Last time

- Internet Application Security and Privacy
  - Link-layer security: WEP, WPA, WPA2
  - Network-layer security: VPN, IPSec

This time

- Internet Application Security and Privacy
  - Transport-layer security and privacy: TLS / SSL, Tor
    - The Nymity Slider
  - Application-layer security and privacy: ssh

Transport-layer security and privacy

- Network-layer security mechanisms arrange to send individual IP packets securely from one network to another
- Transport-layer security mechanisms transform arbitrary TCP connections to add security
  - And similarly for “privacy” instead of “security”
- The main transport-layer security mechanism:
  - TLS (formerly known as SSL)
- The main transport-layer privacy mechanism:
  - Tor

TLS / SSL

- In the mid-1990s, Netscape invented a protocol called Secure Sockets Later (SSL) meant for protecting HTTP (web) connections
  - The protocol, however, was general, and could be used to protect any TCP-based connection
  - HTTP + SSL = HTTPS
- Historical note: there was a competing protocol called S-HTTP. But Netscape and Microsoft both chose HTTPS, so that's the protocol everyone else followed
- SSL went through a few revisions, and was eventually standardized into the protocol known as TLS (Transport Layer Security, imaginatively enough)

TLS at a high level

- Client connects to server, indicates it wants to speak TLS, and which ciphersuites it knows
- Server sends its certificate to client, which contains:
  - Its host name
  - Its public key
  - Some other administrative information
  - A signature from a Certificate Authority
- Server also chooses which ciphersuite to use
- Client sends symmetric encryption key *K*, encrypted with server's public key
- Communication now proceeds using *K* and the chosen ciphersuite

The success of TLS

- TLS (including SSL) is the single most successful Privacy Enhancing Technology (PET) ever
- Why?
  - It comes with your computer
    - Which encouraged web server operators to bother paying $$ for their certificates
  - It just works, without you having to configure anything
  - Most of the time, it even protects the privacy of your communications

Privacy Enhancing Technologies

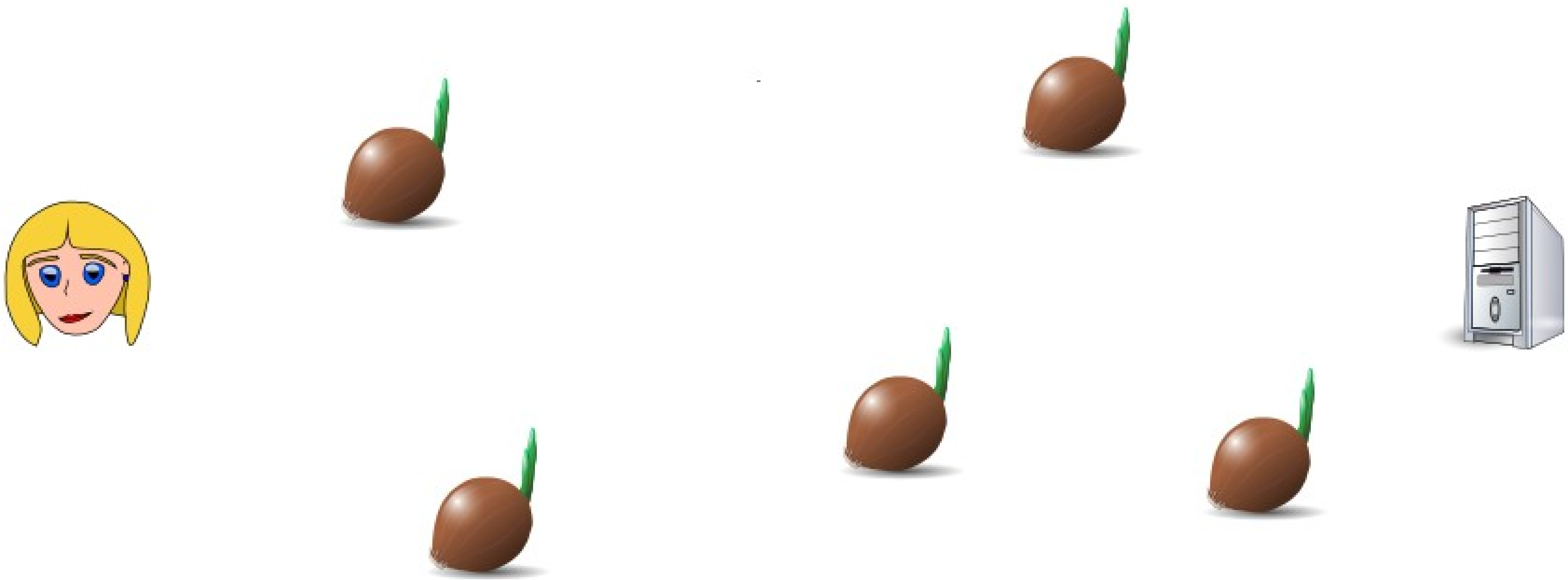
- So far, we've only used encryption to protect the contents of messages
- But there are other things we might want to protect as well!
- We may want to protect the metadata
  - Who is sending the message to whom?
  - If you're seen sending encrypted message to Human Rights Watch, bad things may happen
- We may want to hide the existence of the message
  - If you're seen sending encrypted messages at all, bad things may happen

Tor

- Tor is another successful privacy enhancing technology that works at the transport layer
  - Hundreds of thousands of users
- Normally, any TCP connection you make on the Internet automatically reveals your IP address
  - Why?
- Tor allows you to make TCP connections without revealing your IP address
- It's most commonly used for HTTP (web) connections

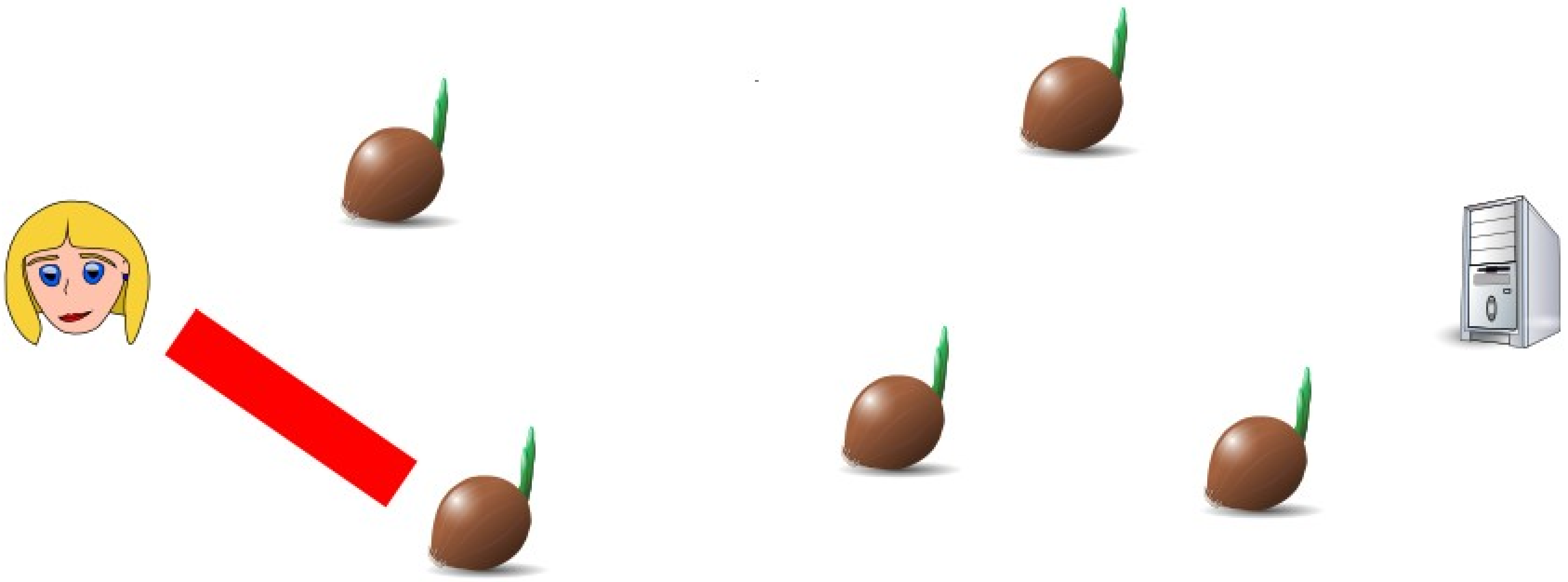
How Tor works

- Scattered around the Internet are about 1000 Tor nodes, also called Onion Routers
- Alice wants to connect to a web server without revealing her IP address



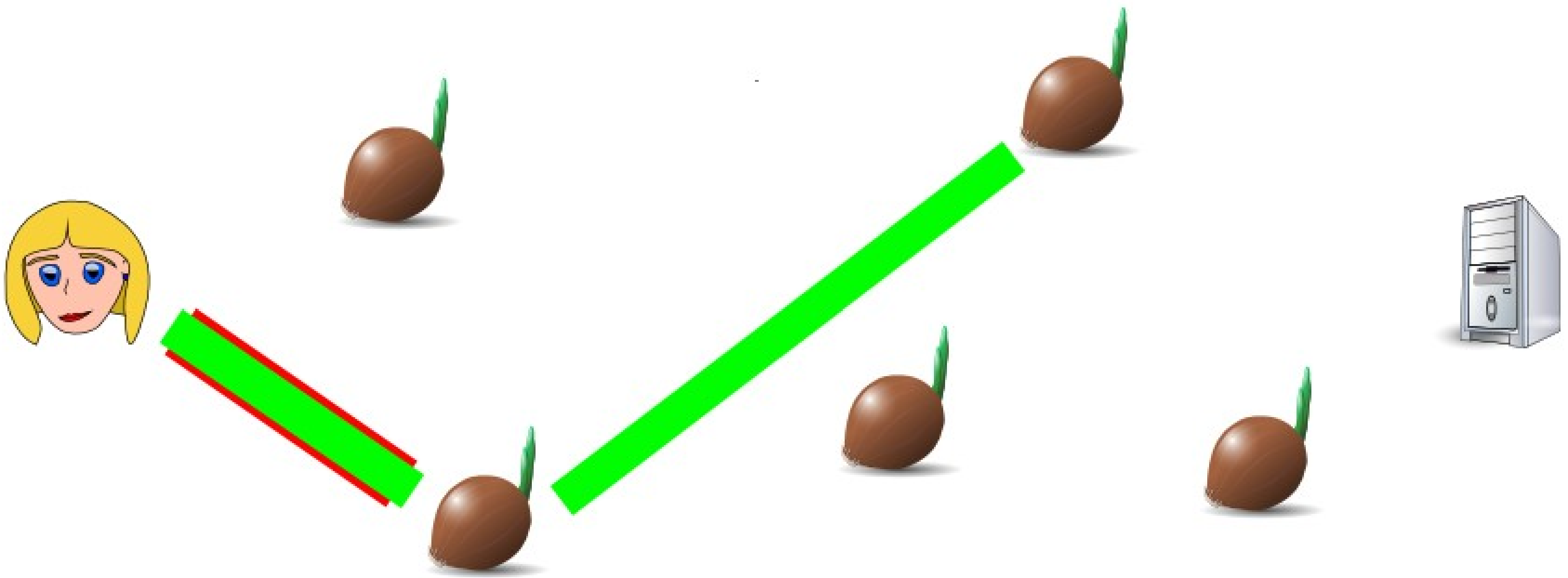
How Tor works

- Alice picks one of the Tor nodes (n1) and uses public-key cryptography to establish an encrypted communication channel to it (much like TLS)



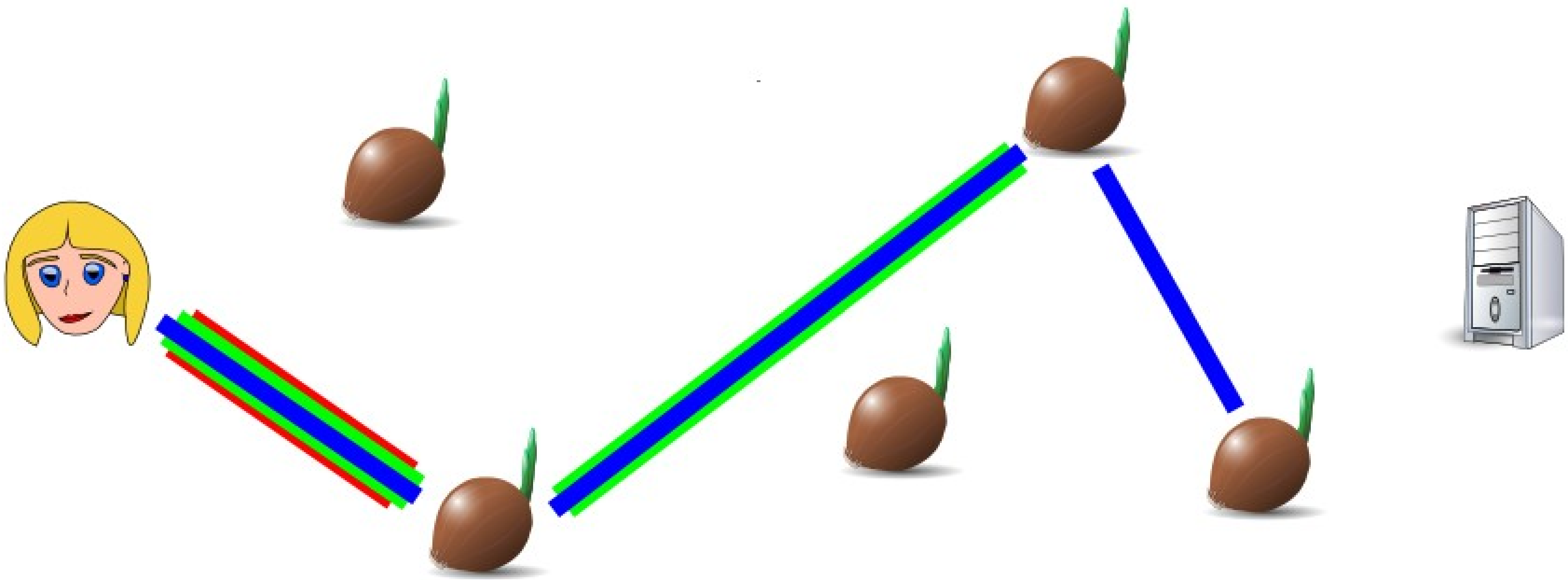
How Tor works

- Alice tells n1 to contact a second node (n2), and establishes a new encrypted communication channel to n2, tunnelled within the previous one to n1



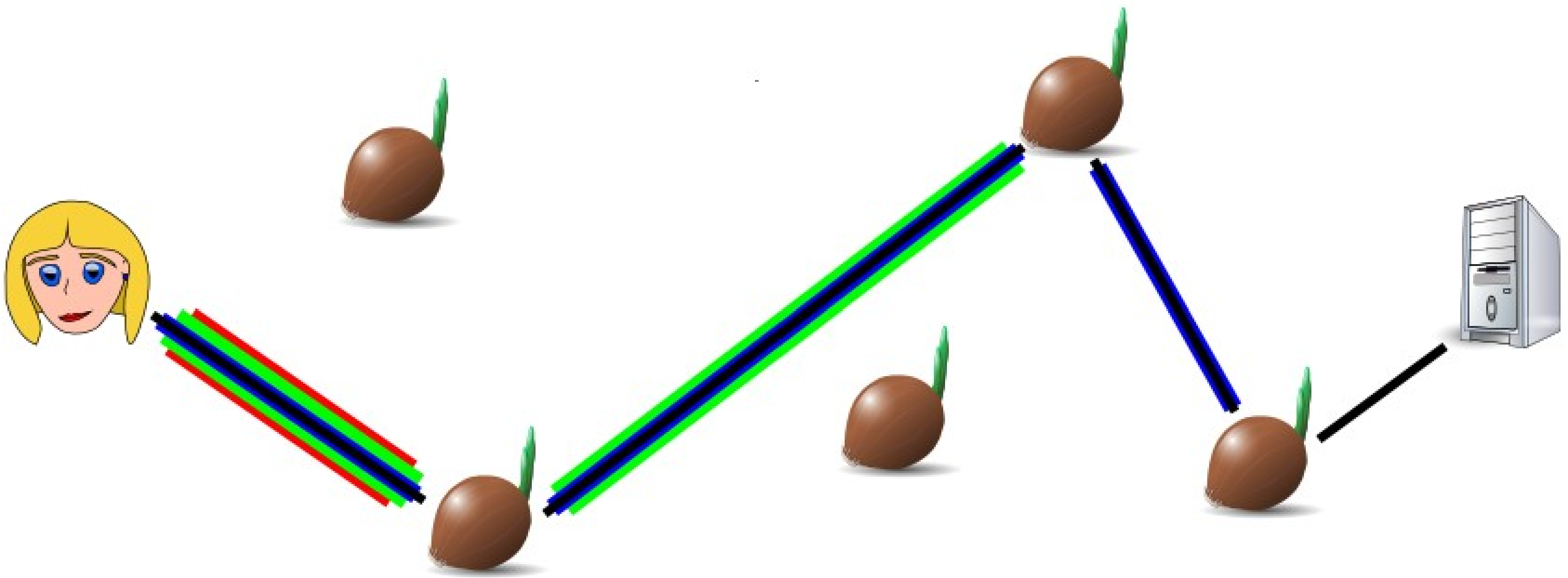
How Tor works

- Alice tells n2 to contact a third node (n3), and establishes a new encrypted communication channel to n3, tunnelled within the previous one to n2



How Tor works

- And so on, for as many steps as she likes (usually 3)
- Alice tells the last node (within the layers of tunnels) to connect to the website



Sending messages with Tor

- Alice now shares three symmetric keys:
  - *K1* with n1
  - *K2* with n2
  - *K3* with n3
- When Alice wants to send a message M, she actually sends $E_{K1}(E_{K2}(E_{K3}(M)))$
- Node n1 uses *K1* to decrypt the outer layer, and passes the result $E_{K2}(E_{K3}(M))$ to n2
- Node n2 uses *K2* to decrypt the next layer, and passes the result $E_{K3}(M)$ to n3
- Node n3 uses *K3* to decrypt the final layer, and sends M to the website

Replies in Tor

- When the website replies with message R, it will send it to node n3
  - Why?
- Node n3 will encrypt R with *K3* and send $E_{K3}(R)$ to n2
- Node n2 will encrypt that with K2 and send $E_{K2}(E_{K3}(R))$ to n1
- Node n1 will encrypt that with K1 and send $E_{K1}(E_{K2}(E_{K3}(R)))$ to Alice
- Alice will use *K1*, *K2*, and *K3* to decrypt the layers of the reply and recover R

Who knows what?

- Notice that node n1 knows that Alice is using Tor, and that her next node is n2, but does not know which website Alice is visiting
- Node n3 knows some Tor user (with previous node n2) is using a particular website, but doesn't know who
- The website itself only knows that it got a connection from Tor node n3

- Note: the connection between n3 and the website is not encrypted! If you want encryption as well as the benefits of Tor, you should use encryption in addition
  - Like HTTPS

Anonymity vs. pseudonymity

- Tor provides for anonymity in TCP connections over the Internet, both unlinkably (long-term) and linkably (short-term)

- What does this mean?
  - There's no long-term identifier for a Tor user
  - If a web server gets a connection from Tor today, and another one tomorrow, it won't be able to tell whether those are from the same person
  - But two connections in quick succession from the same Tor node are more likely to in fact be from the same person

The Nymity Slider

- We can place transactions (both online and offline) on a continuum according to the level of nymity they represent:
  - Verinymity
    - Government ID, SIN, credit card #, address
  - Persistent pseudonymity
    - Noms de plume, many blogs
  - Linkable anonymity
    - Prepaid phone cards, loyalty cards
  - Unlinkable anonymity
    - Cash payments, Tor

The Nymity Slider

- If you build a system at a certain level of nymity, it's easy to modify it to have a higher level of nymity, but hard to modify it to have a lower level.
- For example:
  - It's easy to add a loyalty card to a cash payment, or a credit card to a loyalty card.
  - It's hard to remove identity information if you're paying by credit card.
- The lesson: design systems with a low level of nymity fundamentally; adding more is easy.

Application-layer security and privacy

- TLS can provide for encryption at the TCP socket level
  - “End-to-end” in the sense of a network connection
  - Is this good enough? Consider SMTPS (SMTP/email over TLS)
- Many applications would like true end-to-end security
  - Human-to-human would be best, but those last 50 cm are really hard!
  - We usually content ourselves with desktop-to-desktop
- We'll look at three particular applications:
  - Remote login, email, instant messaging

Secure remote login (ssh)

- You're already familiar with this tool for securely logging in to a remote machine
- Usual usage (simplified):
  - Client connects to server
  - Server sends its public key
    - The client should verify that this is the correct key
  - Client picks a random session key, encrypts it with server's public key, sends to server
    - All communication from here on in is encrypted and MACd with the session key
  - Client authenticates to server
  - Server accepts authentication, login proceeds (under encryption and MAC)

Authentication with ssh

- There are two main ways to authenticate with ssh:
  - Send a password over the encrypted channel
    - The server needs to know (a hash of) your password
  - Sign a challenge with your private signature key
    - The server needs to know your public key
- Which is better? Why?

Recap

- Internet Application Security and Privacy
  - Transport-layer security and privacy: TLS / SSL, Tor
    - The Nymity Slider
  - Application-layer security and privacy: ssh

Next time

- Internet Application Security and Privacy
  - Application-layer security and privacy: remailers, PGP/gpg, OTR