Breaking and Fixing

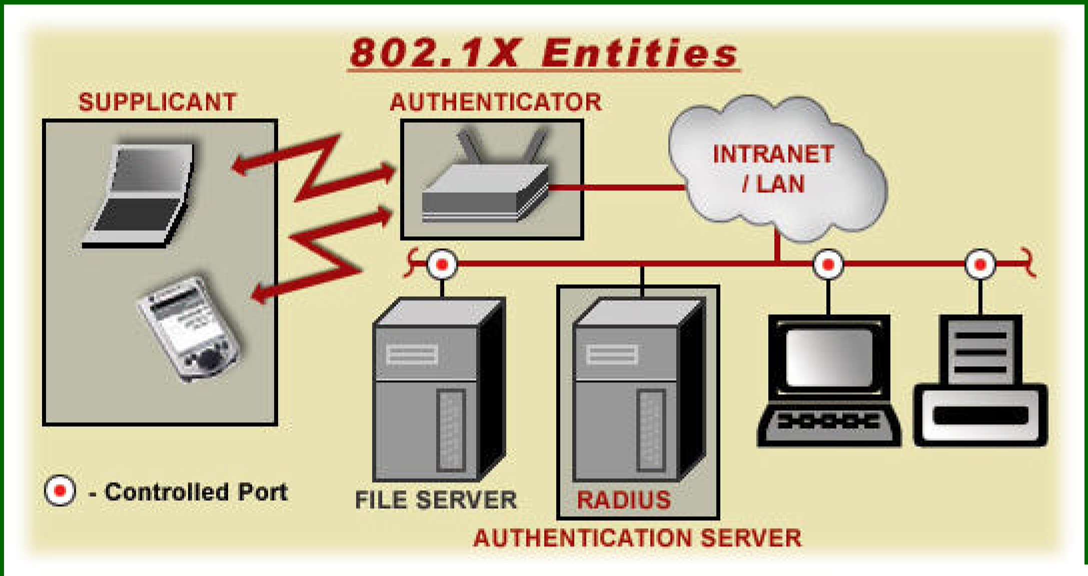
The IEEE 802.11i Wireless Networking Standard

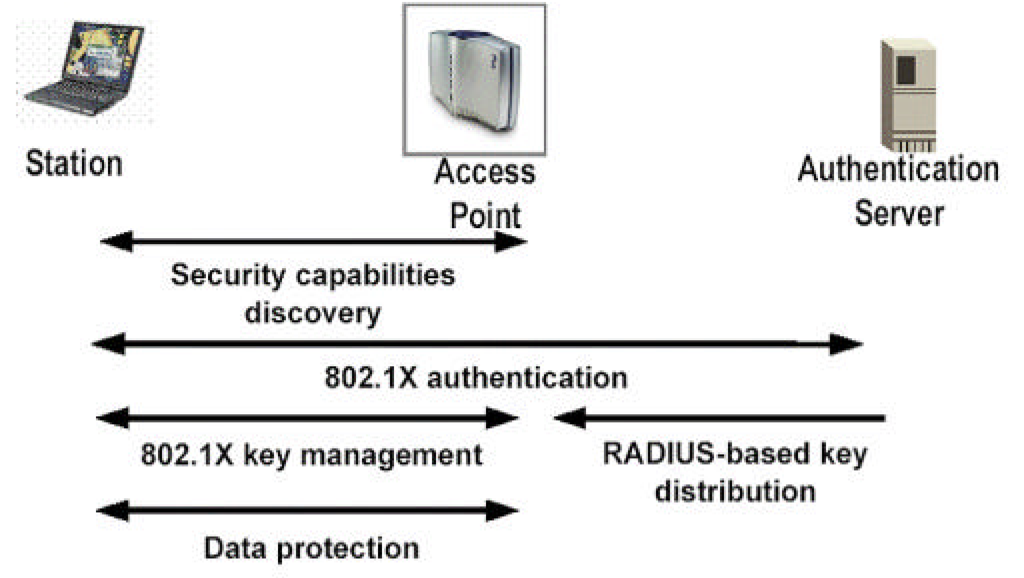
Changhua He, John Mitchell

Mukund Sundararajan, Anupam Datta, Ante Derek

Stanford University

802.11i Wireless Authentication





Wireless Threats

- Passive eavesdropping and traffic analysis
  - Easy, most wireless NICs have promiscuous mode
- Message injection and eavesdropping
  - Easy, some techniques to gen. any packet with common NIC
- Message removal
  - Possible, block packet reception with directional antennas
- Masquerading and malicious AP
  - Easy, MAC address forgeable with available s/w (HostAP)
- Session Hijacking
- Man-in-the-Middle
- Denial-of-Service
  - We propose and use a cost-related evaluation of DoS attacks

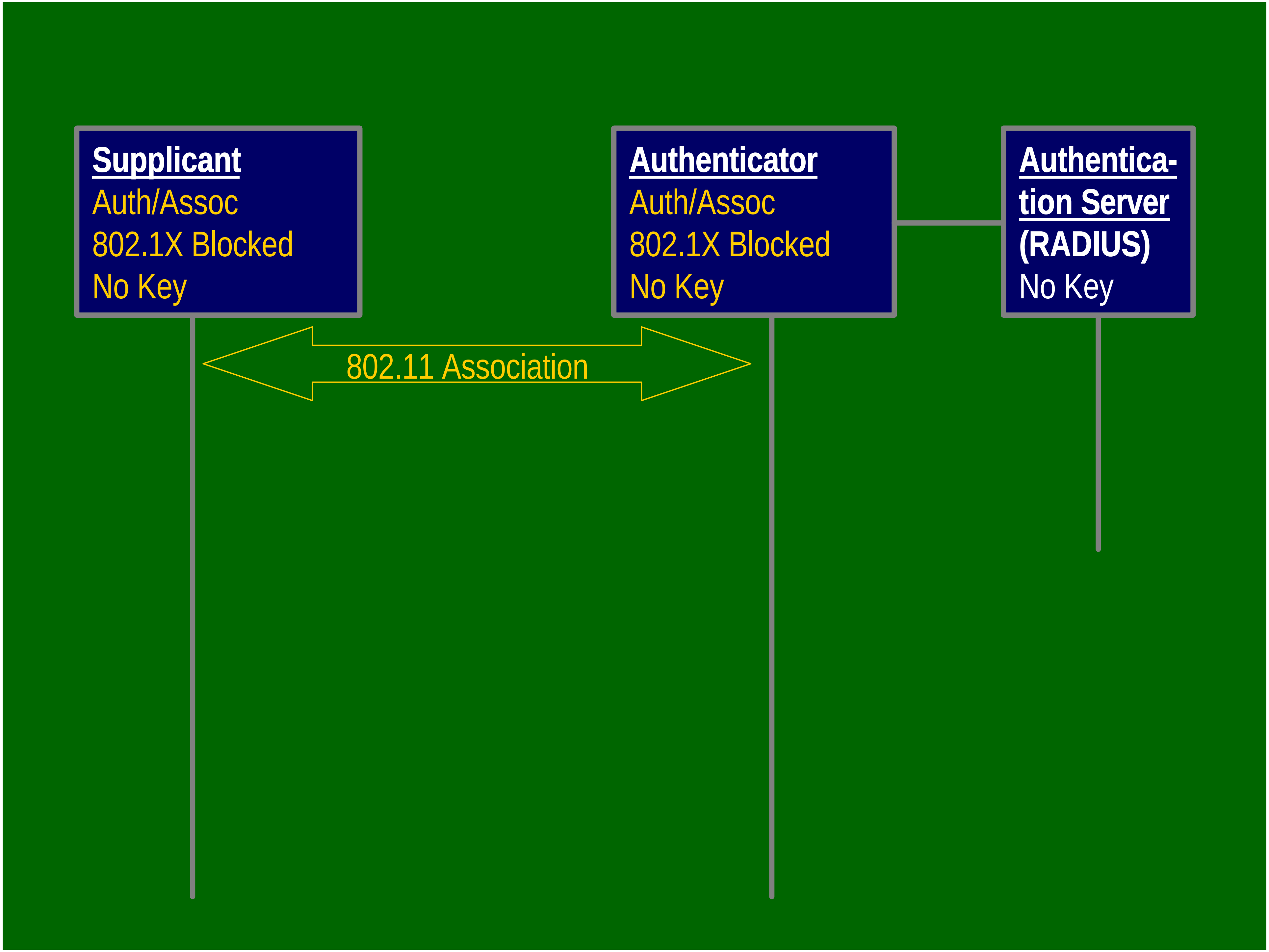
Wireless Security Evolution

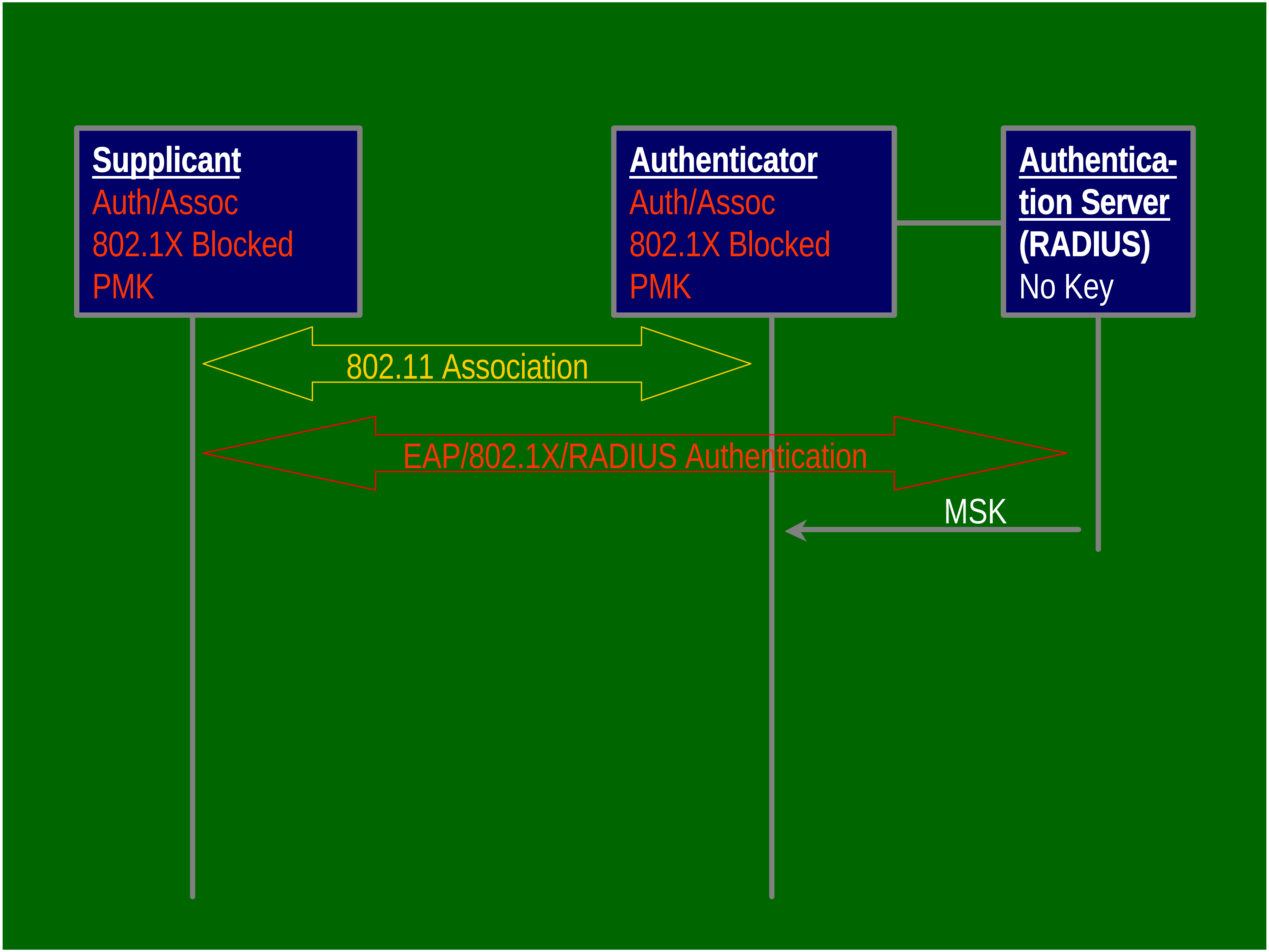
- 802.11 (Wired Equivalent Protocol)
  - Authentication: Open system (SSID) and Shared Key
  - Authorization: some vendors use MAC address filtering
  - Confidentiality/Integrity: RC4 + CRC
  - However, considered insecure – bad use of good crypto
- WPA: Wi-Fi Protected Access
  - Authentication: 802.1X
  - Confidentiality/Integrity: TKIP
  - Reuses legacy hardware, still problematic
- IEEE 802.11i (Ratified on June 24, 2004)
  - Mutual authentication, e.g., EAP/TLS
  - Data confidentiality and integrity: CCMP
  - Key management
  - Availability

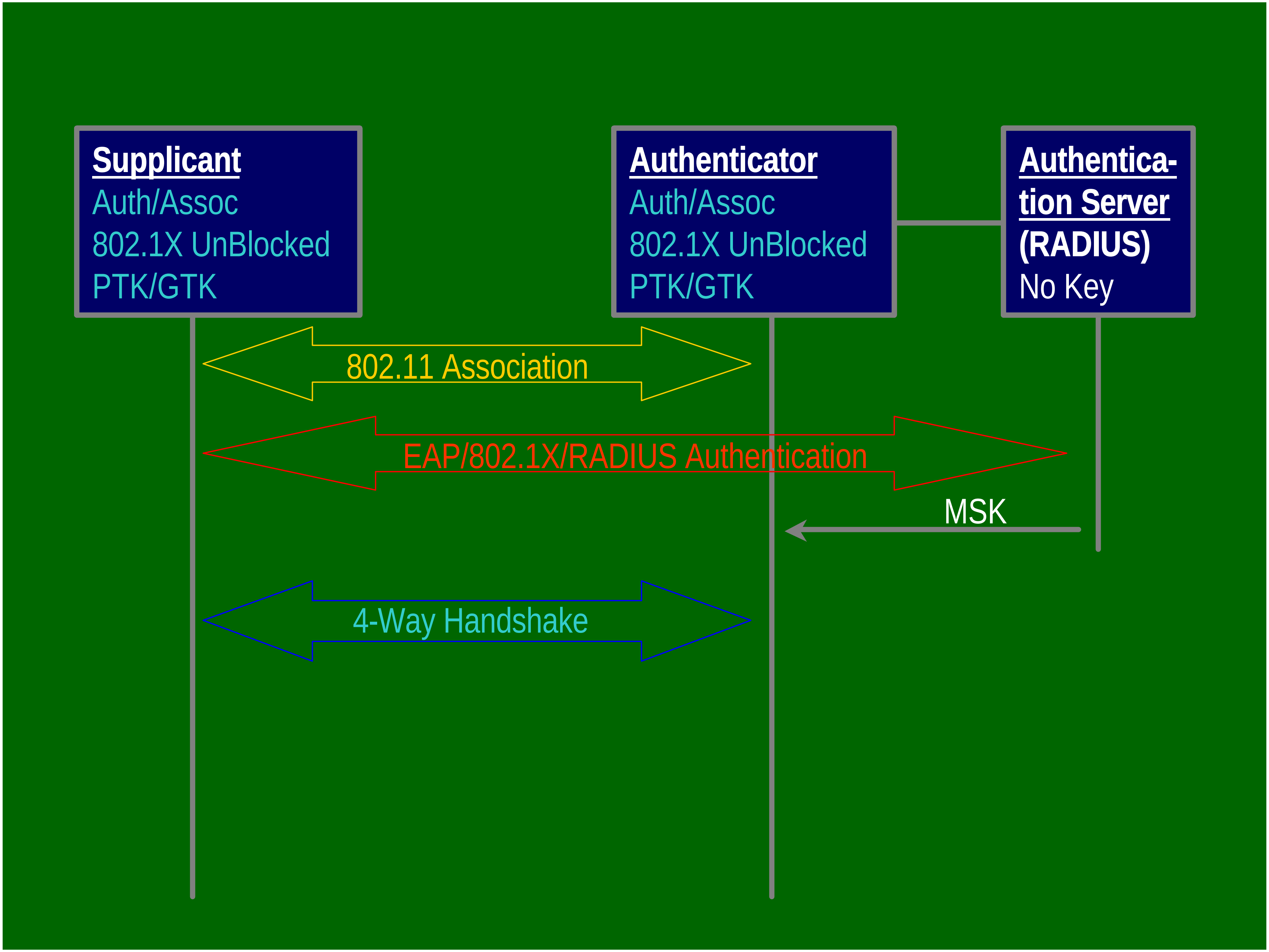
Supplicant
UnAuth/UnAssoc
802.1X Blocked
No Key

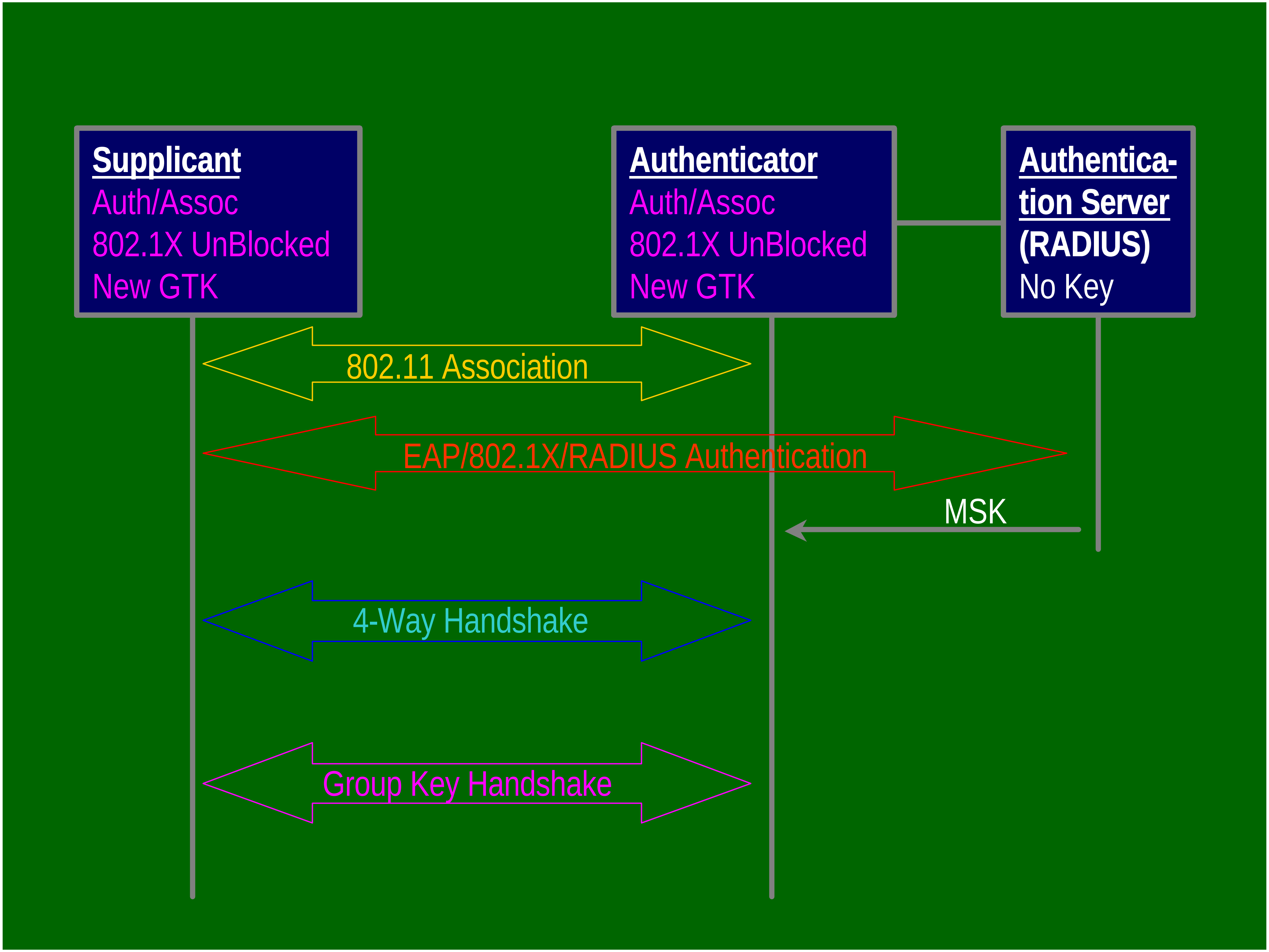
Authenticator
UnAuth/UnAssoc
802.1X Blocked
No Key

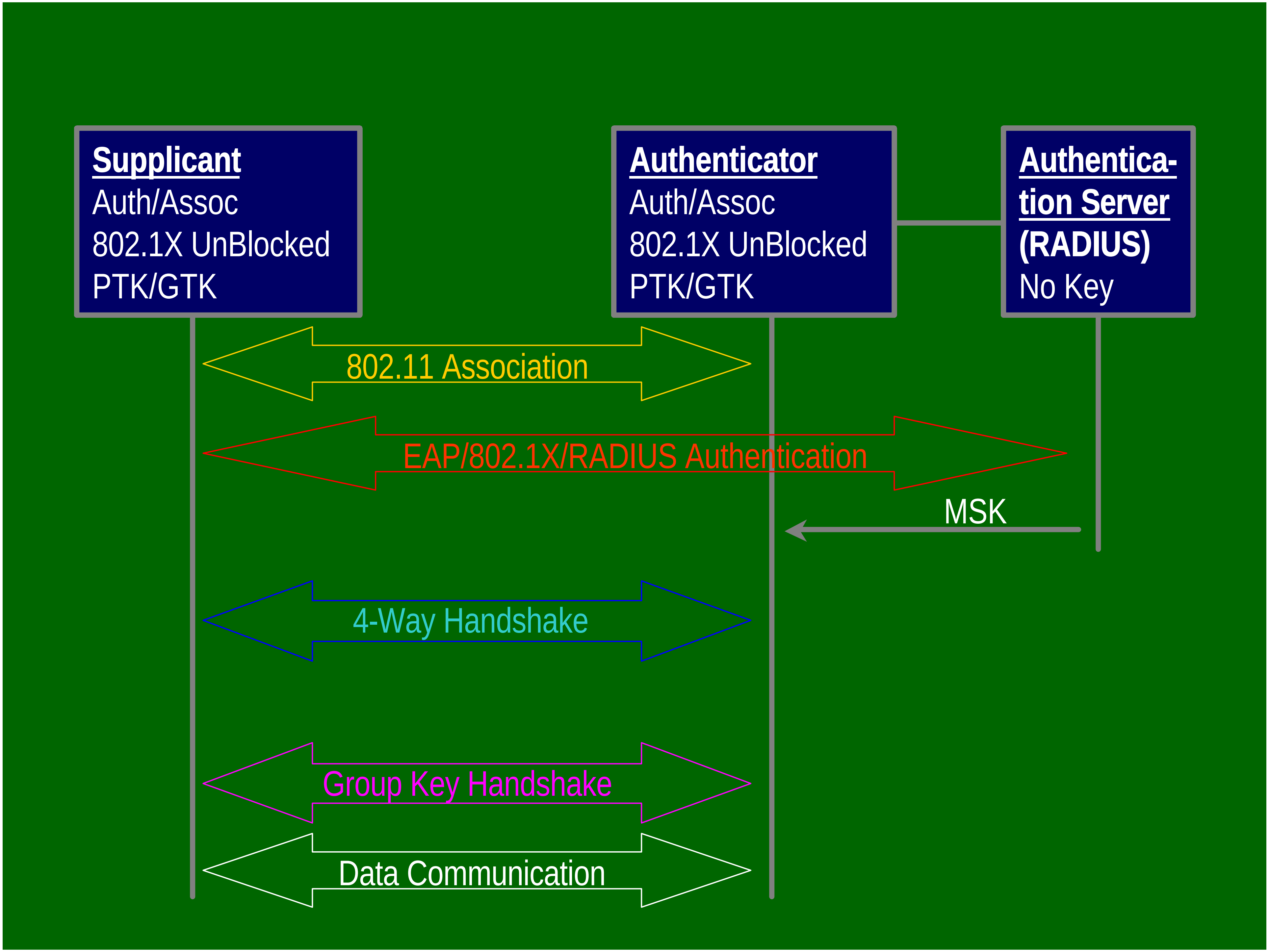
Authentica-tion Server (RADIUS)
No Key


Supplicant
Auth/Assoc
802.1X Blocked
No Key
Authenticator
Auth/Assoc
802.1X Blocked
No Key
Authentica-
tion Server
(RADIUS)
No Key
802.11 Association

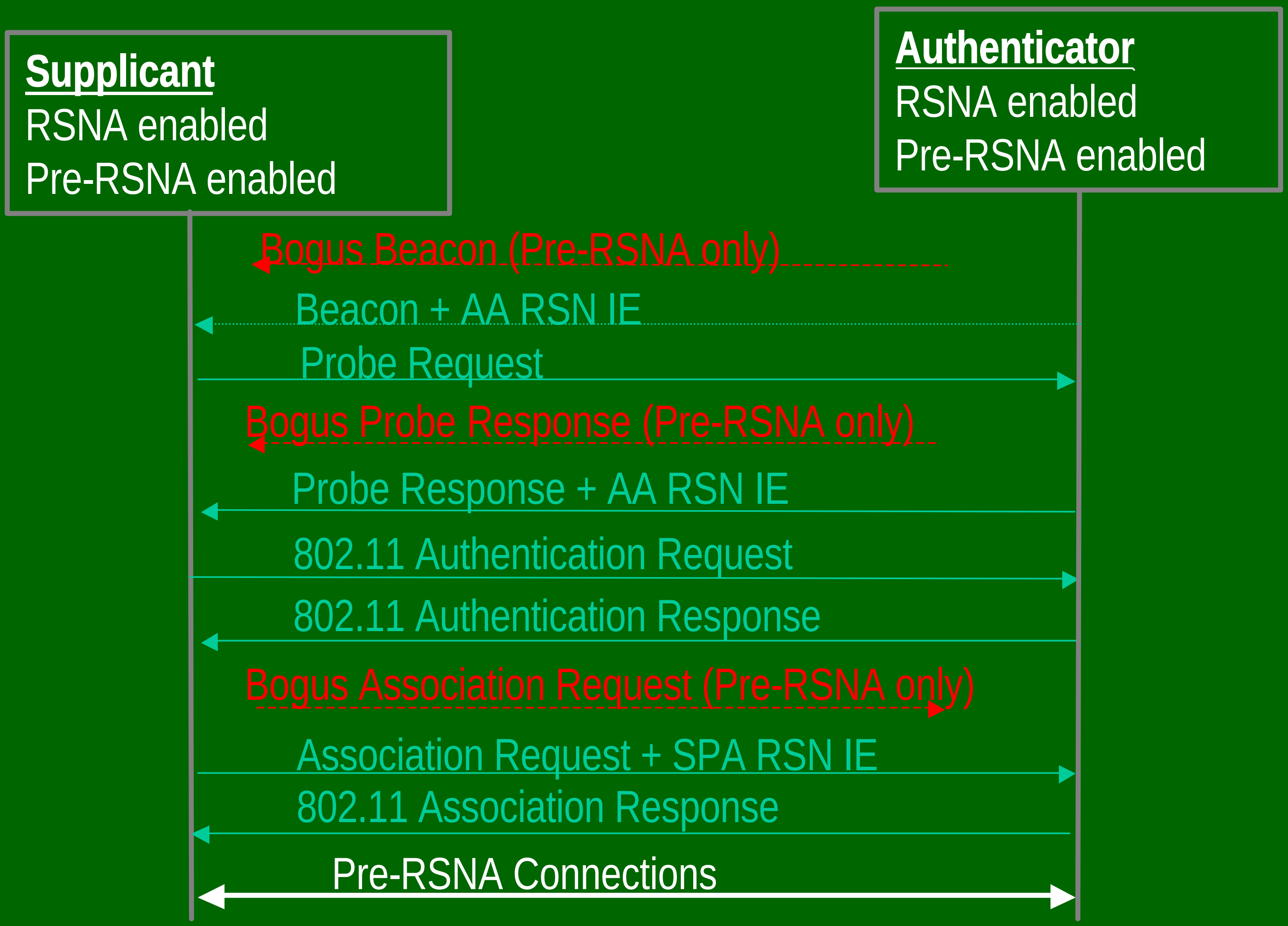

Supplicant
Auth/Assoc
802.1X Blocked
PMK
Authenticator
Auth/Assoc
802.1X Blocked
PMK
Authentica-
tion Server
(RADIUS)
No Key
802.11 Association
EAP/802.1X/RADIUS Authentication
MSK


Supplicant
Auth/Assoc
802.1X UnBlocked
PTK/GTK
Authenticator
Auth/Assoc
802.1X UnBlocked
PTK/GTK
Authentication Server
(RADIUS)
No Key
802.11 Association
EAP/802.1X/RADIUS Authentication
MSK
4-Way Handshake


Supplicant
Auth/Assoc
802.1X UnBlocked
New GTK
Authenticator
Auth/Assoc
802.1X UnBlocked
New GTK
Authentica-
tion Server
(RADIUS)
No Key
802.11 Association
EAP/802.1X/RADIUS Authentication
MSK
4-Way Handshake
Group Key Handshake


Supplicant
Auth/Assoc
802.1X UnBlocked
PTK/GTK
Authenticator
Auth/Assoc
802.1X UnBlocked
PTK/GTK
Authentica-
tion Server
(RADIUS)
No Key
802.11 Association
EAP/802.1X/RADIUS Authentication
MSK
4-Way Handshake
Group Key Handshake
Data Communication

Security Level Rollback Attack

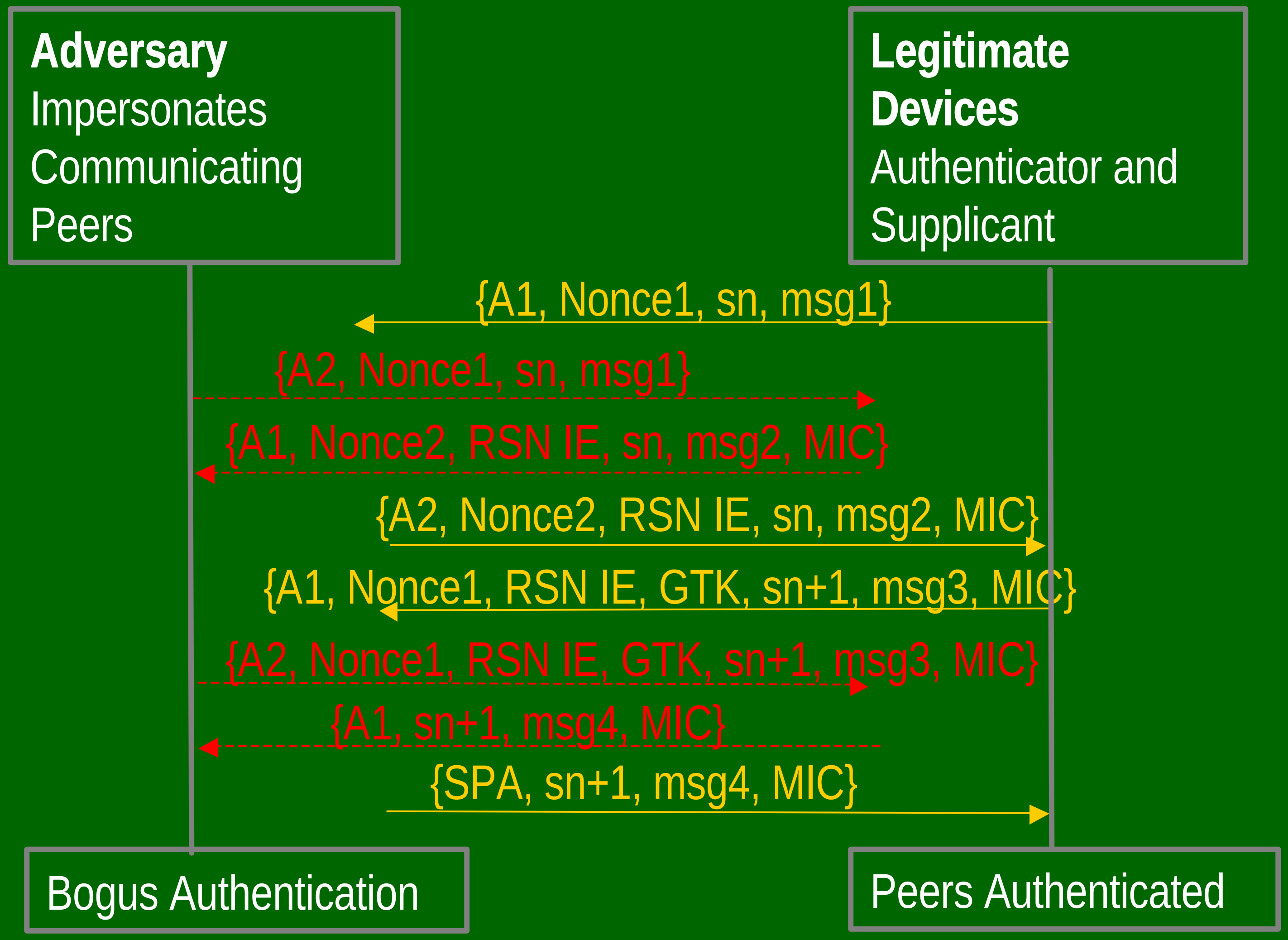


Pre-RSNA connection is INSECURE!

Rollback Solutions

- Consequences of rollback attack
  - Similar to general version rollback attack
  - Destroy security since WEP is insecure
  - Not a vulnerability of 802.11i standard per se, but an important deployment problem
- Solutions
  - Allow only RSNA connections
    - Too strict if Transient Security Network still in use
  - Deploy both, but
    - Allow supplicant to manually choose to accept or deny
    - Authenticator may limit pre-RSNA connections to less sensitive data

Reflection Attack



Reflection Solutions

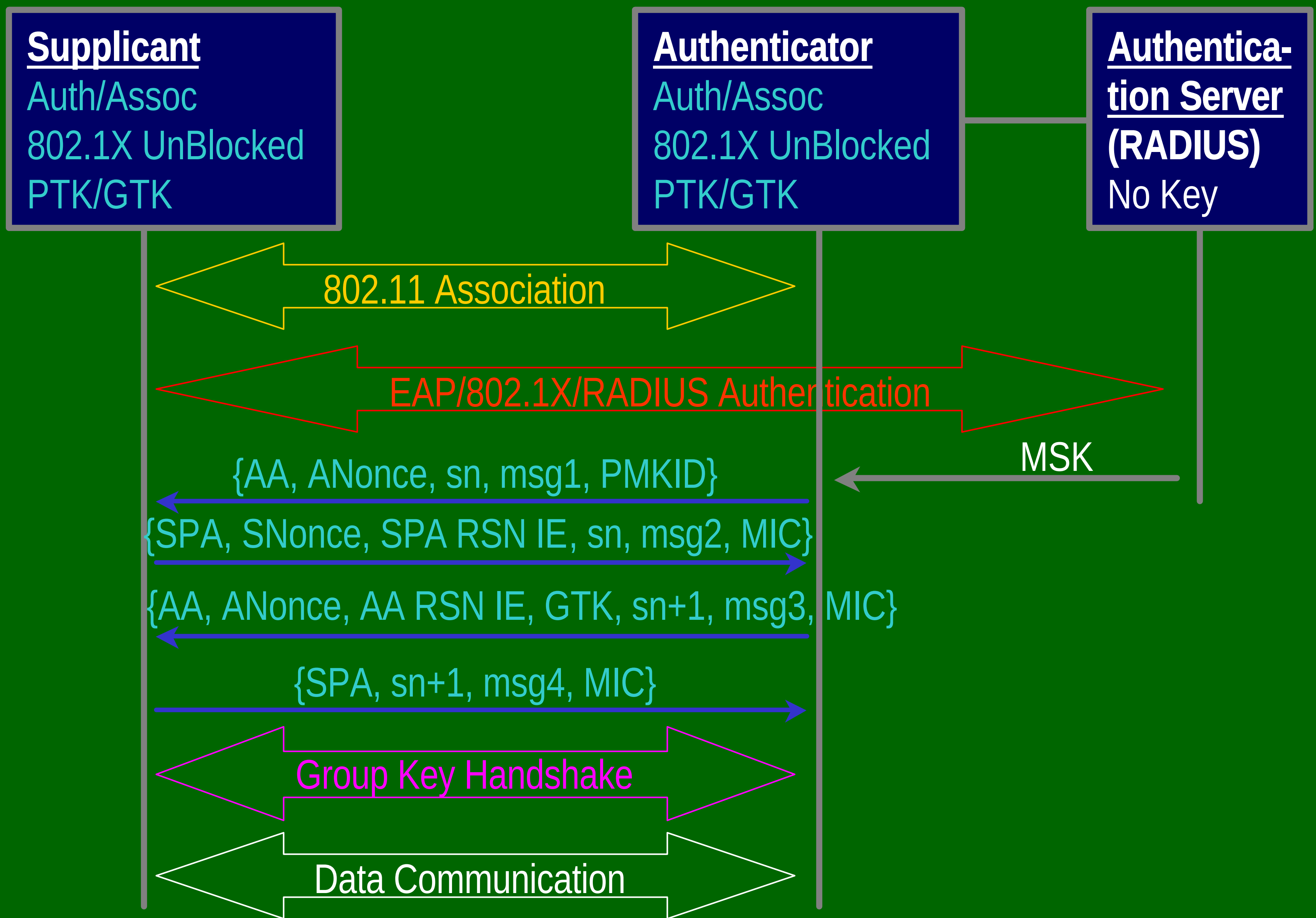
- Consequences of reflection attack
  - Possible in ad hoc networks
  - Violates mutual authentication
- Solutions:
  - Restrict each entity to single role
    - Access point should not act as wireless station
  - Allow one entity to have two roles
    - But require different pairwise master keys (PMK)

802.11i Availability

- Not an original design objective
- Physical Layer DoS attacks
  - Inevitable but detectable, not our focus
- Network and upper Layer DoS attack
  - Depend on protocols, not our focus
- Link Layer attacks
  - Flooding attack: Lots of traffic and power req'd
  - Some Known DoS attacks in 802.11 networks
  - DoS attack on Michael algorithm in TKIP
  - RSN IE Poisoning/Spoofing
  - 4-Way Handshake Blocking
  - Failure Recovery

These risks can be reduced!

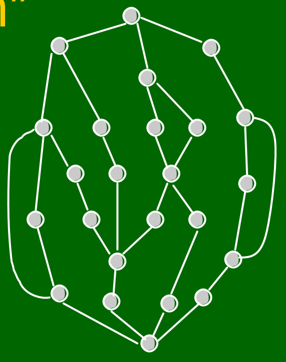
The 4-Way Handshake



Study of 4-way handshake

- Assumption
  - PMK is shared between the Supplicant and the Authenticator
  - The AS transfers the key materials to the Authenticator
- Handshake Goals
  - Confirm the possession of PMK
  - Derive a fresh session key for data transmission

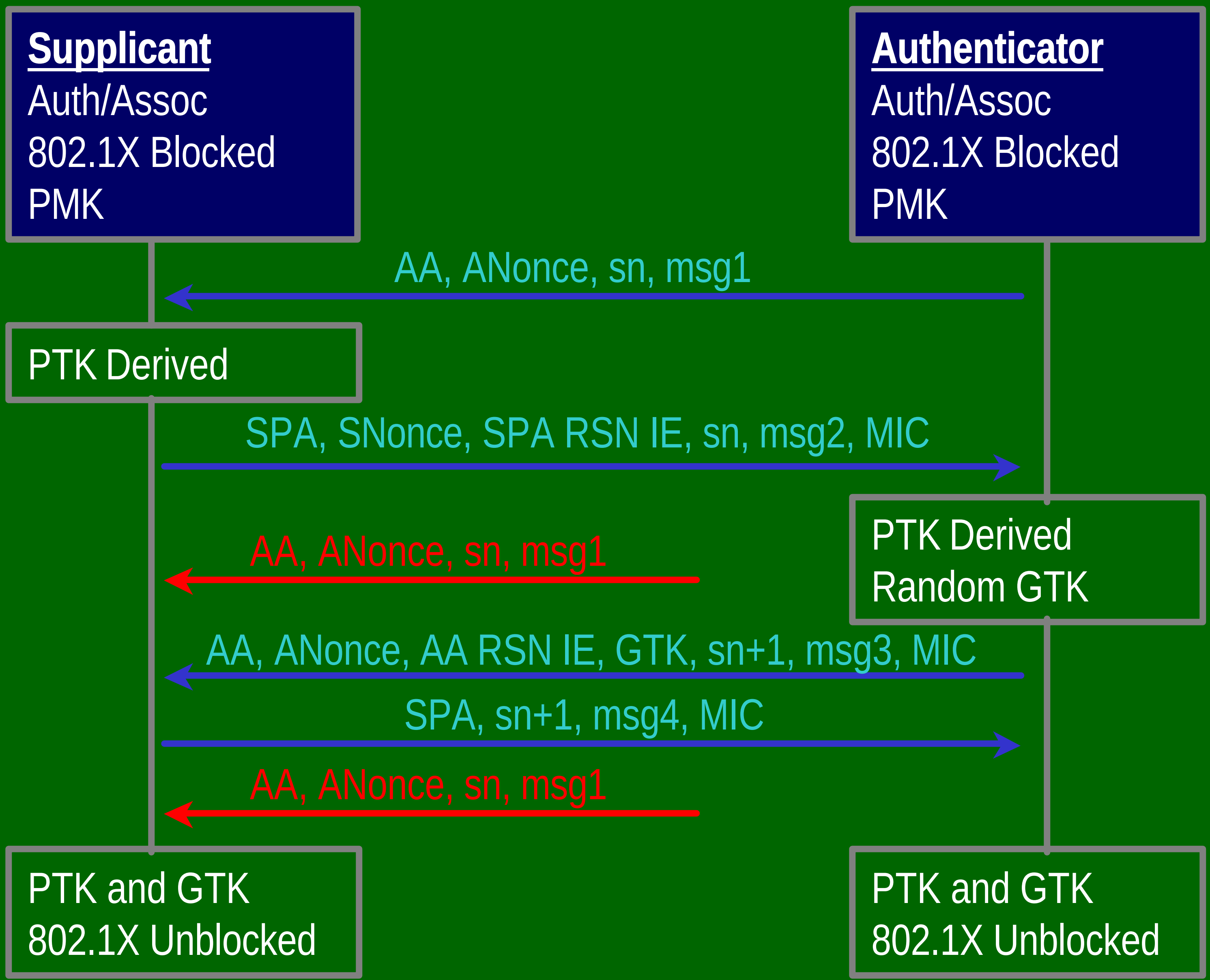
    PTK = PRF{PMK, AA||SPA||ANonce||SNonce}
- Analysis
  - Based on the 2003 specifications of the 4-way handshake
  - Murφ verification using "rationale reconstruction"



Modeling the 4-Way Handshake

- Authenticators/Supplicants:
  - Each authenticator maintain one association with each supplicant, and vice versa
  - Each association has a uniquely shared PMK
  - Multiple sequential legitimate handshakes in one association
- Intruder
  - Impersonates both supplicant and authenticator
  - Eavesdrop, intercept and replay messages
  - Compose messages with known nonce and MIC
  - Forge fresh Message 1
  - Predict and replay nonces for pre-computation of MIC
- Rationale reconstruction
  - Turn on/off fields: nonce, sequence, msg, address

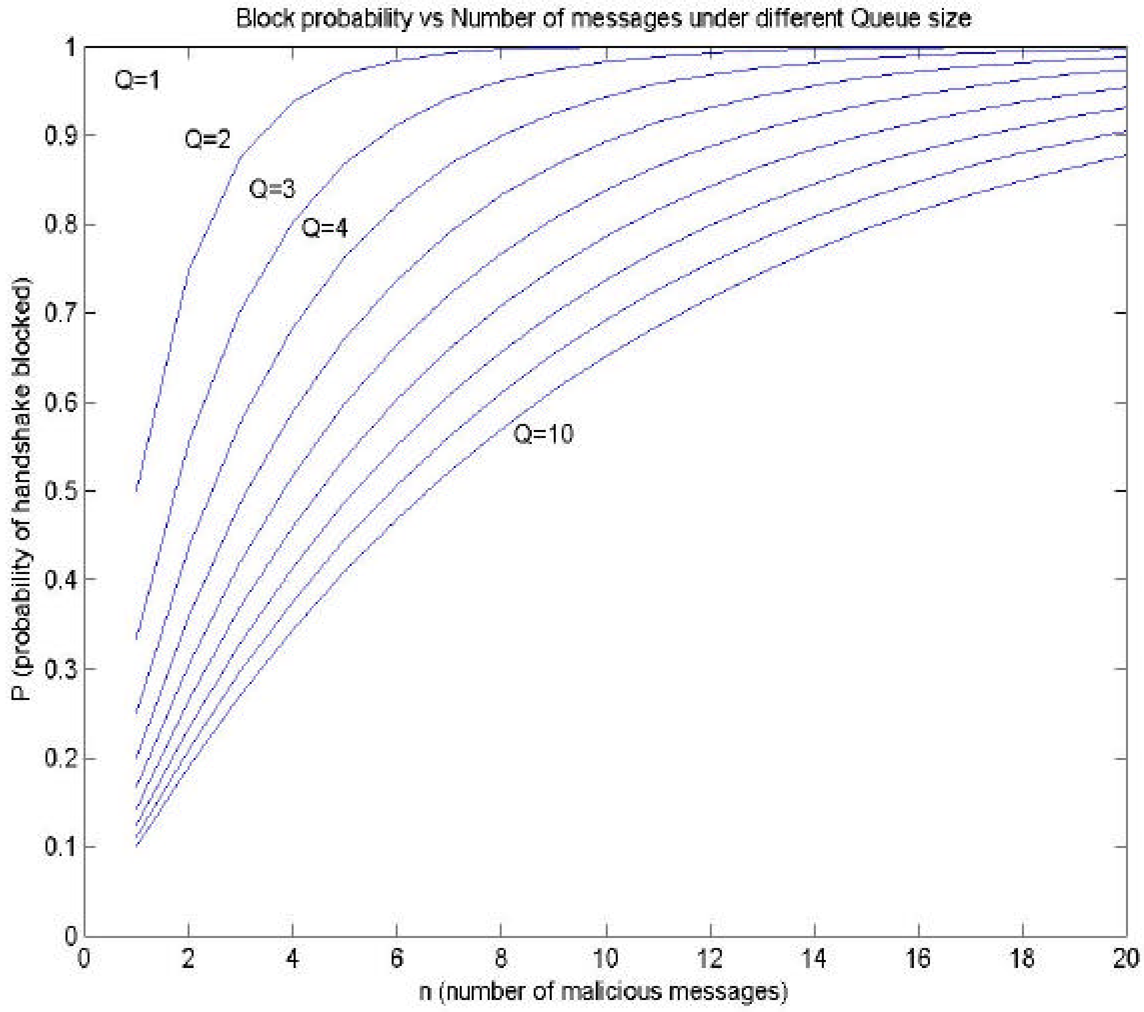
Forged Message 1 Attack



Need for “half-open” handshakes

- TPTK/PTK solution
  - Proposed in the documentation
  - Does not work for all cases
- Keep state for each Message 1 received
  - Memory/CPU exhaustion
  - Similar to TCP SYN flooding attack
- Interleaving handshakes may be required
  - Authenticator can reject unexpected messages
  - Supplicant must accept Msg 1 in all stages
  - Parallel incomplete handshakes are required

Countermeasures (1)



Random-Drop Queue:

Randomly drop a stored entry to adopt the state for the incoming Message 1 if the queue is filled.

Not effective

Countermeasures (2)

- Authenticate Message 1
  - To reuse the algorithm/hardware, set nonces to special values, e.g., 0, and derive PTK.
  - Calculate MIC for Msg 1 using the derived PTK
  - Good solution if PMK is fresh
- If PSK and cached PMK, replay attacks !
  - Add a monotonically increasing global sequence counter
  - Use local time in authenticator side
  - Sufficient space in Message 1 (8-octet sequence field)
  - No worry about time synchronization

Requires modified packet format

Countermeasures (3)

- Reuse Nonce
  - Supplicant reuses SNonce until one 4-way handshake completes successfully
  - Derive correct PTK from Message 3
  - Authenticator may (or may not) re-use ANonce
- Solves the problem, but
  - Attacker might gather more information about PMK by playing with Message 1, recall

    PTK=PRF{PMK, AA||SPA||ANonce||SNonce}
  - May require significant computation in the supplicant

Performance Degradation

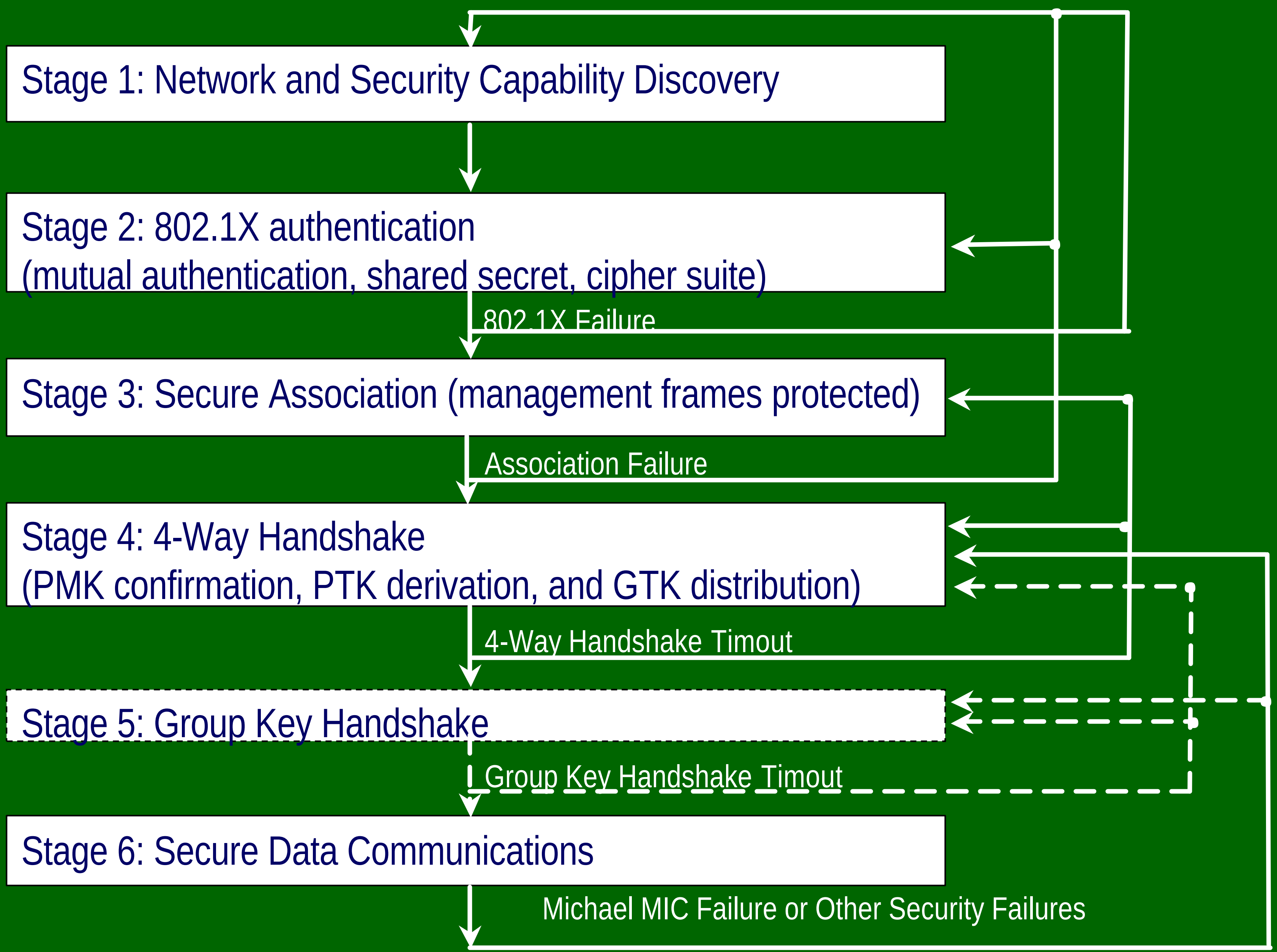
Our Proposal

- Combined solution
  - Supplicant reuse SNonce
  - Store one entry of ANonce and PTK for the first Message 1
  - If nonce in Message 3 matches the entry, use PTK directly; otherwise derive PTK again and use it.
- Advantages
  - Eliminates the memory DoS attack
  - Ensure performance in "friendly" scenarios
  - Only minor modification to the Supplicant algorithm
    - No modifications on the packet format
- Adopted by TGi

802.11i Failure Recovery

- Failure recovery is important
  - Can reduce but not eliminate DoS vulnerabilities
- Current 802.11i method
  - When failure, restart everything: inefficient
- *Better* failure approach
  - If 802.1X does not finish, restart everything
  - Otherwise restart from nearest completed subprotocol
  - ★ Channel scanning time is significantly larger than the protocol execution time

Improved 802.11i



Summary of recommendations

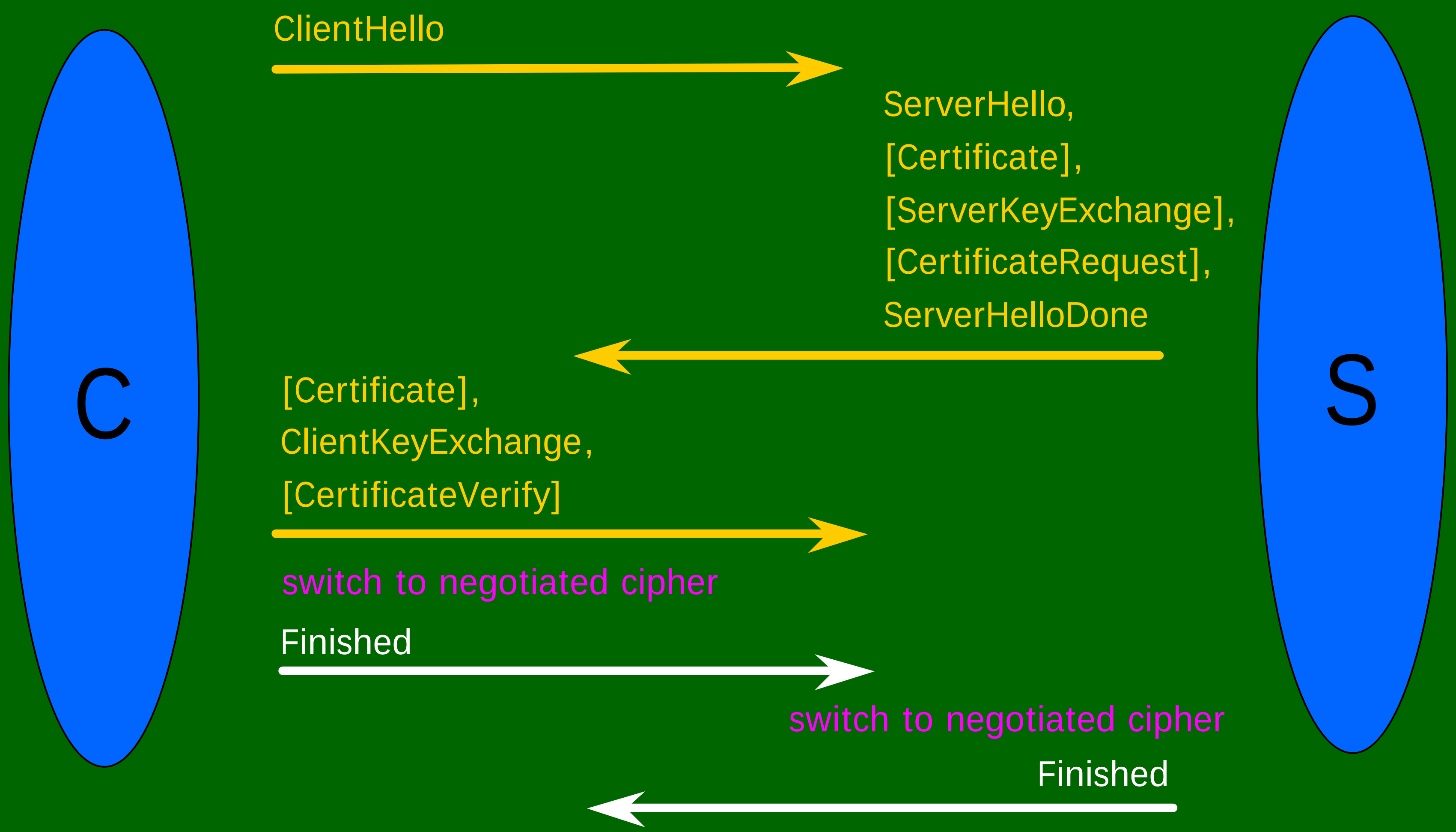
Attack	Solution
security rollback	supplicant *manually* selects security protocol; restricted acces for pre-RSNA connections
reflection attack	authenticator and supplicant must be different devices or use different keys (PMK).
attack on Michael countermeasures	freeze connection for a specific time instead of re-keying and deauthentication; update TSC before MIC and after FCS, ICV are validated.
RSN IE poisoning	Authenticate Beacon and Probe Response frames; Confirm RSN IE in early step but relax the condition of RSN IE confirmation.
4-way handshake blocking	Change supplicant algorithm to reuse a single nonce on all half-open connections, eliminating memory DoS attack

802.11i correctness proof in PCL

- EAP-TLS
  - Between Supplicant and Authentication Server
  - Authorizes supplicant and establishes access key (PMK)
- 4-Way Handshake
  - Between Access Point and Supplicant
  - Checks authorization, establish key (PTK) for data transfer
- Group Key Protocol
  - AP distributes group key (GTK) using KEK to supplicants
- AES based data protection using established keys

Our proof covers subprotocols 1, 2, 3 alone and in various combinations

SSL/TLS



ClientHello
ServerHello,
[Certificate],
[ServerKeyExchange],
[CertificateRequest],
ServerHelloDone
C
S
[Certificate],
ClientKeyExchange,
[CertificateVerify]
switch to negotiated cipher
Finished
switch to negotiated cipher
Finished

TLS Protocol: Client

$$
\begin{aligned}
\text{Client} \quad = \quad & (X, \hat{Y}, VerSUx)[\\
& \texttt{new}\ Nx;\ \texttt{send}\ \hat{X}, \hat{Y}, \hat{X}, Nx, VerSUx; \\
& \texttt{receive}\ \hat{Y}, \hat{X}, Ny, VerSUy, cert;\ \texttt{match}\ cert/SIG_{\hat{CA}}(\hat{Y}, Ky); \\
& \texttt{new}\ secret; \\
& \texttt{send}\ (\hat{X}, \hat{Y}, SIG_{\hat{CA}}(\hat{X}, Vx), SIG_{Vx}(handShake), ENC_{Ky}(secret), HASH_{secret}(handShake, ''client''); \\
& \texttt{receive}\ \hat{Y}, \hat{X}, hash;\ \texttt{match}\ hash/HASH_{secret}(handShake, ''server'');
\end{aligned}
$$

The TLS Server actions also defined by a straight-line sequential process (cord)

TLS Properties

- Authentication: client and the server agree on
  - Master secret
  - Protocol version and crypto suite
  - Each other's identities
  - Protocol completion status
- Secrecy
  - The master secret must *not* be known to any other principal

Theorems: Agreement and Secrecy

$$\text{Honest}(\hat{X}) \wedge \text{Honest}(\hat{Y}) \wedge \text{Honest}(\hat{CA}) \wedge \hat{X} \neq \hat{Y}$$
$$[\textbf{Client}]_X$$
$$\exists Y.(\text{Send}(X, \hat{X}, \hat{Y}, m1)$$
$$< \text{Receive}(Y, \hat{X}, \hat{Y}, m1)$$
$$< \text{Send}(Y, \hat{Y}, \hat{X}, m2)$$
$$< \text{Receive}(X, \hat{Y}, \hat{X}, m2)$$
$$< \text{Send}(X, \hat{X}, \hat{Y}, m3)$$
$$< \text{Receive}(Y, \hat{X}, \hat{Y}, m3)$$
$$< \text{Send}(Y, \hat{Y}, \hat{X}, m4)$$
$$< \text{Receive}(X, \hat{Y}, \hat{X}, m4))$$

$$\text{Honest}(\hat{Y})[\textbf{Client}]_X$$
$$\text{Has}(\hat{Z}, secret)$$
$$\wedge \hat{X} \neq \hat{Z} \supset \hat{Z} = \hat{Y}$$

Client is guaranteed:

- there exists a session of the intended server
- this server session agrees on the values of all messages
- all actions viewed in same order by client and server
- there exists exactly one such server session

Similar specification for server

Invariants required by TLS

$$\Gamma_{tls,2} := (\mathsf{Decrypts}(Y, ENC_{K_Y}(y)) \wedge \mathsf{Contains}(y, secret) \wedge \mathsf{Send}(Y, m) \wedge \mathsf{Contains}(m, secret)) \supset \neg\mathsf{ContainsOut}(secret, m, ENC_{K_Y}(y))$$

Server Side Recommendation: If the server reuses Public Key in a protocol different from TLS, then it should not send decryptions of incoming messages

4-Way handshake: Authenticator

$(X, \hat{Y}, K_{\hat{X}\hat{Y}})$
[new x;
send $\hat{X}, \hat{Y}, x, msg1$;
receive $\hat{Y}, \hat{X}, z$;
match $z/y, msg2, Hash(Hash(K_{\hat{X}\hat{Y}}, x, y), y, msg2)$;
send $\hat{X}, \hat{Y}, x, msg3, Hash(Hash(K_{\hat{X}\hat{Y}}, x, y), x, msg3)$;
receive $\hat{Y}, \hat{X}, z$;
match $z/msg4, Hash(Hash(K_{\hat{X}\hat{Y}}, x, y), msg4)]_X$

Supplicant actions also defined by a straight-line sequential process (cord)

4-Way Handshake properties

- The pairwise key (PTK) is fresh and correctly generated from the PMK
- Messages 2 and 4 assure authenticator that supplicant messages are current (not replay)
- Message 3 assures supplicant that authenticator messages are current (not replay)
- Pairwise key PTK derivation produces shared secret between supplicant and authenticator

4-way Handshake Properties

$$
\begin{aligned}
\phi_1(Key\ Secrecy) &::= \\
\mathsf{Honest}(\hat{X}) \wedge \mathsf{Honest}(\hat{Y}) &\supset \\
((\mathsf{Has}(\hat{Z}, Hash(K_{\hat{X}\hat{Y}}, x, y)) \supset \hat{Z} = \hat{X} \vee \hat{Z} = \hat{Y})) &\wedge \\
\mathsf{Has}(\hat{X}, Hash(K_{\hat{X}\hat{Y}}, x, y)) \wedge \mathsf{Has}(\hat{Y}, Hash(K_{\hat{X}\hat{Y}}, x, y)) &\\
\phi_2(Session\ Authentication) &::= \\
\mathsf{Send}(X, \hat{X}, \hat{Y}, x, msg1) &< \\
\mathsf{Receive}(Y, \hat{X}, \hat{Y}, x, msg1) &< \\
\mathsf{Send}(Y, \hat{Y}, \hat{X}, y, msg2, Hash(Hash(K_{\hat{X}\hat{Y}}, x, y), y, msg2)) &< \\
\mathsf{Receive}(X, \hat{Y}, \hat{X}, y, msg2, Hash(Hash(K_{\hat{X}\hat{Y}}, x, y), y, msg2)) &< \\
\mathsf{Send}(X, \hat{X}, \hat{Y}, x, msg3, Hash(Hash(K_{\hat{X}\hat{Y}}, x, y), x, msg3)) &< \\
\mathsf{Receive}(Y, \hat{X}, \hat{Y}, x, msg3, Hash(Hash(K_{\hat{X}\hat{Y}}, x, y), x, msg3)) &< \\
\mathsf{Send}(Y, \hat{Y}, \hat{X}, msg4, Hash(Hash(K_{\hat{X}\hat{Y}}, x, y), msg4)) &< \\
\mathsf{Receive}(X, \hat{Y}, \hat{X}, msg4, Hash(Hash(K_{\hat{X}\hat{Y}}, x, y), msg4)) &
\end{aligned}
$$

Similar specification for server

4-Way : Relating invariants to deployment

$$\Gamma_{4way,3} := (\mathsf{Honest}(\hat{X}) \wedge \mathsf{Receive}(X, Message\ 1) \supset \neg\mathsf{Send}(X, Message\ 3)) \wedge$$
$$(\mathsf{Honest}(\hat{X}) \wedge \mathsf{Send}(X, Message\ 1) \supset \neg(\mathsf{Send}(X, Message\ 2) \wedge \mathsf{Send}(X, Message\ 4)))$$

Recommendation: One Principal should not act as both authenticator and supplicant ! Otherwise, reflection attack. Consider careful deployment in Sensor Network scenarios

Group-Key Protocol

$$
\begin{aligned}
\text{GroupKey : Client} \quad = \quad & (X, \hat{Y}, CurrSeqNo, K_{XY})[\\
& \texttt{receive } \hat{Y}, \hat{X}, x; \\
& \texttt{match } x/AA, Succ(CurrSeqNo), grp1, ENC_{XY}(GTK), \\
& HASH_{K_{XY}}(AA, Succ(CurrSeqNo), grp1, ENC_{XY}(GTK)); \\
& \texttt{send } \hat{X}, \hat{Y}, SPA, Succ(CurrSeqNo), grp2, ENC_{XY}(GTK), \\
& HASH_{K_{XY}}(SPA, Succ(CurrSeqNo), grp2, ENC_{XY}(GTK)); \;]_X \\
\text{GroupKey : Server} \quad = \quad & (Y, \hat{X}, CurrSeqNo, K_{XY})[\\
& \texttt{send } \hat{Y}, \hat{X}, AA, Succ(CurrSeqNo), grp1, ENC_{XY}(GTK), \\
& HASH_{K_{XY}}(AA, Succ(CurrSeqNo), grp1, ENC_{XY}(GTK)); \\
& \texttt{receive } \hat{X}, \hat{Y}, y; \\
& \texttt{match } y/SPA, Succ(CurrSeqNo), grp2, ENC_{XY}(GTK), \\
& HASH_{K_{XY}}(SPA, Succ(CurrSeqNo), grp2, ENC_{XY}(GTK));]_Y
\end{aligned}
$$

Group key handshake

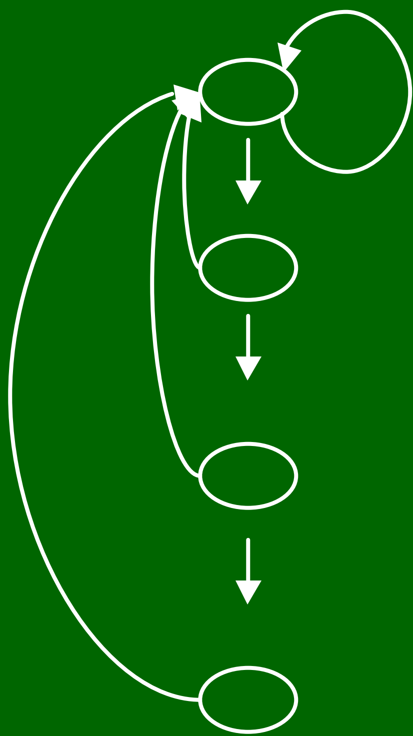
- Authenticator guarantee: If principal has the group key, then it must have a shared PTK with the authenticator
- Supplicant guarantee: the GTK received was transmitted by the Access Point, and correctly supersedes any GTK from earlier handshakes (4-Way or Group Key)

Observation: For assurance of GTK freshness, important that the first handshake uses 4-Way protocol; one principal should not be authenticator and supplicant, as in the 4-way handshake.

Composition

- All necessary invariants are satisfied by basic blocks of all the sub-protocols
- The postconditions of TLS imply the preconditions of the 4-Way handshake
- The postconditions of 4-Way handshake imply the preconditions of the Group Key protocol

Complex Control Flows



Simple Flow

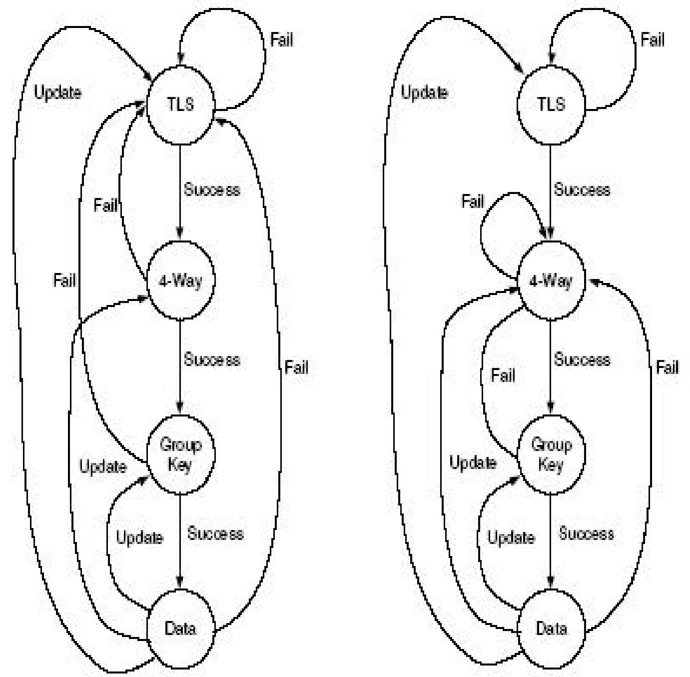


Figure 1: The Control Flow of 802.11i RSNA Establishment Procedure

Complex Flow

Use Staged Composition Theorem

- If the preconditions of each subprotocol are preserved by subsequent subprotocols, then preconditions will hold at entry of each subprotocol

802.11i: All error recovery methods shown are correct, so error handling can be chosen according to deployment conditions.

Composition also guarantees correctness of hybrid modes of deployment, such as pre-shared Key, and cached PMK options.

Conclusions

- Protocol analysis methods
  - Model checking is fairly easy to apply
  - Ready for industrial use
- Wireless 802.11i improvements
  - Automated study led to improved standard
  - Deployment recommendations also
- Correctness proof in PCL
  - Correctness proof for TLS, 4Way, GroupKey
  - Staged composition theorem
  - Covers implementation and configuration options

Future protocol studies

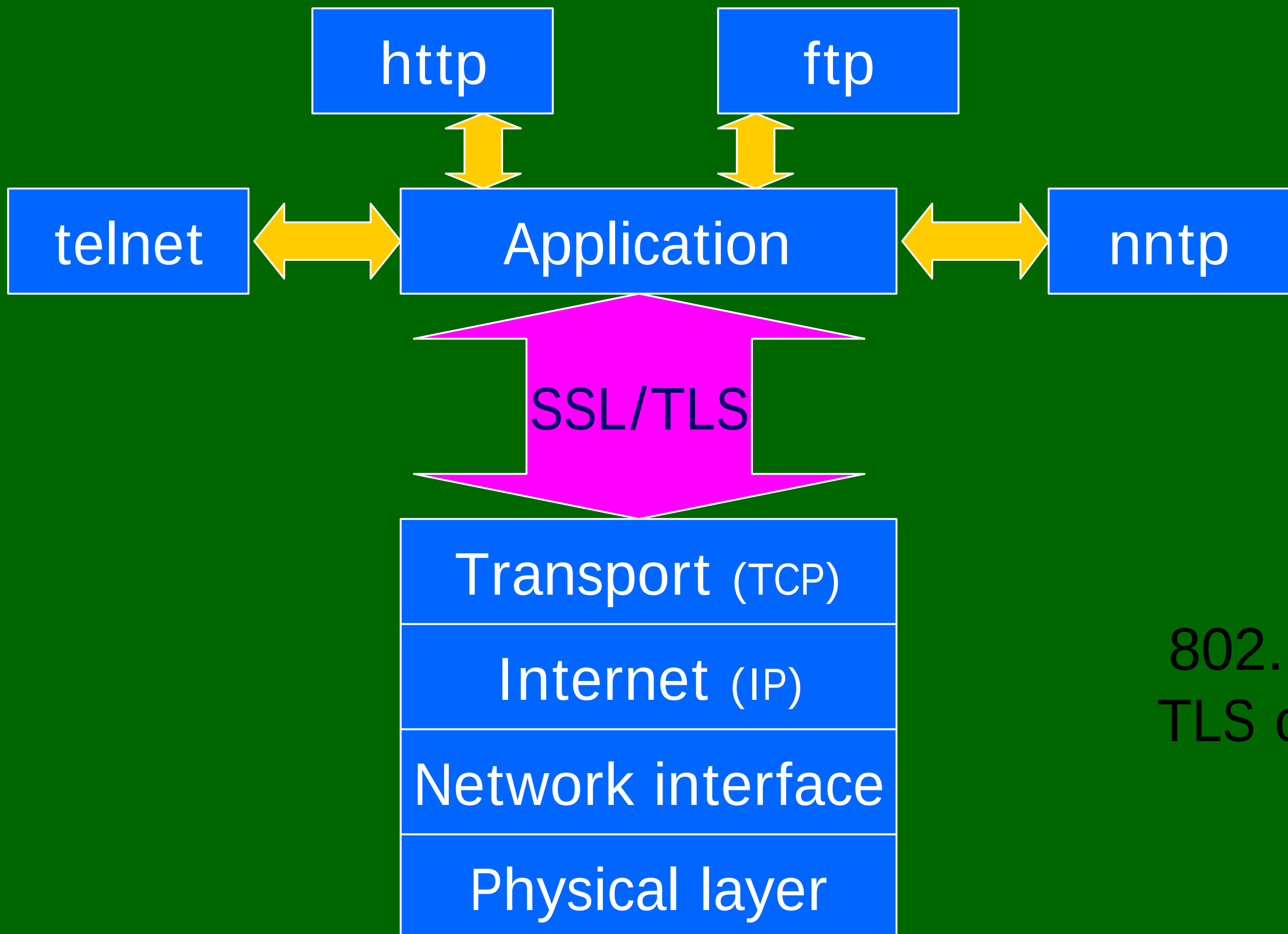
- Mobility
  - IPv6 binding update to eliminate triangle routing
- Wireless 802.16e

  -----Original Message-----
  From: Bernard Aboba [mailto:aboba@internaut.com]
  Sent: Thursday, May 05, 2005 11:14 AM
  To: mitchell@cs.stanford.edu
  Subject: Security review of 802.16e?

  The Chair of 802.16 (Roger Marks of NIST) has sent a liaison letter to the IETF asking for a security review of 802.16e, among other things. I'm looking to organize a group to do the review...

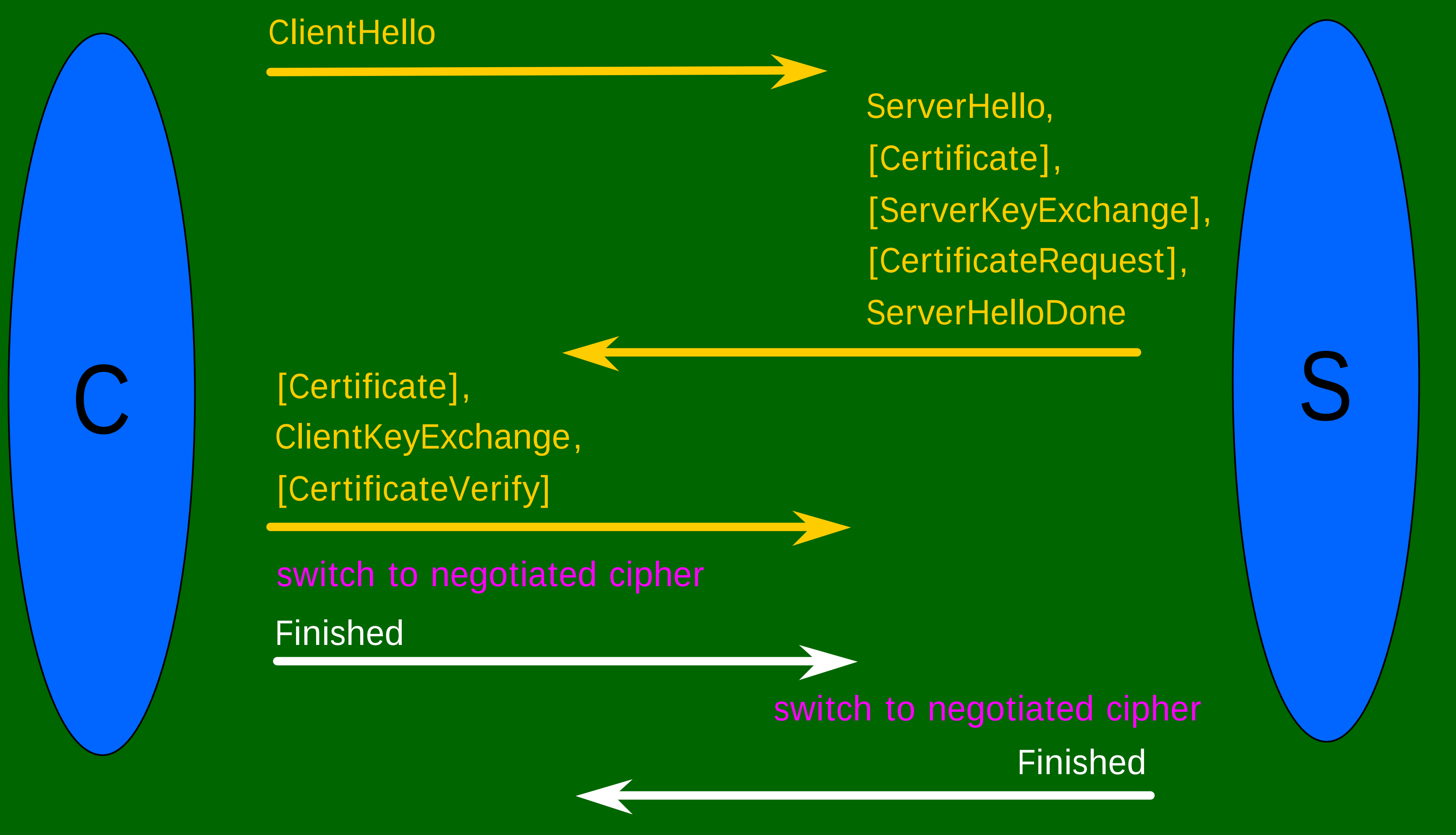
Case studies: find errors, fix protocol, derive correctness proof

TLS protocol layer over TCP/IP



802.11i uses
TLS over EAP

SSL/TLS


ClientHello
ServerHello,
[Certificate],
[ServerKeyExchange],
[CertificateRequest],
ServerHelloDone
C
S
[Certificate],
ClientKeyExchange,
[CertificateVerify]
switch to negotiated cipher
Finished
switch to negotiated cipher
Finished

Handshake Protocol

ClientHello $C \rightarrow S$ $C, Ver_C, Suite_C, N_C$

ServerHello $S \rightarrow C$ $Ver_S, Suite_S, N_S, sign_{CA}\{ S, K_S \}$

ClientVerify $C \rightarrow S$ $sign_{CA}\{ C, V_C \}$

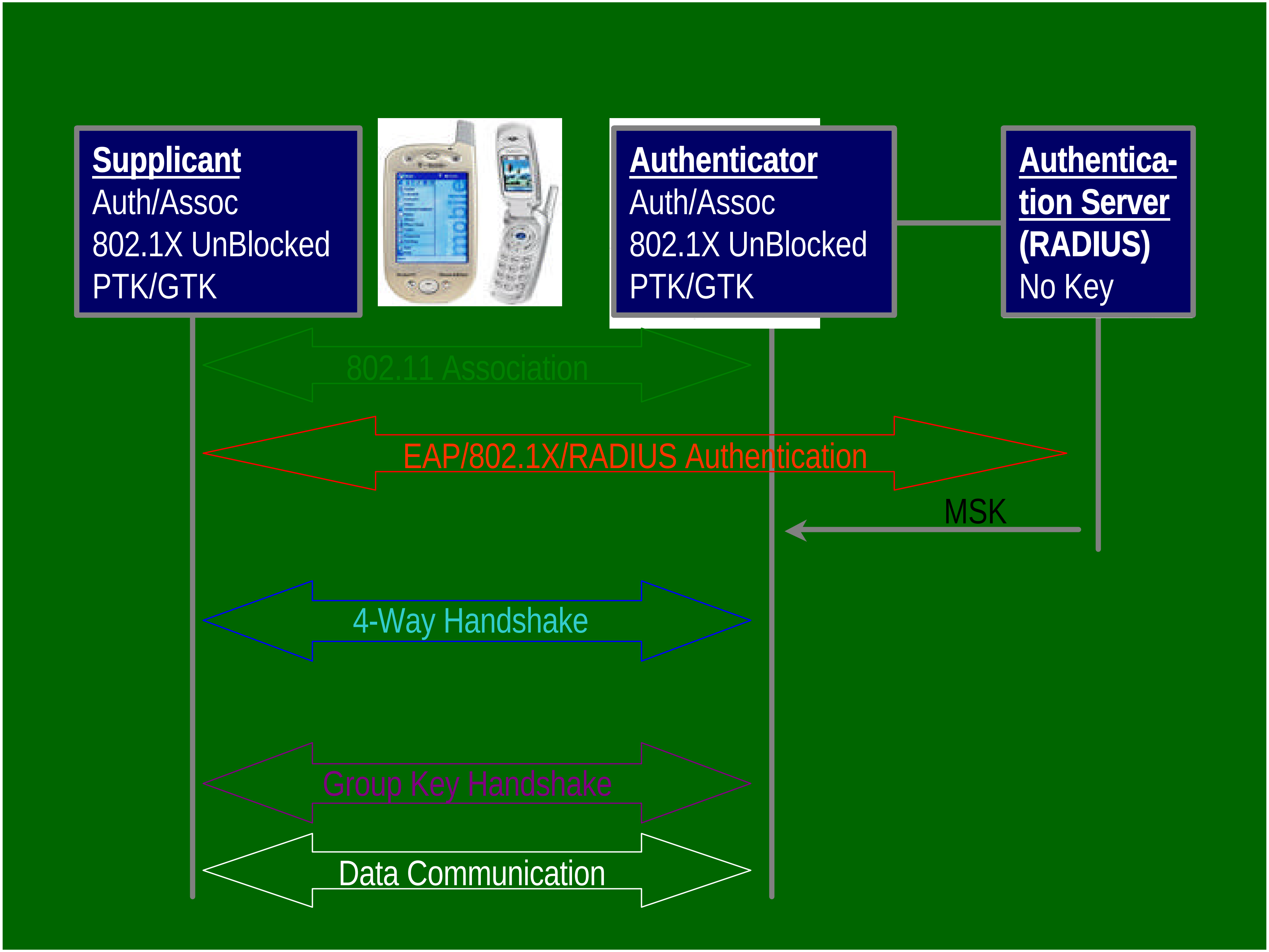
$\{ Ver_C, Secret_C \}_{K_S}$

$sign_C \{ Hash(Master(N_C, N_S, Secret_C) + Pad_2 +$
$Hash(Msgs + C + Master(N_C, N_S, Secret_C) + Pad_1)) \}$

(Change to negotiated cipher)

ServerFinished $S \rightarrow C$ $\{ Hash(Master(N_C, N_S, Secret_C) + Pad_2 +$
$Hash(Msgs + S + Master(N_C, N_S, Secret_C) + Pad_1))$
$\}_{Master(N_C, N_S, Secret_C)}$

ClientFinished $C \rightarrow S$ $\{ Hash(Master(N_C, N_S, Secret_C) + Pad_2 +$
$Hash(Msgs + C + Master(N_C, N_S, Secret_C) + Pad_1))$
$\}_{Master(N_C, N_S, Secret_C)}$


Supplicant
Auth/Assoc
802.1X UnBlocked
PTK/GTK
Authenticator
Auth/Assoc
802.1X UnBlocked
PTK/GTK
Authentica-
tion Server
(RADIUS)
No Key
802.11 Association
EAP/802.1X/RADIUS Authentication
MSK
4-Way Handshake
Group Key Handshake
Data Communication