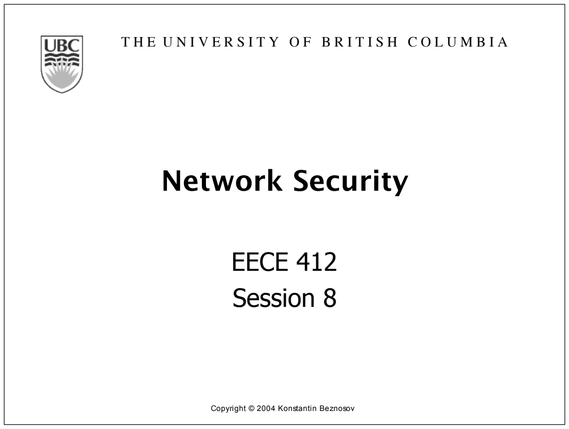
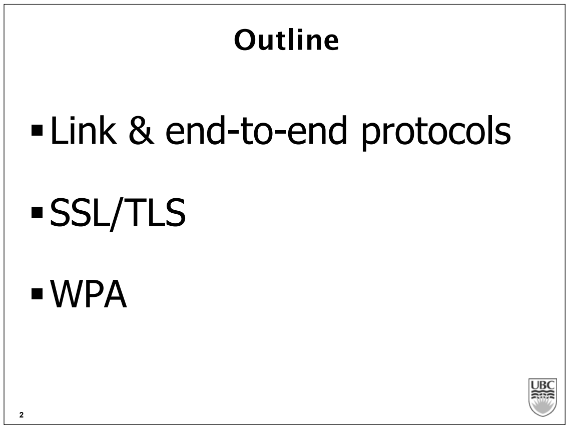


THE UNIVERSITY OF BRITISH COLUMBIA

Network Security

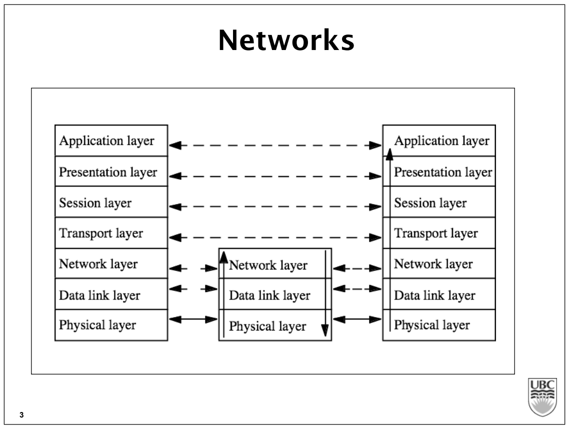
EECE 412
Session 8



Outline

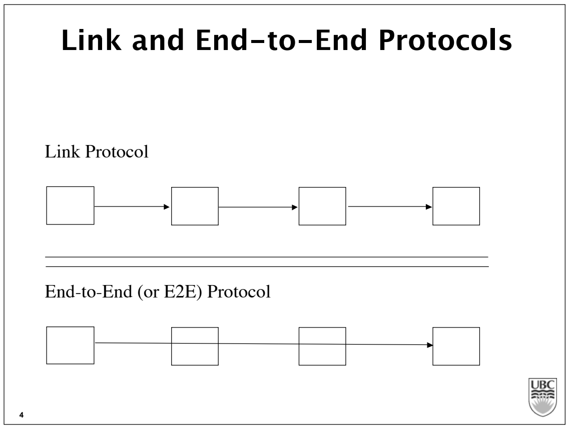
- Link & end-to-end protocols
- SSL/TLS
- WPA

2



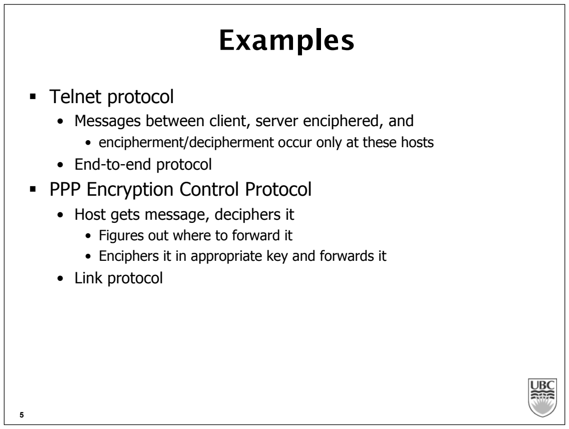
Networks

3



Link and End-to-End Protocols

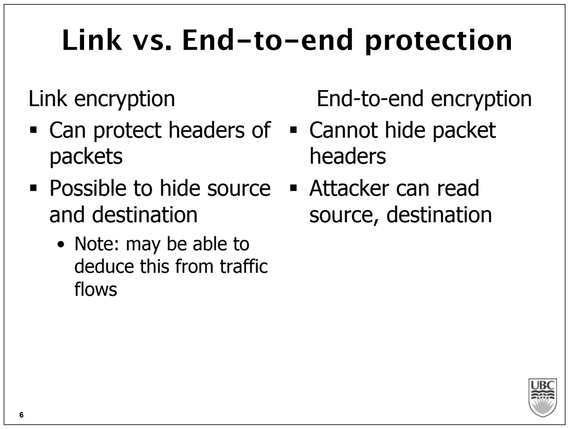
4



Examples

- Telnet protocol
  - Messages between client, server enciphered, and
    - encipherment/decipherment occur only at these hosts
  - End-to-end protocol
- PPP Encryption Control Protocol
  - Host gets message, deciphers it
    - Figures out where to forward it
    - Enciphers it in appropriate key and forwards it
  - Link protocol

5



Link vs. End-to-end protection

Link encryption

- Can protect headers of packets
- Possible to hide source and destination
  - Note: may be able to deduce this from traffic flows

End-to-end encryption

- Cannot hide packet headers
- Attacker can read source, destination

6

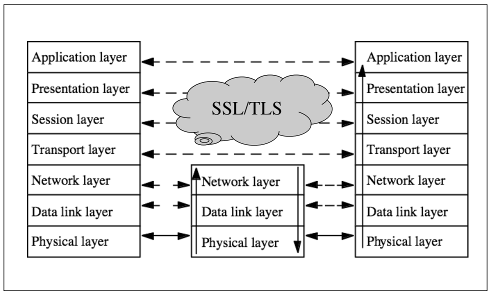
Example Protocols

- Privacy-Enhanced Electronic Mail (PEM)
  - Applications layer protocol
  - Bishop
- Secure Socket Layer (SSL)/Transport Layer Security (TLS)
  - Transport layer protocol
- IP Security (IPSec)
  - Network layer protocol
  - Bishop
- Wi-Fi Protected Access (WPA)
  - Data layer protocol
  - Today session

THE UNIVERSITY OF BRITISH COLUMBIA

Secure Socket Layer (SSL)
a.k.a.
Transport Layer Security (TLS)

Networks



SSL Session

Association between two peers

- Two peers may have several sessions
- Information for each association:
  - Unique session identifier
  - Peer’s X.509v3 certificate, if needed
  - Compression method
  - Cipher spec for cipher and MAC
  - “Master secret” shared with peer
    - 48 bits

SSL Connection

Describes how data exchanged with peer in a session

- Several connections per session
- Information for each connection
  - Random data
  - Write keys (used to encipher data)
  - Write MAC key (used to compute MAC)
  - Initialization vectors (IVs) for ciphers, if needed
  - Sequence numbers

Supporting Crypto

- All parts of SSL use them
- Initial phase: public key system exchanges keys
  - Messages enciphered using classical ciphers, and MACed
  - Only certain combinations allowed
    - Depends on algorithm for key exchange cipher
  - Key exchange (a.k.a., interchange) algorithms:
    - RSA
    - Diffie-Hellman
    - Fortezza

RSA: Cipher, MAC Algorithms

Interchange cipher	*Classical cipher*	*MAC Algorithm*
RSA, key ≤ 512 bits	*none*	MD5, SHA
	RC4, 40-bit key	MD5
	RC2, 40-bit key, CBC mode	MD5
	DES, 40-bit key, CBC mode	SHA
RSA	*None*	MD5, SHA
	RC4, 128-bit key	MD5, SHA
	IDEA, CBC mode	SHA
	DES, CBC mode	SHA
	DES, EDE mode, CBC mode	SHA

13

D-H: Cipher, MAC Algorithms

Interchange cipher	*Classical cipher*	*MAC Algorithm*
Diffie-Hellman, DSS Certificate	DES, 40-bit key, CBC mode	SHA
	DES, CBC mode	SHA
	DES, EDE mode, CBC mode	SHA
Diffie-Hellman, key ≤ 512 bits RSA Certificate	DES, 40-bit key, CBC mode	SHA
	DES, CBC mode	SHA
	DES, EDE mode, CBC mode	SHA

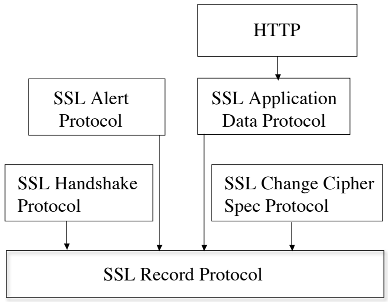
14

Fortezza: Cipher, MAC Algorithms

Interchange cipher	*Classical cipher*	*MAC Algorithm*
Fortezza key exchange	*none*	SHA
	RC4, 128-bit key	MD5
	Fortezza, CBC mode	SHA

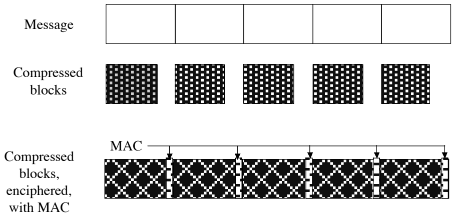
15

SSL Protocols



16

SSL Record Layer

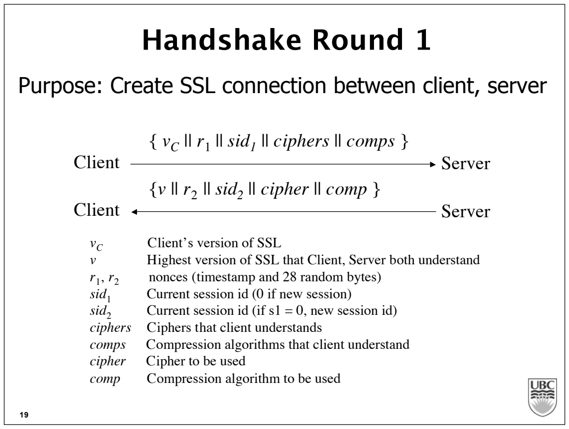


17

Overview of Handshake Rounds

1. Create SSL connection between client, server
2. Server authenticates itself
3. Client validates server, begins key exchange
4. Acknowledgments all around

18



Handshake Round 1

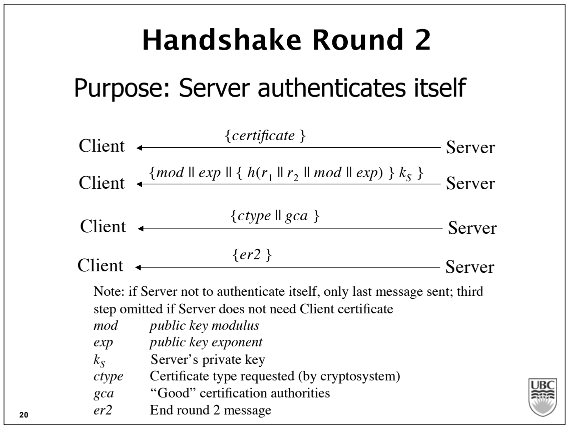
Purpose: Create SSL connection between client, server

$$\text{Client} \xrightarrow{\{\, v_C \parallel r_1 \parallel \mathit{sid}_1 \parallel \mathit{ciphers} \parallel \mathit{comps} \,\}} \text{Server}$$

$$\text{Client} \xleftarrow{\{v \parallel r_2 \parallel \mathit{sid}_2 \parallel \mathit{cipher} \parallel \mathit{comp} \,\}} \text{Server}$$

v_C	Client’s version of SSL
v	Highest version of SSL that Client, Server both understand
r_1, r_2	nonces (timestamp and 28 random bytes)
sid_1	Current session id (0 if new session)
sid_2	Current session id (if s1 = 0, new session id)
ciphers	Ciphers that client understands
comps	Compression algorithms that client understand
cipher	Cipher to be used
comp	Compression algorithm to be used

19



Handshake Round 2

Purpose: Server authenticates itself

$$\text{Client} \xleftarrow{\{\mathit{certificate}\,\}} \text{Server}$$

$$\text{Client} \xleftarrow{\{\mathit{mod} \parallel \mathit{exp} \parallel \{\, h(r_1 \parallel r_2 \parallel \mathit{mod} \parallel \mathit{exp})\,\}\, k_S\,\}} \text{Server}$$

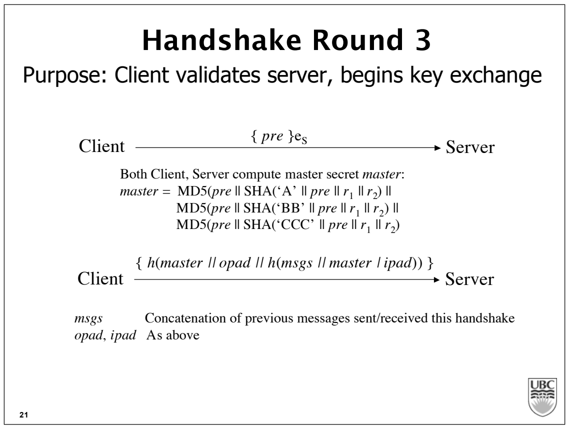
$$\text{Client} \xleftarrow{\{\mathit{ctype} \parallel \mathit{gca}\,\}} \text{Server}$$

$$\text{Client} \xleftarrow{\{\mathit{er2}\,\}} \text{Server}$$

Note: if Server not to authenticate itself, only last message sent; third step omitted if Server does not need Client certificate

mod	*public key modulus*
exp	*public key exponent*
k_S	Server’s private key
ctype	Certificate type requested (by cryptosystem)
gca	“Good” certification authorities
er2	End round 2 message

20



Handshake Round 3

Purpose: Client validates server, begins key exchange

$$\text{Client} \xrightarrow{\{\, \mathit{pre}\,\}\mathrm{e}_S} \text{Server}$$

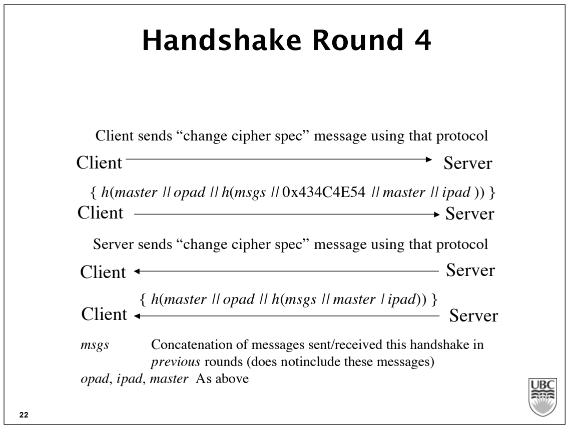
Both Client, Server compute master secret *master*:

$$\begin{aligned} \mathit{master} = {} & \mathrm{MD5}(\mathit{pre} \parallel \mathrm{SHA}(\text{‘A’} \parallel \mathit{pre} \parallel r_1 \parallel r_2) \parallel \\ & \mathrm{MD5}(\mathit{pre} \parallel \mathrm{SHA}(\text{‘BB’} \parallel \mathit{pre} \parallel r_1 \parallel r_2) \parallel \\ & \mathrm{MD5}(\mathit{pre} \parallel \mathrm{SHA}(\text{‘CCC’} \parallel \mathit{pre} \parallel r_1 \parallel r_2) \end{aligned}$$

$$\text{Client} \xrightarrow{\{\, h(\mathit{master} \,//\, \mathit{opad} \,//\, h(\mathit{msgs} \,//\, \mathit{master} \,/\, \mathit{ipad}))\,\}} \text{Server}$$

msgs	Concatenation of previous messages sent/received this handshake
opad, *ipad*	As above

21



Handshake Round 4

Client sends “change cipher spec” message using that protocol

$$\text{Client} \xrightarrow{\hspace{6em}} \text{Server}$$

$$\text{Client} \xrightarrow{\{\, h(\mathit{master} \,//\, \mathit{opad} \,//\, h(\mathit{msgs} \,//\, \mathrm{0x434C4E54} \,//\, \mathit{master} \,//\, \mathit{ipad}\,))\,\}} \text{Server}$$

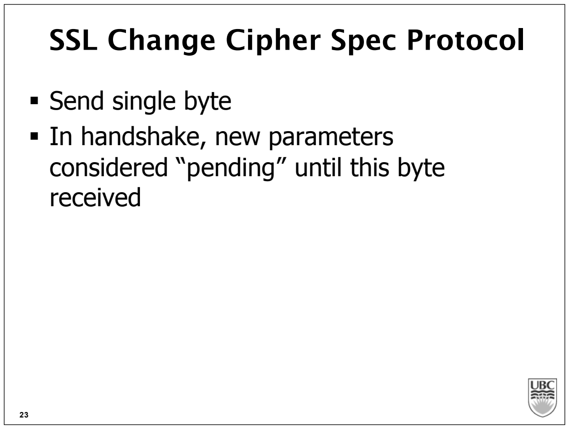
Server sends “change cipher spec” message using that protocol

$$\text{Client} \xleftarrow{\hspace{6em}} \text{Server}$$

$$\text{Client} \xleftarrow{\{\, h(\mathit{master} \,//\, \mathit{opad} \,//\, h(\mathit{msgs} \,//\, \mathit{master} \,/\, \mathit{ipad}))\,\}} \text{Server}$$

msgs	Concatenation of messages sent/received this handshake in *previous* rounds (does notinclude these messages)
opad, *ipad*, *master*	As above

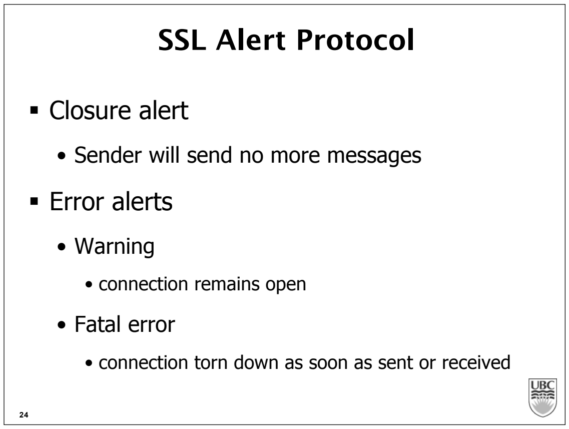
22



SSL Change Cipher Spec Protocol

- Send single byte
- In handshake, new parameters considered “pending” until this byte received

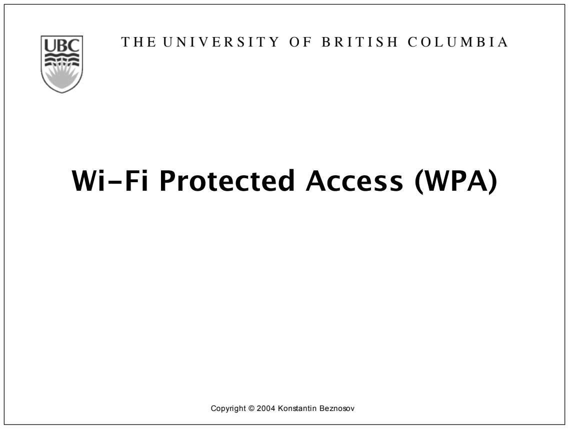
23



SSL Alert Protocol

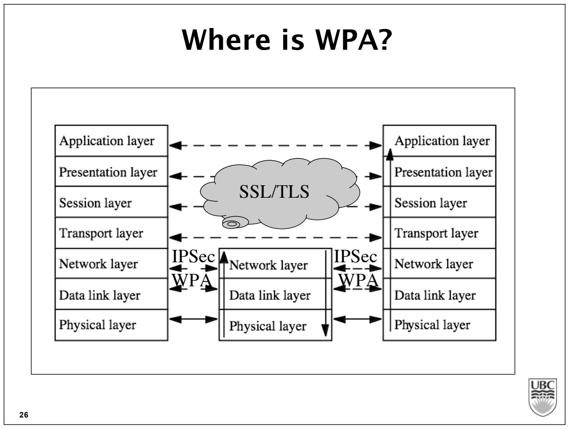
- Closure alert
  - Sender will send no more messages
- Error alerts
  - Warning
    - connection remains open
  - Fatal error
    - connection torn down as soon as sent or received

24



THE UNIVERSITY OF BRITISH COLUMBIA

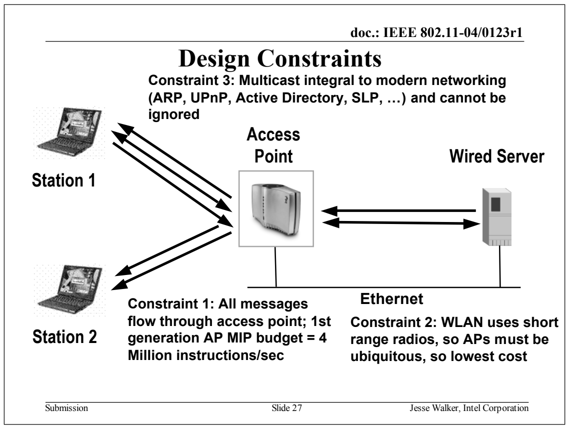
Wi-Fi Protected Access (WPA)



Where is WPA?

Application layer
Presentation layer
Session layer
Transport layer
Network layer
Data link layer
Physical layer

26



doc.: IEEE 802.11-04/0123r1

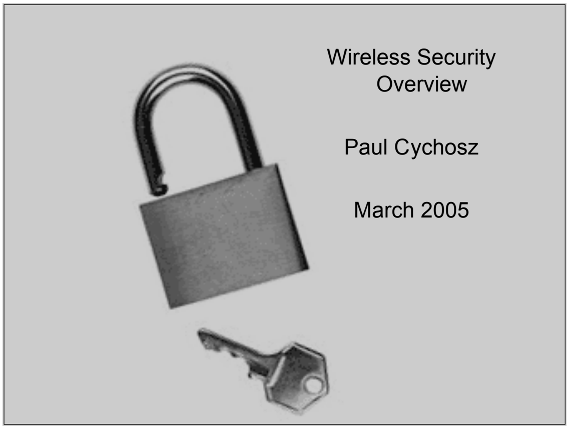
Design Constraints

Constraint 3: Multicast integral to modern networking (ARP, UPnP, Active Directory, SLP, …) and cannot be ignored

Constraint 1: All messages flow through access point; 1st generation AP MIP budget = 4 Million instructions/sec

Constraint 2: WLAN uses short range radios, so APs must be ubiquitous, so lowest cost

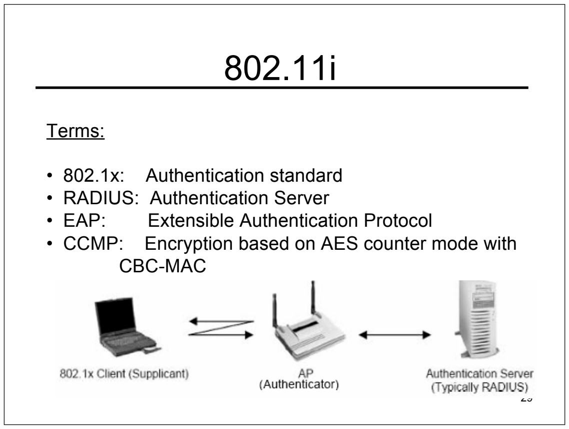
Submission Slide 27 Jesse Walker, Intel Corporation



Wireless Security Overview

Paul Cychosz

March 2005



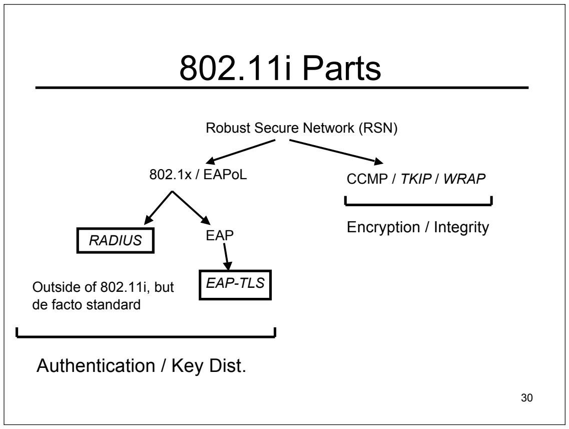
802.11i

Terms:

- 802.1x: Authentication standard
- RADIUS: Authentication Server
- EAP: Extensible Authentication Protocol
- CCMP: Encryption based on AES counter mode with CBC-MAC

802.1x Client (Supplicant)
AP
(Authenticator)
Authentication Server
(Typically RADIUS)

29

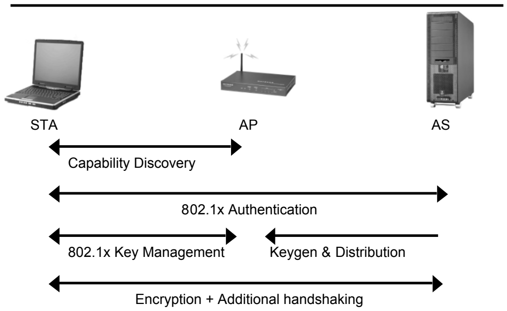


802.11i Parts

Robust Secure Network (RSN)
802.1x / EAPoL
CCMP / *TKIP* / *WRAP*
Encryption / Integrity
RADIUS
EAP
EAP-TLS
Outside of 802.11i, but de facto standard
Authentication / Key Dist.

30

802.11i – First half



STA

AP

AS

Capability Discovery

802.1x Authentication

802.1x Key Management

Keygen & Distribution

Encryption + Additional handshaking

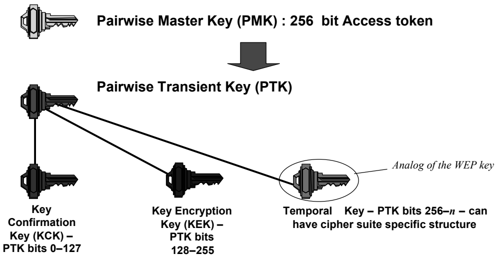
31

THE UNIVERSITY OF BRITISH COLUMBIA

WPA Key Managment

doc.: IEEE 802.11-04/0123r1

802.11i Pairwise Key Hierarchy



Pairwise Master Key (PMK) : 256 bit Access token

Pairwise Transient Key (PTK)

Analog of the WEP key

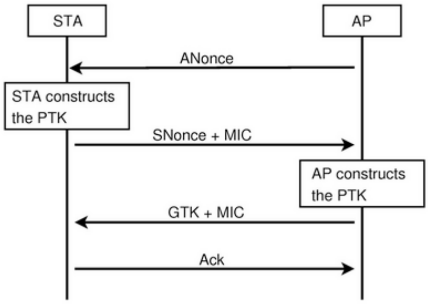
Key Confirmation Key (KCK) – PTK bits 0–127

Key Encryption Key (KEK) – PTK bits 128–255

Temporal Key – PTK bits 256–*n* – can have cipher suite specific structure

Submission Slide 33 Jesse Walker, Intel Corporation

Session Key Establishment



STA

AP

ANonce

STA constructs the PTK

SNonce + MIC

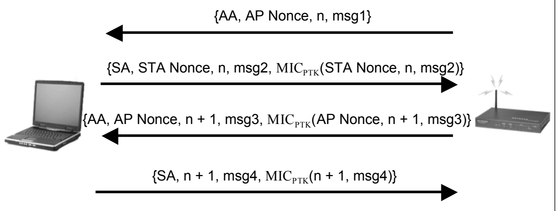
AP constructs the PTK

GTK + MIC

Ack

34

Handshake Details



{AA, AP Nonce, n, msg1}

{SA, STA Nonce, n, msg2, $\mathrm{MIC}_{\mathrm{PTK}}$(STA Nonce, n, msg2)}

{AA, AP Nonce, n + 1, msg3, $\mathrm{MIC}_{\mathrm{PTK}}$(AP Nonce, n + 1, msg3)}

{SA, n + 1, msg4, $\mathrm{MIC}_{\mathrm{PTK}}$(n + 1, msg4)}

35

Message 1

- not protected, doesn't matter though

AP → STA: {AA, AP Nonce, n, msg1}

AA: MAC Address of AP

AP Nonce: random value

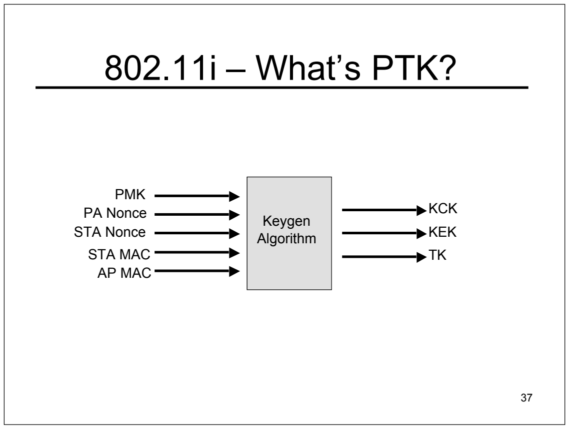
n: sequence identifier

msg1: PMKID = HMAC-SHA1-128(PMK, "PMK Name" || AA || SPA).

- Client uses AP Nonce and PMK to compute PTK

PTK = 802.11i-PRF(
PMK,
min(AP Nonce, STA Nonce) || max(AP nonce, STA Nonce) ||
min(AP MAC Addr, STA MC Addr) || max(AP MAC Addr, STA MAC Addr))

36



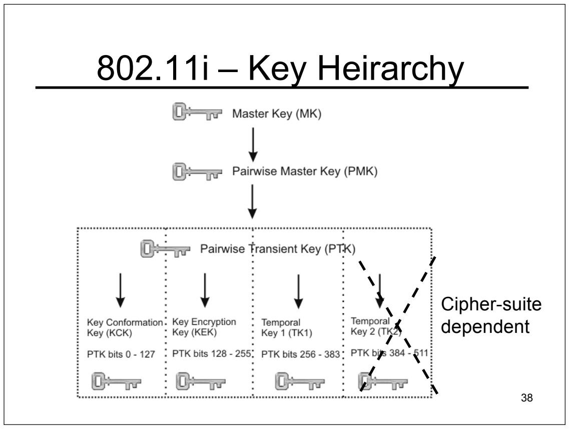
802.11i – What’s PTK?

PMK
PA Nonce
STA Nonce
STA MAC
AP MAC

Keygen Algorithm

KCK
KEK
TK

37



802.11i – Key Heirarchy

Master Key (MK)

Pairwise Master Key (PMK)

Pairwise Transient Key (PTK)

Key Conformation Key (KCK)
PTK bits 0 - 127

Key Encryption Key (KEK)
PTK bits 128 - 255

Temporal Key 1 (TK1)
PTK bits 256 - 383

Temporal Key 2 (TK2)
PTK bits 384 - 511

Cipher-suite dependent

38

Message 2

STA → AP: {SA, STA Nonce, n, msg2, MIC_{PTK}(STA Nonce, n, msg2)}

SPA: MAC Address of STA

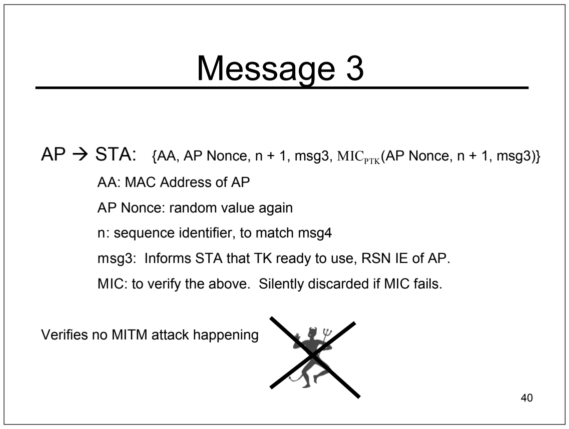
SNonce: random value

n: sequence identifier, matches msg1

msg2: RSN IE of STA

- AP uses STA Nonce and PMK to compute PTK

39



Message 3

AP → STA: {AA, AP Nonce, n + 1, msg3, MIC_{PTK}(AP Nonce, n + 1, msg3)}

AA: MAC Address of AP

AP Nonce: random value again

n: sequence identifier, to match msg4

msg3: Informs STA that TK ready to use, RSN IE of AP.

MIC: to verify the above. Silently discarded if MIC fails.

Verifies no MITM attack happening

40

Message 4

STA → AP: {SPA, n + 1, msg4, MIC_{PTK}(n + 1, msg4)}

SPA: MAC Address of STA

n: sequence identifier, to match msg3

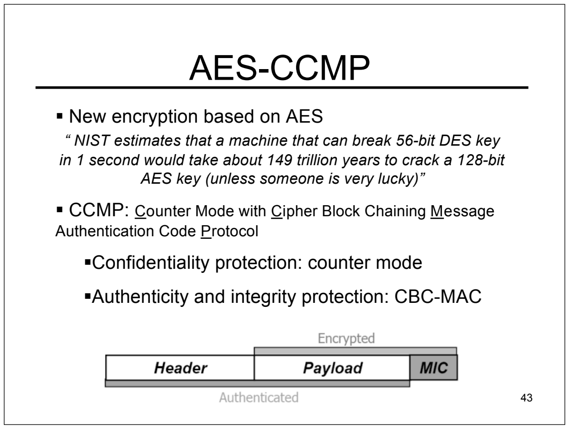
MIC: to verify the above. Silently discarded if MIC fails.

- This message dropped in some implementations.
- Only kept for convention

41

WPA Data Protection

42



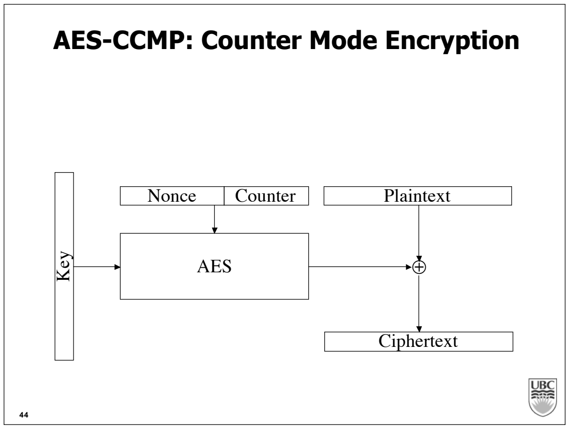
AES-CCMP

- New encryption based on AES

  " NIST estimates that a machine that can break 56-bit DES key in 1 second would take about 149 trillion years to crack a 128-bit AES key (unless someone is very lucky)"

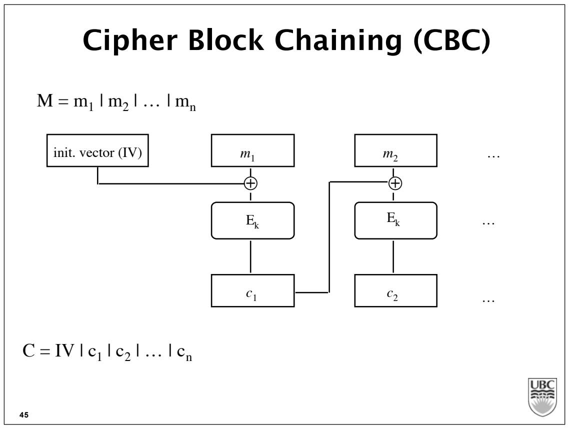
- CCMP: Counter Mode with Cipher Block Chaining Message Authentication Code Protocol
  - Confidentiality protection: counter mode
  - Authenticity and integrity protection: CBC-MAC

43



AES-CCMP: Counter Mode Encryption

44

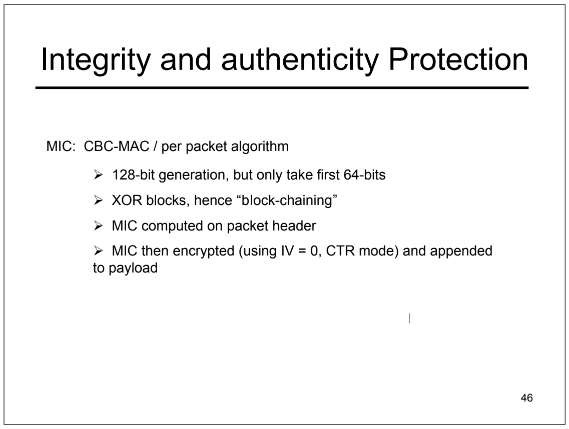


Cipher Block Chaining (CBC)

$M = m_1 \mid m_2 \mid \ldots \mid m_n$

$C = IV \mid c_1 \mid c_2 \mid \ldots \mid c_n$

45



Integrity and authenticity Protection

MIC: CBC-MAC / per packet algorithm

- 128-bit generation, but only take first 64-bits
- XOR blocks, hence "block-chaining"
- MIC computed on packet header
- MIC then encrypted (using IV = 0, CTR mode) and appended to payload

46