Chapter 10 | Key Management and the Public Key Revolution

Tuesday, October 31, 2023 10:50 AM

10.1 Key Distribution and Key Management

Story:

- In Chapter 1-7 we have seen
  how private key cryptography
  can be used to ensure
  secrecy and integrity
  for two parties
  communicating over an insecure channel
  —assuming the two parties are in possision of a shared, secret key.
    - The question we have deferred since Ch 1, however, is

      How can the parties share a secret key in the first place?
    - Clearly
      the key cannot simply be sent over the public channel
      because
      an eavesdropping adversary would then be able to
      observe it en route.
    - Some other mechanism
      must be used instead.
- In some situations
  the parties may have access to a secure channel
  that they can use to reliably share a secret key.
    - e.g.
      two parties are co-located at some time
    - Alternatively
      the parties might use a trusted courier service (as a secure channel)
    - Stress: private key crypto is not useless—the secure channel may not be available
      at all times (or may be more expensive to use repeatedly)
- The above approaches have been used
  to share keys in government, diplomatic and military contexts.

  E.g.
  the "red phone" connecting Moscow and Washington
  in the 1960s
  was encrypted using a one-time pad

with keys shared by couriers
who flew from one country t othe other
carrying briefcases full fo print-outs.

Such approaches can also be used in corporations
e.g. to setup a shared key b/w a central databse
and
a new employee on his/her first day fo work
(we return to this example in the next section).

- Relying on a secure channel to distirbute keys
  however
  does not work well in many other situtions
  - E.g.
    consider a large MNC in which *every pair*
    of employees
    might need the ability to communicate securely,
    with their communacion protected from other employees as w.ll
    - It will be inconveninent
      for each pari of employees to meet so they
      can securely share a key

      Especially an issue if a new employee joins
      again have t oshare keys with everyone
    - Assumping these N employees are somehow
      able to securely share keys with each other
      another significant drawback si that
      each employee will have to manage and store $N-1$
      secret keys
      (one for each other employee).
      - In fact this may significantly undercount the keys
        —need keys for secure communication with
        remote resources such as
        databases, servers, printers etc.
    - The profiliration of so many keys is a
      significant logistical problem.
    - Moreover,
      these keys must be stored securely
      (harder when there are so many keys)
- Storing keys is anyway a concern
  - Smart cards, e.g., can be used
    but their memories are limited
    on how many keys they can store.
- Concerns above

can be addressed in "closed" organisations

but

"open interactions"

(e.g. sending an email to a new person

or

buying something from a merchant

for the first time)

- In the latter, private key crypto
  does not provide a solution.

Summary:
Private key cryptography has three problems

- Key distribution
- Key management (many keys arise)
- Inapplicability to open systems

10.2 A partial solution: Key-distribution Centres

- One way to address the concerns listed previously
  is to use a Key Distribution Centre—to share keys.
- Idea
  - KDC is a trusted entity in an organisation
  - When teh ith employee joins
    KDC creates a key b/w itself and this new employee
    also creates $k_1 \ldots k_{i-1}$ keys
    these are for letting the ith (new) employee communicate with
    all other employees

    keys to the $1 \ldots (i-1)$ employees is
    sent by the KDC to the employees
    by encrypting using the key the KDC already shares
    with the existing employees
  - Now, everyone can communicate with each other.
- Better Idea
  - KDC sets up a key with each new employee
  - Whenever user A wants to talk to user B
    they talk to the KDC and it issues a "session key"
  - When the users are done talking
    they end the session
  - Advantages
    - simplifies key distribution
    - reduces key storage complexity
  - Issues:
    - KDC is a high value target (for attacks)
    - KDC is a single point of failure

- Could consider replicating the KDC
  but then this means more points of failure
  and more keys/updates etc.

Protocols for key distribution using a KDC.

Story:

- Many protocols exist for secure key distribution using a KDC
  - E.g.
    Needham-Schroeder protocol
    which forms th ecore of
    Kerberos
    an important and widely used service
    for performing authentication and
    supporting secure communication
    (Kerberos is used in many universities
    and coroporations
    and is the default mechanism for
    supporting secure networked authentictaion and communication
    in Windows and many UNIX sysetms).

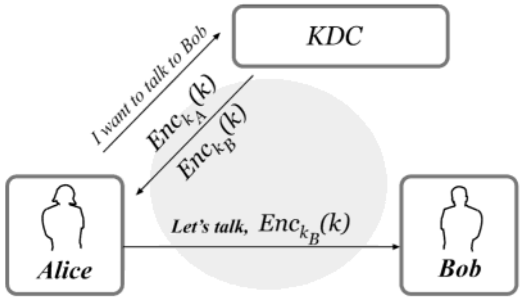
  - We only highlight one feature of this protocol.

  - When Alice contacts the KDC
    and asks to communicate with bob
    the KDC does not send the encrypted session key
    to both Alice nad Bob (like we described earlier).

  - Instead
    the KDC sends to Alice the sessien key
    encrypted under Alice's key
    in addition to
    the session key encrypted under Bob's key

  - Alice then forwards the second ciphertext
    to Bob as in the figure below

    - The second ciphertext is sometimes called a ticket
      and can be viewed as a credential
      that allows Alice to talk to Bob
      (and allows Bob to be assured that he is talking to Alice)



- Indeed
    - although we have not stressed this point
  - a KDC-based approach can provide a useful means of performing
    - authentication as well.

- Note also that Alice and Bob need not both be users
    - Alice might be a sure
        - and
    - Bob a resource
        - such as a server or a remote disk etc.

- The protocol was desigend in this way
    - to reduce the load on the KDC.

    - In the protocol as described
        - the KDC does not need to initiate a second
            - connection to Bob
        - and need not worry whether
            - Bob is online when Alice initiates the protocol

    - Moreover
        - if Alice retains the ticket (and her copy of the session key)
      - then she can re-initae secure communication with Bob
        - by simply re-sending the ticket to Bob
            - without the invlovement of the KDC at all
      - (In practice
        - tickets expire and eventually need to be renewed.
      - But a session could be re-established
        - within some acceptable time period).

- We conclude by noting that
    - in practice the key that Alice shares with the KDC
        - might be a short, easy-to-memorise passowrd.

    - In this case
        - many additional security problemms arise that must b edealt with.

    - We have also been implicitly assuming an attacker
        - who only passivel eavesdrops
            - rather than one who might actively triy to
        - interfere with the protocol

- We refer the interested reader to the references (at the end) for more info
  on how to address these.

10.3 Key Exchange and the Diffie-Hellman Protocol

Story:

- KDCs and protocols like Kerberos are commonly used in practice

  - But these approaches t othe key-distributino problem
    still require (at some point)
    a private and authenticated channel
    that can be used to share keys.

    (In particular
    we assemed the existence of such a channel b/w KDC and the
    employees on their first day).

  - Thus
    they still cannot solve the problem of key distribution
    in open systems, like the Internet
    where there may be no private channel available
    b/w two users who wish to communicate.

- To achieve private communication
  without ever communicating over a private channel
  a radically different approach is needed.
  - In 1976,
    Whitfield Diffe and Martin Hellman
    published a paper
    with the innocent-looknig title
    "New Directions in Cryptography".

  - In that work
    they observed that there is often *assymetry* in the world
    In particular
    there certain actions that can be easily performed
    but not
    easily reversed.

    E.g.
    padlocked without a key (i.e. easily)
    but then cannot be reopened (easily).

    more strikingly
    it is easy to shatter a glass vase
    but
    extremely difficult to put it back together again

Algorithmically (and more germane for our purposes)
it is easy to multiply
two large primes
but
difficult to recover those primes
from their product.

This is exactly the factoring problem
discussed in previous chapters.

- Diffie and Hellman
  realised that such phenomena could be used
  to derive interactive protocols
  for *secure key exchange*
  that allow two parties to
  share a scret key
  via communication over a public channel
  by having the parties perform operations that
  they can reserve but that
  an eavesdropper cannot.

- The exsitence of
  secure key-exchange protocols is quite amazing
  - It means that you
    and a friend could agree on a secret
    by simply shouting across a room
    (and performing some local computation);

    the secret would be unknown to anyone else
    even if they had listened to everything that was said.

  - Indeed,
    until 1976, it was generally beleived that
    secure communication could not be achivede
    without first sharing some secret information
    using a private communication channel.

- The influence of Diffe and Hellman's paper was enormous
  - In addition to introducing a fundamentally new way of looking at cryptography
    it was one of the first steps
    towards moving cryptography out of the private domain
    and into the public one.

    Quote (first two paragraphs of their paper):

> *We stand today on the brink of a revolution in cryptography. The development of cheap digital hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade cryptographic devices down to where they can be used in such commercial applications as remote cash dispensers and computer terminals.*
>
> *In turn, such applications create a need for new types of cryptographic systems which minimize the necessity of secure key distribution channels. ... At the same time, theoretical developments in information theory and computer science show promise of providing provably secure cryptosystems, changing this ancient art into a science.*

- Diffie and Hellman were not exaggerating
  and the revolution they spoke of was
  due in great part to their work.

- In this section
  we present the Diffie-Hellman key-exchange protocol.

  We prove its security against eavesdropping adversaries
  (or equivalently)
  under the assumption that the parties
  communicate over a public but *authenticated* channel

  (so an attacker cannot interfere with their communication).

  - Security against an eavesdropping adversary is a
    relatively weak guarantee
    and in practice
    key-exchange protocols must satisfy stronger notinos
    of security
    that are beyond our present scope

    (moreover, we are interested here in the setting
    where the communicating parties have *no* prior shared information
    in which case
    there is nothing that can be done
    to prevent an adversary from
    impersonating one of the parties—we return to this point later).

The setting and definition of security.

Story:

- We consider a setting with two parties—Alice and Bob—
  who run a probabilistic protocol Π
  in order to generate a shared secret key

  - Π can be viewd as the set of instructions
    for Alice and Bob in the protocol.

- Alice and Bob begin by holding the security parameter 1^n
  - They then run Π using independent random bits.
  - At the end of the protocol
    Alice and bob output keys
    $k_A, k_B \in \{0,1\}^n$
  - The basic correctness requirement is that $k_A = k_B$
    - Since we only deal with protocols that satisfy this requirement
      we speak simply of *the* $k = k_A = k_B$
      generated by an honest execution of Π.

- We now turn to defining security.
  - Intutively, a key-exchange protocol is
    secure if
    the key output by Alice and Bob is completely unguessable
    by an eavesdropping adversary.
  - This is formally defined by requiring that
    an adversary who has eavesdropped on an execution of the protocol
    should be unable to distinguish
    the key k generated by that execution
    from
    a *uniform* key of length n.
  - This is much stronger than
    simply requiring that the adversary be
    unable to compute k *exactly*,
    and this stronger notion is necessary if
    the parties will subsequently use k for some
    cryptographic application
    (e.g. as a key for a private-key encryption scheme).

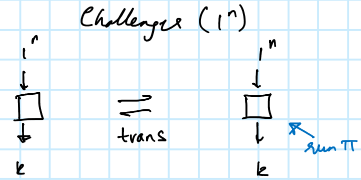
- Formalising the above
  - let Π be a key-exchange protocol
  - $\mathcal{A}$ be an adversary
  - and n the security parameter.

Consider the following experiment.

The Key Exchange Experiment $KE^{eav}_{\mathcal{A},\Pi}(n)$:

Adversary — Challenger (1^n)

1.



$b \xleftarrow{\$} \{0,1\}$, $r \leftarrow \{0,1\}^n$

2. $\hat{k} := \begin{cases} k & \text{if } b = 0 \\ r & \text{if } b = 1 \end{cases}$

3. $\xleftarrow{\text{trans},\ \hat{k}}$

3. $\xleftarrow{\text{trans}, \hat{k}}$

4 $\xrightarrow{b'}$ output 1 if $b' = b$

0 else

The key-exchange experiment $\mathsf{KE}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n)$:

1. *Two parties holding 1^n execute protocol Π. This results in a transcript* trans *containing all the messages sent by the parties, and a key k output by each of the parties.*
2. *A uniform bit $b \in \{0,1\}$ is chosen. If $b = 0$ set $\hat{k} := k$, and if $b = 1$ then choose $\hat{k} \in \{0,1\}^n$ uniformly at random.*
3. *$\mathcal{A}$ is given* trans *and $\hat{k}$, and outputs a bit b'.*
4. *The output of the experiment is defined to be 1 if $b' = b$, and 0 otherwise. (In case $\mathsf{KE}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n) = 1$, we say that $\mathcal{A}$* succeeds.*)*

Remarks:

- $\mathcal{A}$ is given trans to capture the fact that $\mathcal{A}$ eavesdrops on the entire execution of the protocol and
  thus
  sees all messages exchanged by the parties.
  - In the real wordl
    $\mathcal{A}$ would not be given any key;
    in the experiment
    the adversary is given $\hat{k}$ only as a means
    of defining what it means for $\mathcal{A}$
    to "break" the security of Π.
    - That is,
      the adversary succeeding in "breaking" Π
      if it can correctly determine
      whether the key $\hat{k}$ is the real key corresponding
      to the given execution of the protocol
      OR
      whether $\hat{k}$ is a uniform key that is
      independent of teh transcript.
    - As expected,
      we say Π is secure if
      the adverasry succeeds with
      probability that is at most negligibly greater than 1/2.
    - i.e.

Definition 10.1

A key-exchange protocol Π is *secure in the presence of an eavesdropper*
if
for all PPT adversaries $\mathcal{A}$
there is a negligible function negl such that

$$\Pr\left[\mathsf{KE}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n) = 1\right] \leq \frac{1}{2} + \mathsf{negl}(n).$$

Story:

- The aim of a key-exchange protocol
  is almost always to generate a shared key k
  that will be used by the parties for
  some further cryptographic purpose
  e.g. to encrypt and authenticate their
  subsequent communication using
  say
  an authenticated encryption scheme.
  - Intuitively
    using a shared key generated by a
    secure key exchange protocol
    should be "as good as" using a key shared
    over a private channel.
  - It is possible to prove this formally; see Exercise 10.1

10.1 Let Π be a key-exchange protocol, and $(\mathsf{Enc}, \mathsf{Dec})$ be a private-key encryption scheme. Consider the following *interactive* protocol Π' for encrypting a message: first, the sender and receiver run Π to generate a shared key k. Next, the sender computes $c \leftarrow \mathsf{Enc}_k(m)$ and sends c to the other party, who decrypts and recovers m using k.

(a) Formulate a definition of indistinguishable encryptions in the presence of an eavesdropper (cf. Definition 3.8) appropriate for this interactive setting.

(b) Prove that if Π is secure in the presence of an eavesdropper and $(\mathsf{Enc}, \mathsf{Dec})$ has indistinguishable encryptions in the presence of an eavesdropper, then Π' satisfies your definition.

The Diffie-Hellman key-exchange protocol

Story:

- We now describe
  the key-exchange protocol that
  appeared in the original paper by
  Diffie and hellman
  (although they were less formal than we will be here).

- Let $\mathcal{G}$ be a PPT algorithm that on input 1^n

outputs

- a description of a cyclic group $\mathbb{G}$
- its order q (with $||q|| = n$) and
- a generator $g \in \mathbb{G}$.

< guessing $||q||$ refers to the number of bits needed to store n.

- The Diffe-Hellman key exchange is as follows

Construction 10.2

- Common input: Security parameter 1^n
- The protocol:

Alice

Bob

1. Runs $\mathcal{G}(1^n)$ to obtain $(\mathbb{G}, q, g)$

2. Chooses $x \leftarrow \mathbb{Z}_q$ uniformly and computes $h_A \coloneqq g^x$.

3. **Sends** $(\mathbb{G}, \boldsymbol{g}, \boldsymbol{q}, \boldsymbol{h_A})$

4. Recieves $(\mathbb{G}, g, q, h_A)$

   Chooses $y \leftarrow \mathbb{Z}_q$ uniformly
   computes $h_B \coloneqq g^y$.

   Sends $\boldsymbol{h_B}$ to Alice
   Outputs the key $k_B \coloneqq h_A^y$

5. Receives h_B and outputs the key $k_A \coloneqq h_B^x$

(NB for babies: both $k_A = k_B = g^{xy}$)

Story:

- In our description
  we assumed that Alice generates $(\mathbb{G}, q, g)$
  and
  sends these parameters to Bob
  as part of her first message.

- In practice,
  these paremeters are *standardised*
  and
  are fixed
  and
  are known to both parties
  before the protocol begins.

- In that case, Alice need only send
  h_A and
  Bob need not wait to recieve Alice's message
  before computing and sending h_B.

- It is not hard to see that
  the protocol is correct:
  (noted above already).

  - The observant reader will note
    that the shared key is a group element
    not a bit-string
    We return to this point later

Side remark

- Claim: with only classical communication
  and an unbounded eavesdropper
  no Π for exchanging keys
  can be secure.

- Proof idea:
  The adversary can simply simulate
  Alice and Bob
  consistent with the transcript
  and thus learn the
  keys they generate.

- Diffie and Hellman
  did not prove security of their protocol
  —indeed, the appropriate notions
  (both the defintional framework
  as well as the idiea of formulating
  precise assumptions)
  were not yet in place.

  - Let's see what sort of assumption
    will be needed in order for the protocol to be secure.

  - A first observation (due to DH)
    is that a *minimal* requirement for security here
    is that
    the **discrete-logarithm problem** be hard
    relative to $\mathcal{G}$
    (recall: $\mathcal{G}$ was generating the tuple $(\mathbb{G}, q, g)$).

  - If not,
    then an adversary given the trasnscript
    (which in particular includes h_A)
    can compute the secret value of
    one of the parties (i.e. x)
    and
    then easily compute the shared key
    using that value.

- So,
  hardness of the discrete-log problem
  is necessary for the protocol to be secure.

- It is, not, however
  sufficient—as it is posible that there are
  other ways of computing the key
  $k_A = k_B$ without
  explicitly computing x or y.

- **Computational Diffie-Hellman assumption**
  requires that the key g^{xy} is hard to compute in its *entirety*
  from the transcript.
  - NB: this does not suffice either

- What is required by Definition 10.1 is that the shared key g^{xy} should be *indistinguishable from uniform* for any
  adversary given g, g^x, g^y.
  - This is the **decisional Diffie-Hellman assumption**
    introduced in Section 8.3.2

- As we will see,
  a proof of security for the protocol follows
  almost immediately from the decisional Diffie-Hellman assumption.
  - This should not be surprising
    as the Diffie-Hellman assumptions were introduced
    well after Diffie and Hellman published their paper

    as a way of abstracting the properties underlying the
    (conjectured) security of the Diffie-Hellman protocol.

  - Given this
    it is fair to ask whether anything is gained by
    defining and proving security here.

  - By this point in the book
    hopefully you're convinced the answer is *yes*.

    (and I was still not! □)

- Precisely defining secure key exchange
  forces us to think about
  exactly what security properties we require
  specifying a precise assumption
  (namely the decisional Diffie-Hellman assumption)
  means
  we can study this assumption
  independently of any particular application
  and—once we are convinced of its plausibility—
  construct other protocols based on it.

- finally proving security shows that
  the assumption does, indeed, suffice
  for the protocol to meet our
  desired notion of security.

- In our proof of security
  we used a modified version of Definition 10.1
  in which it is required that the shared key
  be indistinguishable from a
  uniform elemnt of $\mathbb{G}$
  rather than a
  uniform n-bit string.

  - This discrepancy will need to be addressed
    before the protocol can be used in practice
    —after all,
    group elements are not typically useful as
    cryptographic keys
    and the representation of a uniform group element
    will not (in general)
    be a uniform bit-string—
    and we briefly discuss one standard way to do so
    following the proof.

  - For now
    we let $\widehat{\mathrm{KE}}^{\mathrm{Eav}}_{\mathcal{A},\Pi}(n)$ denote a
    modified experiment
    where if $b = 1$ the
    adversary is given $\hat{k}$ chosen
    uniformly from $\mathbb{G}$
    instead of a uniform n-bit string.

Theorem 10.3

If the decisional Diffie-Hellman problem is hard
relative to $\mathcal{G}$
then the Diffie-Hellman key-exchange protocol Π
is secure in the presence of
an eavesdropper

(with respect to the modified experiment $\widehat{\mathrm{KE}}^{\mathrm{eav}}_{\mathcal{A},\Pi}$)

Proof:

Let $\mathcal{A}$ be a PPT adversary.

Since $\Pr[b=0] = \Pr[b=1] = 1/2$

it holds that

$$\Pr\left[\widehat{KE}^{eav}_{\mathcal{A},\Pi}(n)=1\right] = \frac{1}{2}\Pr\left[\widehat{KE}^{eav}_{\mathcal{A},\Pi}(n)=1 \,\middle|\, b=0\right] + \frac{1}{2}\Pr\left[\widehat{KE}^{eav}_{\mathcal{A},\Pi}(n)=1 \,\middle|\, b=1\right].$$

In experiment $\widehat{\mathrm{KE}}$

the adversary $\mathcal{A}$ recieves

$(\mathbb{G}, q, g, h_A, h_B, \hat{k})$

where the first part (excluding $\hat{k}$)

is the transcript

and $\hat{k}$ is either the actual key or a uniform group element (depending on b being 0 or 1, resp.).

Distinguishing b/w these two cases is

exactly equilavent to solving the decisiniol Diffie-Hellman problem.

Here are the details

(basically relates distinguishing

to the security game here)

$$
\begin{aligned}
&\Pr\left[\widehat{\mathsf{KE}}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n)=1\right] \\
&= \frac{1}{2}\cdot \Pr\left[\widehat{\mathsf{KE}}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n)=1 \mid b=0\right] + \frac{1}{2}\cdot \Pr\left[\widehat{\mathsf{KE}}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n)=1 \mid b=1\right] \\
&= \frac{1}{2}\cdot \Pr[\mathcal{A}(\mathbb{G},g,q,g^x,g^y,g^{xy})=0] + \frac{1}{2}\cdot \Pr[\mathcal{A}(\mathbb{G},q,g,g^x,g^y,g^z)=1] \\
&= \frac{1}{2}\cdot \left(1-\Pr[\mathcal{A}(\mathbb{G},g,q,g^x,g^y,g^{xy})=1]\right) + \frac{1}{2}\cdot \Pr[\mathcal{A}(\mathbb{G},q,g,g^x,g^y,g^z)=1] \\
&= \frac{1}{2}+\frac{1}{2}\cdot \left(\Pr[\mathcal{A}(\mathbb{G},g,q,g^x,g^y,g^z)=1]-\Pr[\mathcal{A}(\mathbb{G},q,g,g^x,g^y,g^{xy})=1]\right) \\
&\le \frac{1}{2}+\frac{1}{2}\cdot \left|\Pr[\mathcal{A}(\mathbb{G},g,q,g^x,g^y,g^z)=1]-\Pr[\mathcal{A}(\mathbb{G},q,g,g^x,g^y,g^{xy})=1]\right|,
\end{aligned}
$$

where the probabilities are all taken over $(\mathbb{G},q,g)$ output by $\mathcal{G}(1^n)$, and uniform choice of $x,y,z\in\mathbb{Z}_q$. (Note that since g is a generator, g^z is a uniform element of $\mathbb{G}$ when z is uniformly distributed in $\mathbb{Z}_q$.) If the decisional Diffie–Hellman assumption is hard relative to $\mathcal{G}$, that exactly means that there is a negligible function negl for which

$$\left|\Pr[\mathcal{A}(\mathbb{G},g,q,g^x,g^y,g^z)=1]-\Pr[\mathcal{A}(\mathbb{G},q,g,g^x,g^y,g^{xy})=1]\right| \le \mathsf{negl}(n).$$

We conclude that

$$\Pr\left[\widehat{\mathsf{KE}}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n)=1\right] \le \frac{1}{2}+\frac{1}{2}\cdot\mathsf{negl}(n),$$

completing the proof. ■

□

Uniform group elements vs uniform bit-strings

(quick version)

- To go from indistinguishability from uniform group elements
  to that from uniform strings
  the DH protocol can be modified by
  having the parties apply an appropriate
  key-derivation function
  to their shared group element g^{xy}.

Active adversaries

- Active adversary—modify the communication b/w Alice and Bob
- Two classes of active adversaries informally considered
  - Impersonation—act as the other player
  - Man-in-the-middle—Alice and Bob communicate but the adversary intercpets and changes the messages
- We don't study these in the book
  - involved
  - cannot be achieved without *some* information
    shared between the parties in advance
- DH is completel insecure against
  man-in-the-middle—there is an attack
  where Alice and Bob end up with differentkeys
  k_A and k_B

both known to the adversary
but neither party detecting th eadversary.

Diffie–Hellman key exchange in practice. The Diffie–Hellman protocol in its basic form is typically not used in practice due to its insecurity against man-in-the-middle attacks, as discussed above. This does not detract in any way from its importance. The Diffie–Hellman protocol served as the first demonstration that asymmetric techniques (and number-theoretic problems) could be used to alleviate the problems of key distribution in cryptography. Furthermore, the Diffie–Hellman protocol is at the core of standardized key-exchange protocols that are resilient to man-in-the-middle attacks and are in wide use today. One notable example is TLS; see Section 12.8.