Unit 5: Key Exchange

Diffie-Hellman
Diffie-Hellman Weaknesses
Elliptic Curve Diffie-Hellman (ECDH)
Passing Key Using Public Key
Key Distribution Centre (KDC)

Prof Bill Buchanan OBE

http://asecuritysite.com/crypto05
http://asecuritysite.com/encryption

Unit 5: Key Exc

Diffie-Hellman
Diffie-Hellman Weaknesses
Elliptic Curve Diffie-Hellman (ECI
Passing Key Using Public Key
Key Distribution Centre (KDC)

Prof Bill Buchanan OB

http://asecuritysite.com/cryptoC
http://asecuritysite.com/encrypt

Data Centre ▸ Networks

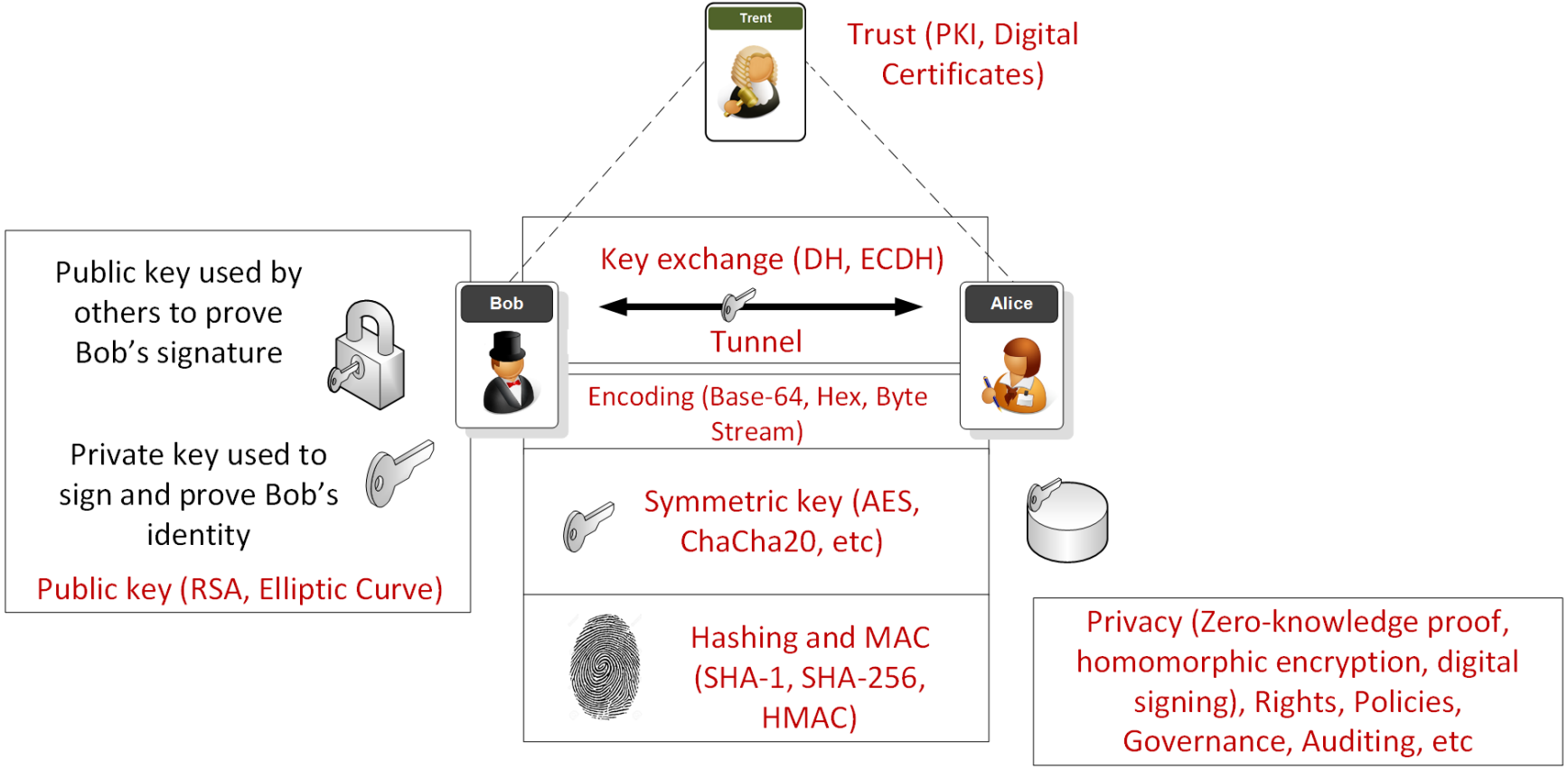
Internet's safe-keepers forced to postpone crucial DNSSEC root key signing ceremony – no, not a hacker attack, but because they can't open a safe

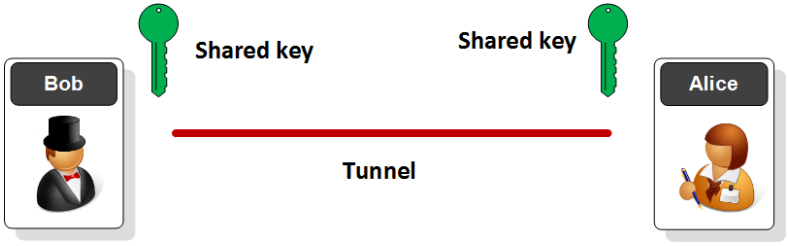
Online security process stalled by offline security screw-up

By Kieren McCarthy in San Francisco 13 Feb 2020 at 06:09 59 SHARE ▼

No	Date	Subject	Lab
1	17 Jan 2020	Ciphers and Fundamentals Unit	Lab [Link] Demo [Link]
2	24 Jan 2020	Symmetric Key Unit	Lab [Link] Demo [Link]
3	31 Jan 2020	Hashing and MAC Unit	Lab [Link]
4	7 Feb 2020	Asymmetric (Public) Key Unit	Lab [Link]
5	14 Feb 2020	Key Exchange Unit	Lab [Link]
6	21 Feb 2020	Guest lecture	Mini-project/Coursework [Link]
7	28 Feb 2020	Trust and Digital Certificates Unit	Lab [Link]
8	6 Mar 2020	Tunnelling Unit	Lab [Link]
9	13 Mar 2020	Test 1 (Units 1-5) [Study guide]	
10	20 Mar 2020	Blockchain Unit	Lab [Link]
11	27 Mar 2020	Future Cryptography Unit	Lab [Link]
12	3 April 2020	Tokens, Authorization and Docker Unit	Lab [Link]
13	10 April 2020	Trusted Hosts Unit	
Easter Break			
14	Week beginning 27 April 2020 (TBC)	Test 2 (Units 6-10)	
15	Week beginning 4 May 2020 (TBC)	Coursework Hand-in [Draft]	

Overview


Trent
Trust (PKI, Digital Certificates)
Public key used by others to prove Bob's signature
Private key used to sign and prove Bob's identity
Public key (RSA, Elliptic Curve)
Bob
Alice
Key exchange (DH, ECDH)
Tunnel
Encoding (Base-64, Hex, Byte Stream)
Symmetric key (AES, ChaCha20, etc)
Hashing and MAC (SHA-1, SHA-256, HMAC)
Privacy (Zero-knowledge proof, homomorphic encryption, digital signing), Rights, Policies, Governance, Auditing, etc



Test Scenario and Properties	Application					
	Signal	WhatsApp	Wire	Viber	Riot	Telegram
Setup and Registration						
Phone Registration	●	●	●	●	○	●
E-mail Registration	○	○	●	○	●	○
Access SMS Inbox	●	●	○	●	○	●
Contact list Upload	●	●	●	●	●	●
Verification by SMS	●	●	●	●	○	●
Verification by Phone Call	●	●	●	●	○	●
Initial Contact						
Trust-On-First-Use	●	●	○	○	○	○
Notification About E2E Encryption	○	●	○	○	●	●
Message After a Key Change						
Notification about key changes	●	●	○	○	●	○
Blocking message	●	○	○	○	○	○
Key Change While a Message Is In Transit						
Re-encrypt and Send Message	○	●	○	○	○	○
Details About Transmission of Message	●	●	●	○	●	●
Verification Process						
QR-Code	●	●	○	○	○	●
Verify By Phone Call	●	●	●	●	●	○
Share Keys Through 3rd Party	●	●	○	○	○	○
Verified Check	○	○	●	●	●	○
Other Security Implementations						
Two-Step Verification	○	●	○	○	○	●
Passphrase/Code	●	○	○	○	○	●
Screen Security	●	○	○	○	○	●
Clear Trusted Contacts	○	○	○	●	○	○
Delete Devices From Account	○	○	●	●	●	●

●: Has the property; ○: Does not have the property.

Paper

> Use anything by Open Whisper Systems.
>
> **Edward Snowden**, Whistleblower and privacy advocate

> Signal is the most scalable encryption tool we have. It is free and peer reviewed. I encourage people to use it everyday.
>
> **Laura Poitras**, Oscar-winning filmmaker and journalist

> I am regularly impressed with the thought and care put into both the security and the usability of this app. It's my first choice for an encrypted conversation.
>
> **Bruce Schneier**, internationally renowned security technologist

Unit 5: Key Exchange

Diffie-Hellman

Diffie-Hellman Weaknesses

Elliptic Curve Diffie-Hellman (ECDH)

Passing Key Using Public Key

Key Distribution Centre (KDC)

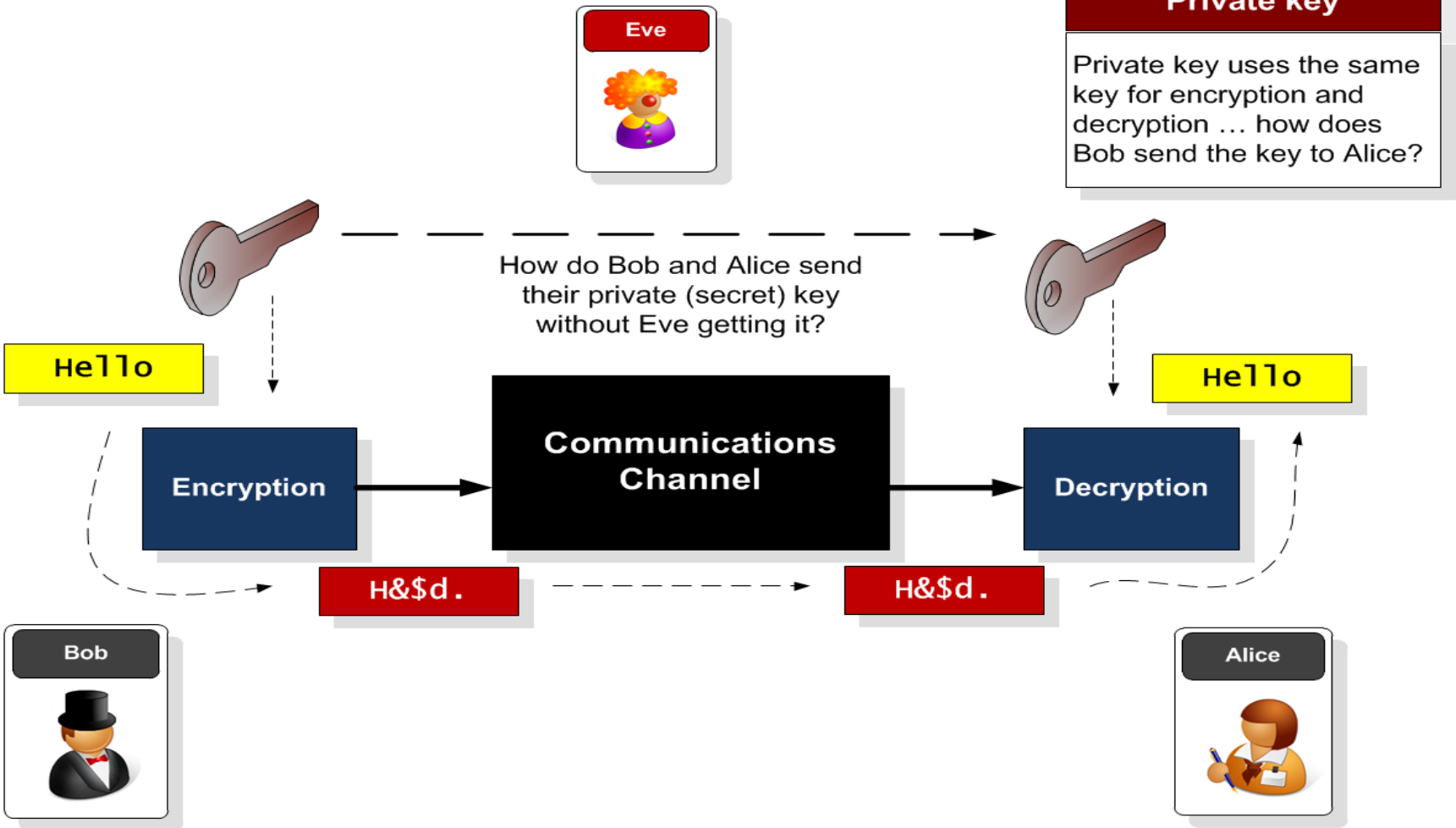
Prof Bill Buchanan OBE

http://asecuritysite.com/crypto05

http://asecuritysite.com/encryption

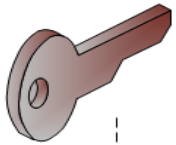
Private key

Private key uses the same key for encryption and decryption … how does Bob send the key to Alice?

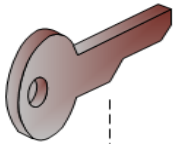


Diffie-Hellman

One of the most widely method for creating a secret key which is the same for Bob and Alice



How do Bob and Alice send their private (secret) key without Eve getting it?



Hello

Encryption

Communications Channel

Decryption

Hello

H&$d.

H&$d.

This problem was solved by Whitfield Diffie, who created the Diffie-Hellman algorithm, which is the most widely used method for passing secret keys

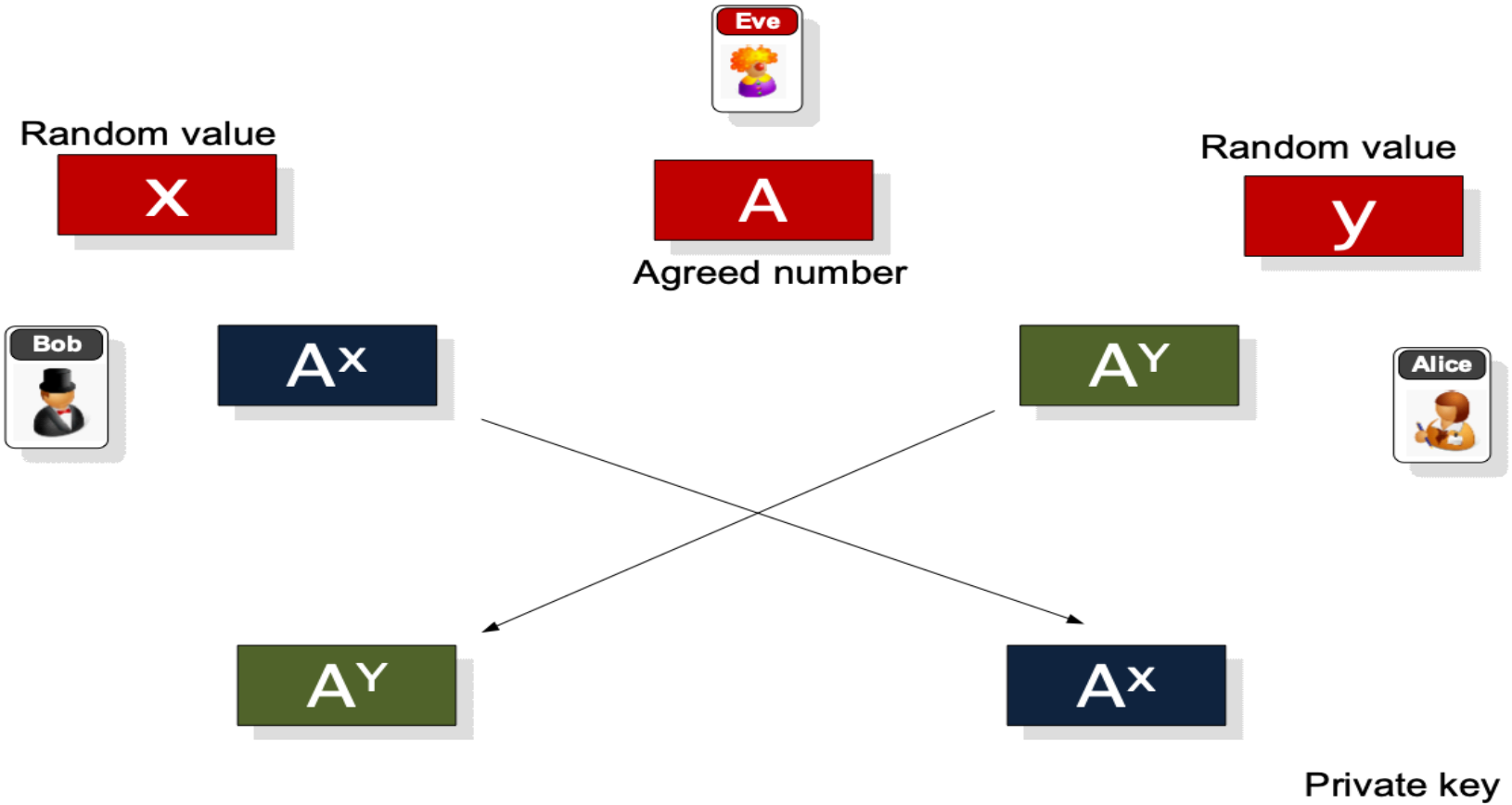
Key Exchange

- **Forward secrecy** (FS), which means that a comprise of the long-term keys will not compromise any previous session keys. A leakage of the public key of the server would cause all the sessions which used this specific public key to be compromised. FS thus aims to overcome this by making sure that all the sessions keys could not be compromised, even though the long-term key was compromised.
- **Ephemeral**. With some key exchange methods the same key will be generated if the same parameters are used on either side. This can cause problems as an intruder could guess the key, or even where the key was static and never changed. With ephemeral methods, a different key is used for each connection, and, again, the leakage of any long-term would not cause all the associated session keys to be breached.

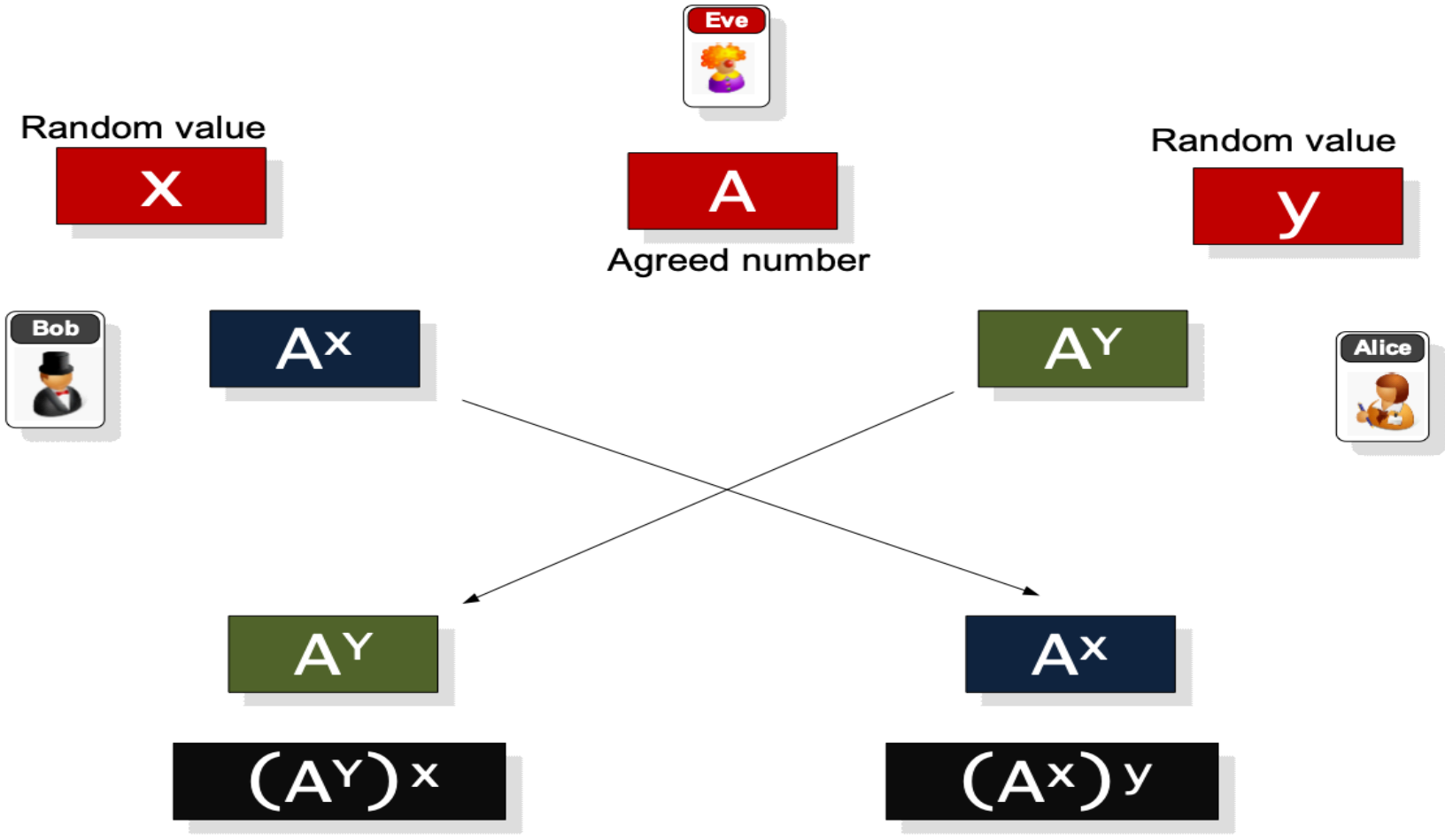
$A^x A^y \rightarrow A^{(x+y)}$

$(A^x)^y \rightarrow A^{xy}$

Author: Prof Bill Buchanan



Author: Prof Bill Buchanan



Author: Prof Bill Buchanan

Diffie-Hellman

Eve can listen to the values of A and B, but should not be able to determine the secret key

1. Both nodes agree on two values (G and *n*)

2. Generate a random value (*x*)

3. $A = G^x \bmod n$

2. Generate a random value (y)

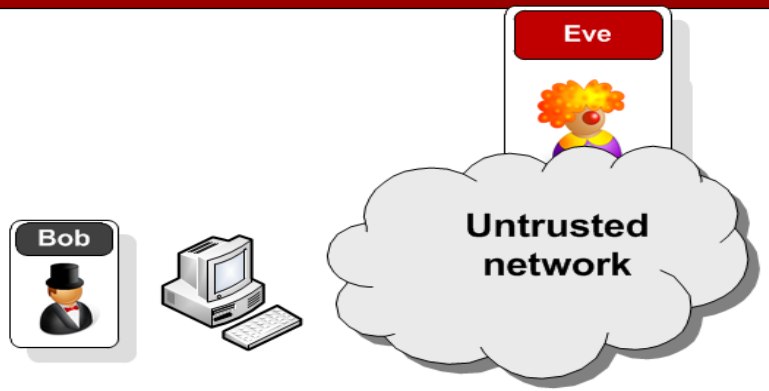
3. $B = G^y \bmod n$

4. A and B values exchanged

5. $K1 = B^x \bmod n$

5. $K2 = A^y \bmod n$

K1 and *K2* should be the **same** and are the secret key



Diffie-Hellman

Eve can listen to the values of A and B, but should not be able to determine the secret key

1. Both nodes agree on two values (5 and 7)

2. Generate a random value (2)

3. $A = 5^2 \bmod 8 = 25 \bmod 7 = 4$

2. Generate a random value (3)

3. $B = 5^3 \bmod 7 = 125 \bmod 7 = 6$

4. A and B values exchanged

6

4

5. $K1 = 6^2 \bmod 7 = 36 \bmod 7 = 1$

5. $K2 = 4^3 \bmod 7 = 64 \bmod 7 = 1$

K1 and *K2* should be the **same** and are the secret key

Example

Diffie-Hellman Generator

$$Y = G^x \bmod p$$

p	11							
Generator	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
x	**g^x mod p**	**g^x mod p**	**g^x mod p**	**g^x mod p**	**g^x mod p**	**g^x mod p**	**g^x mod p**	**g^x mod p**
2	4	9	5	3	3	5	9	4
3	8	5	9	4	7	2	6	3
4	5	4	3	9	9	3	4	5
5	10	1	1	1	10	10	10	1
6	9	3	4	5	5	4	3	9
7	7	9	5	3	8	6	2	4
8	3	5	9	4	4	9	5	3
9	6	4	3	9	2	8	7	5
10	1	1	1	1	1	1	1	1

Picking G

Diffie-Hellman Generation

```
C:\> openssl dhparam -out dhparams.pem 768 –text
C:\> type dhparams.pem
Diffie-Hellman-Parameters: (768 bit)
    prime:
        00:d0:37:c2:95:64:02:ea:12:2b:51:50:a2:84:6c:
        71:6a:3e:2c:a9:80:e2:65:b2:a5:ee:77:26:22:31:
        66:9e:fc:c8:09:94:e8:9d:f4:cd:bf:d2:37:b2:fb:
        b8:38:2c:87:28:38:dc:95:24:73:06:d3:d9:1f:af:
        78:01:10:6a:7e:56:4e:7b:ee:b4:8d:6b:4d:b5:9b:
        93:c6:f1:74:60:01:0d:96:7e:85:ca:b8:1f:f7:bc:
        43:b7:40:4d:4e:87:e3
    generator: 2 (0x2)
-----BEGIN DH PARAMETERS-----
MGYCYQDQN8KVZALqEitRUKKEbHFqPiypgOJlsqXudyYiMWae/
MgJlOid9M2/0jey
+7g4LIcoONyVJHMG09kfr3gBEGp+Vk577rSNa021m5PG8XRgAQ2WfoXKu
B/3vEO3
QE1Oh+MCAQI=
-----END DH PARAMETERS-----
```

- **DH Group 5**: 1,536 bit prime.
- **DH Group 2**: 1,024 bit prime.
- **DH Group 1**: 768-bit prime.

Unit 5: Key Exchange

Diffie-Hellman

Diffie-Hellman Weaknesses

Elliptic Curve Diffie-Hellman (ECDH)

Passing Key Using Public Key

Key Distribution Centre (KDC)

Prof Bill Buchanan OBE

http://asecuritysite.com/crypto05

http://asecuritysite.com/encryption

Diffie-Hellman Weaknesses

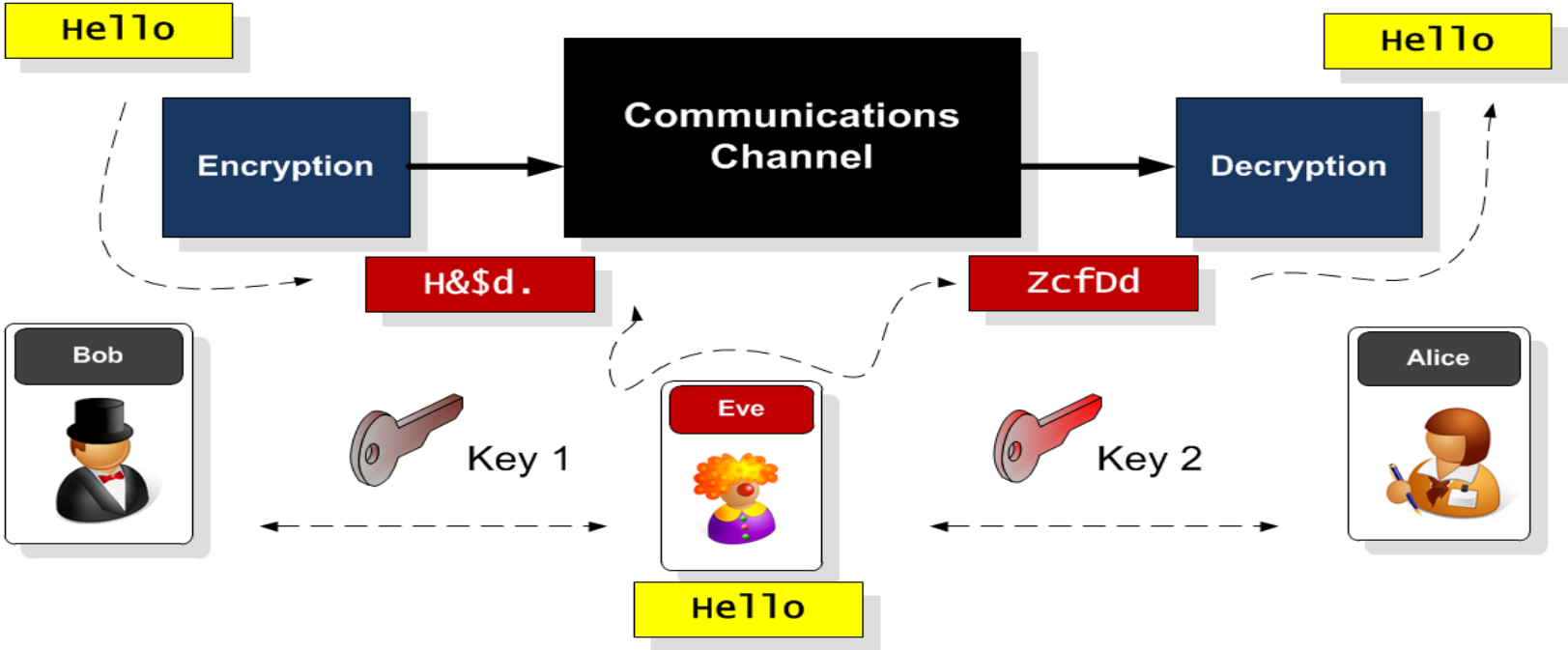
- In 2015, a paper entitled *Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice* – showed that it was fairly easy to precompute on values for two popular Diffie-Hellman parameters (and which use the DHE_EXPORT cipher set).
- The research team found that one was used as a default in the around 7% of the Top 1 million web sites and was hard coded into the Apache httpd service. Overall, at the time, it was found that over 3% of Web sites were still using the default.

- Diffie-Hellman-Parameters: (512 bit)
- prime:
- 00:9f:db:8b:8a:00:45:44:f0:04:5f:17:37:d0:ba:
- 2e:0b:27:4c:df:1a:9f:58:82:18:fb:43:53:16:a1:
- 6e:37:41:71:fd:19:d8:d8:f3:7c:39:bf:86:3f:d6:
- 0e:3e:30:06:80:a3:03:0c:6e:4c:37:57:d0:8f:70:
- e6:aa:87:10:33
- generator: 2 (0x2)

Diffie-Hellman suffers from a man-in-the-middle attack, where Eve negotiates for each side, and creates two encryption channels

Man-in-the-middle

Diffie-Hellman suffers from Eve intercepting the key interchange, so that Bob thinks he's talking to Alice for the key exchange.



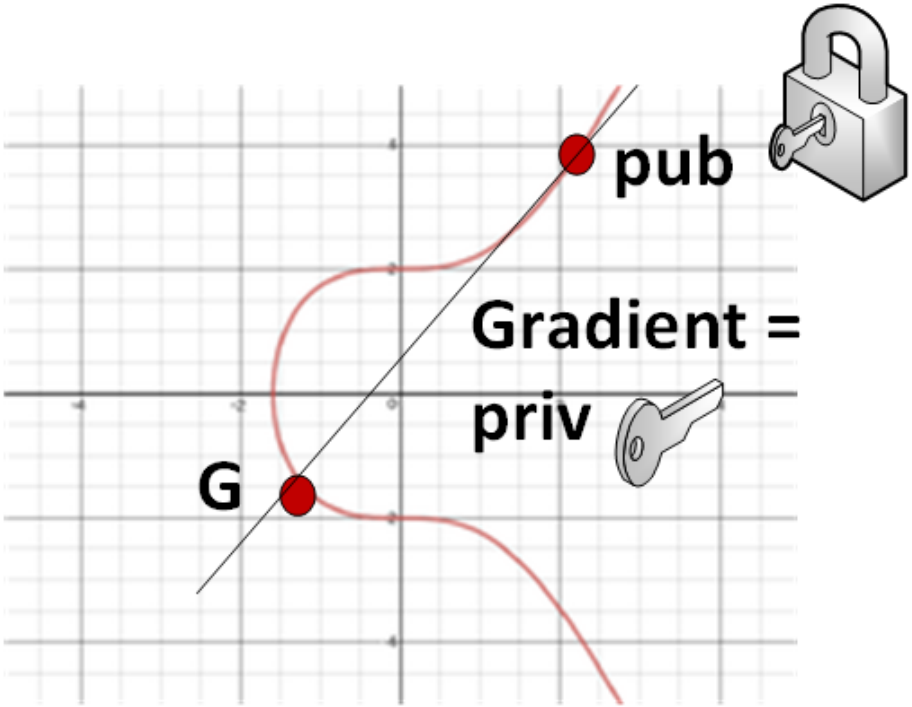
Unit 5: Key Exchange

Diffie-Hellman
Diffie-Hellman Weaknesses
Elliptic Curve Diffie-Hellman (ECDH)
Passing Key Using Public Key
Key Distribution Centre (KDC)

Prof Bill Buchanan OBE

http://asecuritysite.com/crypto05
http://asecuritysite.com/encryption

Public and private keys with ECC



Private key:
0xc9f4f55bdeb5ba0bd337f2dbc952a5439e20ef
9af6203d25d014e7102d86aaeeL
Public key:
0xc44370819cb3b7b57b2aa7edf550a9a5410c23
4d27aff497458bbbfec8b6a327,
0x52a1a3e222cd89cbd2764b69bd9b0ea5c4fd6c
a28861e1f2140eeff9c2e76487

G:
(506626302227734366957871889516853432625060345377759417550018736038911672924 0L
,
326705100207588169780830851305070431844712733806592432759389043357573374824 2L)

Public and private keys with EC

```
-----BEGIN EC PARAMETERS-----
BgUrgQQACg==
-----END EC PARAMETERS-----
-----BEGIN EC PRIVATE KEY-----
MHQCAQEEIEa56GG2PTUJyIt4FydaMNItYsjNj6ZIbd7jXvDY4ElfoAcGBSuBBAAK
oUQDQgAEJQDn8/vd8oQpA/VE3ch0lM6VAprOTiV9VLp38rwfOog3qUYcTxxX/sxJ
l1M4HncqEopYIKkkovoFFi62Yph6nw==
-----END EC PRIVATE KEY-----
Private-Key: (256 bit)
priv:
    34:7c:b7:89:8c:d9:5f:eb:00:73:94:3e:bc:b9:97:
    89:67:ef:e7:f5:04:ba:04:a3:1b:4f:ec:b0:63:1c:
    66:10
pub:
    04:29:b0:50:38:a6:32:fc:89:a1:2b:65:80:7b:14:
    8e:20:f3:03:2a:34:85:1f:3a:63:1a:fc:30:96:83:
    c0:e0:b7:72:77:17:cf:0c:53:2f:b9:e9:48:de:c2:
    70:1c:74:12:8d:04:8d:18:55:af:34:ff:9b:0b:a4:
    8a:bb:1d:3b:b9
ASN1 OID: secp256k1
Field Type: prime-field
```

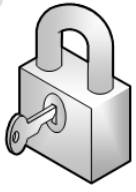
$Y^2 == X^3 + ax + b) \bmod p$

```
Prime:
    00:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:
    ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:fe:ff:
    ff:fc:2f
A:    0
B:    7 (0x7)
Generator (uncompressed):
    04:79:be:66:7e:f9:dc:bb:ac:55:a0:62:95:ce:87:
    0b:07:02:9b:fc:db:2d:ce:28:d9:59:f2:81:5b:16:
    f8:17:98:48:3a:da:77:26:a3:c4:65:5d:a4:fb:fc:
    0e:11:08:a8:fd:17:b4:48:a6:85:54:19:9c:47:d0:
    8f:fb:10:d4:b8
Order:
    00:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:
    ff:fe:ba:ae:dc:e6:af:48:a0:3b:bf:d2:5e:8c:d0:
    36:41:41
Cofactor:  1 (0x1)
```

ECDH

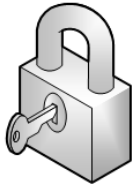
Private key (d_A)

Public key:
$Q_A = d_A \times G$

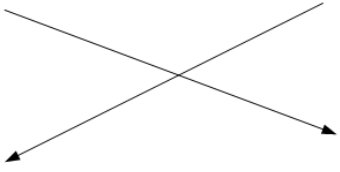


Private key (d_B)

Public key:
$Q_B = d_B \times G$



Q_A **Q_B**



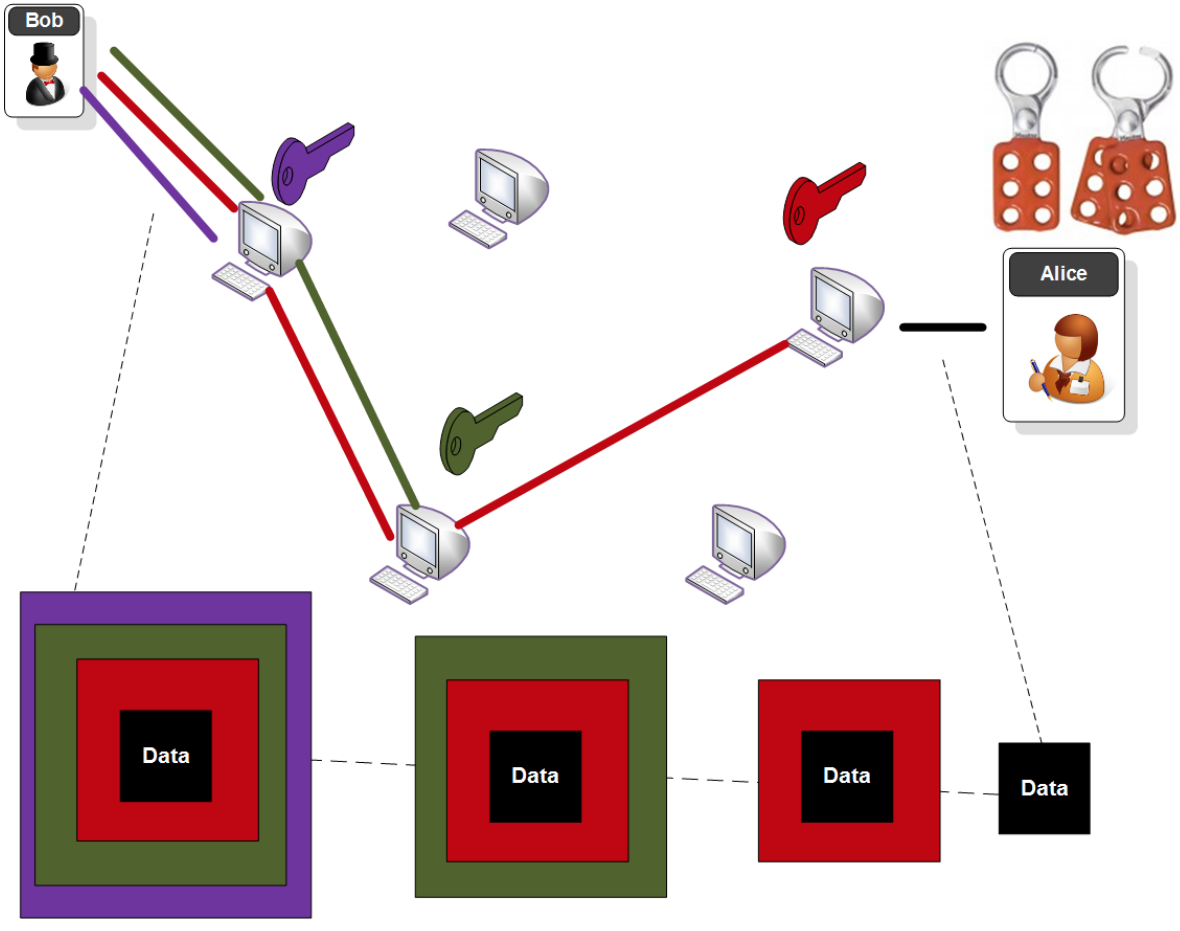
Shared key:
Share = $d_A \times Q_B$

Shared key:
Share = $d_A \times d_B \times G$

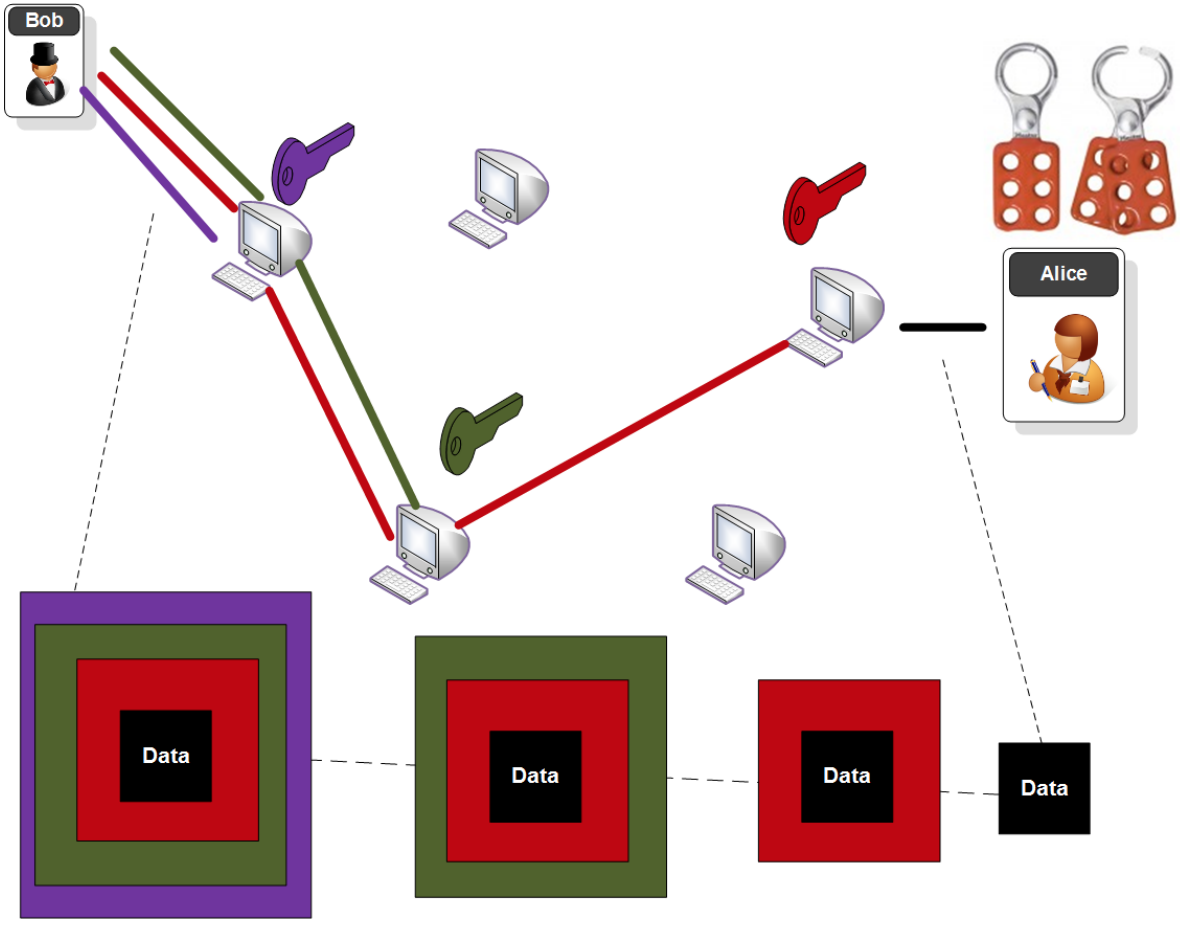
Shared key:
Share = $d_B \times Q_A$

Shared key:
Share = $d_B \times d_A \times G$

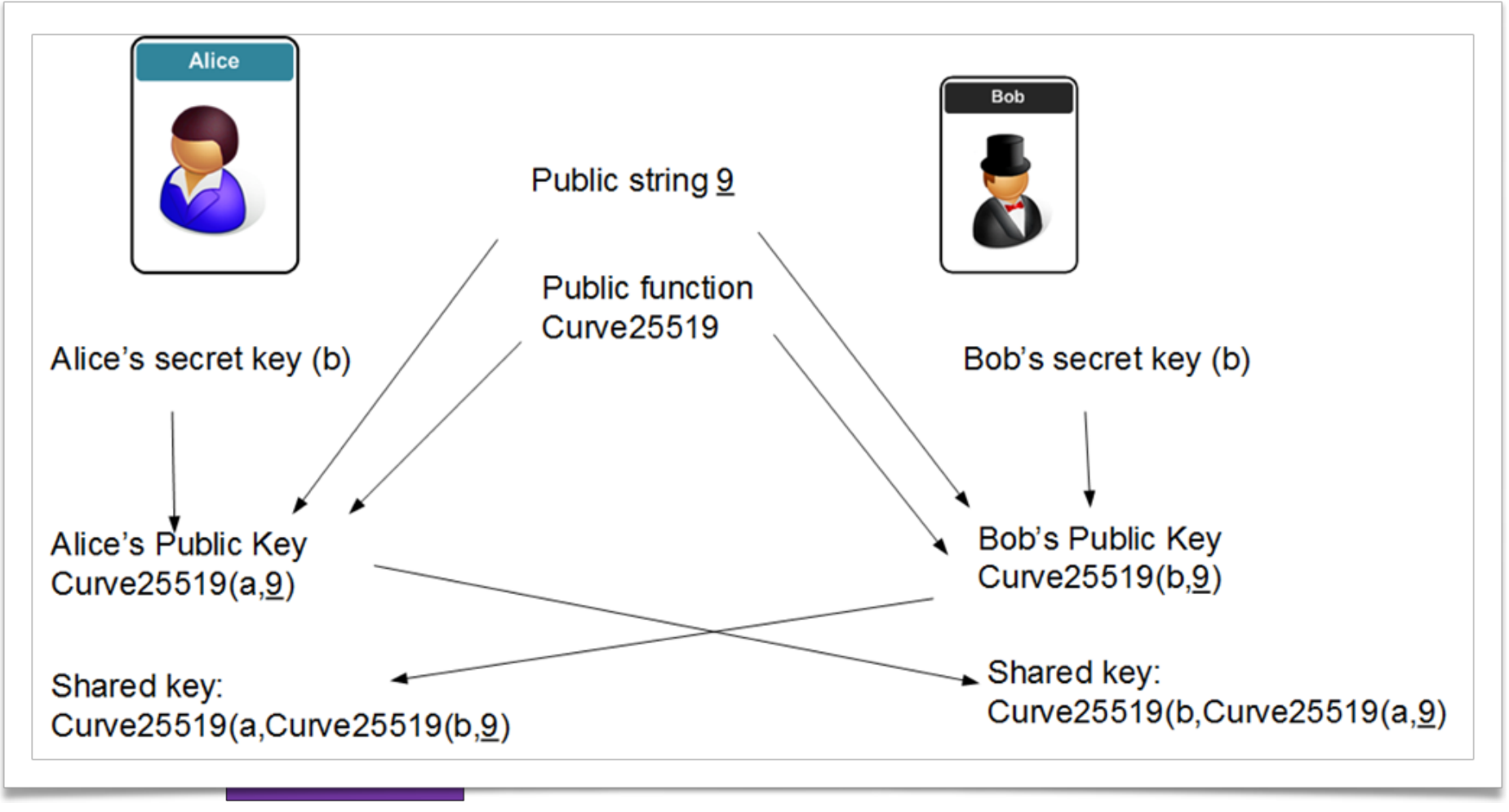
Elliptic Curve Diffie Hellman (ECDH)



Elliptic Curve Diffie Hellman (ECDH)



Elliptic Curve Diffie Hellman (ECDH)



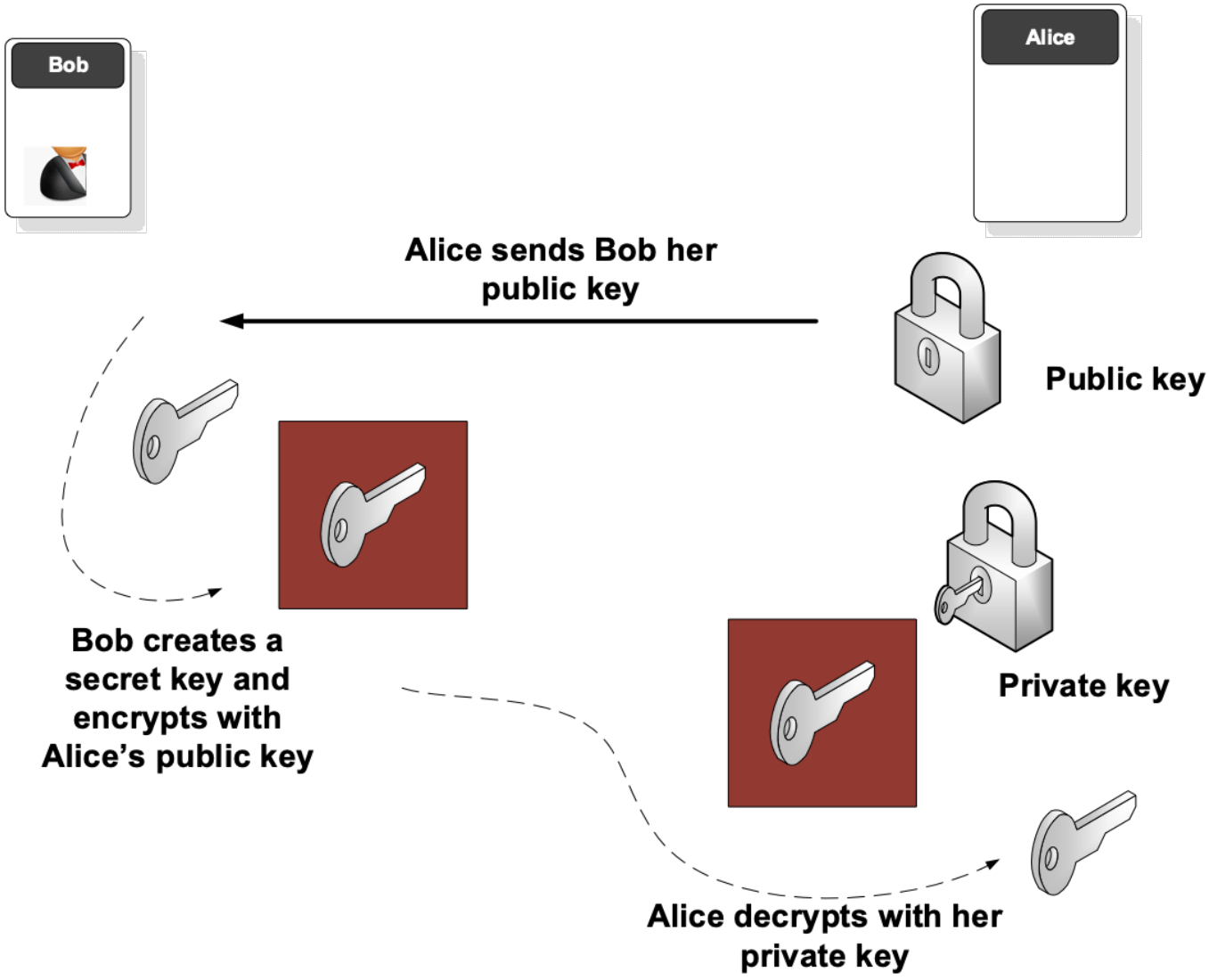
Unit 5: Key Exchange

Diffie-Hellman
Diffie-Hellman Weaknesses
Elliptic Curve Diffie-Hellman (ECDH)
Passing Key Using Public Key
Key Distribution Centre (KDC)

Prof Bill Buchanan OBE

http://asecuritysite.com/crypto05
http://asecuritysite.com/encryption

Key Exchange with Public Key



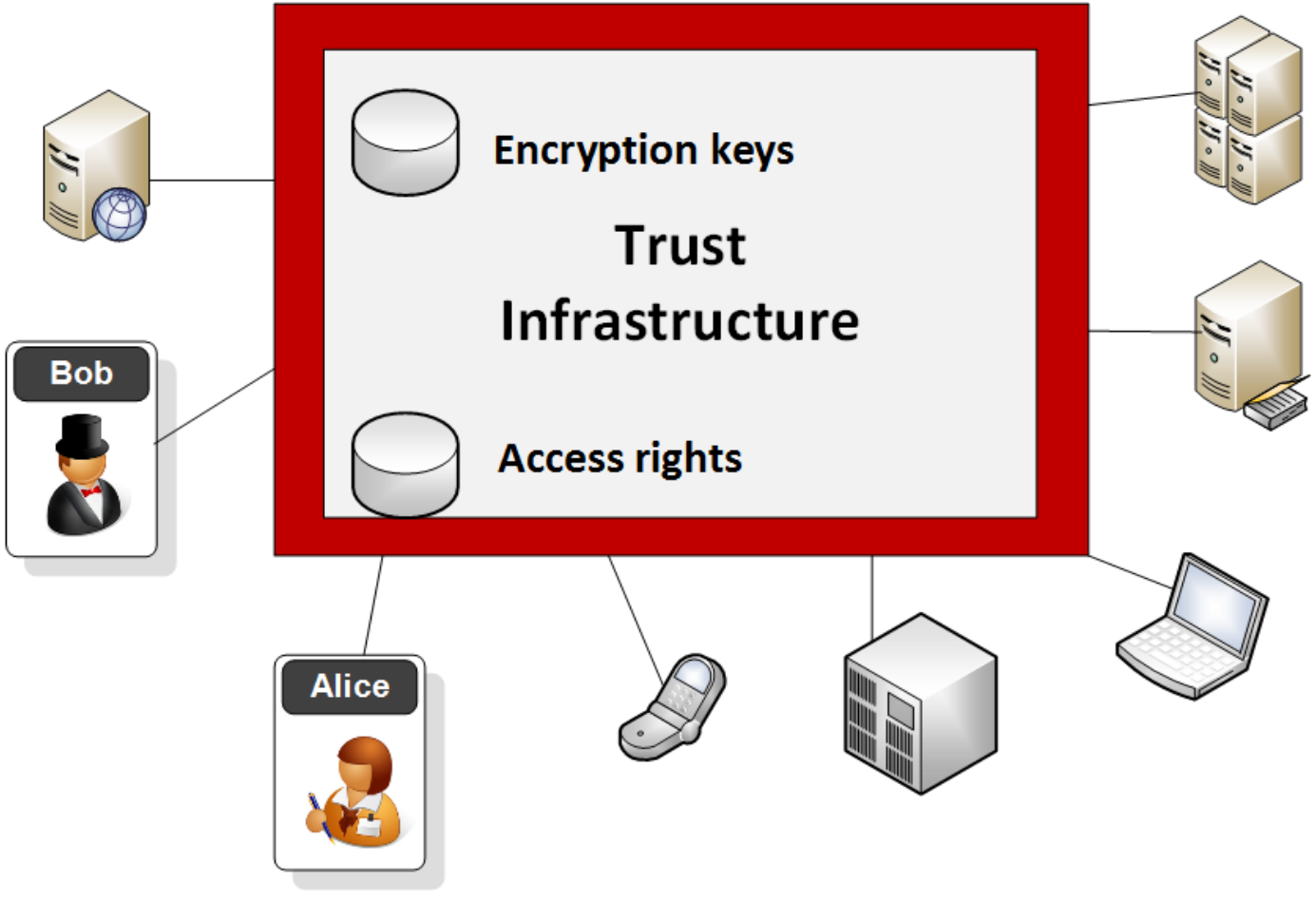
Unit 5: Key Exchange

Diffie-Hellman
Diffie-Hellman Weaknesses
Elliptic Curve Diffie-Hellman (ECDH)
Passing Key Using Public Key
Key Distribution Centre (KDC)

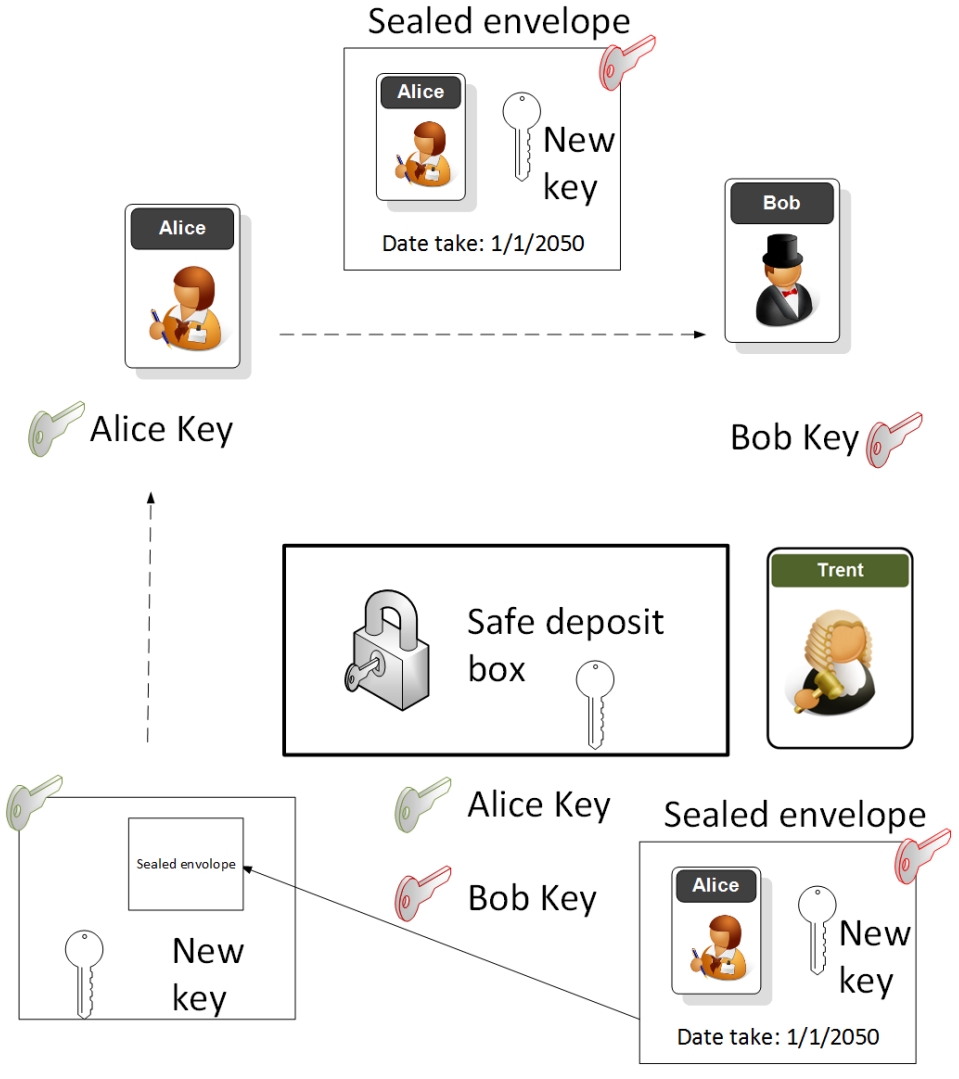
Prof Bill Buchanan OBE

http://asecuritysite.com/crypto05
http://asecuritysite.com/encryption

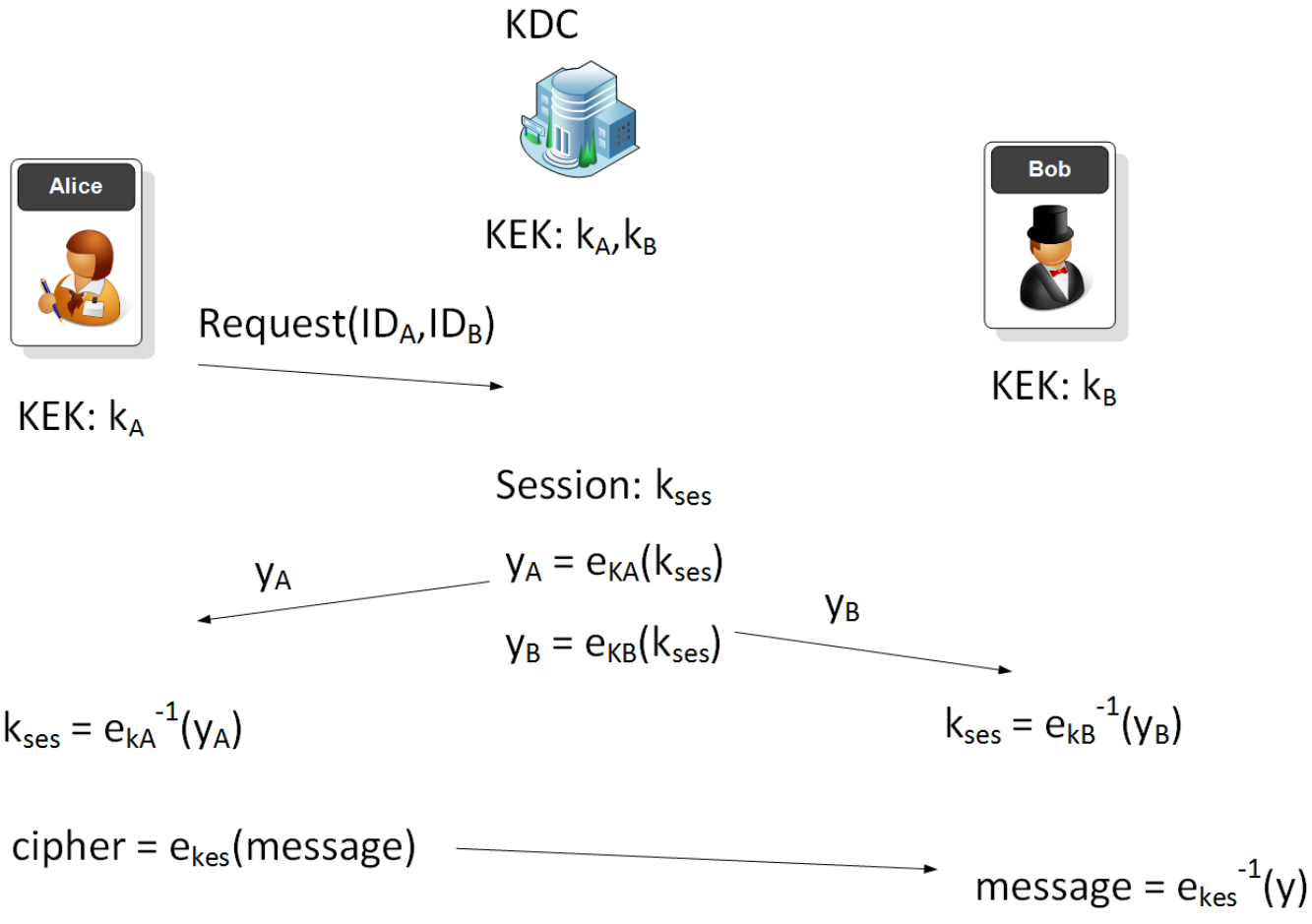
Trust Infrastructures



Trust Infrastructures



Simple KDC



Link

Simple KDC

KEK: k_A

Request(ID_A,ID_B)

y_A

$k_{ses} = e_{kA}^{-1}(y_A)$

cipher = e_{kes}(message)

message = e_{kes}^{-1}(y)

```
rnd = random.randint(1,2**128)
keyA= hashlib.md5(str(rnd)).digest()

rnd = random.randint(1,2**128)
keyB= hashlib.md5(str(rnd)).digest()

print 'Long-term Key Alice=',binascii.hexlify(keyA)
print 'Long-term Key Bob=',binascii.hexlify(keyB)

rnd = random.randint(1,2**128)

keySession= hashlib.md5(str(rnd)).hexdigest()
ya = encrypt(keySession,keyA,AES.MODE_ECB)
yb = encrypt(keySession,keyB,AES.MODE_ECB)

print "Encrypted key sent to Alice:",binascii.hexlify(ya)
print "Encrypted key sent to Bob:",binascii.hexlify(yb)

decipherA = decrypt(ya,keyA,AES.MODE_ECB)
decipherB = decrypt(yb,keyB,AES.MODE_ECB)
print "Session key:",decipherA print "Session key:",decipherB
```

[Link]

Simple KDC

KEK: k_A

Request(ID_A,ID_B

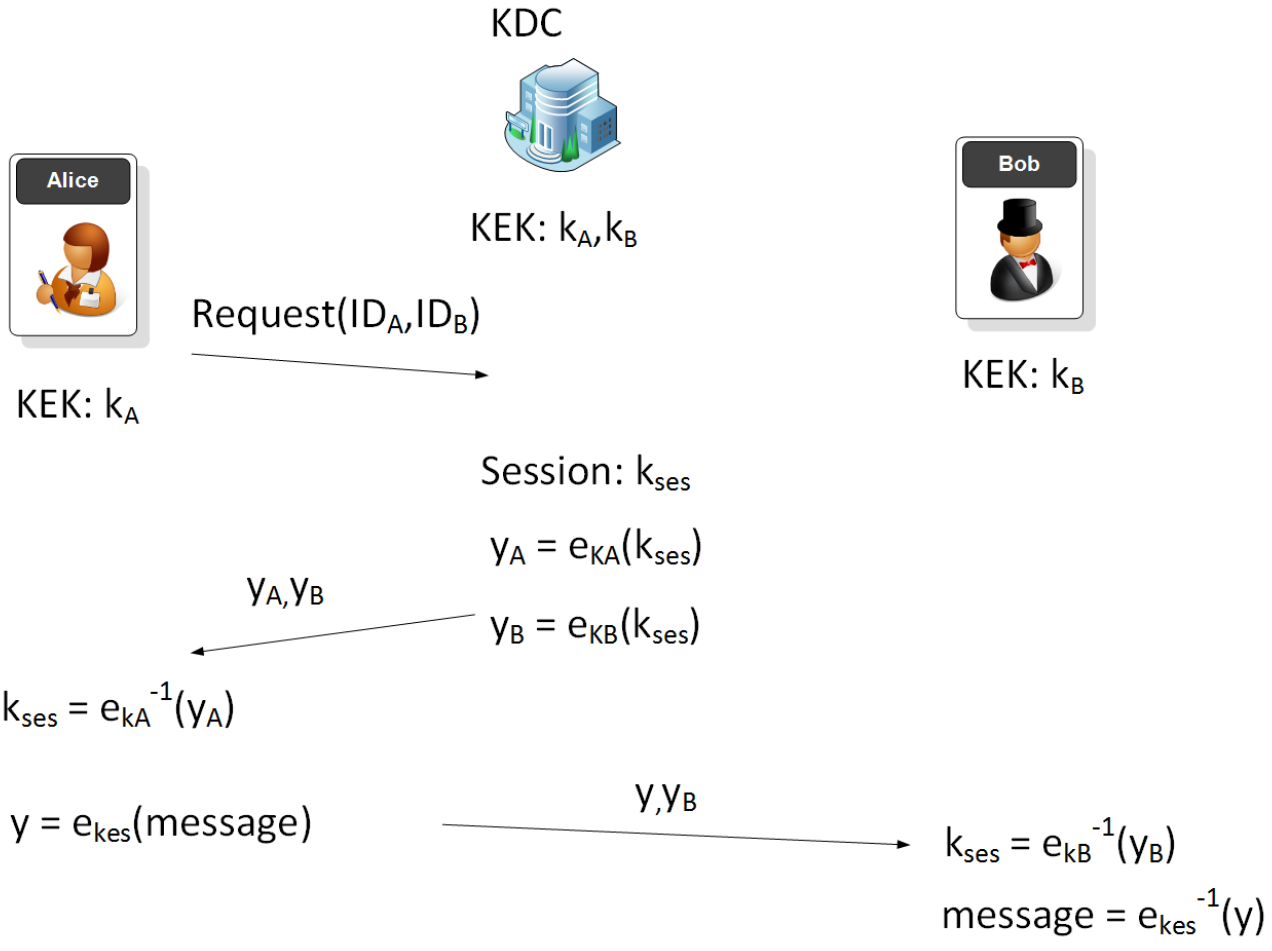
```
rnd = random.randint(1,2**128)
keyA= hashlib.md5(str(rnd)).digest()

rnd = random.randint(1,2**128)
keyB= hashlib.md5(str(rnd)).digest()

print 'Long-term Key Alice=',binascii.hexlify(keyA)
print 'Long-term Key Bob=',binascii.hexlify(keyB)

rnd = random.randint(1,2**128)

keySession= hashlib.md5(str(rnd)).hexdigest()
ya = encrypt(keySession,keyA,AES.MODE_ECB)
yb = encrypt(keySession,keyB,AES.MODE_ECB)

o Alice:",binascii.hexlify(ya)
o Bob:",binascii.hexlify(yb)

yA,AES.MODE_ECB)
yB,AES.MODE_ECB)
erA print "Session key:",decipherB
```

```
Long-term Key Alice= 9997205ef32f910d094b11b5f02ffe23
Long-term Key Bob= c2d0b8ac3567ac1f7305d223cecf3cbe
Encrypted key sent to Alice:
2c4ebf8e8748cb11065bb3cba7869c1af9d877c08805b2232ad4c7d8b2f987d5
Encrypted key sent to Bob:
8d9d34a0cf2385a328643e83c11f5523b9d241db50d2e534c563dbbac9bb08ba
Session key: 0ff7556d5a49f4f84f1b7e7c61c7c869
Session key: 0ff7556d5a49f4f84f1b7e7c61c7c869
```

$message = e_{kes}^{-1}(y)$

[Link]

Simple KDC (enhanced)



Link

Unit 5: Key Exchange

Diffie-Hellman
Diffie-Hellman Weaknesses
Elliptic Curve Diffie-Hellman (ECDH)
Passing Key Using Public Key
Key Distribution Centre (KDC)

Prof Bill Buchanan OBE

http://asecuritysite.com/crypto05
http://asecuritysite.com/encryption