CSC429 – Computer Security

LECTURE 12
NETWORK SECURITY

Mohammed H. Almeshekah, PhD
meshekah@ksu.edu.sa

Network Security

Application Level

Domain Name System

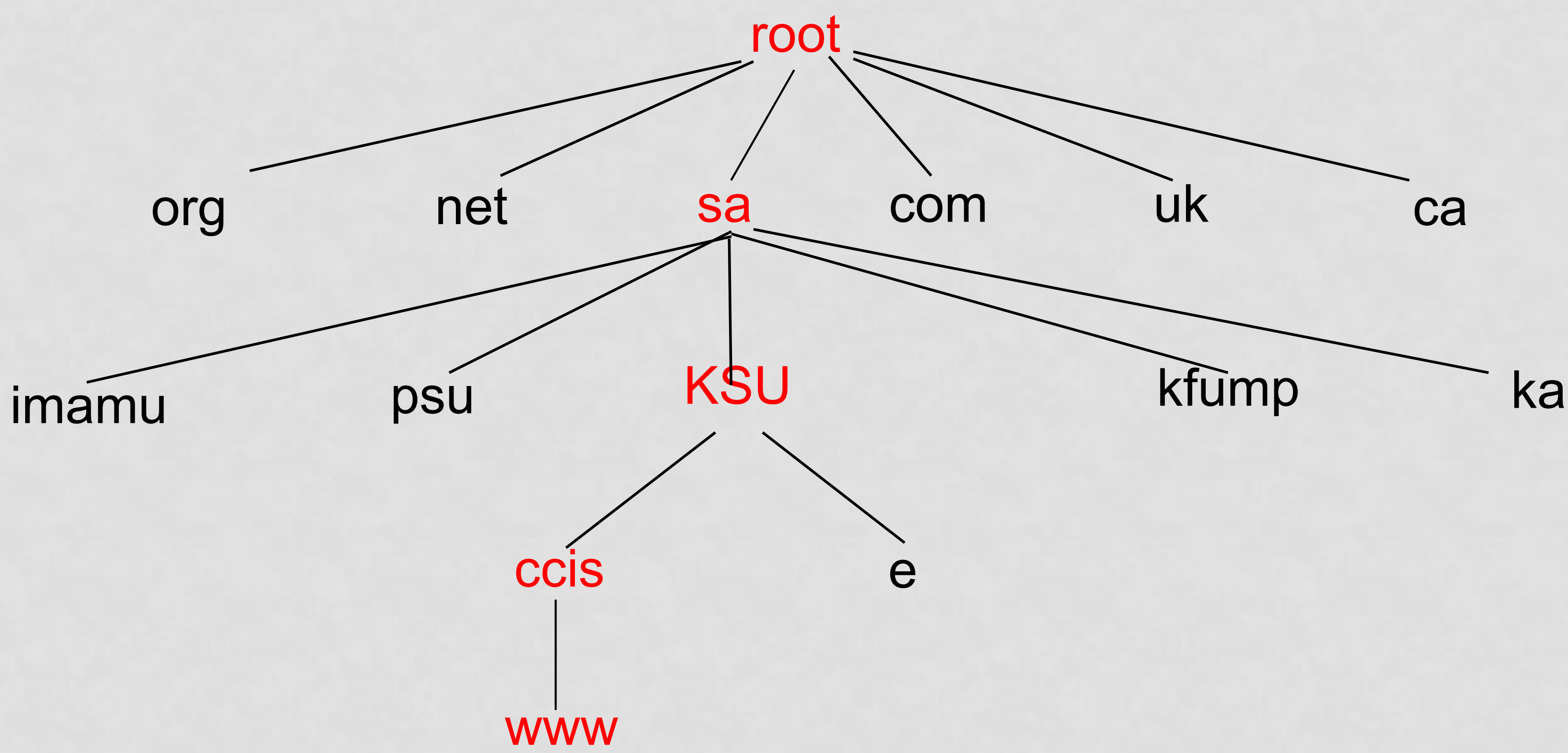
- Translate host names to IP addresses
  - E.g., www.xyz.com ➔ 74.125.91.103
  - Why is needed?
    - E.g. akami.
- And back
  - From IP addresses to DNS name

DNS is a Distributed Database

- Information is stored in a distributed way
- Highly dynamic
- Decentralized authority

Domain Name System

- Hierarchical Name Space



Domain Name System

Domain Name Servers

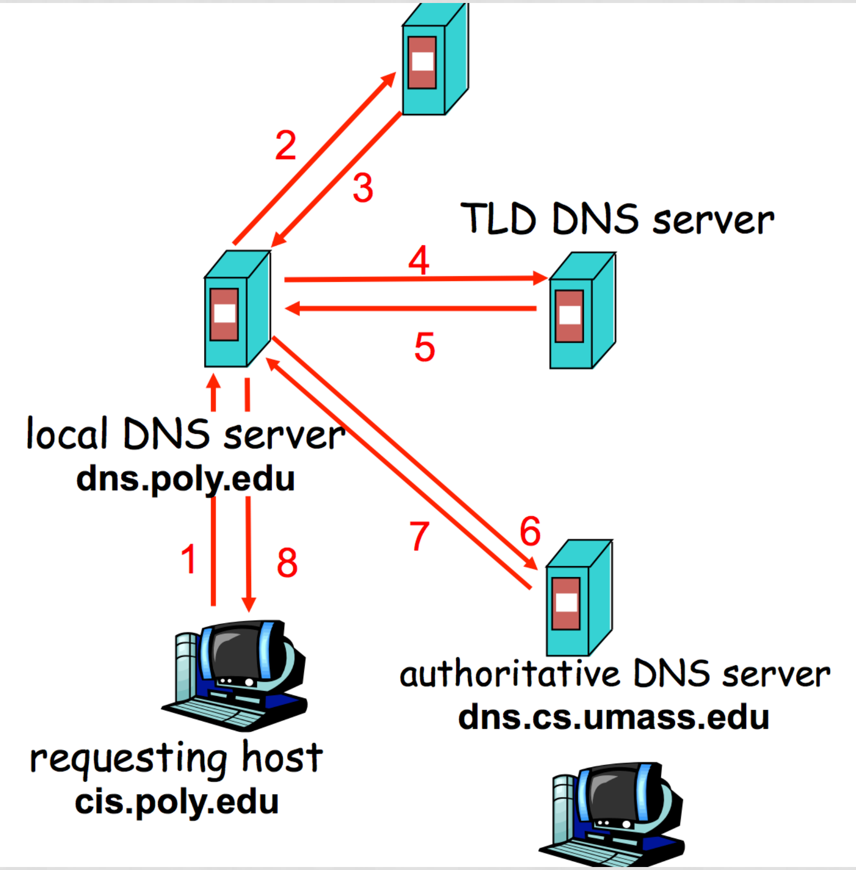
- Top-level domain (TLD) servers:
  - responsible for com, org, net, edu, etc, and all top-level country domains, e.g. uk, fr, ca, jp.
  - Network Solutions maintains servers for ".com"
- Authoritative DNS servers:
  - organization's DNS servers, providing authoritative hostname to IP mappings for organization's servers.
  - can be maintained by organization or service provider.
- Local Name Server
  - does not strictly belong to hierarchy
  - each organization (company) has one.

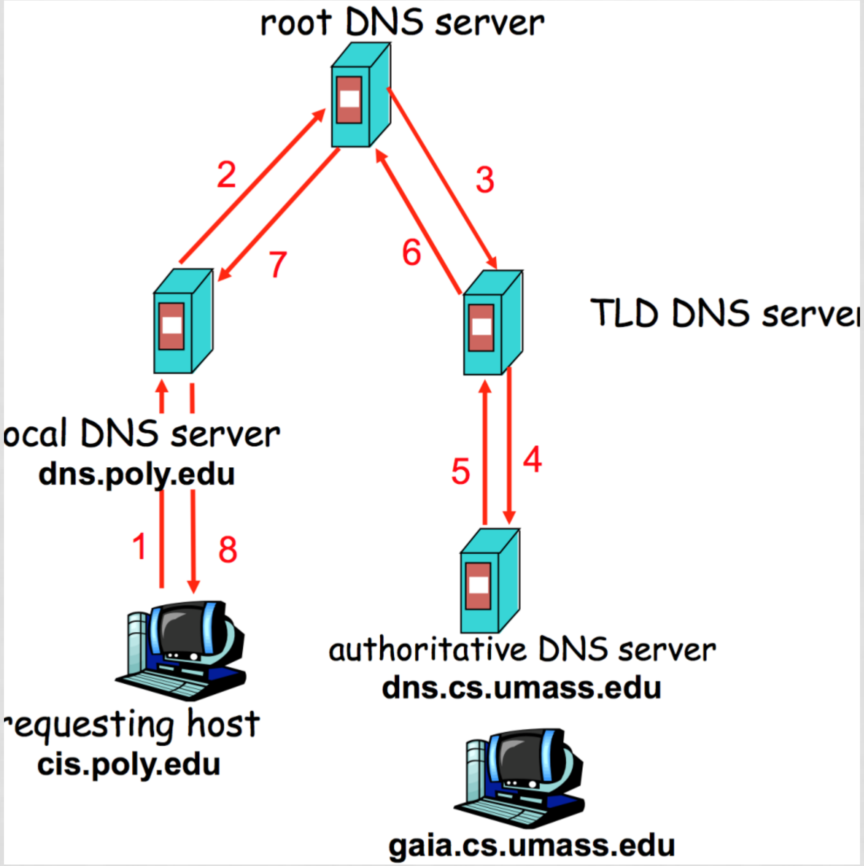
DNS Resolving

- When host makes DNS query, query is sent to its local DNS server.
  - acts as proxy, forwards query into hierarchy.

- Two resolving schemes:
  - Iterative, and
  - Recursive.

DNS Resolving

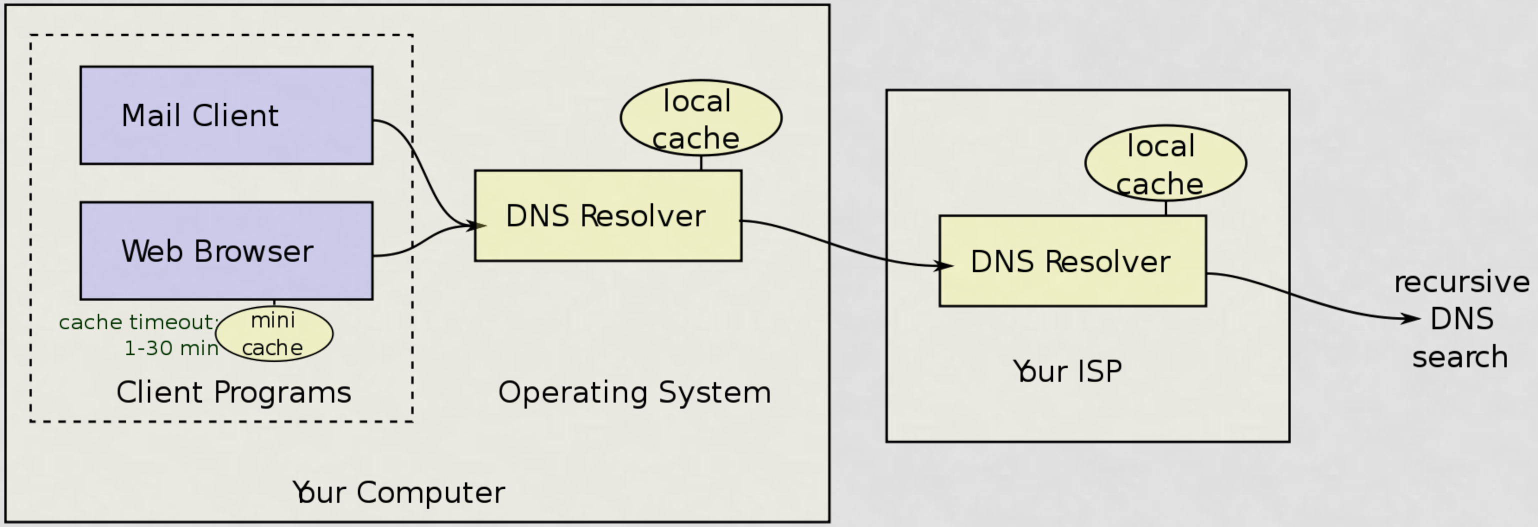




Caching

- DNS responses are cached
  - Quick response for repeated translations
- Negative results are also cached
  - Save time for nonexistent sites, e.g. misspelling
- Cached data periodically times out
  - Each record has a TTL field

Caching


Mail Client
Web Browser
cache timeout:
1-30 min
mini
cache
Client Programs
local
cache
DNS Resolver
Operating System
Your Computer
local
cache
DNS Resolver
Your ISP
recursive
DNS
search

Inherent DNS Vulnerabilities

- Users/hosts typically trust the host-address mapping provided by DNS
  - What bad things can happen with wrong DNS info?
- DNS resolvers trust responses received after sending out queries.
- Obvious problem
  - No authentication for DNS responses

User Side Attack - Pharming

- DNS poisoning attack
  - Change IP addresses to redirect URLs to fraudulent sites
  - Potentially more dangerous than phishing attacks
    - Why?

- DNS poisoning attacks have occurred:
  - January 2005, the domain name for a large New York ISP, Panix, was hijacked to a site in Australia.
  - In November 2004, Google and Amazon users were sent to Med Network Inc., an online pharmacy

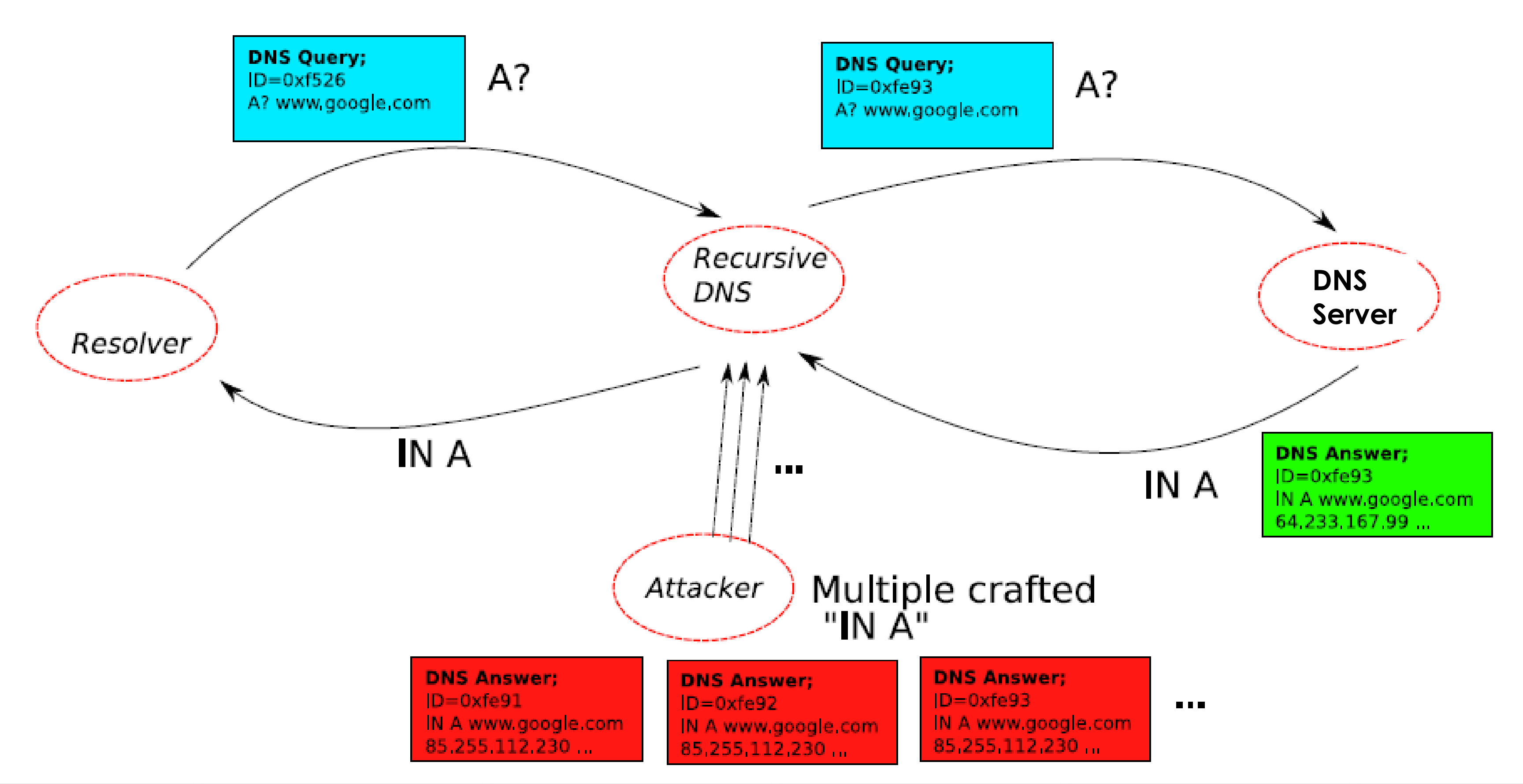
DNS Cache Poisoning

- Attacker wants his IP address returned for a DNS query
- When the resolver asks ns1.google.com for www.google.com, the attacker could reply first, with his own IP
- What is supposed to prevent this?
- Transaction ID
  - 16-bit random number
  - The real server knows the number, because it was contained in the query
  - The attacker has to guess

DNS Cache Poisoning

- An attacker can guess when a DNS cache entry times out and a query has been sent, and provide a fake response.
  - Responding before the real nameserver
- The fake response will be accepted only when its 16-bit transaction ID matches the query
- Fixed by using random transaction IDs

DNS Cache Poisoning – Racing to Respond First



DNS Short-Term Defenses

- Difficulty to change the protocol
  - Protocol stability (embedded devices)
  - Backward compatibility.

- Short-term
  - Only change the recursive server (local DNS).
  - Easy to adopt

Short-Term Defenses

- Source port randomization
  - Add up to 16 bits of entropy
  - NAT could de-randomize the port
- DNS 0x20 encoding
  - From Georgia tech, CCS 2008
- Tighter logic for accepting responses

DNS-0x20 Bit Encoding

- DNS labels are case insensitive
- Matching and resolution is entirely case insensitive
- A resolver can query in any case pattern
  - E.g., WwW.ExAmpLe.cOM
  - It will get the answer for www.example.com

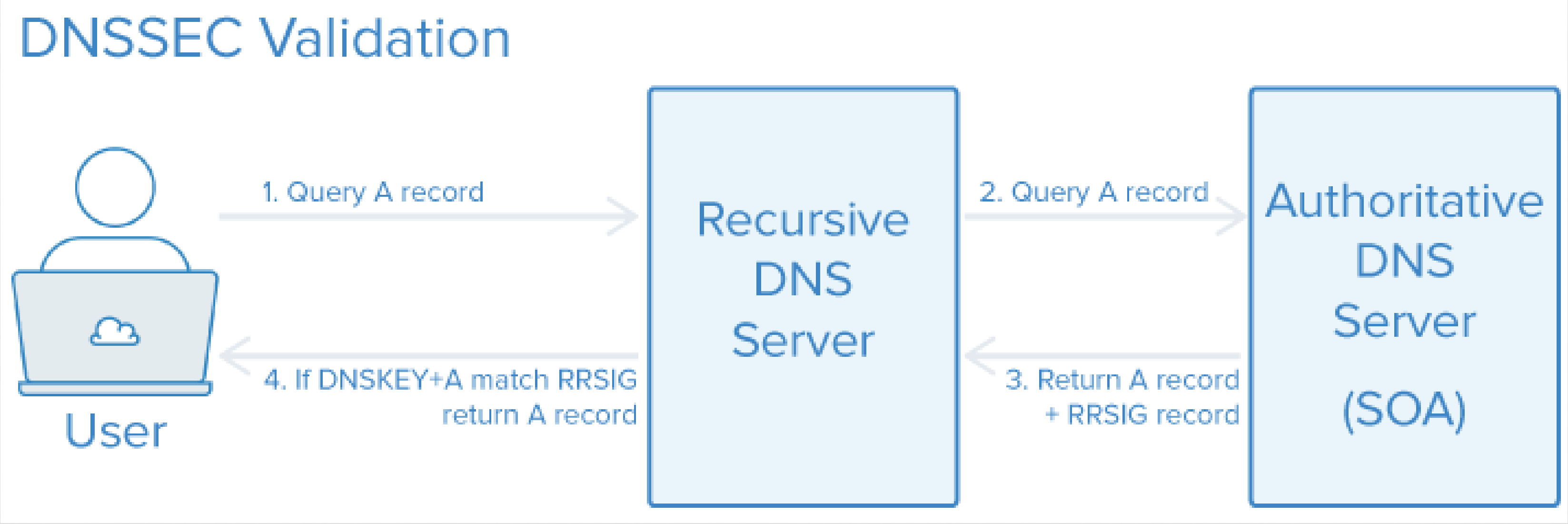
DNS-0x20 DNS Encoding

- A DNS response contains the query being asked
- When generating the response, the query is copied from the request exactly into the response
  - The case pattern of the query is preserved in the response
- The mixed pattern of upper and lower case letters constitutes a channel, which can be used to improve DNS security
  - Only the real server knows the correct pattern
  - Adds randomization.

DNS-0x20 Encoding Analysis

- Not every character is 0x20 capable
- Improve the forgery resistance of DNS messages only in proportion to the number of upper or lower case characters
  - cia.gov — 6-bit entropy
  - licensing.disney.com — 18-bit entropy
  - 163.com — 3-bit entropy

DNSSEC


DNSSEC Validation
User
1. Query A record
Recursive DNS Server
2. Query A record
Authoritative DNS Server
(SOA)
3. Return A record + RRSIG record
4. If DNSKEY+A match RRSIG return A record

DNS Long-Term Defenses

- DNSSEC:
  - Cryptographic protections
    - Authenticate responses.
  - A multi-year process
  - Google DNS now is enabled by default.

- Challenges in deployment:
  - Response is large, might no longer fit in single UDP message.
  - Legacy software and machines.

Network Security

Security Toolbox

Cryptographic Network Protection

- Solutions above the transport layer
  - Examples: SSL and SSH
  - Protect against session hijacking and injected data
  - Do not protect against denial-of-service attacks caused by spoofed packets

- Solutions at network layer
  - E.g. IPsec
  - Can protect against
    - session hijacking and injection of data.
    - denial-of-service attacks using session resets.

The “Secure Channel” Concept

- We achieve this by building a “secure channel” between two end points on an insecure network.

- Typically this channel will offer:
  - Data origin authentication
  - Data integrity
  - Confidentiality

- But usually not:
  - Non-repudiation
  - Any security services once data has been received

Typical Cryptographic Primitives Used

- Symmetric encryption algorithms.
  - Almost universally used for performance reasons.
- MAC algorithms
  - Usually built from hash functions, block ciphers, or possibly a dedicated design.
- Key-Exchange Algorithms, e.g. Diffie-Hellman
- Hash Functions:
  - E.g. For key derivation.

Other Common Techniques Used

- Sequence numbers are widely used to prevent replay attacks and ensure correct data ordering.
  - These need to be cryptographically protected.

- Nonces and timestamps are used to provide freshness in entity authentication exchanges.

Where to Place Security in the Network?

- Data Link (Network Interface) layer:
  - ✓ Covers all traffic on that link, independent of protocols above.
    - ✓ e.g. link level encryptor.
    - ✓ Cannot be compromised even if communicating hosts are.
  - ✓ Typically runs at line-speed of link.
  - ✘ Protection only for one "hop" (point-to-point).
    - ✘ Doesn't scale well.
  - ✘ Usually implemented using moderate to high cost special-purpose hardware.

Where to Place Security in the Network?

- Network (Internet) layer:
  - ✓ Covers all traffic carried by IP.
  - ✓ Can be end-to-end.
  - ✓ Transparent to applications.
  - ✓ Cost of authentication/key exchange protocols can be amortized over many applications.
  - ✘ Little application control over security that gets applied.
    - ✘ Application has no visibility of Internet layer.
  - ✘ May be unnatural place to apply security.
    - ✘ Network layer is stateless and unreliable.
      - ✘ Detecting and preventing replays therefore technically impossible, without maintaining extra state or via a layer violation.
      - ✘ Order of data in secure channel may be crucial; difficult to maintain if IP datagrams are dropped, re-ordered,…

Where to Place Security in the Network?

- Transport layer:
  - ✓ End-to-end protocol.
  - ✓ Covers all traffic using the protected transport protocol.
  - ✓ Applications can control when it's used.
    - ✓ Application can choose to select secure transport layer or not.
  - ✓ Transport layer may be naturally stateful (TCP).
    - ✓ Makes provision of some security services easier.
  - ✘ Each application must be modified or proxied to take advantage of the security provided by secure transport layer option.
  - ✘ May be protocol-specific.
    - ✘ E.g. SSL/TLS only implemented over TCP, not UDP.

Where to Place Security in the Network?

- Application layer:
  - ✓ Security can be tuned to application requirements.
    - ✓ Different applications may have different needs.
      - e.g. VoIP applications vs. sensitive data transfer.
  - ✓ Easy access to user credentials (e.g. private keys).
  - ✓ Possible to provide non-repudiation services at application level.
    - ✓ May not make sense at lower layers.
  - ✘ But no leveraging effect.
    - ✘ Every application must handle its own security.
    - ✘ Plenty of room for errors, redundancy, and security holes.