Identifier Based PKC - Potential Applications

Dr. Ian Levy

CESG
Traditional PKC vs. Identifier PKC

Private Key → RSA, Diffie-Hellman → Public Key

Public Key → IDPKC → Private Key

Process requiring system secret information
Identifier PKC Methods

- Quadratic Residuosity Method - Cocks, CESG
- Elliptic Curve Pairing Method - Boneh & Franklin, Stanford
- Efficient Tate Pairing Method - Galbraith & Harrison, Royal Holloway University & HP Labs, Bristol
- Low bandwidth Weil Pairing Method - Pinch & Cocks, CESG
Communities of Interest

- The process of converting an identifier to a public key depends on the public system information.

- An identifier can therefore have a number of public/private key pairs associated with it. The appropriate authority must generate each ‘community private key’ unique to the identity.

- These keys provide for cryptographic separation between the communities (defined by the system public information) for a given identifier.
Communities of Interest - A Real Example

Judge

- Clerk

  - Prosecution Barrister
    - Solicitor
      - Paralegal
    - Solicitor
      - Paralegal
    - Solicitor
  
  - Defence Barrister
    - Solicitor
      - Paralegal
    - Solicitor
      - Paralegal
Communities of Interest - A Real Example

Judge

Clerk

Prosecution Barrister

John Smith Solicitor Solicitor

Para-legal Para-legal

Defence Barrister

Solicitor Solicitor

John Smith Para-legal
Communities of Interest - A Real Example
What about the authority?

- A single authority presents an obvious point of compromise or system failure.
- A single authority can masquerade as any given entity.
- Solution: split the authority into two or more co-operating parties.
Split authority

The authorities perform a one-time setup in order to share the system secrets in a secure manner.
Split authority

The user proves himself to each authority

Sys\(_1\)
Authority 1

User

Sys\(_2\)
Authority 2

I am Bob!

I am Bob!
Split authority
Each authority returns their part of the private key
Split authority
The user calculates his private key by combining the parts

Sys$_1$
Authority 1

I am Bob!

S$_{Bob}^1$

Sys$_2$
Authority 2

I am Bob!

S$_{Bob}^2$

User

S$_{Bob} = S_{Bob}^1 \times S_{Bob}^2$
Benefits of split authority

- No single point of compromise or failure.

- The private key can remain split until use, providing for more security.

- If the system secret is split between \( n \) authorities, then no \( n-1 \) of them can generate a key or compromise the system.

- Can have different authorities for different COIs. E.g. one registration authority and multiple community authorities.
Split authority registration

Authority 1 (Post Office)

Sys$_A^1$
Sys$_B^1$
Sys$_C^1$

Authority 2 (Tax office)

Sys$_A^2$

Authority 3 (DVLA)

Sys$_B^2$

Authority 4 (Social Security)

Sys$_D^2$
Split authority registration

Authority 1
(Post Office)
Sys$A_1$
Sys$B_1$
Sys$C_1$
Sys$D_1$
Sys$E_1$
Sys$F_1$

Authority 2
(Tax office)
Sys$A_2$

Authority 3
(DVLA)
Sys$B_2$

Authority 4
(Social Security)
Sys$D_2$

Physical Registration

Bob
Split authority registration

Authority 1
(Post Office)

Sys$_A^1$
Sys$_B^1$
Sys$_C^1$

Sys$_D^1$
Sys$_E^1$
Sys$_F^1$

Physical Registration

Bob

High assurance keys
Bob$_A^1$, Bob$_B^1$, ....

Authority 2
(Tax office)

Sys$_A^2$

Authority 3
(DVLA)

Sys$_B^2$

Authority 4
(Social Security)

Sys$_D^2$
Split authority registration

High assurance keys

Bob\textsuperscript{A}\textsubscript{1}, Bob\textsuperscript{B}\textsubscript{1}, \ldots.

Email registration

Bob

\textbf{Authority 2 (Tax office)}

\textbf{Authority 3 (DVLA)}

\textbf{Authority 4 (Social Security)}
Split authority registration

High assurance keys
Bob\(^A\)_1, Bob\(^B\)_1, ....

Email registration
Low assurance key
(by email) Bob\(^A\)_2

Authority 2 (Tax office)

Authority 3 (DVLA)

Authority 4 (Social Security)
Split authority registration

High assurance keys
Bob^{A_1, B_1}, ....

Bob

Web registration

Sys^{A_2}_2
Authority 2
(Tax office)

Sys^{B_2}_2
Authority 3
(DVLA)

Sys^{D_2}_2
Authority 4
(Social Security)
Split authority registration

High assurance keys
Bob^{A_1}, Bob^{B_1},….

Medium assurance key
(by physical post) Bob^{B_2}

Bob

Web registration

Sys^{A_2}
Authority 2
(Tax office)

Sys^{B_2}
Authority 3
(DVLA)

Sys^{D_2}
Authority 4
(Social Security)
Split authority registration

High assurance keys
Bob\(^A_1\), Bob\(^B_1\),….

Bob

Physical registration

Sys\(^A_2\)
Authority 2
(Tax office)

Sys\(^B_2\)
Authority 3
(DVLA)

Sys\(^D_2\)
Authority 4
(Social Security)
Split authority registration

Bob

High assurance keys
Bob\textsuperscript{A1}, Bob\textsuperscript{B1}, ....

Physical registration

 Sys\textsuperscript{A2}  
Authority 2  
(Tax office)

 Sys\textsuperscript{B2}  
Authority 3  
(DVLA)

 Sys\textsuperscript{D2}  
Authority 4  
(Social Security)

High assurance key
(face-to-face) Bob\textsuperscript{D2}
Security Paradigm

- An important culture shift is that the system (rather than the user/sender) determines whether the recipient is able to read a particular message. This has some interesting consequences.
- With traditional PKI, sending a message implies that the recipient can read it since the private key must exist. This is not true in an IDPKC.
New Key Semantics

- Since the public key is derived from some arbitrary string, a sender can use it to determine the hoops a recipient needs to jump through to get his private key.

- Since the associate private key doesn’t necessarily exist, these conditions need not be pre-arranged and can be ephemeral down to a single transaction.
New Key Semantics - Example 1

- We’ll create an anonymous payment system between me, a retailer and my bank.
- I don’t want the retailer to know my bank details, but he does want to ensure that he gets paid and gets paid only the right amount.
- I want to be able to track payments and the bank and the retailer want to be able to ensure I can’t lie about buying something.
- The bank is the sole authority in the system and the public system parameters are known to all.
Payment Protocol

The purchaser sends the retailer an order and random number encrypted under the identity comprised of his payment details and a second random number (to avoid replays)

\[ E_{[Alice|Bob|3081\ldots1626|\£20|R_2]} (Order|R_1), R_2 \]

Purchaser (Alice) \rightarrow Retailer (Bob)
Payment Protocol

The retailer tells the bank his merchant number, that the order is from Alice and the value of $R_2$ that Alice sent him.

\[ E_{\text{[Alice|Bob|3081…1626|£20|R_2]}} (\text{Order|R_1}), R_2 \]

The order is from Alice.
My merchant number is 12345
Order value is £20
The value of $R_2$ is…. 
Payment Protocol

The bank creates the private key. He uses the purchaser name that Bob purports, the name of the merchant related to the merchant number that Bob gives. He also looks up the credit card information for Alice. $R_2$ is used as is.

$$E_{[\text{Alice}|\text{Bob}|3081\ldots1626|\£20|R_2]}(\text{Order}|R_1), R_2$$

Purchaser (Alice) → Retailer (Bob)

Bank

The order is from Alice.  
My merchant number is 12345  
Order value is £20  
The value of R2 is….
Payment Protocol

The bank returns the private key to Bob who can decrypt the order.

\[ E_{[\text{Alice}|\text{Bob}|3081\ldots1626|\£20|R_2]} (\text{Order}|R_1), R_2 \]

- **Purchaser** (Alice)
- **Retailer** (Bob)
- **Bank**

Private key

The order is from Alice.
My merchant number is 12345
Order value is £20
The value of R2 is….
Payment Protocol - analysis

- Bob never sees Alice’s credit card details (or anything else in the key specification) unless Alice allows him.
- Alice can’t cheat by using Carol’s credit card number. If she does, the credit card number that the bank looks up doesn’t match that used by Alice in the key construction. Bob can’t decrypt the order.
- Eve, who’s listening to Bob’s communications can’t subvert the order. If she does, she must present her merchant number to the bank who’ll look up ‘Eve’ instead of ‘Bob’ when constructing the private key. Eve won’t be able to decrypt the order.
- If Eve present’s Bob’s merchant number and then tries to cash-in, the bank’s records won’t match and they won’t pay (hopefully!).
Role Based Access

- For example, releasing information dependent on an ephemeral role and other conditions.
- The sender constructs a key that contains conditions that must be satisfied by the recipient.
- The key generation authority implicitly validates the conditions on the recipient when generating the identifier string that results in a private key.
- Example: Releasing sensitive hospital patient records to a GP surgery.
Role Based Access

Hospital sends a discharge letter to the patient’s GP’s surgery. It’s encrypted and we want only a doctor who’s not a locum and is seeing the patient today to be able to read it.

\[ E_{[Fred Bloggs|Appointment|Not Locum|Doctor]}(Letter) \]
Role Based Access

Let’s say the practice nurse (who has an appointment with Fred) tries to read the letter. She asks the surgery authority to create her a key.

\[ E_{[\text{Fred Bloggs}\mid \text{Appointment}\mid \text{Not Locum}\mid \text{Doctor}]} \text{(Letter)} \]

Key for Carol to see Fred Bloggs’ information please.
Role Based Access

The authority checks her credentials: she has an appointment with Fred, she’s not a locum and she’s a nurse. So, it creates the key using those credentials. Obviously, she can’t read it.

\[E_{\text{[Fred Bloggs|Appointment|Not Locum|Doctor]}} \text{(Letter)}\]

Hospital → GP Surgery

Key generated from:
Fred Bloggs|Appointment|Not Locum|Nurse

Key for Carol to see Fred Bloggs’ information please.
Role Based Access

Now, let’s assume the doctor (who’s not a locum and has an appointment with Fred) tries to read the letter.

\[ E_{[Fred Bloggs|Appointment|Not Locum|Doctor]} \text{ (Letter)} \]

Key for Dr. Bob to see Fred Bloggs’ information please.
Role Based Access

The authority check his credentials and creates the key using those credentials. Dr. Bob can read the letter

\[ E_{\text{[Fred Bloggs|Appointment|Not Locum|Doctor]}} \text{(Letter)} \]

Key generated from:
Fred Bloggs|Appointment|Not Locum|Doctor

Key for Dr. Bob to see Fred Bloggs’ information please.
Role Based Access - analysis

- The sender determines the conditions that the potential recipient must meet in order to read the information.
- Those conditions needn’t be true at the time of information issue - they’re checked when the recipient wishes to see the data. That’s when you need to check it - the doctor may have been struck off in the meantime.
- If each surgery has its own community of interest (that the hospital knows about), or the identifier specification has the surgery name in it, then mis-routing of messages doesn’t compromise patient confidentiality.
The Real World - implementations

Execution times of optimised implementations of Tate pairing based elliptic curve IDPKC system created by Keith Harrison, HP Labs Bristol.

<table>
<thead>
<tr>
<th>Curve Size</th>
<th>Pentium III 650Mhz</th>
<th>PPC G4 667MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2^{241}$</td>
<td>12ms</td>
<td>8.9ms</td>
</tr>
<tr>
<td>$2^{271}$</td>
<td>18ms</td>
<td>14ms</td>
</tr>
<tr>
<td>$2^{283}$</td>
<td>20ms</td>
<td>15ms</td>
</tr>
<tr>
<td>$3^{97}$</td>
<td>34ms</td>
<td>17ms</td>
</tr>
<tr>
<td>$3^{163}$</td>
<td>114ms</td>
<td>54ms</td>
</tr>
</tbody>
</table>
What are we doing?

- Pilots - CESG, HP Labs Bristol and Viacode are piloting IDPKC/IBE systems for e-Government use.
- Research - we’re continuing research into IDPKC algorithms, the protocols they enable and applications.
- Partnering - talking to Industry and working with them to further IDPKC applications
- Standards - we hope that standards for submission to appropriate bodies will come out of our pilots.
Questions, anyone?

Keep an eye on www.cesg.gov.uk for developments!