *Authentication* und wird nach dem IP Header angeordnet.

ESP besteht aus zwei Teilen, dem Header und dem Trailer. Zwischen diesen beiden sind alle Daten verschlüsselt. Zusätzlich kann noch der ESP Auth. verwendet werden. Dieser gewährleistet, dass die ESP Nachricht nicht verändert worden ist.

Beide Header können getrennt oder gemeinsam angewendet werden:

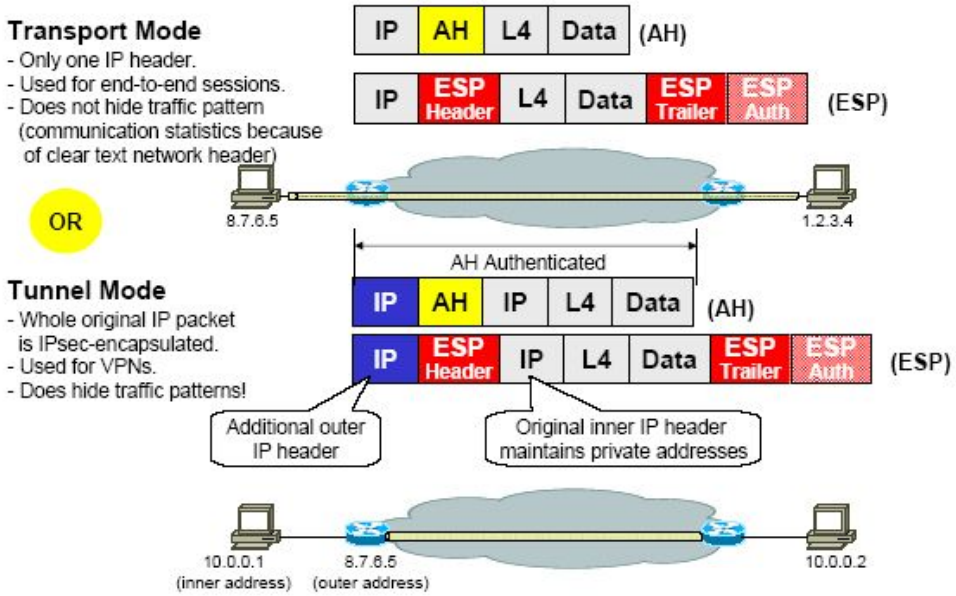


- **Welche Modes gibt es?**

IPSEC kann in 2 Modi betrieben werden: im **Transport Mode** und **Tunnel Mode**.

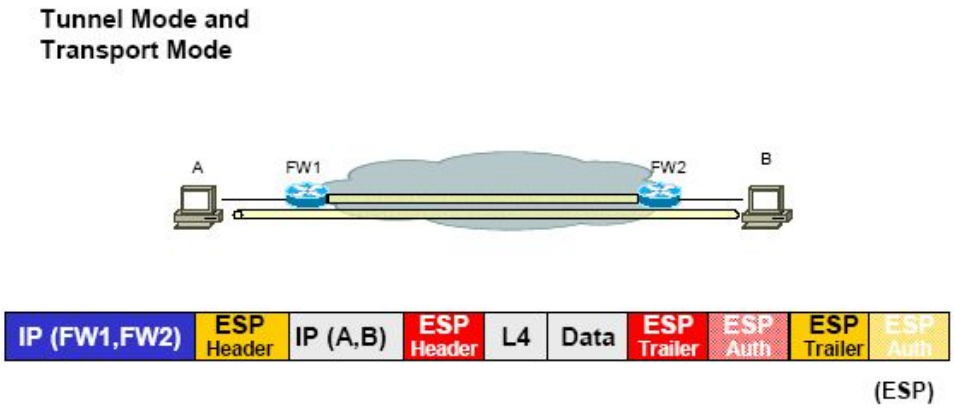
Beim Transport Mode werden Pakete wie oben beschrieben übertragen.

- ◊ Nur ein IP Header, der im Klartext übertragen wird
- ◊ Wird in End to End Sessions verwendet



Beim Tunnel Mode wird ein zusätzlich IP Header vorangestellt. Dadurch wird das Verkehrsmuster des originalen IP Headers versteckt. Dieser wird nicht in End to End Sessions verwendet.

- wird in Virtual Private Networks verwendet
- Das gesamte IP Packet ist in IPSEC eingeschlossen



Gemeinsame Nutzung von Transport Mode und Tunnel Mode

- **Aus welchen Phasen besteht IKE?**

IKE = Internet Key Exchange
Vor dem eigentlichen Start einer verschlüsselten Verbindung muss man sich auf die zu verwendenden Schlüssel und Algorithmen einigen. Hierfür ist IKE gedacht. IPsec arbeitet mit verschiedenen symmetrischen wie asymmetrischen Schlüsseln. Diese können manuell oder automatisch ausgehandelt werden.
IKE basiert auf UDP und nutzt standardmäßig den Port 500.

Phase 1: Main Mode oder Aggressive Mode
Phase 2: Quick Mode

IKE arbeitet in zwei Phasen, in der ersten Phase wird ein sicherer Tunnel geschaffen über den in der zweiten Phase die SA vereinbart werden.

- **Was wird in den einzelnen Phasen prinzipiell bewerkstelligt?**

Internet Key Exchange will ein vereinfachtes Verfahren zum Aufbau sicherer, authentifizierter Verbindungen standardisieren, wohl auch, um die Komplexität von anderen Verfahren zu vermeiden, welche auf Entwickler eher abschreckend erscheint.

IKE unterscheidet Modes, in denen Schlüssel ausgetauscht werden, welches in einer oder zwei Phasen geschieht. In der ersten Phase wird eine sichere, authentisierte Verbindung aufgebaut, in der zweiten Phase werden die in den verschiedenen Protokollen benötigten Schlüssel ausgetauscht, wobei in der Regel einzelne Schlüssel (Verschlüsselung, Hashen) von einem Masterschlüssel abgeleitet werden.

Beschreiben Sie im Detail Phase 1 Aggressive Mode Preshared Secret-Key und Phase 2 Quick Mode.

Main Mode (1. Phase)

Der Main Mode kann in der ersten Phase der Internet Key Exchange genutzt werden. Hierbei handeln der Initiator (derjenige, der die Verbindung aufnehmen will) und der Antwortende miteinander SAs für ISAKMP aus. Diese "Verhandlung" geschieht in folgenden sechs Schritten:

- Initiator sendet einen oder mehrere Vorschläge mit Authentifizierungs- und Verschlüsselungsalgorithmen
- Antwortender wählt einen Vorschlag aus und bestätigt
- Initiator sendet öffentlichen Teil der Diffie-Hellmann-Schlüsselvereinbarung und einen zufälligen Wert (Nonce)
- Antwortender schickt ebenfalls öffentlichen Teil der Diffie-Hellmann-Schlüsselvereinbarung und einen zufälligen Wert (Nonce)
- Initiator berechnet Signatur und sendet diese mit seiner Identität an Antwortenden. Diese Daten werden mit einem symmetrischen Schlüssel verschlüsselt.
- Antwortender schickt gleiche Daten von seiner Seite an den Initiator

Aggressive Mode (1. Phase)

Im Aggressive Mode werden die obigen Schritte auf drei zusammengefasst. Hierbei fällt dann die Verschlüsselung des obigen fünften Schrittes weg. Stattdessen werden die Werte im Klartext übertragen. Daher sollte man diesen Modus nach Möglichkeit nicht verwenden.

Quick Mode (2. Phase)

Der Quick Mode wird in der zweiten Phase von IKE zur Anwendung gebracht. Die gesamte Kommunikation in dieser Phase erfolgt verschlüsselt. Wie in der ersten Phase wird zunächst ein Vorschlag (Proposal) gemacht. Dieser wird zusammen mit einem Hashwert und der Nonce übertragen. Später werden die Schlüssel neu berechnet, und es gehen keinerlei Informationen aus den zuvor generierten SAs ein. Dies stellt sicher, dass niemand von den zuvor generierten Schlüsseln auf die neuen schließen kann.

Wozu dient eine PKI prinzipiell? Beschreiben Sie die Elemente, die zum Aufbau einer PKI Infrastructur prinzipiell benötigt werden. Wie kann dabei bei einer hierarchischen PKI Struktur durch „Certifcate Chaining" die Verifizierung erfolgen? Was sind „Trust Anchors"?

PKI:

Als **PKI** (**Public Key Infrastructure**) bezeichnet man in der Kryptologie und Kryptografie ein System, welches es ermöglicht, digitale Zertifikate auszustellen, zu verteilen und zu prüfen. Die innerhalb einer PKI ausgestellten Zertifikate sind meist auf Personen oder Maschinen festgelegt und werden zur Absicherung computergestützter Kommunikation verwendet.

Der zu Grunde liegende Gedanke ist der folgende: Mit Hilfe eines Public-Key-Verschlüsselungsverfahrens können Nachrichten im Internet signiert und verschlüsselt werden. Das Signieren garantiert, dass die Nachricht in dieser Form wirklich vom angegebenen Absender stammt. Allerdings benötigt man hierzu den Public-Key des Absenders. Dieser könnte z.B. per E-Mail versendet werden. Es stellt sich genau an diesem Punkt aber die Frage, wie man sicher ist, dass es sich tatsächlich um den Schlüssel des Absenders handelt und nicht um eine Fälschung eines Betrügers. Hierzu kann der zu verschickende Schlüssel selbst wieder mit einem vertrauenswürdigen Schlüssel signiert sein. Auf diese Weise lässt sich eine Hierarchie aus vertrauenswürdigen Institutionen aufbauen. Auf die Echtheit der Schlüssel der obersten Institutionen dieser Hierarchie muss man sich aber verlassen können. Sie sind oft in die verarbeitende Computer-Software integriert.

- **Wesentliche Bestandteile einer (minimalen) PKI:**

- **Digitale Zertifikate:**

  Digital signierte elektronische Daten, die sich zum Nachweis der Echtheit von Objekten verwenden lassen.

- **Certification Authority (CA):**

  Organisation, welche die Bereitstellung von Zertifikaten übernimmt.

- **Registration Authority (RA):**

  Organisation, bei der Personen und Maschinen Zertifikate beantragen können.

- **Certificate Revocation Lists (CRL):**

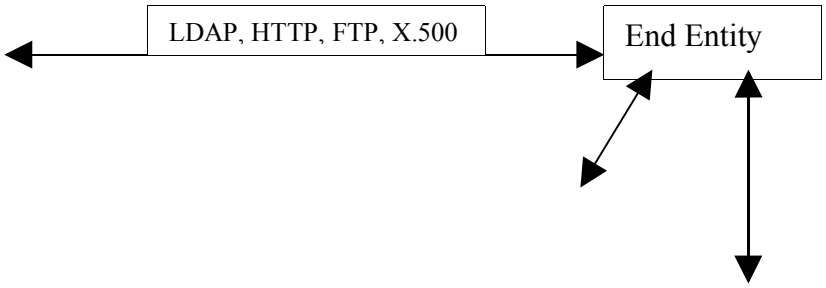
  (Sperrliste) Listen mit zurückgezogenen, abgelaufenen und für ungültig erklärten Zertifikaten.

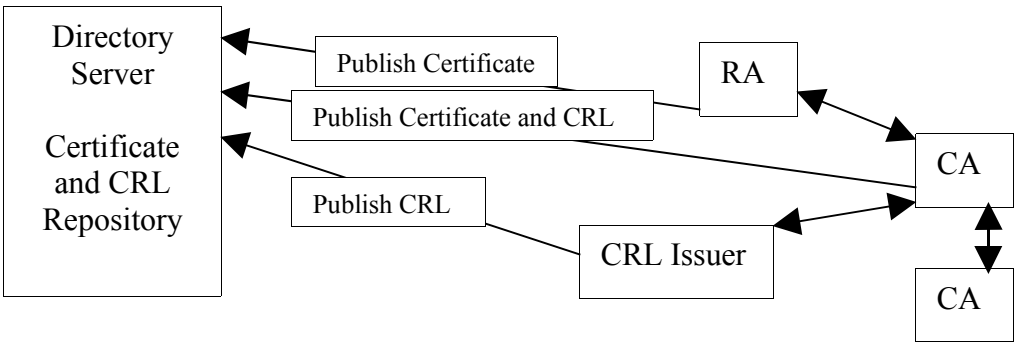
◊ **Verzeichnisdienst:**

  ein durchsuchbares Verzeichnis welches ausgestellte Zertifikate enthält, meist ein LDAP-Server, seltener ein X.500-Server.

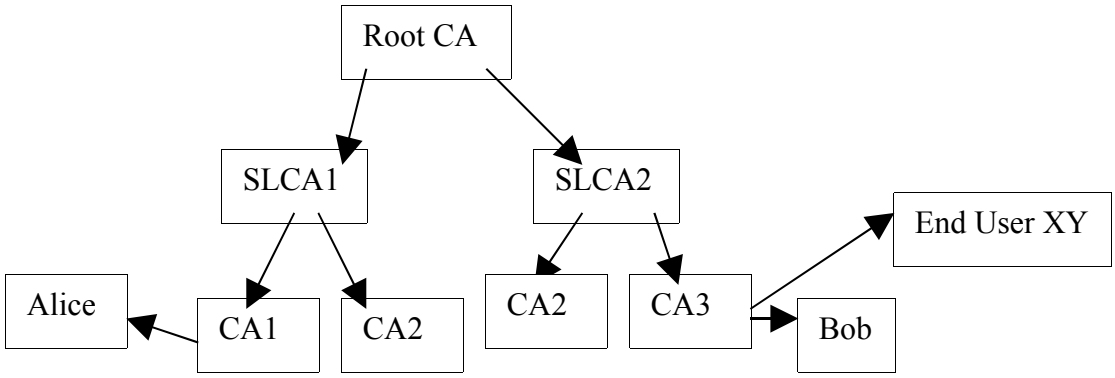
- **Validierungsdienst:**

  Ein Dienst, der die Überprüfung von Zertifikaten in Echtzeit ermöglicht.





Es gibt eine Root Certificate Authority, die an der Spitze der Hierarchie steht. Darunter kommen die Second Level CAs, die SLCAs. Unter ihnen folgen die eigentlichen CAs, die Zertifikate an die End User ausstellen. Die Echtheit eines Zertifikates wird nun mit Certificate Chaining bewiesen.



1) Alice asks Bob for Public Key Bob

Signature of CA3
PK Bob
Bob is approved

2) Alice asks SLCA for PK of CA3

Signature of SLCA2
PK CA3
CA3 is approved

3) Alice asks Root for PK SLCA2

Signature of Root
PK SLCA2
SLCA2 is approved

4) Alice verifies PK SLCA2 with Signature of Root

Signature of SLCA2
PK CA3
CA3 is approved

5) Alice verifies PK CA3 with the Signature of SLCA2

Signature of CA3
PK Bob

Bob is approved

In der Praxis gibt es nicht eine einzige Root CA weltweit, sondern viele. Problematisch ist bei PKI-Systemen, dass Computer-Programme bereits mit einer Vielzahl von "Root-Zertifikaten" ausgeliefert werden, die von Organisationen ausgestellt werden, deren Existenz und deren Integrität nicht gewährleistet ist. Weiterhin kann keine Aussage über die Anforderungen getroffen werden, die zur Ausstellung der Zertifikate erforderlich sind.

- **Ein Trust Anchor**

ist nichts anderes, als ein Public Key, der bekannt und vertrauenswürdig ist und an der Spitze der Hierarchie steht. Diese Trust Anchors werden meist mit der Software mitgeliefert (Z.B. bei Browsern) (Laut dem Buch „Network Security", dort stand auch nicht wirklich mehr, auf seinen Folien ebenfalls nicht, nachfragen ob das genügt!)

Was ist Kerberos und wozu dient es? Welche Server Funktionen sind notwendig? Wie ist die Handhabung der long-term und short-term secret-keys? Beschreiben Sie die drei Phasen der Kerberos Abläufe? Warum ist die AS und TGS Funktion voneinander getrennt angelegt? Wozu dienen Kerberos Realms?

- **Was und wozu Kerberos:**

- Verschlüsselungsstandard
- Authentifizierungsverfahren nach dem digitale Signaturen erzeugt werden können
- Dient zur Authentifizierung einzelner Nachrichten
- Dient zur Verschlüsselung einzelner Nachrichten
- Dient zur Identifikation eines Clients beim Aufbau einer Verbindung zwischen C/S
- Verschlüsselung-Sicherheitssystem:
  - gewährleistet Authentifizierung zwischen Server und Client
  - bei normalen Verschlüsselungssystemen müsste jeder Server im LAN mit Passwörtern der User konfiguriert werden > Kerberos nicht, da es hierzu ein Ticketsystem verwendet

- **Welche Serverfunktionen werden benötigt:**

- **Authentification Server:**
  Dieser wird beim Einloggen ins System benutzt, er speichert eine Passphrase für jeden User, der User authentifiziert sich über die Passphrase (wird gehashed über das Netz gesendet) und erhält ein Ticket für den
- **Ticket Granting Server**
  User fragt um ein Ticket für die Service Server im LAN an. Er benutzt dafür das Ticket, das er vom Authentification Server erhalten hat. Hier und auch später ist keine Passwortabfrage nötig > mit diesem Ticket kann der User nun die Services der anderen Server im Netz nutzen

Im Detail:

Principal (zb Benutzer) fragt Kerberos Dienst nach einem Ticket für ein „Ticket Granting Service" (TGS) > Dieses Ticket (TGT) wird mit geheimen Schlüssel des Principals verschlüsselt (da geheimer Schlüssel nicht gespeichert ist Eingabe notwendig: Tastatur, Smartcard) > Benutzer kann Autorisierung auf Server vornehmen zb Telnet: Telnet-Client bittet unter Vorlage des TGT beim Kerberos Server um ein Telnet-Ticket, welches

gemeinsam mit dem Authentikator dem Telnet-Server vorgelegt wird > Anmeldung am Server kann erfolgen.

- **Handhabung der long-term und short-term secret-keys:**

long-term secret-keys:

- Schlüssel die für längere Zeit im System bleiben und gültig sind.
- In Kerberos betrifft dies nur die „pass-phrase"
  - Sie ist „off-line" beim AS eingetragen worden und wird nie über das Netzwerk gesendet.
  - Sie wird nur für das einloggen ins System verwendet.

short-term secret-keys:

- Schlüssel, die nur kurze Zeit Gültigkeit haben.
- Das sind alle Tickets die vom Authentification Server und Ticket Granting Server verteilt werden.

- **3 Phasen der Kerberos Abläufe:**

- **Einloggen und Authentifizierung**
  - Alice fordert ein Ticket Granting Ticket (TGT) an
  - AS Server generiert einen short-term session-key K_T und ein TGT Ticket. Dieses ist mit dem Schlüssel Ticket Granting Servers K_{TGS} verschlüsselt. Im TGT Ticket befindet sich der short-term session-key und die IP-Adresse von Alice.
  - Der K_T und das TGT werden mit dem Secret-Key von Alice verschlüsselt und an Alice geschickt
  - Alice gibt ihre Passphrase ein und kann dadurch den K_T und das TGT entschlüsseln (nicht den Inhalt des TGT!!)

- **Ticket für Service anfordern**
  - Alice sendet das TGT (Sicherstellung dass es wirklich Alice ist) an den Ticket Granting Server TGS. Zusammen mit einer Aufforderung welchen Dienst sie nutzen will (Alice wants Bob) und einem Zeitstempel, der mit K_T verschlüsselt ist (Verhinderung einer Replay Attacke)
  - Der TGS geniert nun einen short-term session-key K_{AB} (dieser gilt nur für diese Session zwischen Alice und Bob). Dieser Schlüssel und die Adresse von Bob werden über den K_T verschlüsselt.
  - Weiters generiert der TGS ein Ticket für den Service, das TB. Es enthält die Adresse von Alice und den session-key K_{AB}. Verschlüsselt wird es über den Schlüssel von Bob K_B (so kanns nur er entschlüsseln)

- **Service nutzen**
  - Alice sendet das Ticket für den Service von Bob T_B an Bob (ich bin sicher Alice, weil ohne Passphrase hätte ich Tickets nicht bekommen können), zusammen mit einem über den session-key K_{AB} verschlüsselten Zeitstempel
  - Bob antwortet mit Zeitstempel+1 verschlüsselt mit session-key K_{AB}
  - Alice vergleicht Zeitstempel, alle weiteren Nachrichten werden mit K_{AB} verschlüsselt

- **Warum ist die AS und TGS Funktion voneinander getrennt angelegt:**

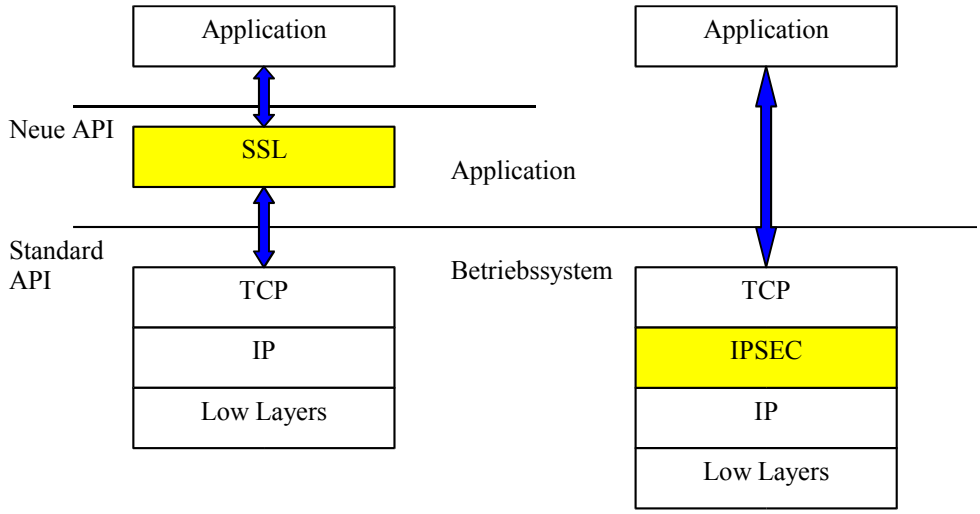
- **Wozu dienen Kerberos Realms:**

Es ist technisch unmöglich wenn weltweit nur ein AS und ein TGS existiert. Deshalb gibt es Realms (Reiche), jede mit seinem eigenen AS und TGS. Will ein User einen Service in einem entfernten Realm nutzen, beantragt er ein TGT bei seinem lokalen TGS, das der entfernte TGS akzeptiert. Mit diesem TGT kann Alice nun ein Ticket TB für Bob im entfernten Realm anfordern.

Wo ist IPSec angesiedelt und was wird abgedeckt? Aus welchen Komponenten besteht IPsec prinzipiell? Welche Header spielen bei IPsec eine Rolle und wo sind diese angeordnet? Welche Modes gibt es? Aus welchen Phasen besteht IKE? Was wird in den einzelnen Phasen prinzipiell bewerkstelligt? Beschreiben Sie im Detail Phase 1 Main Mode Public-Key Signatures und Phase 2 Quick Mode. [130 Punkte]

- **Wo ist IPSec angesiedelt?**

Während SSL zwischen Applikationslayer und TCP Layer angesiedelt ist, liegt IPSEC zwischen IP und TCP. Daraus folgt, dass für die Verwendung von IPSEC nur das Betriebssystem angepasst werden muss, wogegen man für SSL die Applikation ändern muss.



Bei IPSEC können die Applicationen weiterhin das gleiche Application Programming Interface ansprechen.

IPSEC deckt damit die Bereiche **Authentication** (= ich kenne Identität des Senders), **Integrität** (=Nachricht wurde nicht verändert) sowie **Vertraulichkeit oder Privatsphäre** (=niemand außer der autorisierte Empfänger kann die Nachricht lesen) ab. Diese wird durch HMAC, Digitale Signaturen, Verschlüsselung sowie Schlüsselverteilungstechniken erreicht.

IPSEC deckt jedoch NICHT den Bereich **non-repudation** (=Sender kann Nachrichten nicht nachträglich abstreiten).

- **Aus welchen Komponenten besteht IPsec prinzipiell?**

- Sicherheits Associations
- Sicherheitsprotokollen (AH, ESP, Secret Key Algorithmen)
- Management von Sicherheitsverbänden und Schlüsseln (manuell oder automatisch durch ISAKMP)

- Algorithmen für Vertraulichkeit und Verschlüsselung

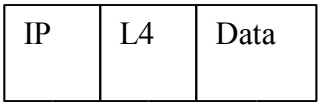
- ***Welche Header spielen bei IPsec eine Rolle und wo sind diese angeordnet?***

- Authentication Header (AH)
- Encapsulation Security Payload (ESP)

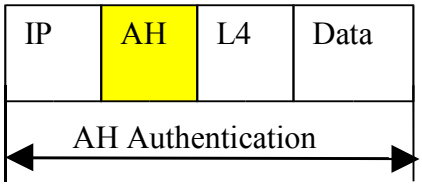
Der AH dient der Authentication und ist wird nach dem IP Header hinzugegeben.
ESP besteht aus zwei teilen, dem Header und dem Trailer. Zwischen diesen beiden sind alle Daten verschlüsselt. Zusätzlich kann noch den ESP Auth. verwendet werden. Dieser gewährleistet, dass die ESP Nachricht nicht verändert worden ist.

Beide Header können getrennt oder gemeinsam angewendet werden:

„Normale" Nachricht:

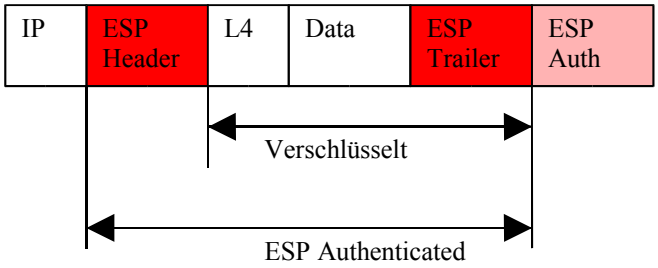


AH Header:



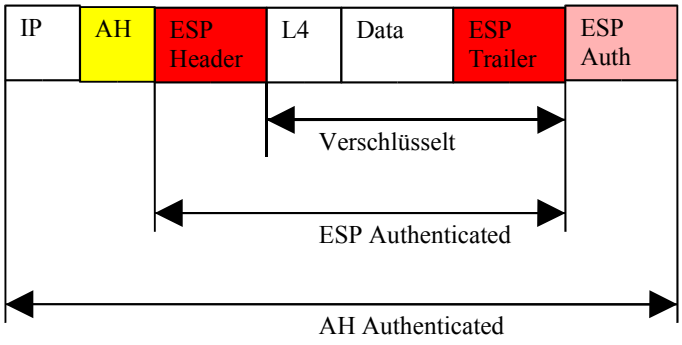
Nur Authentication!

ESP Header:



Authentication optional

Beides zusammen:
Zuerst ESP, dann AH



- ***Welche Modes gibt es?***

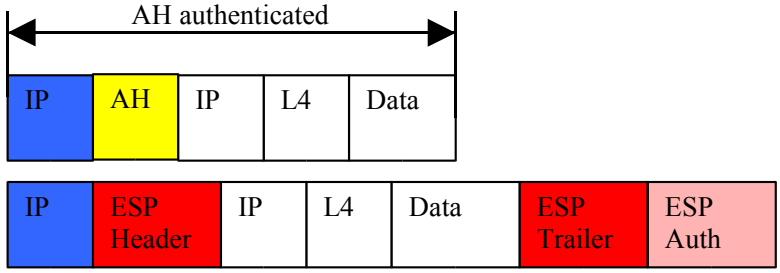
IPSEC kann in 2 Modi betrieben werden: im **Transport Mode** und **Tunnel Mode**.

Beim Transport Mode werden Pakete wie oben beschrieben übertragen.

- Nur ein IP Header, der im Klartext übertragen wird
- Wird in End to End Sessions verwendet

Beim Tunnel Mode wird ein zusätzlich IP Header vorangestellt. Dadurch wird das Verkehrsmuster des originalen IP Headers versteckt. Dieser wird nicht in End to End Sessions verwendt.

- wird in Virtual Private Networks verwendet
- Das gesamte IP Packet ist in IPSEC eingeschlossen



Natürlich können beide Modi gemeinsam eingesetzt werden. Die Pakete sehen dann folgendermaßen aus: