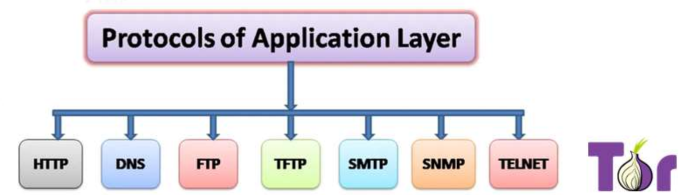
Lecture 15: Application Layer Security

Stephen Huang

Content

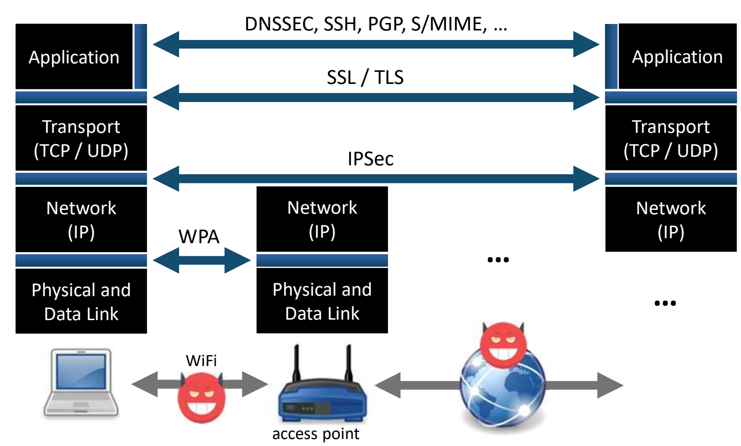
1. Application Layer Protocol
2. Domain Name System (DNS)
3. DNS Secure Extensions (DNSSEC)
4. Anonymous Communication: Tor



1. Application Layer Protocol

- Application layer protocols directly support the features of network applications like web browsers and e-mail readers.
- Application layer protocols define the language that network applications speak to fulfill user requests.
  - They define the format of any data exchanged.
  - They specify how clients and servers should react.

Protocol Stack in Practice



Application Layer

- It is the closest layer to the end user.
- It provides hackers with the largest threat surface.
- Application layer security attacks:
  - Distributed Denial-Of-Service (DDoS) attacks
  - HTTP flooding
  - SQL injections
  - Cross-site scripting

Types of Application Layer Protocol

- Remote login to hosts: telnet,
- File transfer: File Transfer Protocol (FTP), Trivial FTP,
- Electronic mail: Simple Mail Transfer Protocol (SMTP),
- Network Support: Domain Name System (DNS),

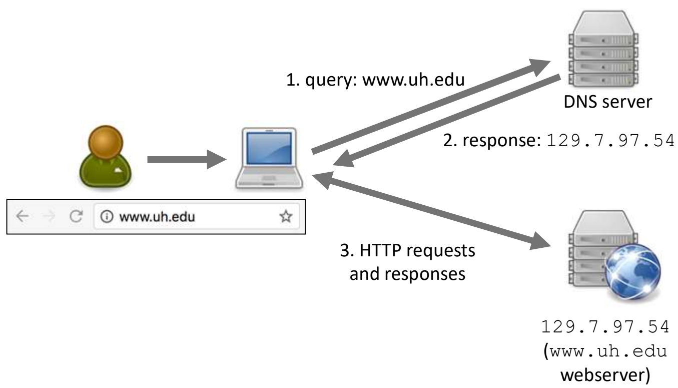
Security-Enhanced ALP

- Secure Shell (SSH), a secure replacement for telnet and ftp.
- DNS Security, or DNSSEC, refers to a set of security extensions and enhancements for DNS.
- Cryptographic file systems have been developed, e.g., the Cryptographic File System (CFS) and the Andrew File System (AFS).
- PGP
- S/MIME

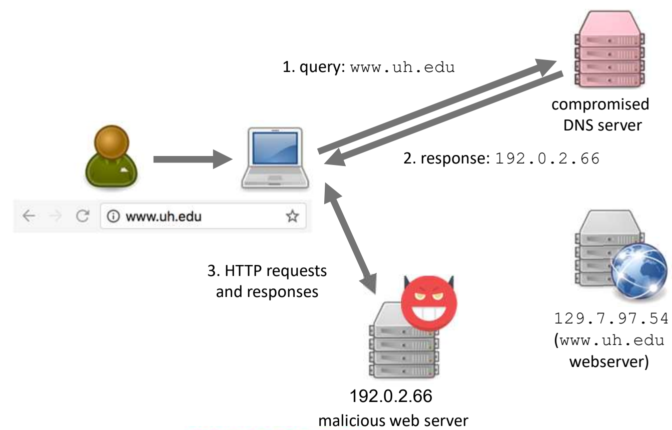
2. Domain Name System

- A Domain Name System (DNS) turns domain names (e.g., `www.uh.edu`) into IP addresses (e.g., `129.7.97.54`), which allow browsers to get to websites and other internet resources.
- Every device on the internet has an IP address, which other devices can use to locate the device.
- Instead of memorizing a long list of IP addreswebsite's nameses, people can enter the website's name and the DNS gets the IP address for them.

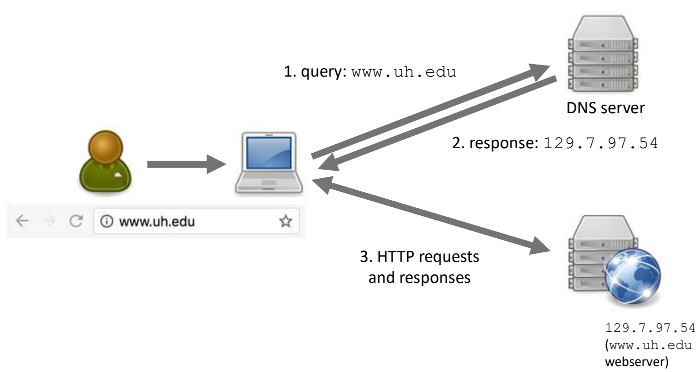
Domain Name Resolution



Domain Name Resolution

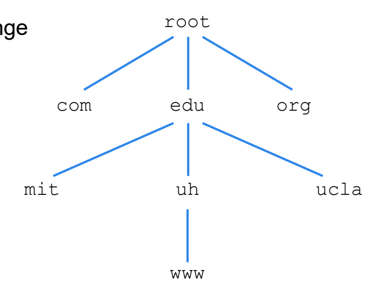


Domain Name Resolution

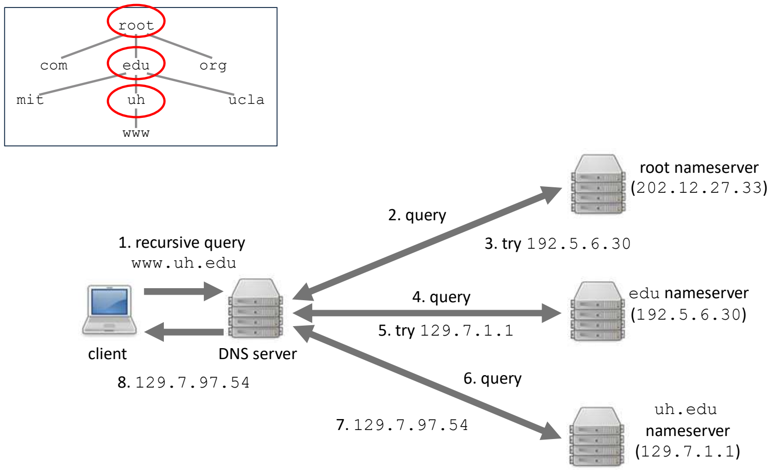


Domain Name System (DNS)

- Millions of domain names ⟶ distributed database
  - dynamic data
    ⟵ IP address for a host may change
  - decentralized authority
    ⟵ each name has an owner
- Hierarchical name space
  - domain: node in the DNS tree
  - for each domain, there is an authoritative server
- Authoritative server
  - responds to queries about the domains for which it is responsible
  - may refer to other authoritative servers for a **subdomain**



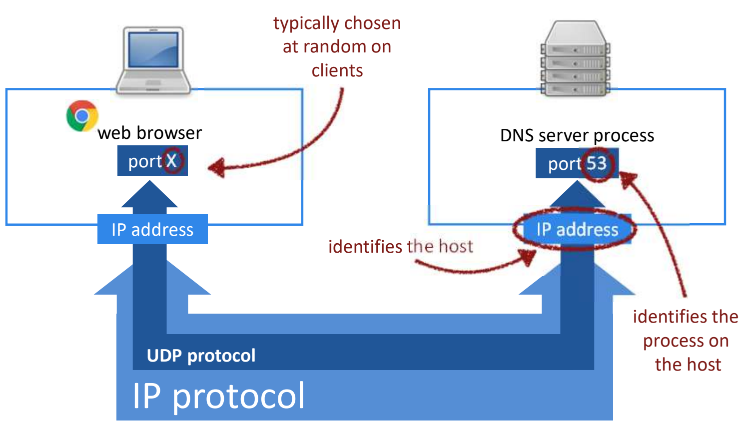
DNS Recursive Resolution: Example



DNS Queries and Responses

- Transport protocol
  - UDP port 53
  - TCP for long responses (and some tasks between nameservers)

Reminder: Network Ports



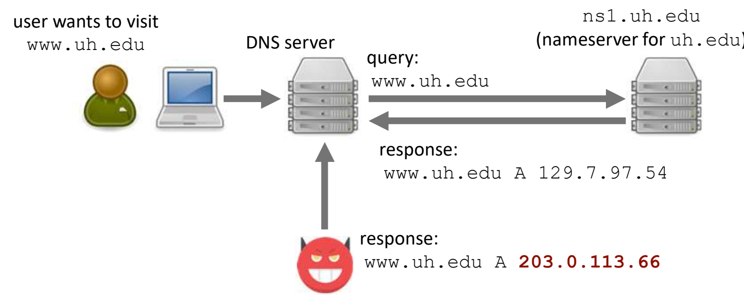
DNS Queries and Responses

- Transport protocol
  - UDP port 53
  - TCP for long responses (and some tasks between nameservers)
- Messages: <u>query</u> and <u>reply</u>
  - 16-bit identification field: match queries with replies
  - *example reply from* `edu` *nameserver:*
    `uh.edu NS ns1.uh.edu` ← NS: Name Server
    `ns1.uh.edu A 129.7.1.1` ← A: Address
  - *example reply from* `uh.edu` *nameserver:*
    `www.uh.edu A 129.7.97.54`
- Caching
  - received responses are cached by servers
  - each record has a Time-to-Live field, after expiry it must be queried again

DNS Weaknesses

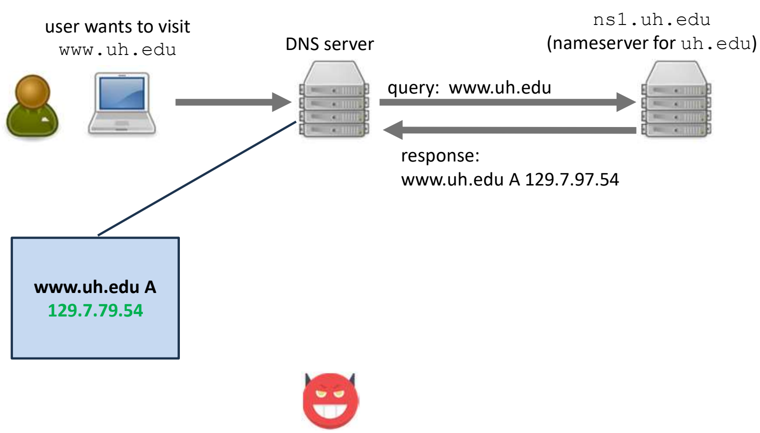
- DNS responses are not authenticated
  - responses can be sent over UDP transport protocol
    → anyone can respond from a spoofed (i.e., fake) IP address to a query
- DNS is a key infrastructure
  - resolvers trust responses, users trust resolvers
  - by tampering with responses, an attacker may direct users to malicious websites or direct e-mail to malicious servers
- DNS cache poisoning
  - attacker sends malicious response to a DNS server, which caches it
    → malicious response is served to all clients using the server
  - attacker does not have to be man-in-the-middle

Race to Respond First

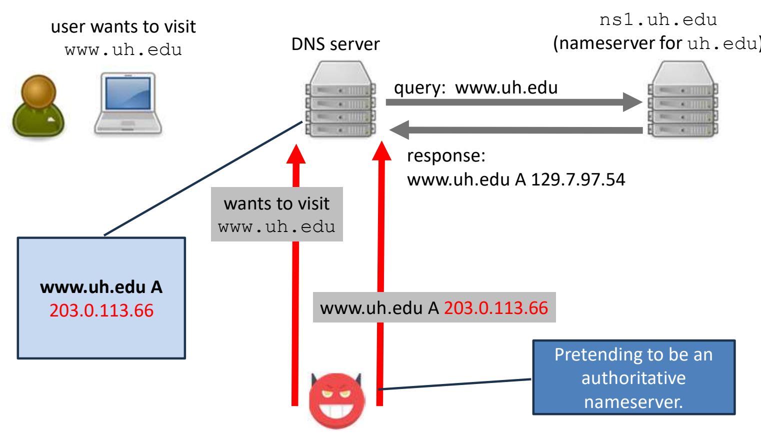


- If the attacker responds before the authoritative server, the DNS cache is poisoned with the malicious IP address
  - fake response is a single UDP packet from the spoofed (i.e., forged) IP address of the authoritative server

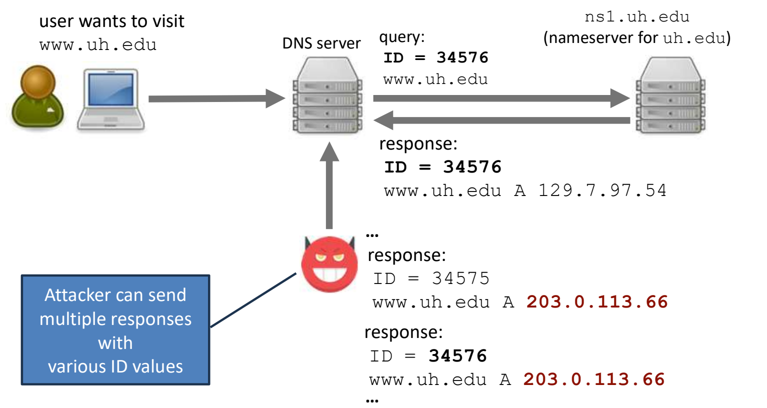
Race to Respond First



Race to Respond First



Race to Respond with Transaction ID

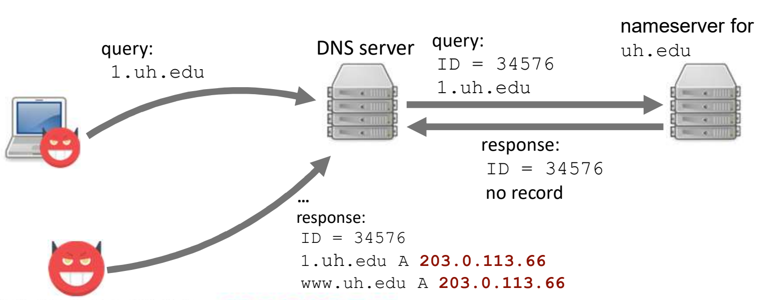


Practical Aspects of Race to Respond First

- Transaction ID
  - used to match responses to queries
  - 16-bit value → non-negligible probability of guessing
- Attack using multiple queries
  - when a cache entry expires, attacker has many clients send the same DNS request to the server (at the same time)
    → many queries sent to the nameserver with various ID values
  - attacker sends many replies with various ID values
  - birthday paradox
    → with high probability, one of the attacker's replies matches one of the queries
- Attacker can guess the transaction ID (or flood the server with responses), but it also needs to know when a cache entry times out

Kaminsky Attack

- In the previous approach, the attacker must wait for the cached record to expire
- Vulnerability discovered in 2008 by Dan Kaminsky
  – attacker does not need to wait for cache expiry

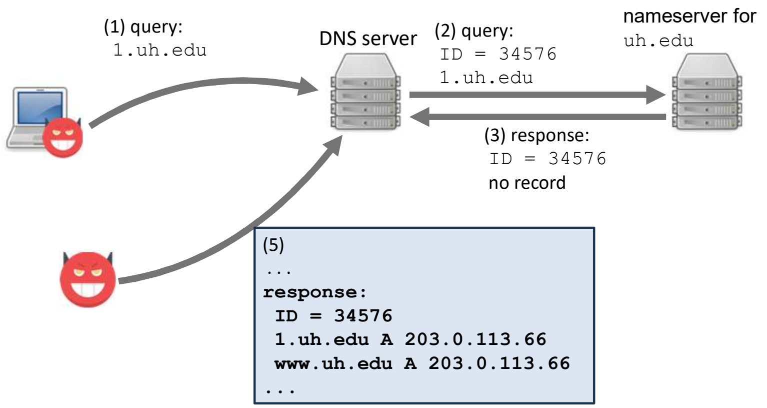


Kaminsky Attack (contd.)

- Attacker sends many queries to the DNS server for
  `1.uh.edu`
  `2.uh.edu`
  …
- DNS server sends out a query for each of these domains to the authoritative server
  – since these are not actual domains, they cannot be in the cache
    → no need to wait for cache expiry
  – for each query, attacker can send many poisoning responses
    → high probability of success
- Mitigation: randomize source UDP port on the DNS server
  – attacker needs to guess both the ID and the port
    → much lower probability of success (but not zero)

  or use DNSSEC, DoT, DoH

Kaminsky Attack



3. DNS Security Extensions

- Set of extensions defined by the IETF to the DNS protocol.
- Guarantees origin authenticity and data integrity of DNS replies.
- Backwards compatible: responses can be interpreted by DNS servers and clients that do not support DNSSEC.
  – of course, no guarantees are provided for these servers and clients
- Does not provide confidentiality.
  – responses are only authenticated but not encrypted.
- Based on public-key cryptography
  – every response is digitally signed

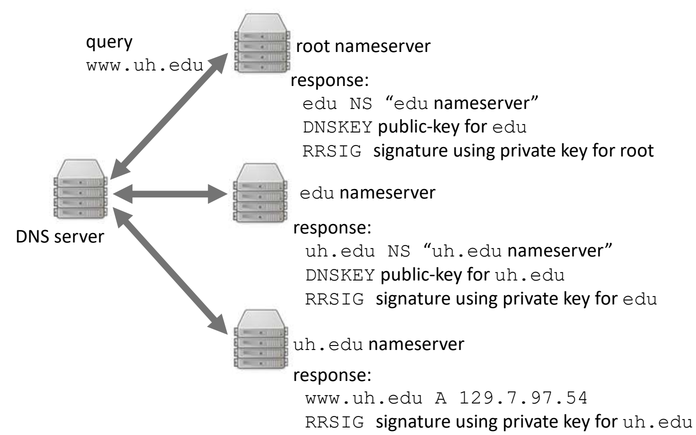
DNSSEC Public Keys and Signatures

- Signature algorithms: RSA-SHA1, RSA-SHA256, ECDSA-SHA256, …
- <u>Trust anchor</u>
  – known public key for an authoritative nameserver
  – typically included in the operating systems
- Authentication chain

trust anchor → `root` $\xrightarrow{\text{signs}}$ `edu` $\xrightarrow{\text{signs}}$ `uh.edu`

- Responses may include
  – RRSIG (Resource Record Signature): digital signature for the contents of the response
  – DNSKEY: public key for a zone
  ← if the response delegates

DNSSEC Resolution Example



DNS over HTTPS and TLS

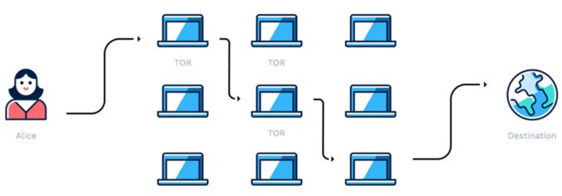
- DNS over HTTPS (DoH): proposed IETF standard RFC 8484 (2018)
  - DNS queries and responses are encoded into HTTPS requests and responses
    - provides integrity and confidentiality
    - server may use HTTP/2 Push to send records in advance
  - adopted recently by multiple web browsers
    - Google Chrome: supported since 2019
    - Mozilla Firefox: default since February 2020, relying on Cloudfare resolvers
  - controversies and criticism: can impede traffic analysis for cybersecurity, may provide a false sense of privacy, can impede traffic filtering by ISPs, …
- DNS over TLS (DoT): proposed IETF standard RFC 7858 (2016)
  - provides integrity and confidentiality
  - adoption: enabled by default on Android 9 (and more recent versions)

Mozilla Firefox and DNS over HTTPS

- 2018: Mozilla partnered with Cloudfare to provide DoH for Firefox users who enable it.
- June 2019: U.K. Internet Service Providers Association nominated Mozilla for its “Internet Villain” award.
  - withdraw nomination in July after community backlash
- February 2020: Mozilla enabled DoH by default for US-based users, relying on Cloudfare resolvers.
  - NextDNS resolvers were also enabled by default for some time, but removed because NextDNS could not handle the volume of traffic.
- June 2020: Mozilla announced that Comcast resolvers would be added to the list of trusted DoH resolvers (enabled by default for Comcast Xfinity users in the US.

4. Anonymous Communication: Tor

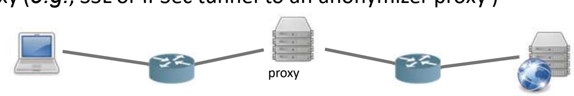
- The Onion Router (Tor)



Anonymous Communication: The Challenge

- Goal: conceal the identities of communication parties
- Simple encrypted connection (*e.g.*, SSL to other endpoint)

  - every single node on the route knows the two endpoints
- Proxy (*e.g.*, SSL or IPSec tunnel to an anonymizer proxy)

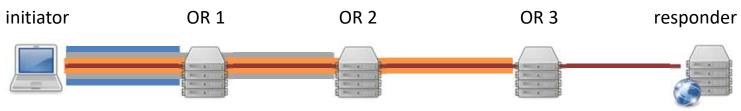


  - only the proxy knows the two endpoints
  - proxy knows the two endpoints

Can the proxy be trusted?

Onion Routing

- Anonymous communication over a computer network
  - packets are encapsulated in multiple layers of encryption
  - concept developed in the mid 1990s at the US Naval Research Laboratory and further developed by DARPA



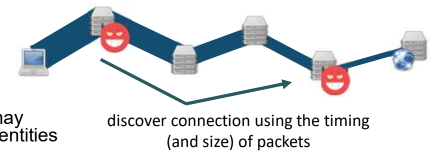
  - select a set of onion routers, and form a “virtual circuit” (i.e., path to the responder)
  - “onion”: packet wrapped in multiple layers of encryption

$$E\left(PU_{OR_1}, OR_2 \mid E\left(PU_{OR_2}, OR_3 \mid E(PU_{OR_3}, \text{responder} \mid \text{data})\right)\right)$$

  - each onion router sees only the preceding and following nodes

Onion Routing in Practice

- Onion routers
  - may be malicious (i.e., trying to infer identities)
  - colluding malicious routers may use traffic analysis to infer identities



- Router directory
  - public list of available onion routers and their public keys
  - initiator needs to select a large and diverse set of routers to prevent analysis
- Virtual circuit setup
  - initiator establishes a connection and a symmetric session key with the first onion router → two-way tunnel
  - tunnel is used to establish a connection and session key with the second router, and so on, until the responder is reached

Tor: The Onion Router

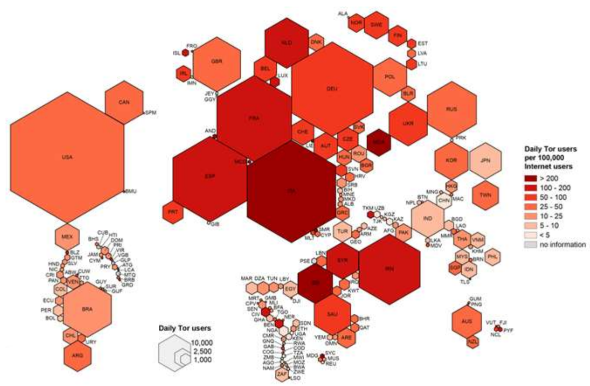
- Tor (The Onion Router)
  - free software implementation of the onion routing concept
  - directs traffic through a worldwide, free, volunteer network of onion routers
  - originally developed at the US Naval Research Laboratory, published in 2002
  - currently maintained by the Tor Project, a non-profit organization
- Trusted Directory Servers
  - public keys and addresses of these servers are hardcoded into the Tor software
  - maintain a list of active onion routers (with their addresses and public keys)
- Onion routers
  - every OR is connected to every other OR through SSL/TLS
  - anyone can install Tor and run a router (but may choose not to be an exit router)

Circuits for Initiator Anonymity

- Goal: user wants to connect to a responder without revealing its identity to the responder or anyone else
- Tor client
  - randomly selects a set of ORs for the circuit (default is three ORs)
  - connects to first OR, establishes AES session key using D-H key exchange
  - uses encrypted tunnel to connect to the next OR, etc. → virtual circuit to exit OR
  - builds new circuit periodically (e.g., every few minutes)
  - provides a SOCKS (Socket Secure) proxy on the client
- A virtual circuit can multiplex and route traffic from multiple SOCKS-aware applications
  - traffic can include HTTP, IRC, etc.
  - Tor Browser: web browser with an automatic Tor client

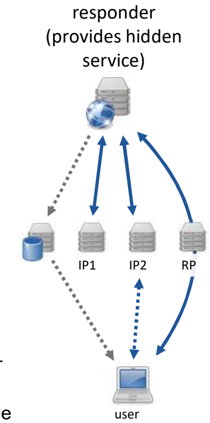
Tor Statistics

- More than 6,000 active ORs (more than 1,000 can be used as exits)
- Estimated number of directly connected clients is over 2 million



Responder Anonymity: Location Hidden Services

- Hidden services are identified by public keys
  - pseudo-domain name: `publickey.onion`
- Responder (i.e., server) selects some ORs to be Introduction Points
  - first, builds circuits to these ORs
  - second, publishes the public key and the list of Introduction Points (signed using its private key in a distributed hash table stored by the ORs)
- User may contact a responder using its public key
  - first, retrieves the Introduction Points from the distributed hash table
  - then, selects a Rendezvous Point and tells the responder about it through one of the Introduction Points
  - finally, the user and responder establish circuit through the Rendezvous Point



Tor Usage

- Tor can be used for licit purposes, such as protecting one's privacy
- However, it is often used for illegal purposes
  - *example:* Silk Road
    - online black market operated as a Tor hidden service
    - launched in February 2011, shut down by the FBI in October 2013
    - transactions were conducted with Bitcoin
- 2015 study by researchers from King's College London
  - hidden services account for around 3 - 6% of overall Tor traffic
  - found 2,723 active websites available as hidden services
  - **57%** of these were **illicit** (423 related to trading illegal drugs, 327 related to money laundering or trading stolen credit cards or accounts, …)

Blocking and De-Anonymization of Tor

- Blocking Tor
  - list of ORs is publicly available → filtering is relatively easy
  - *example:* Wikipedia uses the TorBlock MediaWiki extension to block Tor exit ORs from editing
  - bridges: ORs that are not publicly advertised
    - each client (i.e., IP address) may receive only a very small subset of them
    - can be used only as an entry OR
- De-Anonymization
  - user fingerprinting: users may be identified (e.g., unique mouse movements)
  - some protocols or applications may accidentally leak IP addresses (or other IDs)
  - traffic analysis: Tor does not do traffic-shaping or mixing
- Not steganography: Tor does not hide the existence of communication

Next Topic

- Application Layer Protocol
- Email Security