Assignment 1

- Pick sun.com and one other site. Using whois and ARIN, get as much information as possible about the IP addressing, the DNS and the site (location, owner, etc.)
- Problems (p83): 3.5,c and 3.6
- Due next class March 6
Assignment 1, 3.5c

Plaintext: 8

\[ 8^{17} = \frac{2215799813685248}{77} = 8^{17} \mod 77 = \text{Ciphertext} \]

\[ 57^{53} = 57^{53} \mod 77 = \text{Plaintext} \]

KU = \{e, n\} = \{17, 77\}

KR = \{d, n\} = \{53, 77\}

\[ de = 1 \mod 60 \text{ and } d < 60 \]

\[ 53 \times 17 = 15 \times 60 + 1 \]
Assignment 1, 3.6

Plaintext
M=5

$\Rightarrow M^5 \mod 35 = \text{Ciphertext}$

10

$KU = \{e,n\} = \{5,35\}$

- This is done with brute force, starting with $1^5$, then $2^5$, etc. or, since $n=35$ we can easily determine the factors $p=7$ $q=5$ and then $\phi(n)=6\times4=24$, therefore $d=5$ since $5\times5=1\times24+1$

- Remember that the security of RSA depends wholly on the problem of factoring large numbers
Network Security

Electronic Mail Security
Electronic Mail Security

Agenda:

- Introduction to PGP
- 5 PGP Services
- Key Management
- Use of Trust
- Demo Of PGP In Use
Pretty Good Privacy

- **1991** – Creation of a single person, Phil Zimmermann
- Provides confidentiality and authentication services for electronic mail and file storage applications
Phil Zimmermann

- Target of three year criminal investigation
- Gave software away to friend who put it on the Internet in 1991
- Intended to give individuals “the right to be let alone”
- US export restrictions violated – same class as munitions and nuclear weapons
- Government dropped the case in 1996

“PGP has spread like a prairie fire, fanned by countless people who fervently want their privacy restored in the information age”

- Phil Zimmermann, testifying before the US Senate, 1996
Pretty Good Privacy

- Selected **best** available cryptographic algorithms
- Integrated these algorithms into a general purpose application
- **Source code** and doc freely available on the net
- Agreement with company (Viacrypt) for **low cost commercial version**
Notation

\( K_S \) = session key used in conventional encryption
\( KR_a \) = private key of user A, used in public key encryption
\( KU_a \) = public key of user A, used in public key encryption
\( EP \) = public-key encryption
\( DP \) = public-key decryption
\( EC \) = conventional encryption
\( DC \) = conventional decryption
\( H \) = hash function
\( || \) = concatenation
\( Z \) = compression using ZIP algorithm
\( R64 \) = conversion to radix 64 ASCII format
## Summary of 5 PGP Services

<table>
<thead>
<tr>
<th>Function</th>
<th>Algorithms Used</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital signature</td>
<td>DSS/SHA or RSA/SHA</td>
<td>A hash code of a message is created using SHA-1. This message digest is encrypted using DSS or RSA with the sender's private key, and included with the message.</td>
</tr>
<tr>
<td>Message encryption</td>
<td>CAST or IDEA or Three-key Triple DES with Diffie-Hellman or RSA</td>
<td>A message is encrypted using CAST-128 or IDEA or 3DES with a one-time session key generated by the sender. The session key is encrypted using Diffie-Hellman or RSA with the recipient's public key, and included with the message.</td>
</tr>
<tr>
<td>Compression</td>
<td>ZIP</td>
<td>A message may be compressed, for storage or transmission, using ZIP.</td>
</tr>
<tr>
<td>Email compatibility</td>
<td>Radix 64 conversion</td>
<td>To provide transparency for email applications, an encrypted message may be converted to an ASCII string using radix 64 conversion.</td>
</tr>
<tr>
<td>Segmentation</td>
<td>—</td>
<td>To accommodate maximum message size limitations, PGP performs segmentation and reassembly.</td>
</tr>
</tbody>
</table>
Recall One Way Hash Function

Digital signature

No key distribution

Less computation since message does not have to be encrypted
Recall SHA-1 Secure Hash Function

- Developed by NIST in 1995
- Input is processed in 512-bit blocks
- Produces as output a 160-bit message digest
- Every bit of the hash code is a function of every bit of the input
- Very secure – so far!
Authentication

1. Sender *creates* a message
2. *Generate* a hash code with SHA-1
3. Using sender’s private key and RSA, *encrypt* the hash code and *prepend* to the message
4. Receiver uses sender’s public key to *decrypt* and recover the hash code
5. Receiver *generates* a new hash code for the message and *compares* with the decrypted hash code. If matching, then message is authentic
PGP Cryptographic Functions

(a) Authentication only
Recall Other Public Key Algorithms

- Digital Signature Standard (DSS) – makes use of SHA-1 and presents a new digital signature algorithm (DSA)
- Only used for digital signatures not encryption or key exchange
Authentication

- Other alternatives can be used, e.g., DSS
- Detached signatures are supported
- Good for executables and multi-party signatures (legal contract)
### Summary of 5 PGP Services

<table>
<thead>
<tr>
<th>Function</th>
<th>Algorithms Used</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital signature</td>
<td>DSS/SHA or RSA/SHA</td>
<td>A hash code of a message is created using SHA-1. This message digest is encrypted using DSS or RSA with the sender’s private key, and included with the message.</td>
</tr>
<tr>
<td>Message encryption</td>
<td>CAST or IDEA or Three-key Triple DES with Diffie-Hellman or RSA</td>
<td>A message is encrypted using CAST-128 or IDEA or 3DES with a one-time session key generated by the sender. The session key is encrypted using Diffie-Hellman or RSA with the recipient's public key, and included with the message.</td>
</tr>
<tr>
<td>Compression</td>
<td>ZIP</td>
<td>A message may be compressed, for storage or transmission, using ZIP.</td>
</tr>
<tr>
<td>Email compatibility</td>
<td>Radix 64 conversion</td>
<td>To provide transparency for email applications, an encrypted message may be converted to an ASCII string using radix 64 conversion.</td>
</tr>
<tr>
<td>Segmentation</td>
<td>—</td>
<td>To accommodate maximum message size limitations, PGP performs segmentation and reassembly.</td>
</tr>
</tbody>
</table>
Recall CAST-128

- 1997, Entrust Technologies
- RFC 2144
- Extensively reviewed
- Variable key length, 40-128 bits
- Used in PGP
Recall Conventional Encryption Algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Key Size (bits)</th>
<th>Block Size (bits)</th>
<th>Number of Rounds</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>DES</td>
<td>56</td>
<td>64</td>
<td>16</td>
<td>SET, Kerberos</td>
</tr>
<tr>
<td>Triple DES</td>
<td>112 or 168</td>
<td>64</td>
<td>48</td>
<td>Financial key management, PGP, S/MIME</td>
</tr>
<tr>
<td>AES</td>
<td>128, 192, or 256</td>
<td>128</td>
<td>10, 12, or 14</td>
<td>Intended to replace DES and 3DES</td>
</tr>
<tr>
<td>IDEA</td>
<td>128</td>
<td>64</td>
<td>8</td>
<td>PGP</td>
</tr>
<tr>
<td>Blowfish</td>
<td>variable to 448</td>
<td>64</td>
<td>16</td>
<td>Various software packages</td>
</tr>
<tr>
<td>RC5</td>
<td>variable to 2048</td>
<td>64</td>
<td>variable to 255</td>
<td>Various software packages</td>
</tr>
</tbody>
</table>

We have choices in PGP for confidentiality!
Confidentiality

1. Sender creates a message and random 128bit number for session key
2. Message encrypted using CAST-128 with the session key
3. Session key encrypted with recipient’s public key and prepended to the message
4. Receiver uses it’s private key to decrypt and recover the session key
5. Session key is used to decrypt the message
PGP Cryptographic Functions

(b) Confidentiality only
Confidentiality

- **Alternatives** for conventional encryption: RSA or Diffie-Hellman (ElGamal)
- **Conventional algorithms** are much faster
- Each message is a one time independent event with its own key
- $768 \leq \text{key size} \leq 3072$
Confidentiality & Authentication

- Both services can be used for the same message.
- First, signature is generated for plaintext and prepended.
- Message is encrypted with a session key.
- Session key is encrypted with recipient’s public key.
PGP Cryptographic Functions

(c) Confidentiality and authentication
## Summary of 5 PGP Services

<table>
<thead>
<tr>
<th>Function</th>
<th>Algorithms Used</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital signature</td>
<td>DSS/SHA or RSA/SHA</td>
<td>A hash code of a message is created using SHA-1. This message digest is encrypted using DSS or RSA with the sender's private key, and included with the message.</td>
</tr>
<tr>
<td>Message encryption</td>
<td>CAST or IDEA or Three-key Triple DES with Diffie-Hellman or RSA</td>
<td>A message is encrypted using CAST-128 or IDEA or 3DES with a one-time session key generated by the sender. The session key is encrypted using Diffie-Hellman or RSA with the recipient's public key, and included with the message.</td>
</tr>
<tr>
<td>Compression</td>
<td>ZIP</td>
<td>A message may be compressed, for storage or transmission, using ZIP.</td>
</tr>
<tr>
<td>Email compatibility</td>
<td>Radix 64 conversion</td>
<td>To provide transparency for email applications, an encrypted message may be converted to an ASCII string using radix 64 conversion.</td>
</tr>
<tr>
<td>Segmentation</td>
<td>—</td>
<td>To accommodate maximum message size limitations, PGP performs segmentation and reassembly.</td>
</tr>
</tbody>
</table>
Compression – Save Space

- PGP compresses (ZIP) the message after applying the signature but before encryption (default)
- Better to sign an uncompressed message
- PGP’s compression algorithm is non-deterministic
- Security is greater if message is encrypted after compression
- Appendix 5A - ZIP
PGP Cryptographic Functions
Summary of 5 PGP Services

<table>
<thead>
<tr>
<th>Function</th>
<th>Algorithms Used</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital signature</td>
<td>DSS/SHA or RSA/SHA</td>
<td>A hash code of a message is created using SHA-1. This message digest is encrypted using DSS or RSA with the sender's private key, and included with the message.</td>
</tr>
<tr>
<td>Message encryption</td>
<td>CAST or IDEA or Three-key Triple DES with Diffie-Hellman or RSA</td>
<td>A message is encrypted using CAST-128 or IDEA or 3DES with a one-time session key generated by the sender. The session key is encrypted using Diffie-Hellman or RSA with the recipient's public key, and included with the message.</td>
</tr>
<tr>
<td>Compression</td>
<td>ZIP</td>
<td>A message may be compressed, for storage or transmission, using ZIP.</td>
</tr>
<tr>
<td>Email compatibility</td>
<td>Radix 64 conversion</td>
<td>To provide transparency for email applications, an encrypted message may be converted to an ASCII string using radix 64 conversion.</td>
</tr>
<tr>
<td>Segmentation</td>
<td>—</td>
<td>To accommodate maximum message size limitations, PGP performs segmentation and reassembly.</td>
</tr>
</tbody>
</table>

authentication

confidentiality
E-mail Compatibility

- Part or all of block consists of a stream of arbitrary 8-bit octets
- Many mail systems only allow ASCII text
- PGP converts raw binary stream to a stream of printable ASCII characters
- Radix-64 conversion – 3 binary => 4 ASCII
Stream Of Printable ASCII Chars

-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6.3i
mQBNAi23Dv0AAAAECAMm6GNU3nqebKr3HW/fmrEhM1rFkwuZ6KHIYEat92nYfQIUj
1RLgj3TPHTRIMbswyTdaIJA70vkSgxETLBCExx0ABRG0K0FuZHJ1YXMgUml1Z2Vy
IDwxMDAxMTEuMzU0MEBjb21wdXNlcnZlLmNvbT4=
=8t7f
-----END PGP PUBLIC KEY BLOCK-----
Generic Transmission Diagram

- X ← file
- Signature required?
  - Yes: generate signature X ← signature || X
  - No: Compress X ← Z(X)
- Confidentiality required?
  - Yes: encrypt key, X ← E_{KUB}(K_s) || E_{K_s}(X)
  - No: convert to radix 64 X ← R64[X]
Generic Reception Diagram

convert from radix 64
X ← R64⁻¹[�]

Confidentiality required?
Yes

decrypt key, X
K ← DKRb[Kₜ]; X ← DKₜ[X]

No

Decompress
X ← Z⁻¹(X)

Signature required?
Yes

Strip signature from X
verify signature

No
# Summary of 5 PGP Services

<table>
<thead>
<tr>
<th>Function</th>
<th>Algorithms Used</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital signature</td>
<td>DSS/SHA or RSA/SHA</td>
<td>A hash code of a message is created using SHA-1. This message digest is encrypted using DSS or RSA with the sender’s private key, and included with the message.</td>
</tr>
<tr>
<td>Message encryption</td>
<td>CAST or IDEA or Three-key Triple DES with Diffie-Hellman or RSA</td>
<td>A message is encrypted using CAST-128 or IDEA or 3DES with a one-time session key generated by the sender. The session key is encrypted using Diffie-Hellman or RSA with the recipient’s public key, and included with the message.</td>
</tr>
<tr>
<td>Compression</td>
<td>ZIP</td>
<td>A message may be compressed, for storage or transmission, using ZIP.</td>
</tr>
<tr>
<td>Email compatibility</td>
<td>Radix 64 conversion</td>
<td>To provide transparency for email applications, an encrypted message may be converted to an ASCII string using radix 64 conversion.</td>
</tr>
<tr>
<td>Segmentation</td>
<td>—</td>
<td>To accommodate maximum message size limitations, PGP performs segmentation and reassembly.</td>
</tr>
</tbody>
</table>
Segmentation

- Maximum message length restrictions in e-mail
- PGP automatically subdivides a large message into segments small enough to mail separately
- PGP reassembles entire original block at the receiving end
Summary of 5 PGP Services

- Authentication
- Confidentiality
- Compression
- E-Mail Compatibility
- Segmentation
PGP Cryptographic Keys

- **One-time Session** Conventional Keys
- **Public** Keys
- **Private** Keys
- **Passphrase-Based** Conventional
Key Requirements

- A means of generating unpredictable session keys
- Allow users to have multiple public/private key pairs (*need some kind of identity*)
- Each PGP entity must maintain a file of its and its correspondents public/private pairs
Session Key Generation

- Random **128-bit numbers** are generated using CAST-128
- Input is a stream of 128-bit randomized numbers based on keystroke input from the user
- Produces a **sequence of session keys** that is effectively unpredictable
Key Identifiers

- How does receiver know which public key to use?
- PGP assigns a key ID to each public key
- It has a high probability of being unique within a user ID – 64-bit
What Does A Transmitted Message Look Like?

- **Message component** – actual data plus filename and timestamp
- **Signature component** – timestamp, message digest, leading two octets of MD (checksum), Key ID of sender’s public key
- **Session key component** – session key plus ID of recipient’s public key used to encrypt the session key
PGP Format

Notation:
- $E_{KUb}$ = encryption with user b's private key
- $E_{KrA}$ = encryption with user a's public key
- $E_{Ks}$ = encryption with session key
- ZIP = Zip compression function
- R64 = Radix-64 conversion function
Recall Public Key Encryption

(a) Encryption

Plaintext input → Encryption algorithm (e.g., RSA) → Transmitted ciphertext → Decryption algorithm (reverse of encryption algorithm) → Plaintext output
Recall Public Key Authentication
Key Rings

- PGP provides a pair of data structures at each node – pub/priv key pairs owned by node & public keys of other users
- Private-Key Ring and Public-Key Ring
- Can view the ring as a table – each row represents one of the pub/priv key pairs
Key Ring Structure

### Private Key Ring

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Key ID*</th>
<th>Public Key</th>
<th>Encrypted Private Key</th>
<th>User ID*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ti</td>
<td>KU_i mod 2^64</td>
<td>KU_i</td>
<td>E_{H(P_i)}[K_{R_i}]</td>
<td>User i</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Public Key Ring

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Key ID*</th>
<th>Public Key</th>
<th>Owner Trust</th>
<th>User ID*</th>
<th>Key Legitimacy</th>
<th>Signature(s)</th>
<th>Signature Trust(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ti</td>
<td>KU_i mod 2^64</td>
<td>KU_i</td>
<td>trust_flag_i</td>
<td>User i</td>
<td>trust_flag_i</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* = field used to index table
PGP Message Generation
PGP Message Reception
Public Key Management

- Physically get the key from $B$
- Verify a key by telephone
- Obtain $B$’s public key from a mutually trusted individual $D$
- Obtain $B$’s public key from a trusted certifying authority
Use of Trust

- Associated with each public key is a **key legitimacy field** – extent that PGP will trust that this is a valid public key
- **Signature trust field** – degree PGP user trusts the signer to certify public keys
- **Owner trust field** – degree to which this public key is trusted to sign other public-key certificates
- Contained in a structure referred to as a **trust flag byte**
## Trust Flag Byte

### Contents

<table>
<thead>
<tr>
<th>OwnerTrust Field</th>
<th>KeyLegit Field</th>
<th>SigTrust Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>undefined trust</td>
<td>unknown or undefined trust</td>
<td>undefined trust</td>
</tr>
<tr>
<td>unknown user</td>
<td>key ownership not trusted</td>
<td>unknown user</td>
</tr>
<tr>
<td>usually not trusted to sign other keys</td>
<td>marginal trust in key ownership</td>
<td>usually not trusted to sign other keys</td>
</tr>
<tr>
<td>usually trusted to sign other keys</td>
<td>complete trust in key ownership</td>
<td>usually trusted to sign other keys</td>
</tr>
<tr>
<td>always trusted to sign other keys</td>
<td>this key is present in secret key ring</td>
<td>always trusted to sign other keys</td>
</tr>
<tr>
<td>this key is present in secret key ring (ultimate trust)</td>
<td>WARNONLY bit —set if user wants only to be warned when key that is not fully validated is used for encryption</td>
<td>this key is present in secret key ring (ultimate trust)</td>
</tr>
</tbody>
</table>

| BUCKSTOP bit —set if this key appears in secret key ring | | CONTIG bit —set if signature leads up a contiguous trusted certification path back to the ultimately trusted keyring owner |

---

Hofstra University – Network Security Course, CSC290A

03/21/06
PGP Trust Model Example

- Unknown signatory
- X signed by Y
- Key's owner trusted by you to sign keys
- Partly trusted to sign keys
- Key is deemed legitimate by you
Revoking Public Keys

- A user may wish to *revoke* his public key
- Reasons: compromise suspected or used too long or lost private key
- Owner issues a *key revocation certificate*, signed by the owner
Important URLs

  Good review of PGP, its history and current status
- http://www.pgp.com/
  New home for PGP – This is the commercial version
- http://www.openpgp.org/
  This is the site for OpenPGP
Important URLs


- [http://www.clairewolfe.com/wolfesblog/00001945.html](http://www.clairewolfe.com/wolfesblog/00001945.html) NPR story about how very few people use encryption, and then gives a tutorial on installing and using GNU Privacy Guard and Enigmail with the Thunderbird email program
Download PGP

  Windows version is: GnuPG 1.2.2
- http://enigmail.mozdev.org/download.html
  Enigmail download
Pathetic Demo Attempt
Generating Keys

- **Type:** `gpg -gen-key`
- You should end up with something like this:
Homework

• Read Chapter Five, Section 1, PGP
• S/MIME will be covered later
• Obtain PGP software and install it
• Try sending me an email (vcosta@optonline.net) and your public key
Reminder: Term Paper

- Due Monday, May 1
- Should be about 6-8 pages (9 or 10 font, single space)
- Suggested template: [http://www.acm.org/sigs/pubs/proceed](http://www.acm.org/sigs/pubs/proceed)
- This should be an opportunity to explore a selected area
- Send me your topic by March 20th
Reminder: Term Paper

Possible topics:
- Elliptic Curve Cryptography
- Cyber Forensics
- Digital Rights Management
- Security In Software Development
- Virtualization & Security
- Legal, Ethical Issues Around Security & Privacy
- Wireless/Mobile Security
- Phishing/Identity Theft
- Distributed DoS Attacks
- Electronic Cash
- Anti-Virus Software
- Any Topic Discussed In Class
- Programming Project Can Be Substituted If You Want