Preventing Attackers From Using Verifiers: A-PAKE With PK-Id

Sean Parkinson (sean.parkinson@rsa.com)
RSA, The Security Division of EMC

RSA CONFERENCE EUROPE 2013

Session ID: ARCH-R02

Session Classification: Advanced

Outline

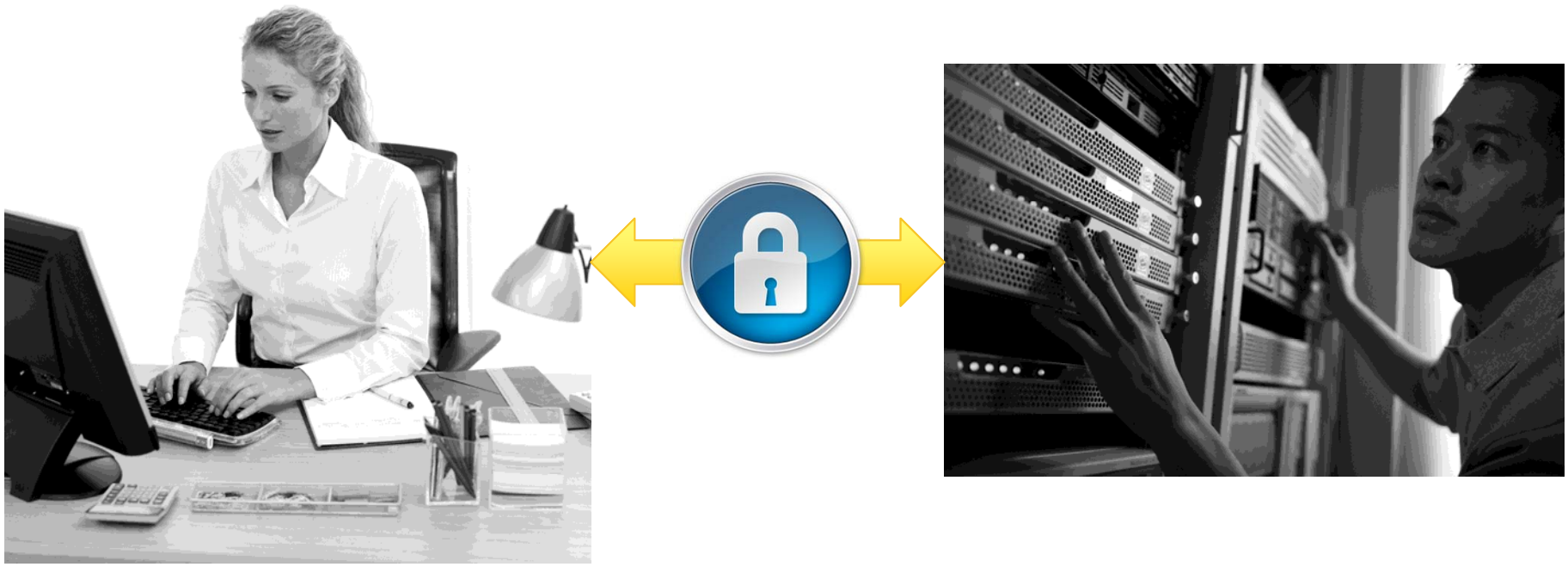
- Introduction
- A-PAKE with PK-Id
- Alternatives and Trade-offs
- Conclusions

Introduction

#RSAC

RSA CONFERENCE EUROPE 2013

Problem Domain



Authentication

Server to User

User to Server

Attacker: Man-in-the-Middle

- Steal password database
- Create a fake server certificate
- Create a fake client certificate

Server Authentication

- Authenticating a computer
- Certificate and Private Key

User Authentication

- Authenticating a person
- User Certificate and Private Key

User Authentication

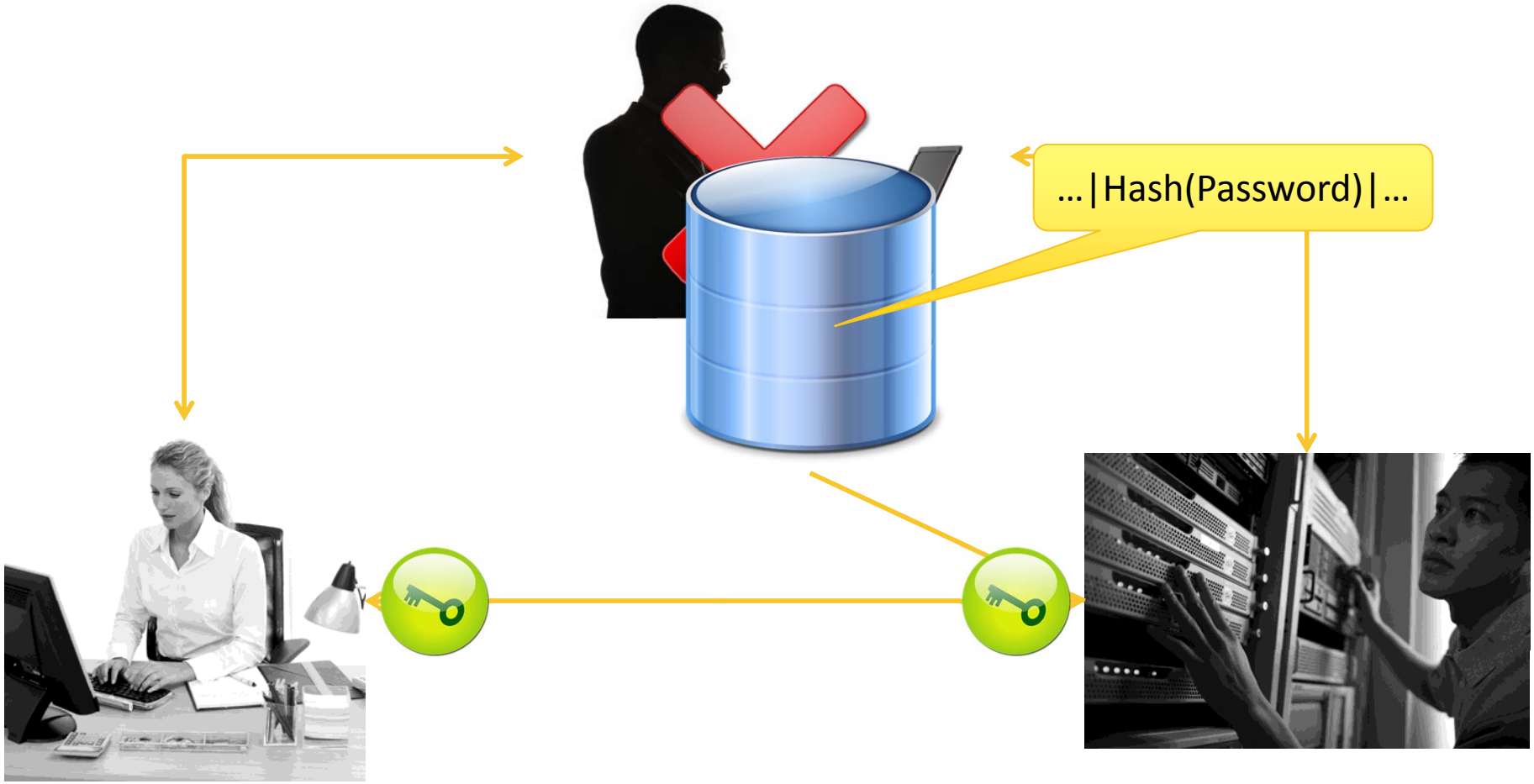
Passwords in the Database

Mechanism	Form	Security
Clear	"password"	None
Hash	H(P)	Depends on length of password
Hash with Salt	H(S, P)	Depends on length of password Different sites, different value
N*Hash with Salt	H(…H(S, P)….)^	Longer to break
Exponentiation	$g^{H(S,P)}$	Brute-force not practical

^ Password Based Key Derivation Function (PBKDF)

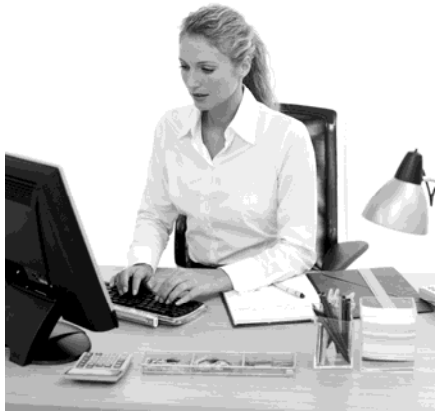
PAKE

Password-Authenticated Key Exchange
“Strong security from weak passwords”



A-PAKE

Augmented Password-Authenticated Key Exchange



…|$g^{H(S,I,P)}$|…

Verifier

Vulnerability

Change Password

A-PAKE with PK-Id: Aim

- New protocol that is standardized
- Easy identity management for User
- Secure storage of password on Server
- Without password Attacker cannot impersonate User or Server

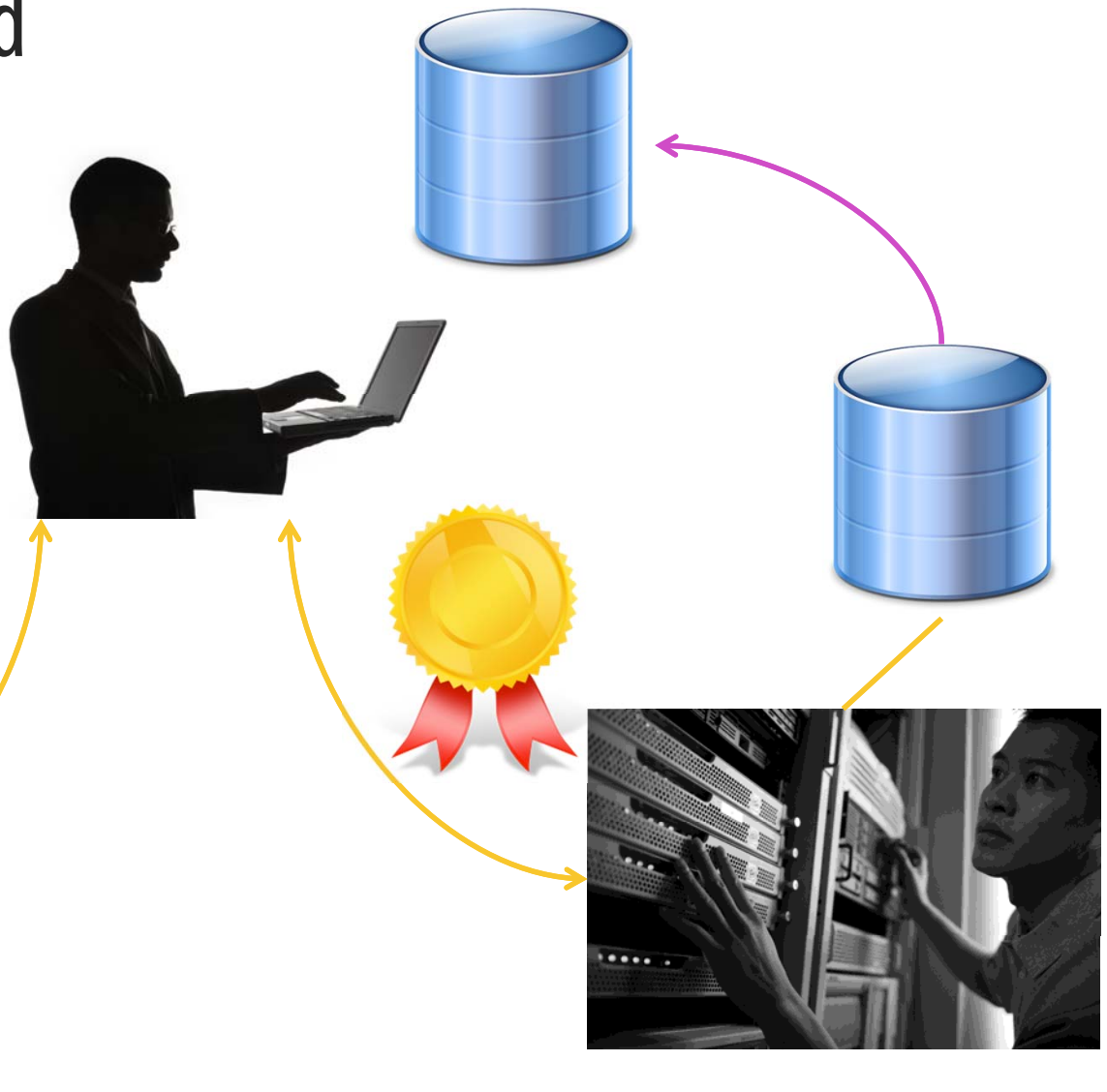
A-PAKE with PK-Id

#RSAC

RSA CONFERENCE EUROPE 2013

A-PAKE with PK-Id

Verifier = $g^{Hash(PK,S,I,P)}$



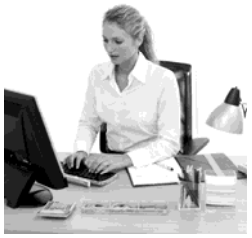
A-PAKE with PK-Id: General

1 Identity → Lookup Parameters

2 Calculate Hash ← User Parameters

3 Shared Key ← Anon Key Exchange → Shared Key

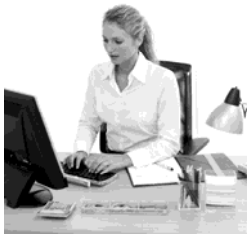
4 Server Verified ← Verification Messages → Client Verified



A-PAKE with PK-Id: SRP-PK

1 Identity → Lookup Parameters

Identity: "Alice Anderson" → Salt = 0x123456…
Verifier = 0xa73b24…



A-PAKE with PK-Id: SRP-PK

2 Calculate Hash

User Parameters

s = Salt

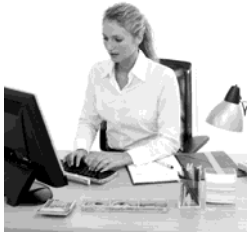
Salt = `0x123456...`

I = Identity

PK = `0xf389da...`

P = `"password"`

x = Hash(s, Hash(I, ":", P))



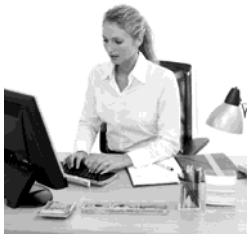
A-PAKE with PK-Id: SRP-PK

3 Shared Key ⟵ Anon Key Exchange ⟶ Shared Key

$A = g^a$ ⟶

⟵ $B = kv + g^b$

Verify(PK, sig, s, g, N, B) ⟵ **sig = Sign(PrivK, s, g, N, B)**



v = Verifier

a,b = random

k = Hash(N,g)

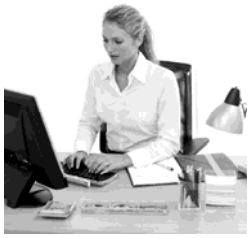
A-PAKE Scheme: SRP-PK

3 Shared Key ⟵ Anon Key Exchange ⟶ Shared Key

$u = Hash(A,B)$ $u = Hash(A,B)$

$S = (B - kg^x)^{a+ux}$ $S = (Av^u)^b$

$K = Hash(S)$ $K = Hash(S)$



$k = Hash(N,g)$

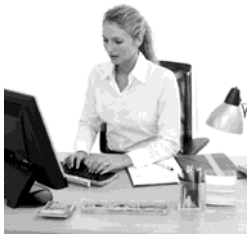
A-PAKE with PK-Id: SRP-PK

4 Server Verified ⟵ Verification Messages ⟶ Client Verified

$M_1 = \text{Hash}(\text{Hash}(N) \oplus \text{Hash}(g), \text{Hash}(I), s, A, B, K)$

M_1 ⟶ (verify M_1)

(verify M_2) ⟵ $M_2 = \text{Hash}(A, M_1, K)$



A-PAKE with PK-Id: Security

- Server signs with the private key
  - Provides proof of ownership
  - Salt is unique to each user and B is random
  - Different signature generated in each handshake
- If Man-in-the-middle replays previously signed B
  - Private random, b, unknown
  - b is required to complete the handshake
- If Man-in-the-middle uses a fake certificate
  - Public key used in verifier
  - Different keys results in different verifiers

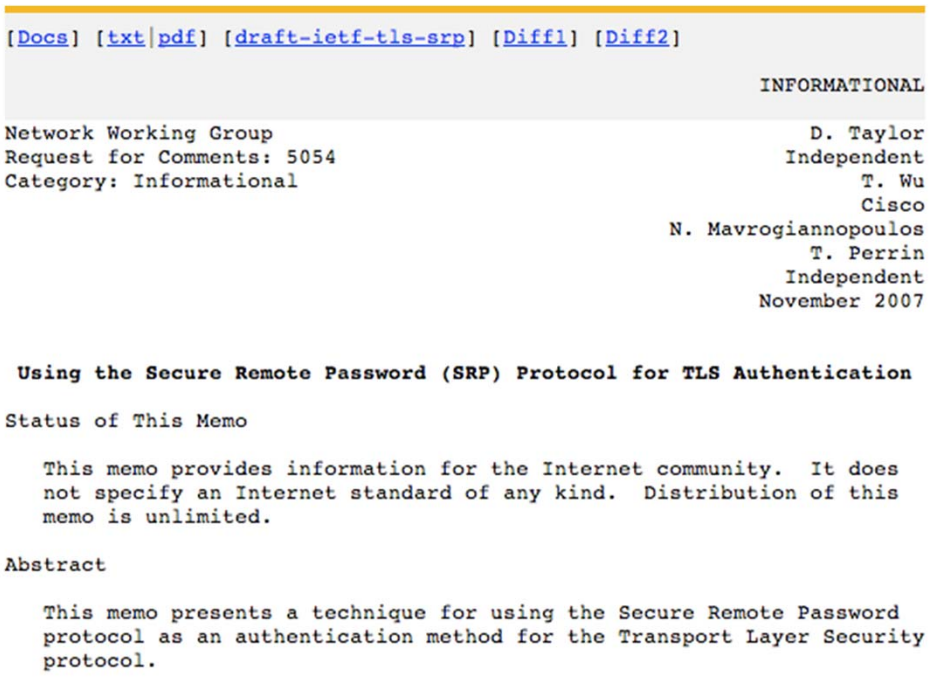
A-PAKE with PK-Id: Optional

- If A sent in clear
  - No security issue

- Client encrypts A
  - A is random
  - Different message generated in each handshake
- Advantage
  - Server knows Client is using the public key
  - Client used the public key in verifier

TLS with SRP

- RFC 5054: Informational only
- Experimental Implementations
- Server certificate optional



[Docs] [txt|pdf] [draft-ietf-tls-srp] [Diff1] [Diff2]

INFORMATIONAL

Network Working Group — D. Taylor, Independent
Request for Comments: 5054 — T. Wu, Cisco
Category: Informational — N. Mavrogiannopoulos, Independent; T. Perrin, Independent
November 2007

Using the Secure Remote Password (SRP) Protocol for TLS Authentication

Status of This Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

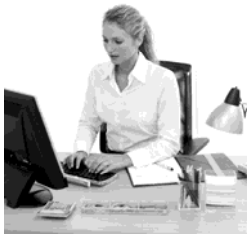
Abstract

This memo presents a technique for using the Secure Remote Password protocol as an authentication method for the Transport Layer Security protocol.

TLS with SRP-PK

1 Client Hello 

Identity (TLS Ext):
"Alice Anderson" 



TLS with SRP-PK

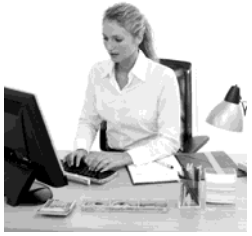
2

Server Certificate

Server Key Exchange

From Server's Certificate

x = Hash(B, **PK**, s, Hash(I,":",P))



Server's Cert Chain

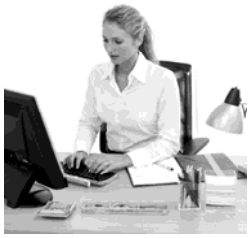
s=0x12…, N=0xF9…
g=0x02, B=0x91…

Signed

TLS with SRP-PK

3
Client Key Exchange 

A = `0x8617E3...` 



A-PAKE-PK with Stored PK

…|Public Key|…

…|Verifier|…

No certificate
No chain verification
[Encrypt A]

Alternatives and Trade-offs

#RSAC

RSA CONFERENCE EUROPE 2013

Alternative: Client Certificate

- Difficult to create/use/deploy
  - Many non-technical users
  - Only useable on a device with private key
  - Separately issued device
- Difficult to verify
  - Chain verification on server
  - Hash lookup in LDAP
  - CRLs and OCSP
- Hackers can create fake certificates

Architecture: User Auth in TLS

- Separation of Layers
  - Which service or website needs authentication?
  - Single authentication domain per server
- TLS Server requires access to
  - Password store
  - Certificate verification infrastructure

Trade-offs: Protocol Timing

Protocol	Client	Server	Round-trips
SRP	3DH	3DH	3
SRP-PK	3DH+1RSA+Cert Chain	3DH+1RSA	3
TLS SRP	3DH(+1RSA+Cert Chain)	3DH(+1RSA)	2
TLS SRP-PK	3DH+1RSA+Cert Chain	3DH+1RSA	2

Secure against
Server Impersonation

Trade-offs: Protocol Timing

Protocol	Client	Server	Round-trips
SRP	3DH	3DH	3
SRP-PK	3DH+1RSA+Cert Chain	3DH+1RSA	3
TLS SRP	3DH(+1RSA+Cert Chain)	3DH(+1RSA)	2
TLS SRP-PK	3DH+1RSA+Cert Chain	3DH+1RSA	2
TLS Client Auth (RSA)	2RSA+Cert Chain	2RSA+Cert Chain	2
TLS Client Auth (DHE-RSA)	2DH+2RSA+Cert Chain	2DH+2RSA+Cert Chain	2

Trade-offs: Protocol Timing

3xDH for 1xRSA Public + Chain Vfy

Protocol	Client	Server	Round-trips
SRP	3DH	3DH	3
SRP-PK	3DH+1RSA+Cert Chain	3DH+1RSA	3
TLS SRP	3DH(+1RSA+Cert Chain)	3DH(+1RSA)	2
TLS SRP-PK	3DH+1RSA+Cert Chain	3DH+1RSA	2
TLS Client Auth (RSA)	2RSA+Cert Chain	2RSA+Cert Chain	2
TLS Client Auth (DHE-RSA)	2DH+2RSA+Cert Chain	2DH+2RSA+Cert Chain	2

1xDH for 1xRSA Public + Chain Vfy

Trade-offs: Password/Certificate

Issue	Password	Client Certificate	Cert on Token
Sensitive Data Safety	Reused	Unique	Physically secure
Attacks	Social engineering	Computer compromise	Steal token (and passcode)
Certificate Expiration	Reset password (overlap period)	User gets new certificate issued	Update token or get new token

Conclusions

#RSAC

RSA CONFERENCE EUROPE 2013

User Authentication Options

Mechanism	Attack	Speed	Management
Hashed Password	Pre-generated values	Fastest	Simplest
Salted Hashed Password	Brute force	Faster	Simpler
Salted Multi-Hashed Password	Brute force (Slower)	Fast	Simpler
Verifier	Server impersonation	Slow	Simple
Verifier PK-Id	Secure	Slower	Simple
Client Certificate	Secure (Fake certs)	Slowest	Difficult

TLS Protocol Comparison

Protocol	Client	Server	Security
TLS A-PAKE-PK	3DH+1RSA +Cert Chain	3DH+1RSA	Protocol secure from Attacker. User must protect password.
TLS Client Auth (RSA)	2RSA +Cert Chain	2RSA +Cert Chain	Only as secure as the Certificate. Lifetime of certificate.
TLS Client Auth (DHE-RSA)	2DH+1RSA +Cert Chain	2DH+1RSA +Cert Chain	Only as secure as the Certificate.

A-PAKE-PK

Verifier = $g^{H(PK,S,I,P)}$


Standards
TLS
IPsec
SSH

#RSAC

RSA CONFERENCE EUROPE 2013

Thank you!

Sean Parkinson

RSA, The Security Division of EMC

sean.parkinson@rsa.com

https://blogs.rsa.com

#RSAC

RSA CONFERENCE EUROPE 2013

TLS with AugPAKE

- IETF Draft - draft-shin-tls-augpake-01
  - Augmented Password-Authenticated Key Exchange for Transport Layer Security (TLS)
  - Expires: March 08, 2014
- No Server Certificate message
- Verifier
  - $W = g^{w} \bmod p$ where w is password **or**
  - $W = g^{w'} \bmod p$ where $w' = H'(0x00 \mid U \mid S \mid w)$

TLS with AugPAKE-PK

- Add Server Certificate message
- Verifier
  - $W = g^{w'} \bmod p$ where $w' = H'(0x00 \mid U \mid S \mid \mathbf{PK} \mid w)$
- Alternatively replace server identity with public key
- Server Key Exchange Message
  - Y is signed

A-EKE-PK

- Server sends public key with identity and cryptographic algorithm proposal
- Client hashes password + public key and calculates verifier
- Client calculates C based on verifier
- Server sends E_B and signature of E_B
- Client verifies E_B
- Client sends encrypted E_A and EP_A
- Server decrypts E_A and EP_A with private key

B-SPEKE-PK

- Server sends public key and signature of (g, p, Q_B, U)
- Client verifies signature
- Client hashes password + salt and public key
- Client sends encrypted Q_A
- Server decrypts Q_A with private key

SSH with SRP

- IETF Draft – draft-nisse-secsh-srp-01
  - Expired in September 2001
- Adds SRP to Transport Layer Protocol
  - Defines new Key Exchange messages
- Verifier
  - $v = g^x$ where $x = Hash(s, Hash(I \mid ":" \mid P))$

SSH and SRP-PK

- Server sends RSA Public key
- Verifier
  - $v = g^x$ where x = Hash(**Pub**, s, Hash(I | ":" | P))

IPsec with A-PAKE

- Internet Key Exchange Protocol Version 2 (IKEv2)
  - RFC 5996
  - Key exchange protocol performs mutual authentication
  - Server sends certificate that client uses to verify authentication fields
- Extensible Authentication Protocol (EAP)
  - Framework for implementing authentication mechanisms
  - A-EKE (RFC 6124) and SRP (expired draft)

IPsec EAP-EKE-PK

- Server must supply a certificate during authentication
  - Server has proven ownership of private key
- Verifier
  - P = prf(0+, password | salt | **pub**)