Email

Guevara Noubir
Fundamentals of Computer Networks
Northeastern University

Email

- One of the most widely used applications of the Internet but still relatively insecure
  - Designed without security concerns
- How does email work?
- How to provide important security services
  - Confidentiality, authentication, integrity, etc.
- Spam resiliency

How Email Works

- Architecture
  - Mail User Agent (MUA): client
  - Mail Sending Agent (MSA): server on sender side
  - Mail Transfer Agent (MTA): server on recipient side
  - Mail Delivery Agent (MDA): responsible for deliver to recipient's MUA
- Operation
  - Client submits message MUA <–> MSA
    - `telnet mail.ccs.neu.edu 25`
  - MSA sends message to MTA ; forwarded to recipient MTA
    - Recipient MTA found through DNS system (`dig @8.8.8.8 ccs.neu.edu MX`)
  - MUA retrieves email using POP3 or IMAP protocols

Security Services: Issues & Solutions

- Confidentiality
  - Traffic not encrypted can be redirected and intercepted (See MITM lab)
- Authentication/Integrity
  - Messages can be fabricated, modified, trust in DNS system
- Additional services:
  - Non-repudiation, proof of submission, proof of delivery, anonymity, message flow confidentiality

- Solutions
  - @Transport Layer: SSL/TLS between sender client/local server/destination server/recipient
    - Implications?
  - @Application Layer: end-to-end confidentiality and integrity protection
    - Authentication of sending user vs. authentication of sending mail transfer agent
    - Examples: PGP, S/MIME, DKIM
    - Implications?

End-to-End Confidentiality

- With symmetric keys
- With asymmetric keys (public key cryptography)
- Single destination, multiple destinations, mailing lists

End-to-end Authentication/Integrity

- With symmetric keys
- With asymmetric keys (public key cryptography)

Additional Security Services

- Non-repudiation
  - With asymmetric keys (public key cryptography)
  - With and without plausible deniability
- Proof of submission
  - With cooperation of MSA/MTA (stronger than regular mail service)
- Proof of delivery
  - Requires cooperation of recipient
  - Not possible to provide a receipt if and only if recipient got the message
- Anonymity
  - Mixing? easier solutions today

PEM, S/MIME, PGP

- PEM, S/MIME PGP allow additional security services
- Privacy Enhanced Mail (RFC 1421- 1424)
  - Provides a way to integrate confidentiality, authentication/integrity services within mail system
  - Not used much today because of CA, evolved into S/MIME
- PEM message

```
-----BEGIN PRIVACY-ENHANCED MESSAGE-----
.
.
.
-----END PRIVACY-ENHANCED MESSAGE-----
```

- Types of data
  - ordinary, unsecured
  - integrity-protected, unmodified (MIC-CLEAR)
  - integrity-protected, encoded (MIC-ONLY)
  - encrypted, integrity-protected, encoded (ENCRYPTED)
- Single root certification authority

Secure/Multipurpose Internet Mail Extension

- MIME specifies a standard way of encoding arbitrary data in email (e.g., picture attachments)
  - S/MIME specifies the security related header
  - Incorporated into MIME => no additional encoding
- Any sequence of sign & encrypt is supported
  - Each as a recursive MIME encapsulation
- Has more options than PEM
- ASN.1 header encoding
- No prescribed certification hierarchy
- Has a good prospect of deployment for commercial & organizational usage

Pretty Good Privacy (PGP)

- Similar to S/MIME
  - with a more complex history
- Major difference: web of trust graph
  - Partial trust, multiple paths
- Issues
  - In theory would be safer than PEM
  - Difficult to operate in practice

Spam

- For years spam has been a major problem of email
  - Estimated to be 94% of emails
  - From a nuisance to a threat
- How?
  - Harvesting/buying addresses
  - Sending through open relays, proxies, creating webmail accounts (circumventing CAPTCHAs), malware, spambots, hijacking IP blocks
- Why? Spam economics
  - Even with a currently estimated conversion rate of 10^{-7} still interesting

Anti-Spam

- Current solutions:
  - Black/white listing IP addresses (e.g., zombie computers, addresses that sent spam to honeypots, ISP willingly hosting spammers)
  - Signatures/content matching rules
  - HashCash: add header

    X-Hashcash: 1:20:101130:noubir@ccs.neu.edu::HdG5s/(oiuU7Ht7b:ePa
  - Distributed Checksum Clearinghouse: message fuzzy checksum is sent to DCC to check how many times it appeared
  - Sender Policy Framework: specify who can send email from a domain (relies on TXT/SPF DNS record)

    dig @8.8.8.8 neu.edu ANY
  - Example of software combining these techniques: spamassassin

Sending MTA Authentication

- DomainKeys Identified Mail (DKIM RFC 4871, 2007 – RFC 6376, 2011)
  - DomainKeys initiated by Yahoo!, today a IETF standard DKIM
- The sending MTA adds a signature to the message
  - MIME header
  - Public key can be retrieved through DNS system

    dig @8.8.8.8 s1024._domainkey.yahoo.com any

    dig @8.8.8.8 gamma._domainkey.gmail.com any
- Example:

```
DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/relaxed;
        d=gmail.com; s=gamma;
        h=domainkey-signature:mime-version:received:received:date:message-id
         :subject:from:to:content-type;
        bh=cvC34ODyPB/uEHubbDQQmwxZfqZboGjW5gpY4W6DuzE=;
        b=ASsElEtXCmM/x3aL38Efnvi9xDrBdleaaBqd24f7XS49pRzhXK/7Vak9+LyLLcN89e
         GZ7SZi7swY2xIlt3zJTiGrGif0bfQdf7LvlP12g53nczhBBRa8McBVtdK9+ImAZByg8o
         oEM4INNjMvdhXi9MVXtntkvmsTmWitAJxZgQQ=
DomainKey-Signature: a=rsa-sha1; c=nofws;
        d=gmail.com; s=gamma;
        h=mime-version:date:message-id:subject:from:to:content-type;
        b=JFWiE0YlmWxu+Sq4OJ9Ef5k3rjbZQ51dGEyaFyvKJYR8NkoGrNoPIUq5f29ld8P0AD
         Lg058evTVeuWxvfPQfa7K65J9AjEQt5U8d9zBKFfxRAz1h5nr7k2kCLRMnhbqVTkiOIS
         OUfxIQeMfgbYz0ydCgerEnfGreKMQIYax+dpo=
```

Summary

- Email application one of the most widely used applications
  - was designed, and deployed without security in mind
- Several security services have been proposed with varying levels of acceptance
  - Transport layer security
  - Application level security
    - End-to-end, MTA authentication, etc.
- More secure today but still vulnerable

  e.g., DNS poisoning