Network Security
(6CCS3NSE - 7CCSMNSE)

Diego Sempreboni
Department of Informatics
King's College London, UK

Second term 2019/20

Lecture 9

Objectives and learning outcomes

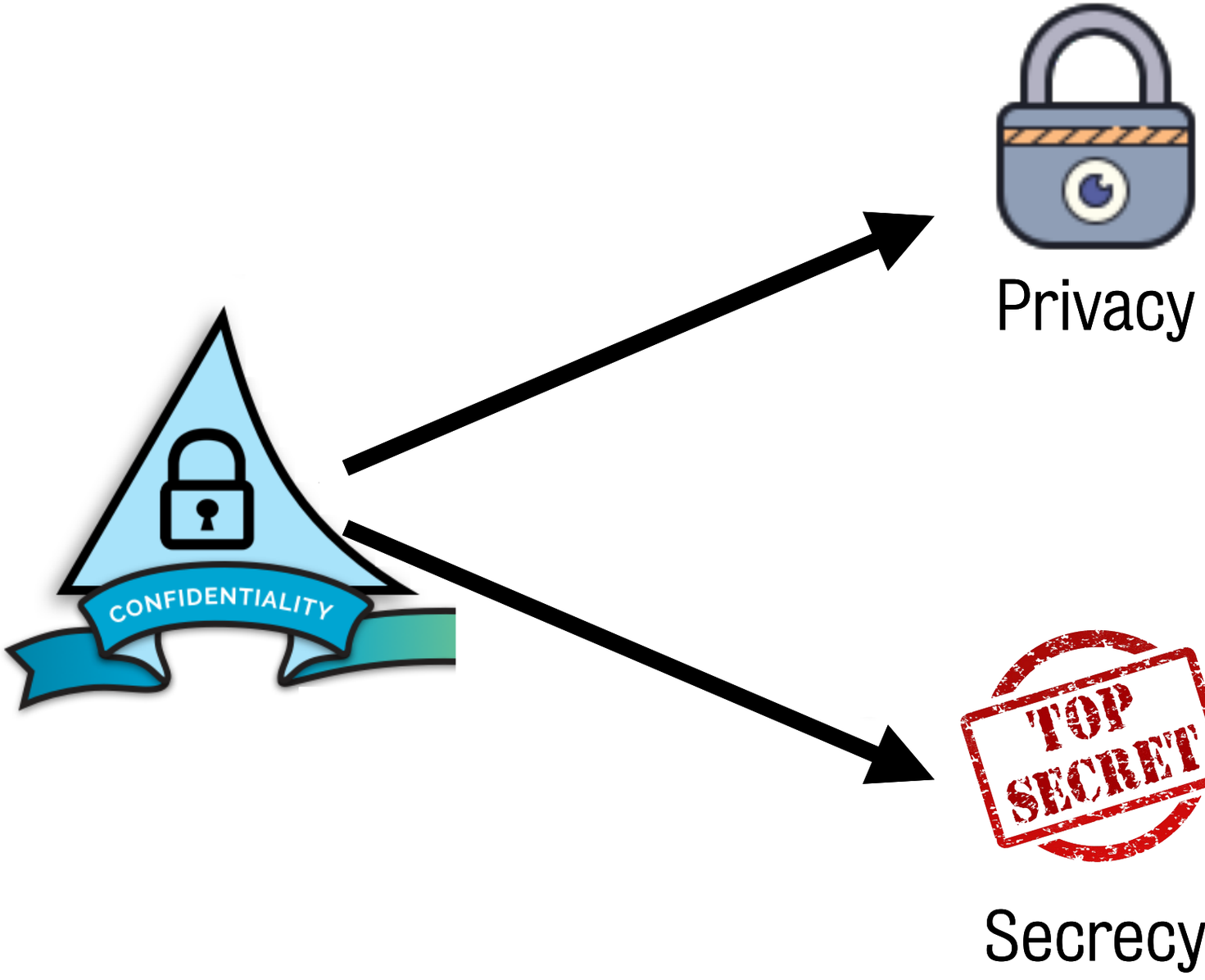
- At the end of this lecture you should understand:
  - What the concept of privacy and anonymity are
  - Privacy and anonymity on public networks
    - (i) Some possible applications of privacy and anonymity
    - (ii) Mix networks
    - (iii) Onion routing
    - (iv) Dining cryptographers

- Privacy: requirements, policy, and mechanisms for data protection

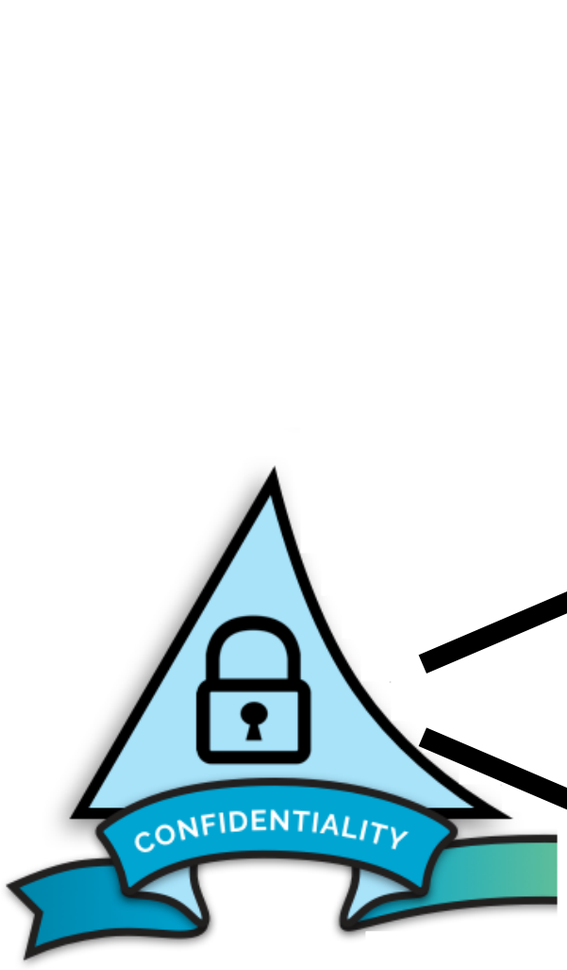
Security Principles

- **Confidentiality**: Assures that private or confidential information is not disclosed to unauthorised individuals.
- **Integrity**: Assures that information and programs are changed only in a specified and authorised manner.
- **Availability**: Assures that systems work promptly and service is not denied to authorised users.

Security Principles



Security Principles



Privacy

Anonymity

Secrecy

Privacy and anonymity

- **Privacy:**
  - According to Oxford English Dictionary: a state in which one is not observed or disturbed by others.
  - You choose what you let other people know
  - Confidentiality of information that you don't want to share

- **Anonymity:**
  - A condition in which your true identity is not known.
  - Confidentiality of your identity.
  - Unobservability of our actions when they occur.

"On the Internet, nobody knows you're a dog."

Privacy and anonymity on public networks

- **Internet is designed as a public network.**
  - Machines on your LAN may see your traffic, network routers see all traffic that passes through them.
  - Email is not a letter but rather a post card! (Everyone can read it along the way.)
- **Routing information is public.**
  - IP packet headers identify source and destination.
  - Even a passive observer can easily figure out who is talking to whom.
- **Encryption does not hide identities.**
  - Encryption hides payload, but not routing information.
  - Even IP-level encryption (tunnel-mode IPsec/ESP) reveals IP addresses of IPsec gateways.

Why is anonymity difficult?

- In a public network:
  - Packet headers identify recipients.
  - Packet routes can be tracked (traffic analysis).
  - Payload, even when encrypted, is visible.

- Challenge is to design technologies to thwart such analysis.

Some possible applications of privacy and anonymity

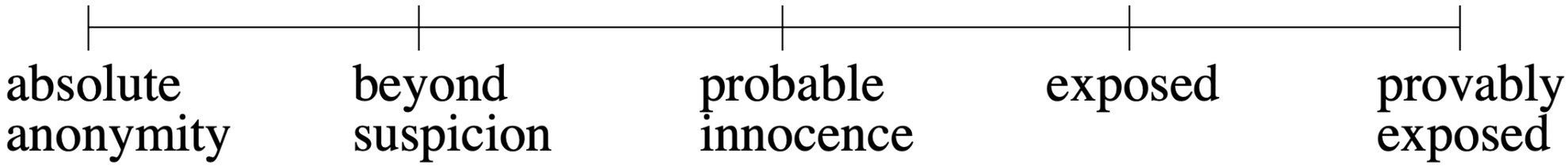
- **Privacy**:
  - Hide online transactions, Web browsing, etc. from intrusive governments, marketers and archivists.
- **Untraceable electronic mail:**
  - Corporate whistle-blowers.
  - Political dissidents.
  - Confidential business negotiations.
- **Digital cash**:
  - Electronic currency with properties of paper money (online purchases unlinkable to buyer's identity).
- **Anonymous electronic voting**.
- **Censorship-resistant publishing**.

ANONYMITY

What is anonymity?

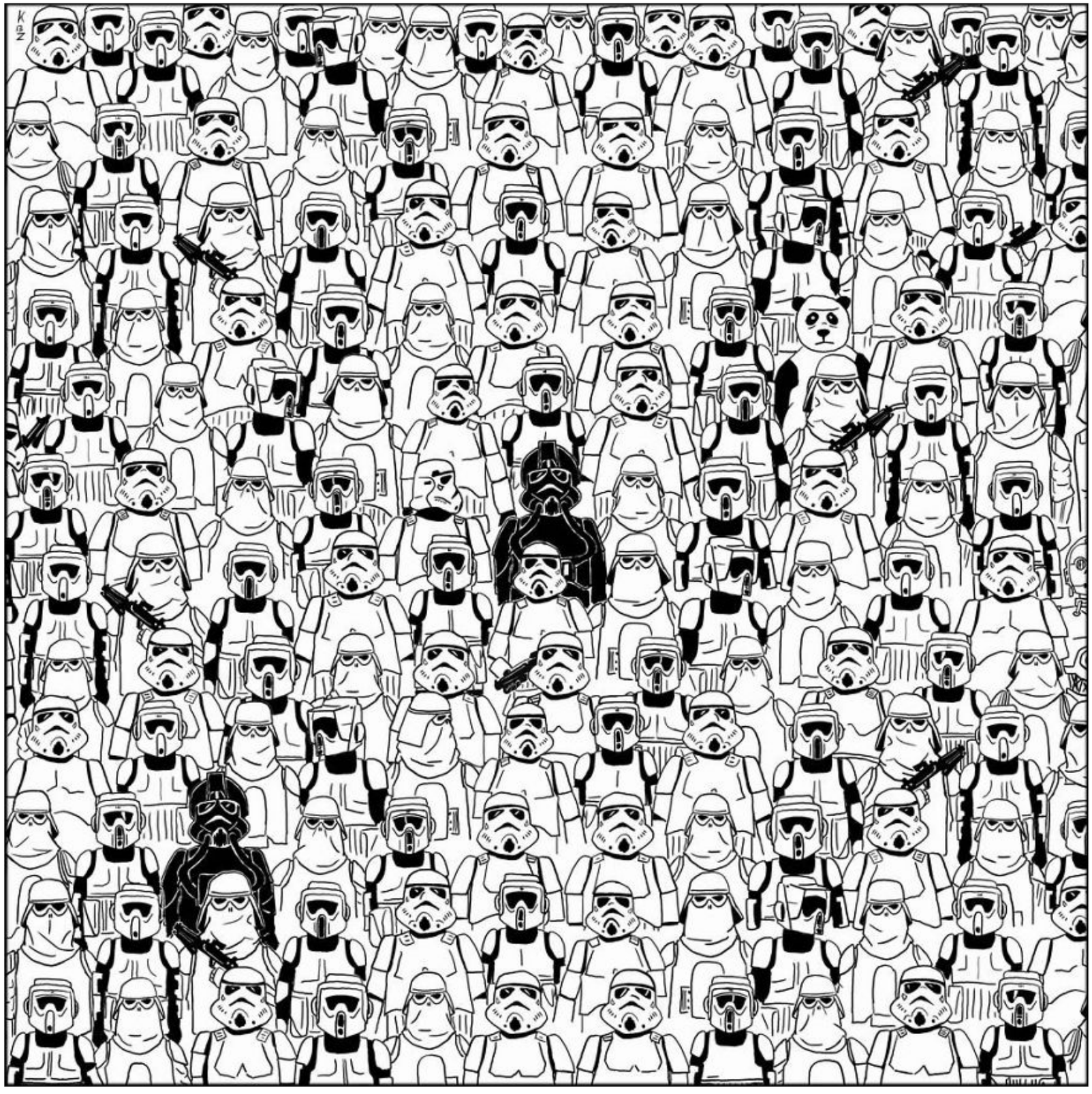
- Assume your actions can be observed (e.g., sending/receiving).

- You are only anonymous within a group if your actions (sending, receiving, communication relationships) cannot be distinguished from the actions of anyone else in a group.
- This group is called the **anonymity set**. The larger, the better.

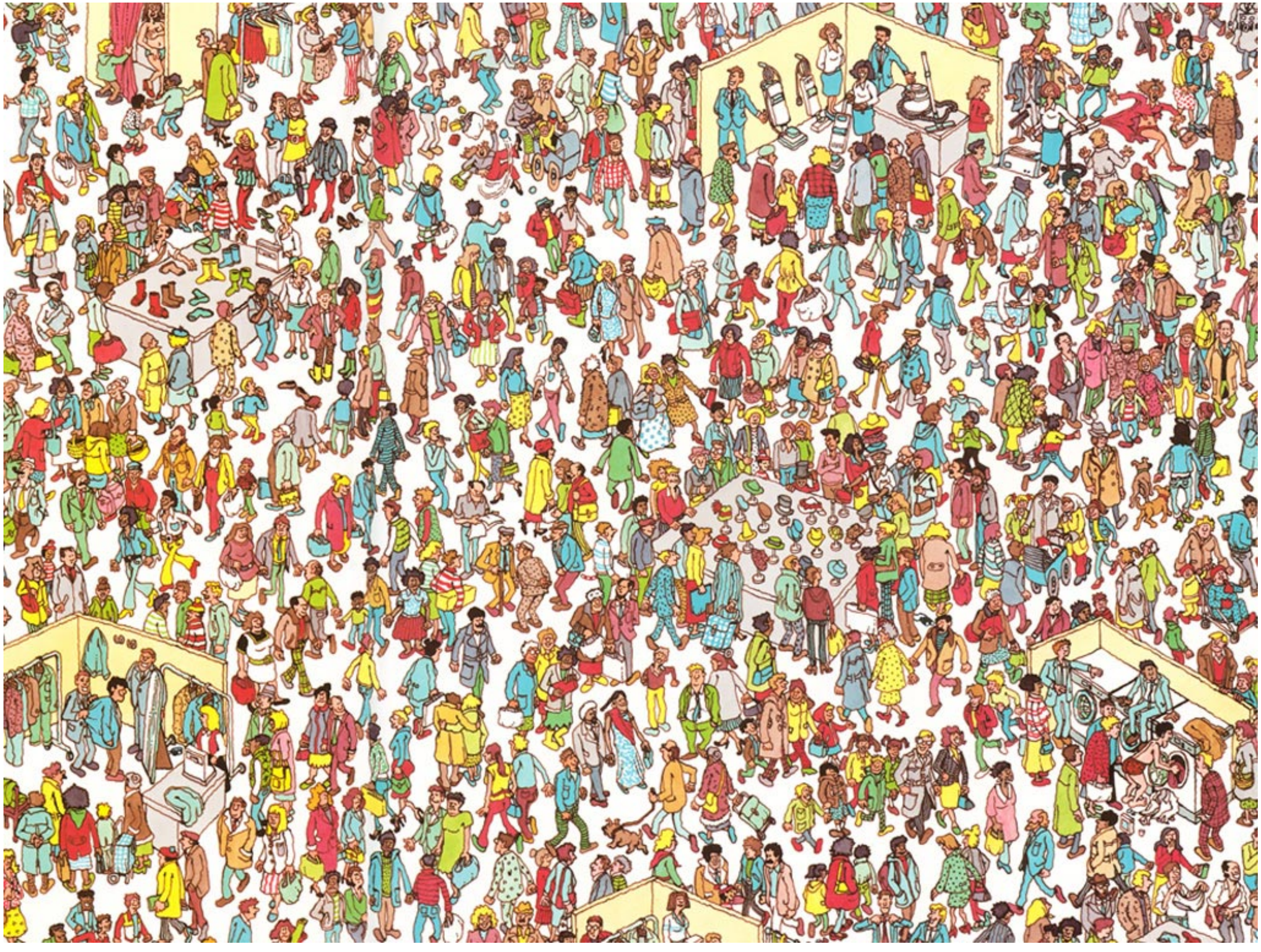


- You cannot be anonymous by yourself!
  - Big difference between anonymity and confidentiality.
- Anonymity is best when anonymizing service attracts many users.
  - All existing technologies have performance/reliability overheads — Usability is central to success.

Anonymity set: find the panda

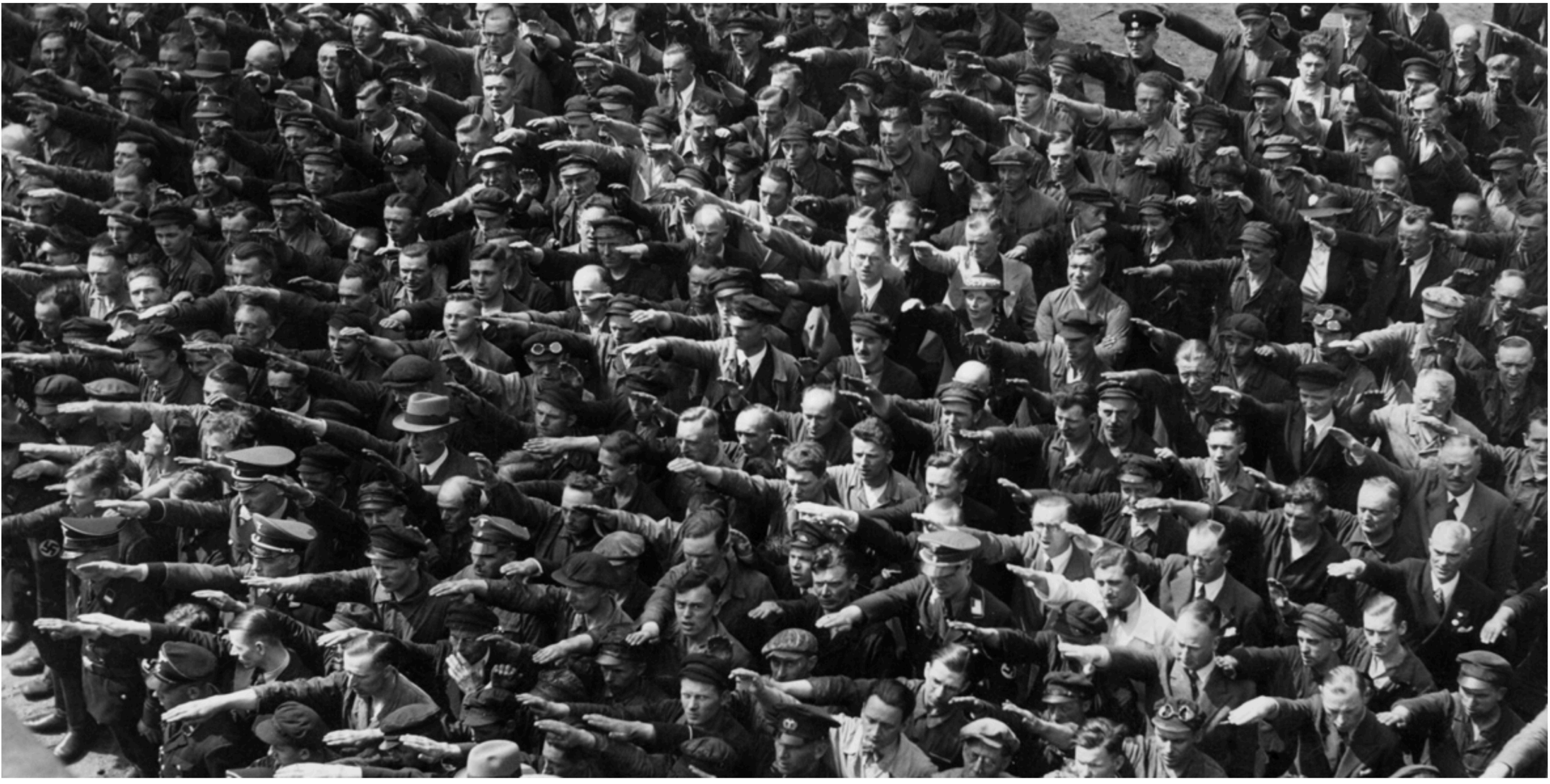


Anonymity set: Where's Waldo?



Anonymity set: August Landmesser

Launch of a German army vessel in 1936 (a ceremony that was attended by Adolf Hitler himself)



Anonymity set: August Landmesser

Anonymity set: Spartacus

Who needs anonymity?

- opposition in autocratic regimes,
- journalists under dictatorship,
- journalists in democracies,
- law enforcement, spies,
- criminals, terrorists,
- citizens under data-retention laws,
- freedom of speech,
- ...

Attacks on anonymity?

- Passive traffic analysis:
  - Infer from network traffic who is talking to whom.
  - To hide your traffic, must carry other people's traffic!
- **Active traffic analysis**:
  - Inject packets or put a timing signature on packet flow.
- **Compromise of network nodes (routers)**:
  - It is not obvious which nodes have been compromised
  - Attacker may be passively logging traffic.
  - Better not to trust any individual node
  - Assume that some fraction of nodes is good, don't know which.

Unobservability/Anonymity

- **Requirements:** Anonymise the sender and/or the receiver. Provide confidentiality of principals' identities.
- **Pseudonyms** as a lightweight mechanism.

John Wayne
(Marion Micheal Morrison)

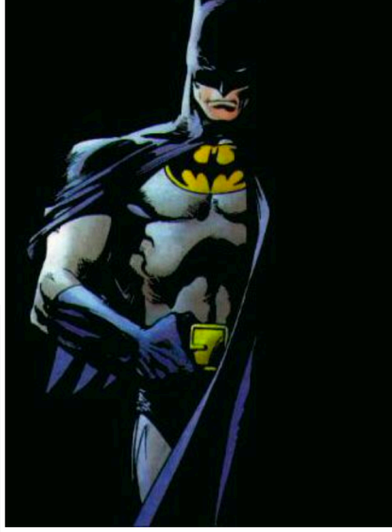
George Sand
(Amantine Aurore Lucile Dupin)

- Linkage to actual identity only in restricted cases.
- IT equivalent: mail and surf from an ISP account, Hotmail, ...
- Pseudonyms need sometimes to be resolved into the proper (underlying) names.
- But naming and name resolution can be quite problematic.

A naming example

- **Q:** Who is Batman? **A:** Bruce Wayne!

Adam West | Michael Keaton | Val Kilmer | G. Clooney | Christian Bale | Ben Affleck | Robert Pattinson



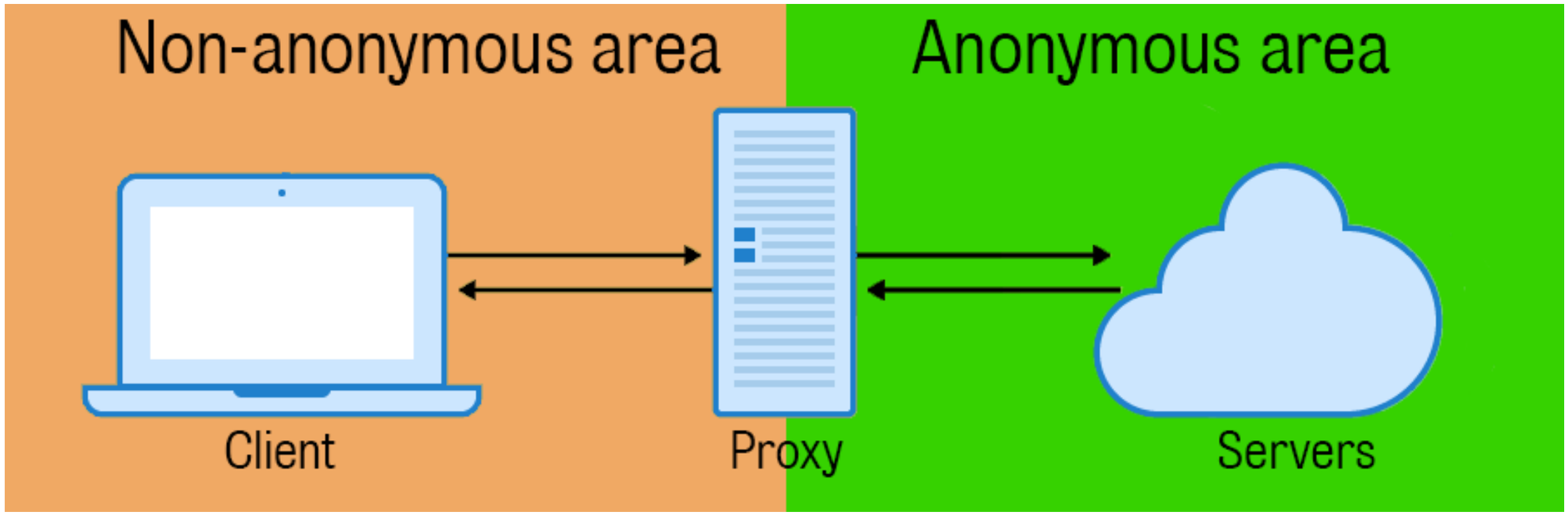
- **Q:** What do Batman and Michael Douglas have in common?

Michael Kirk Douglas. Son of Kirk Douglas (born Issur Danielovitch Demsky).

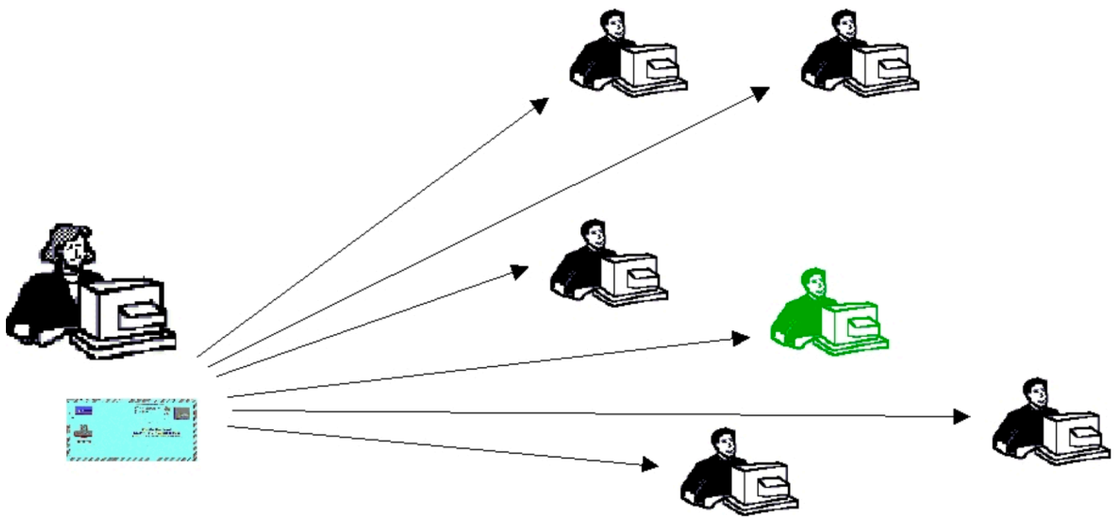
Michael John Douglas, a.k.a. Michael Keaton.

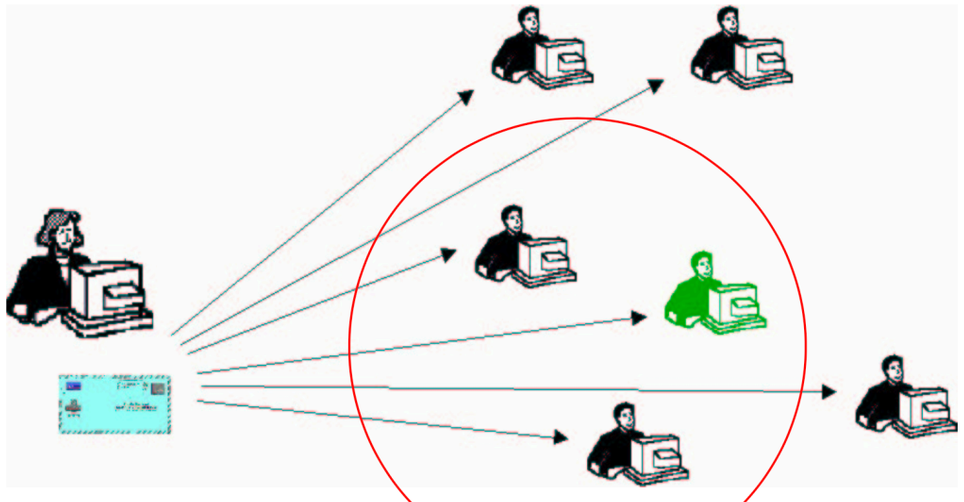
Sender/recipient anonymity: proxies

- Sender: Packets, e.g. HTTP requests, are anonymised by a proxy.



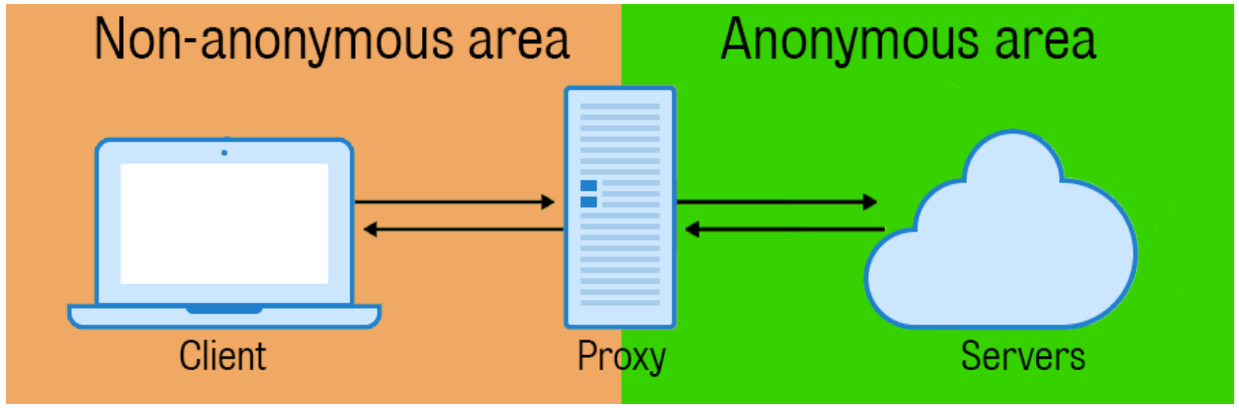
- Receiver: broadcasting or multicasting (broadcast in an anonymity group).

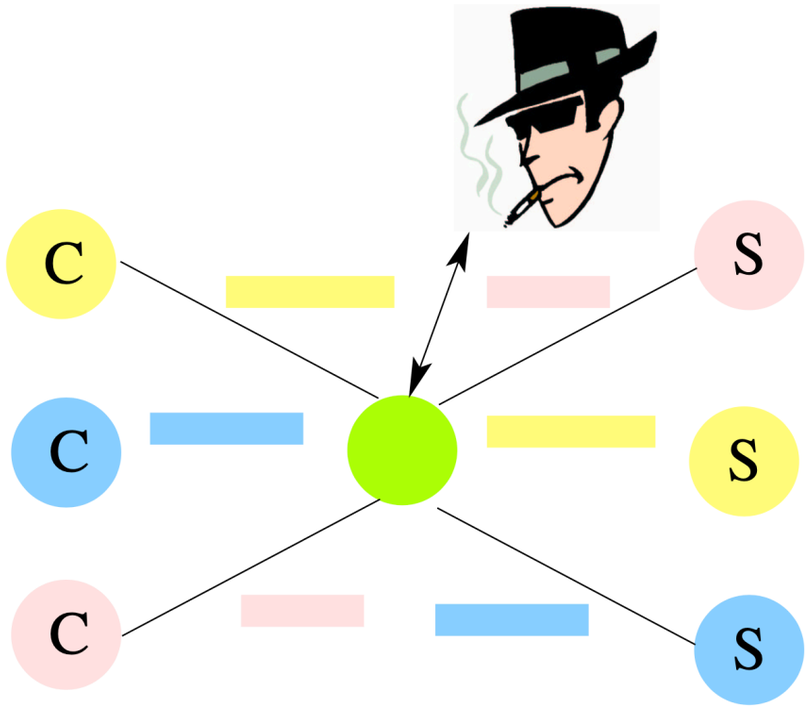




Sender/recipient anonymity: proxies

- Packets, e.g. HTTP requests, are anonymised by a proxy.

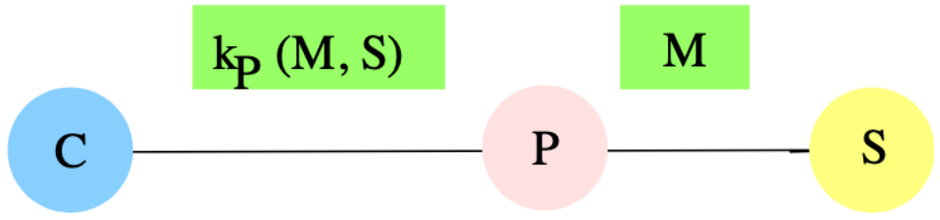




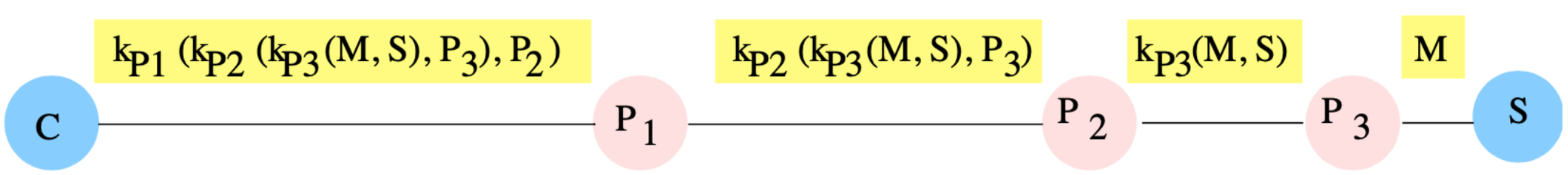
- Weaknesses?
  - The proxy knows everything.
  - Traffic analysis is also possible.
- Solutions: cascaded proxy chains, mix networks, ...

Generalisation: cascaded proxies with encryption

- Assume client shares key with proxy. Single proxy solution is:



- k_P(M, S): client C encrypts message M for proxy using proxy's public key k_P along with server address S.
- Generalise to a chain of proxies with cascaded encryption.



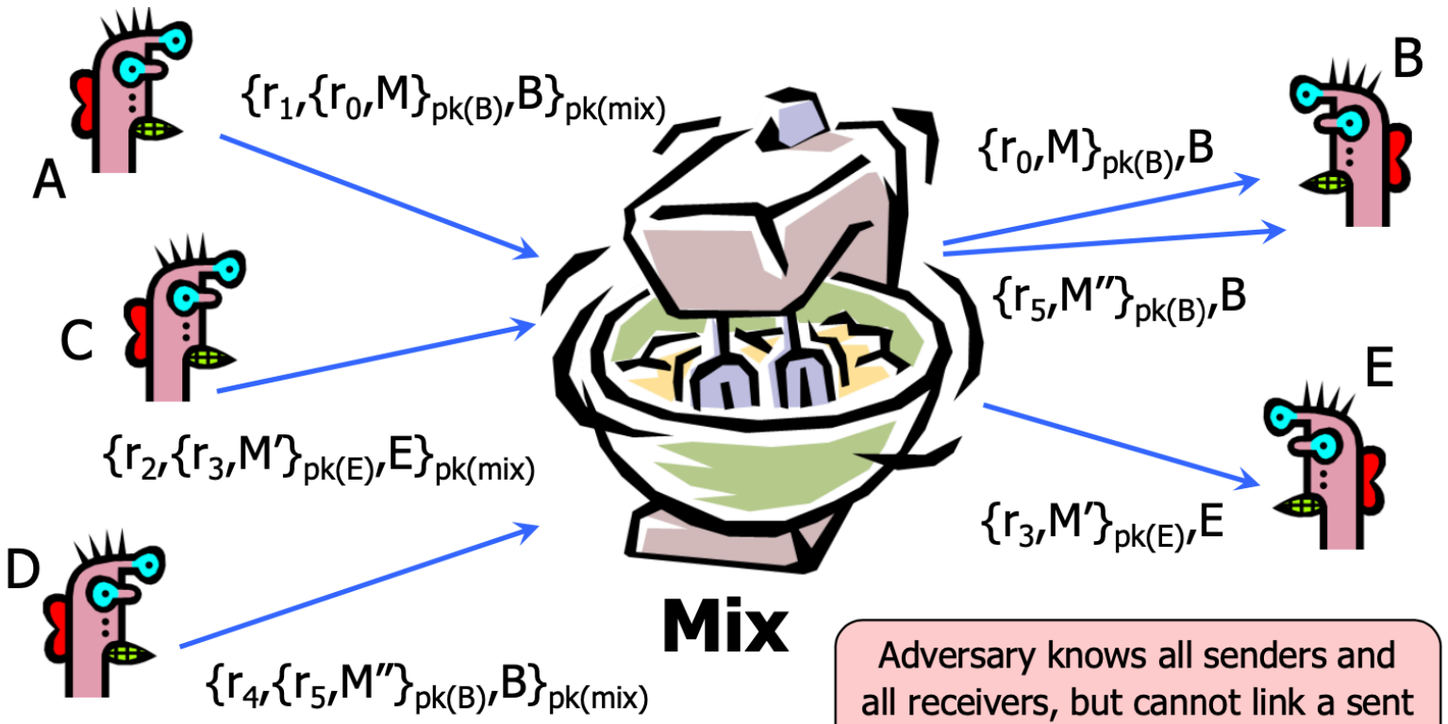
- Improvement: each proxy only knows previous/next hop. So one uncompromised proxy is enough.
- But traffic analysis is still possible. Can this be avoided?

Mix networks

Mix networks

- Developed by David Chaum ("Untraceable electronic mail, return addresses, and digital pseudonyms". Communications of the ACM, February 1981).
  - Mechanism for building an anonymous channel.
- Early proposal for anonymous email.
  - Before spam, people thought anonymous email was a good idea...
- Resulted in a variety of remailers (e.g., Cypherpunk, Mixmaster).
- Public-key cryptography plus trusted re-mailer (Mix).
  - Public keys used as persistent pseudonyms.
  - Untrusted communication medium: designed to work in environment with an active attacker who
    1. can learn origin, destination(s),and representation of all messages in the communication system;
    2. can inject, remove, or modify messages;
    3. however, cannot determine anything about the correspondences between a set of encrypted items and the corresponding set of unencrypted items, or create forgeries.

Using a single Mix



- A **Mix** is a server that processes (mail) items.
- For sending message M to agent at address B

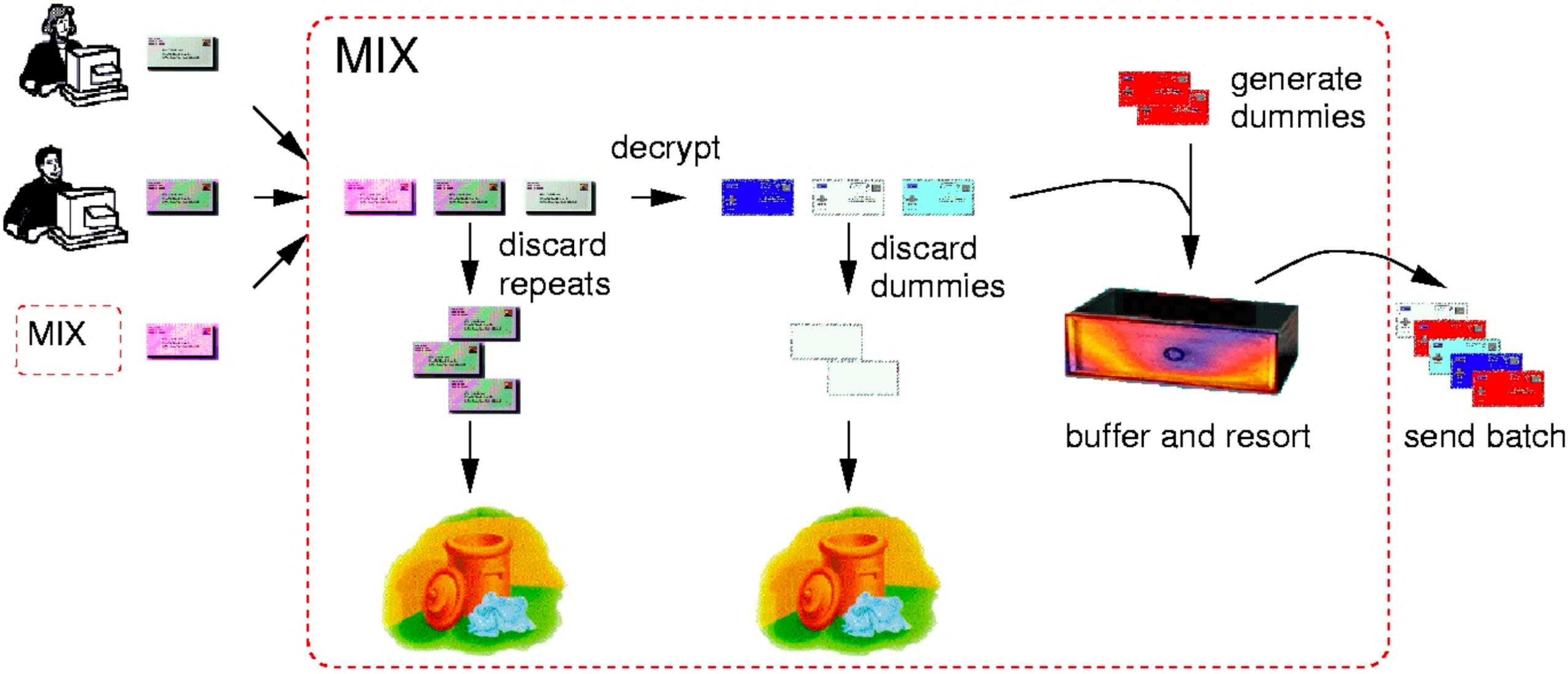
$$\{r_1, \{r_0, M\}_{pk(B)}, B\}_{pk(mix)} \rightarrow \{r_0, M\}_{pk(B)}, B$$

  - input to Mix on LHS of →.
  - Mix generates output on RHS and sends first component to B.
  - r_i are padding strings of random bits.
- This is just a variant of single proxy, with padding and encryption. Mix also performs additional operations to foil traffic analysis.

Foiling traffic analysis: four requirements

1. Agents/mixes work with uniformly sized items.
   - I.e., messages split (or padded) into fixed size blocks.
2. Order of arrival hidden by outputting items in batches.
   - Can use fix ordering (e.g., lexicographic) or random ordering.
3. Repeated information must be blocked.
   - So Mixes must filter duplicates, cross-checking across batches.
   - Or strings r include, e.g., a time stamp.
4. Sufficient traffic from a large anonymity set is required.
   - Few clients sending entails weak anonymity.
   - Solution involves clients regularly sending (and receiving) dummy messages.

Functionality of a Mix



Generating receipts

- If desired, a Mix can return a receipt Y for messages received.

$$Y = \{c, \{r_1, \{r_0, M\}_{pk(B)}, B\}_{pk(mix)}\}_{priv(mix)}$$

where c is some large, known, constant (e.g., string of zeros).

- Participant can later prove he sent message by supplying

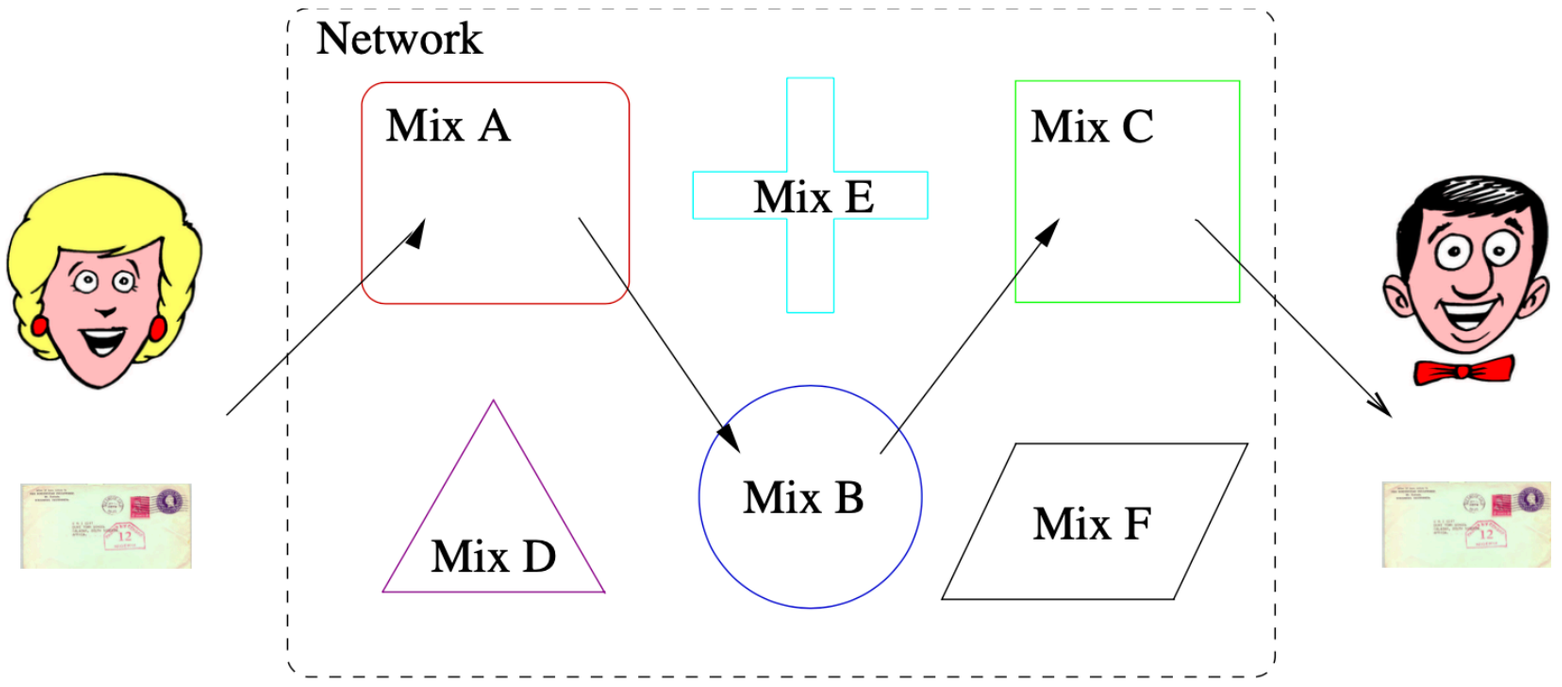
$$X = \{r_0, M\}_{pk(B)}, B$$

as well as retained string r_1.

- Proof of receipt: $\{Y\}_{pk(mix)} = c, \{r_1, X\}_{pk(mix)}$.

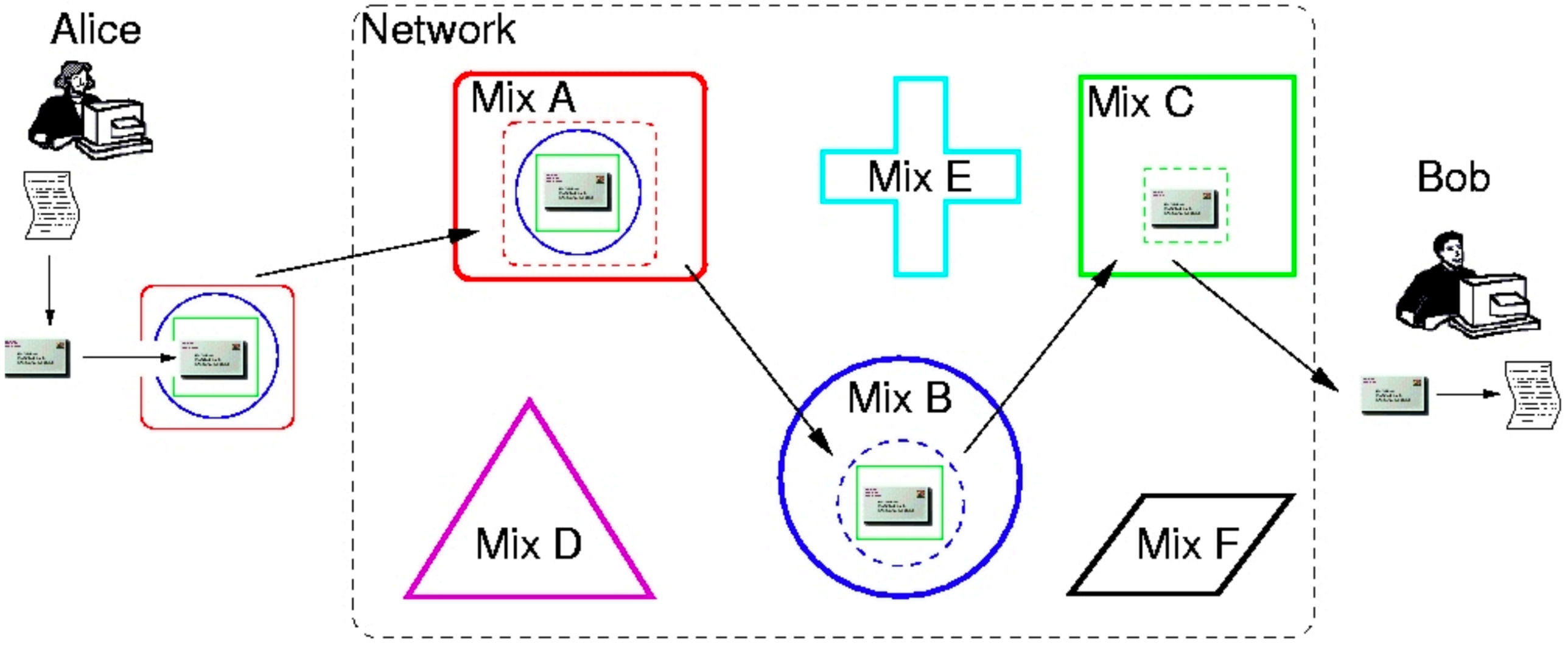
Mix networks

- Single Mix has same weakness as single Proxy.
  - If you compromise it, you know everything.
- Weakness lessened by forming Mix networks (or cascade)
  - Messages are sent through a sequence of mixes-
    - Can also form an arbitrary network of mixes (**mixnet**).
  - Each Mix in a different “administrative domain”.
  - Some of the mixes may be controlled by attacker, but even a single good mix guarantees anonymity



Mix network: schematic

- Alice as (sender/client) chooses Mix-path to Bob (receiver/server). Alice encrypts the messages for every Mix along the path.



Mix network: cryptographic details

- Sender prepares item for every Mix in cascade. E.g., with n Mixes:

$$\{r_n, \{r_{n-1}, \ldots, \{r_1, \{r_0, M\}_{pk(B)}, B\}_{pk(mix_1)} \ldots, B_{n-2}\}_{pk(mix_{n-1})}, B_{n-1}\}_{pk(mix_n)}$$

- Each Mix peals off (decrypts) outermost "layer", forwarding result.

$$\{r_{n-1}, \ldots, \{r_1, \{r_0, M\}_{pk(B)}, B\}_{pk(mix_1)} \ldots, B_{n-2}\}_{pk(mix_{n-1})}$$

- Final Mix processes same data as in single-Mix case.

$$\{r1, \{r0, M\}_{pk(B)}, B\}_{pk(mix1)}$$

Untraceable return addresses

- To respond to an anonymous sender x with a return message M.
- Single Mix case (with key $pk(mix_1)$).
  - Sender includes "return address": $\{r_1, A_x\}_{pk(mix_1)}$, $pk(x)$.
    - r_1 is random string that can also be used as a shared key.
    - $pk(x)$ is a fresh public key, created for this purpose.
    - A_x is x's actual address.
  - Receiver sends to ("response") Mix: $\{r_1,A_x\}_{pk(mix_1)}$,$\{r_0,M\}_{pk(x)}$.
  - Mix transforms this to A_x, $\{\{r_0, M\}_{pk(x)}\}_{r_1}$.
  - Second part sent to: A_x.
- Encryption with r_1 masks input/output correlation.
- Only original sender can decrypt as he created both $pk(x)$ and r_1.

Return addresses — general case

- Generalize return addresses of single Mix case

$$\{r_1, \{r_2, \ldots \{r_n, A_x\}_{pk(mix_n)} \ldots\}_{pk(mix_2)}\}_{pk(mix_1)}, pk(x)$$

- Recipient sends following response to 1st (return) Mix

$$\{r_1, \{r_2, \ldots \{r_n, A_x\}_{pk(mix_n)} \ldots\}_{pk(mix_2)}\}_{pk(mix_1)}, \{r_0, M\}_{pk(x)}$$

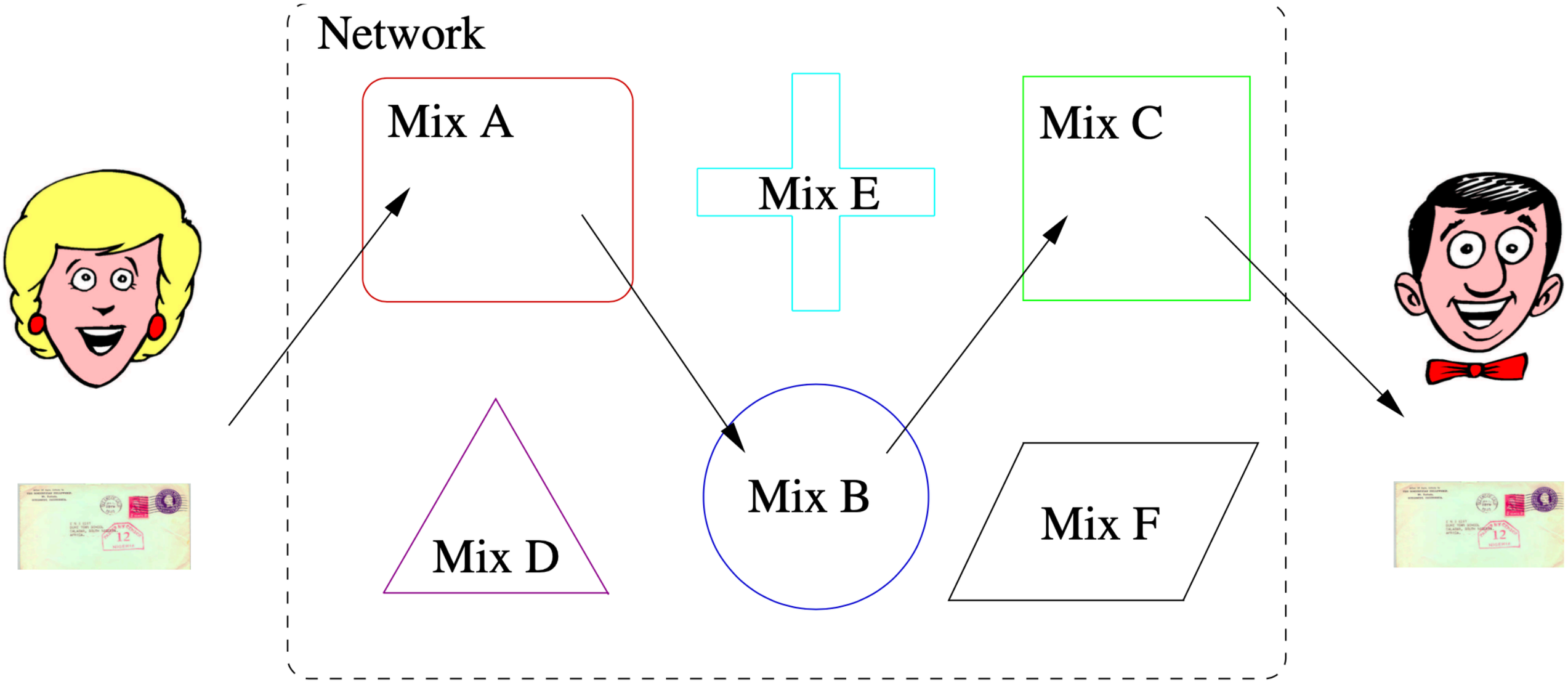
- Result of 1st Mix is

$$\{r_2, \ldots \{r_n, A_x\}_{pk(mix_n)} \ldots\}_{pk(mix_2)}, \{\{r_0, M\}_{pk(x)}\}_{r_1}$$

- Final result after n-1 mixes is

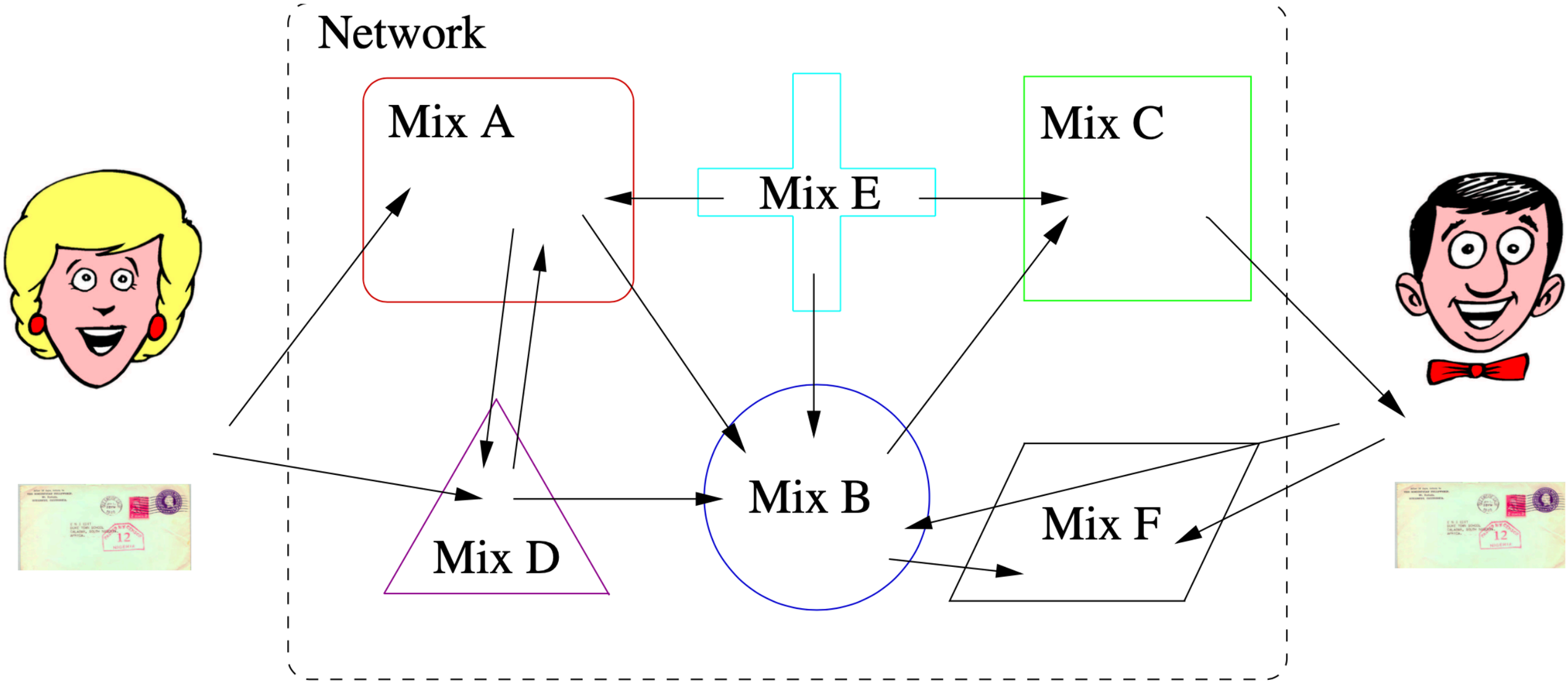
$$A_x, \{\{\{\{r_0, M\}_{pk(x)}\}_{r_1}\}_{r_2} \ldots\}_{r_n}$$

Mix network: attacks



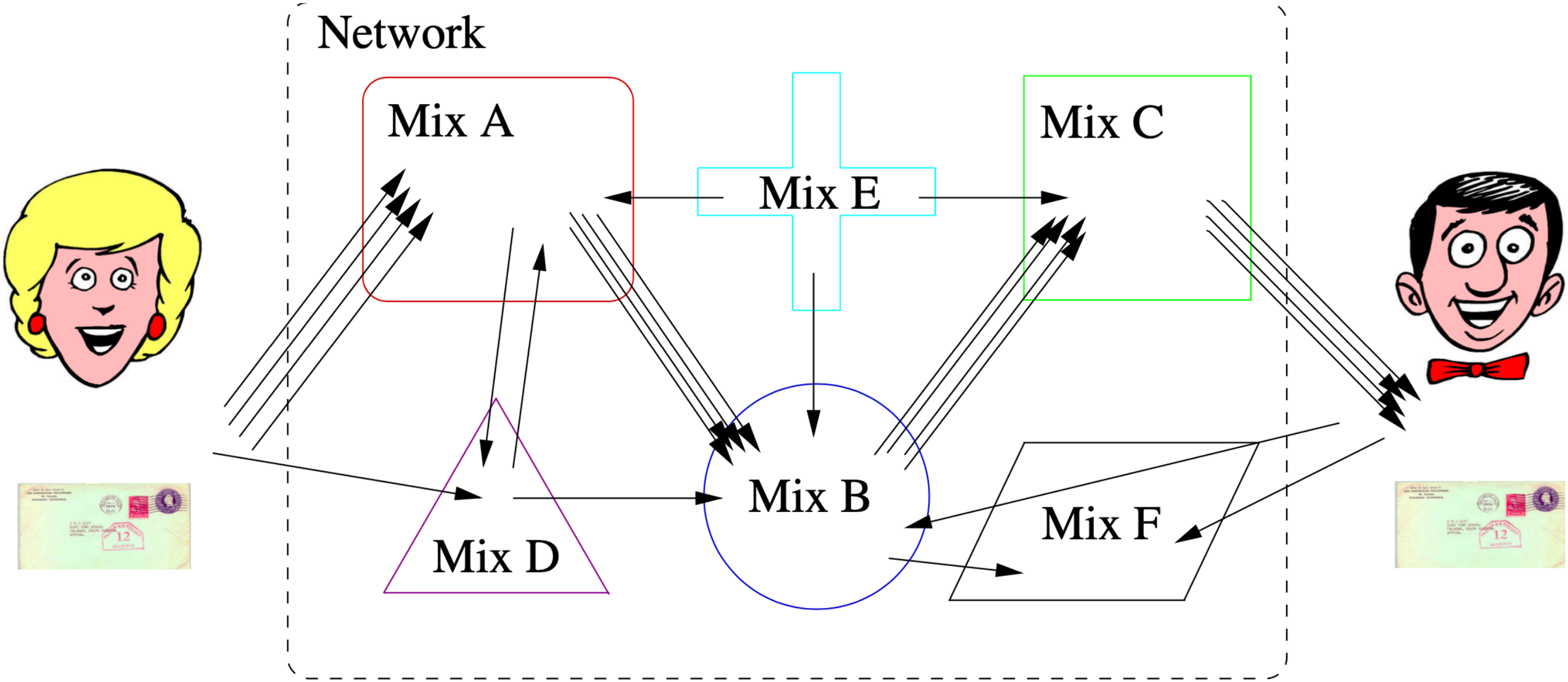
- **Tracing a message**

Mix network: attacks



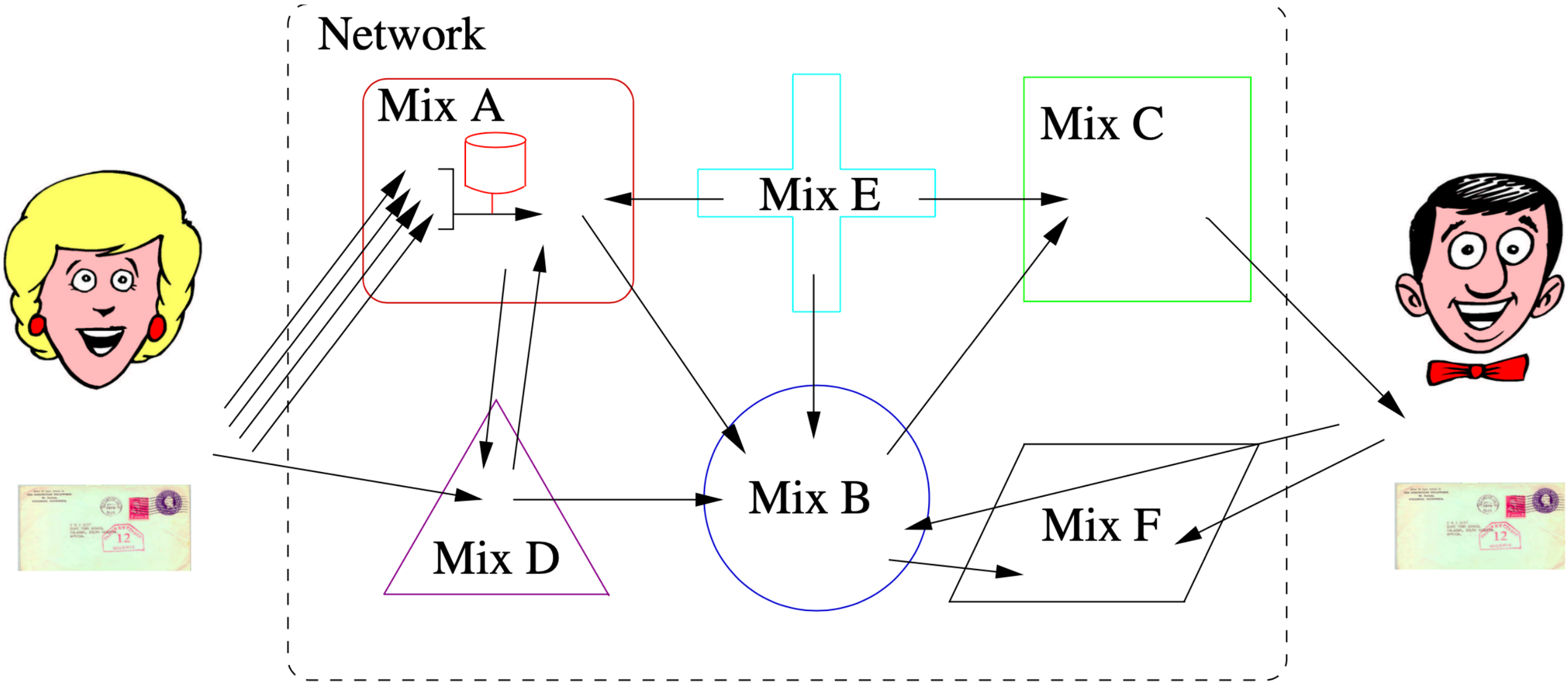
- **Tracing a message** is prevented by dummy messages.

Mix network: attacks



- **Tracing a message** is prevented by dummy messages.
- **Replay attack**

Mix network: attacks



- **Tracing a message** is prevented by dummy messages.
- **Replay attack** is prevented by a replay filter.

Mix network: attacks

- Other attacks are possible, e.g. n - 1 **attack** on a mix:

  What happens if an attacker knows (e.g. has sent himself) n - 1 of the n messages input to a mix?

  What are the countermeasures against this attack?

Mixes in practice

- Zeroknowledge Systems Freedom Network.
  - 150 Mixes in North America, Europe, Japan.
  - Contracts with various ISPs.
  - Acceptable efficiency by avoiding dummy traffic.
  - Shutdown 2001 after 2 years due to high costs/limited returns.
- Numerous volunteer-operated networks. However, no large mix-network operated by volunteers so far.
  - Too expensive for most individuals.
  - Bad press about supporting “terrorists”.
  - Open to law-suits: Church of Scientology vs. anon.penet.fi Possibility that most are owned by the government.
- Current research also is in peer-to-peer based Mix networks.

Mix networks: advantages

- Very high degree of anonymity.
  - No correlation between Mix input and output.
  - Only some nonzero fraction of mixes need be honest.
  - With enough dummy traffic, the anonymity set is entire network.
- Cryptography used in a novel way
  - With anonymous response, sender is anonymous even to receiver.
  - Can also be used for anonymous “certified mail” where receipt is signed by receiver and every Mix along the path.

Mix networks: disadvantages

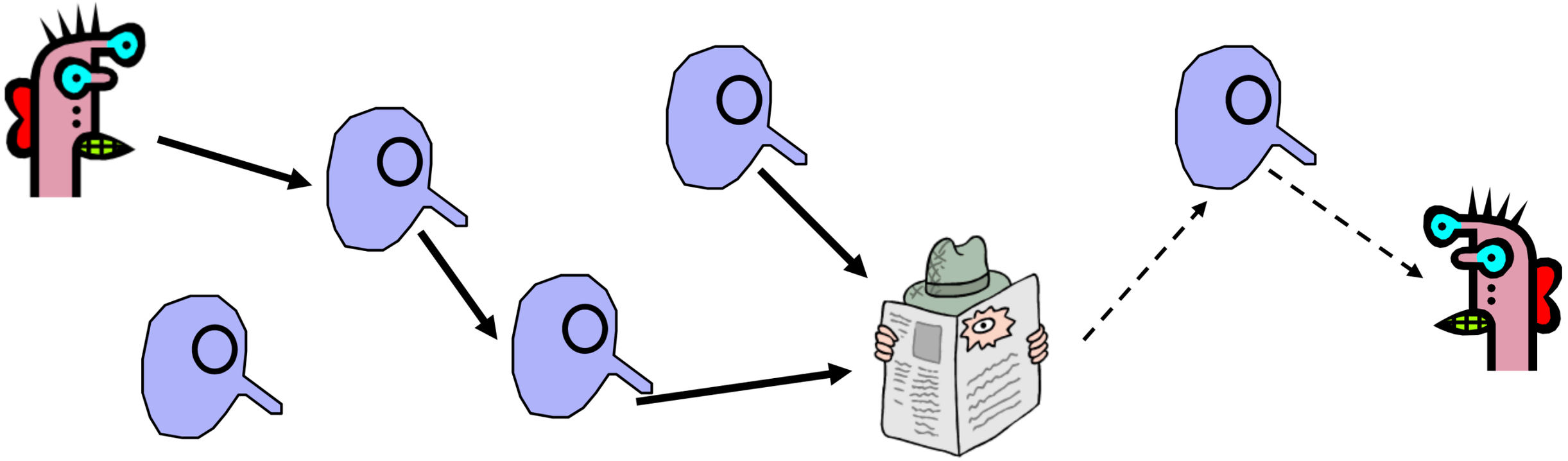
- Public-key encryption and decryption at each mix are **computationally expensive**.
- **Dummy overhead** (but in a network with substantial communication, dummies can be reduced or even eliminated).
- **High latency**: OK for email but not for anonymous web browsing.
- **Challenge**: low-latency anonymity network.

- Use public-key cryptography to establish a "circuit" with pairwise symmetric keys between hops on the circuit.
- Then use symmetric decryption and re-encryption to move data messages along the established circuits.
- Each node behaves like a mix; anonymity is preserved even if some nodes are compromised.

The Thomas Crown Affair: Mix Networks

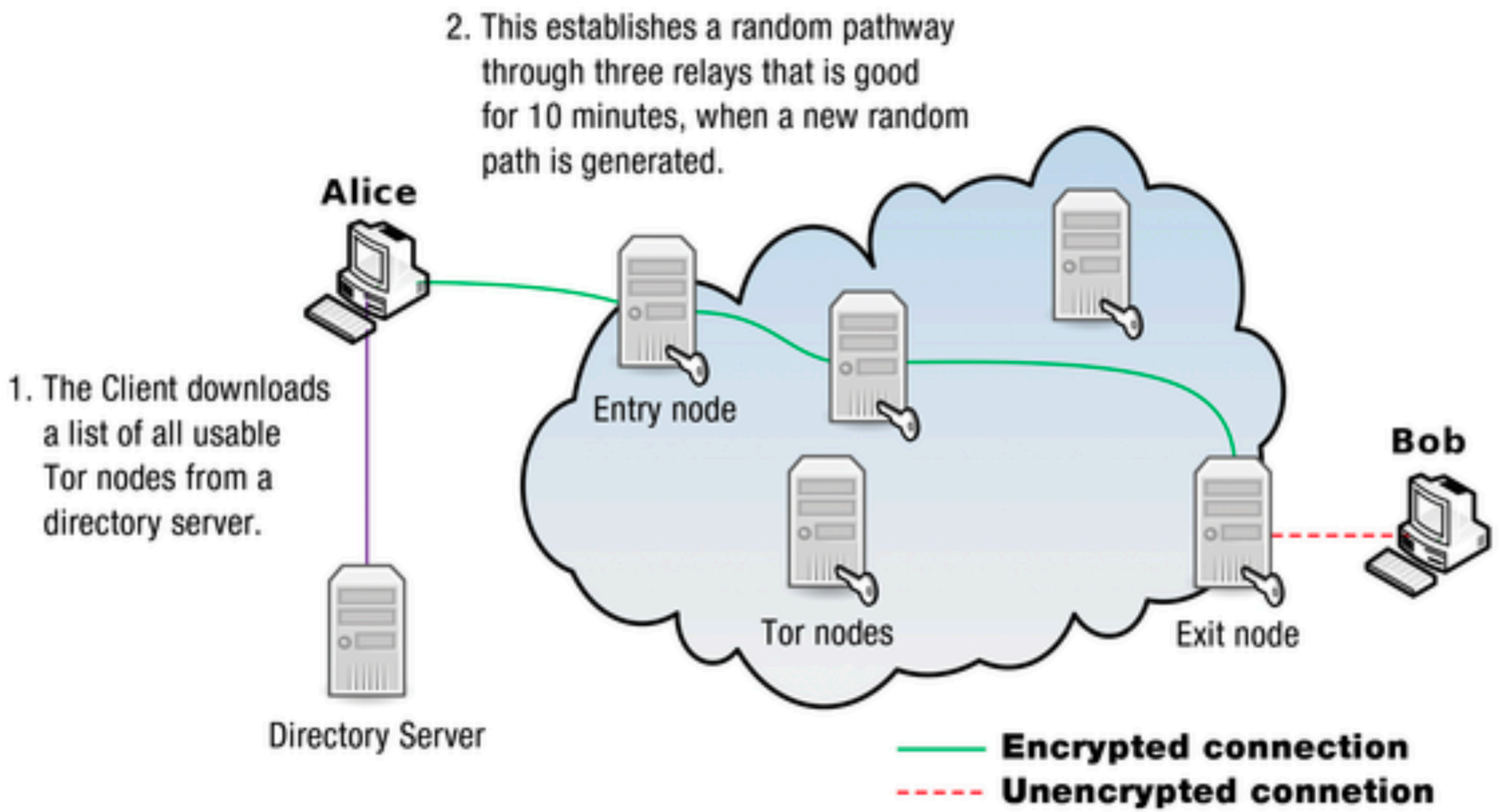
Onion routing

Another Idea: Randomized Routing



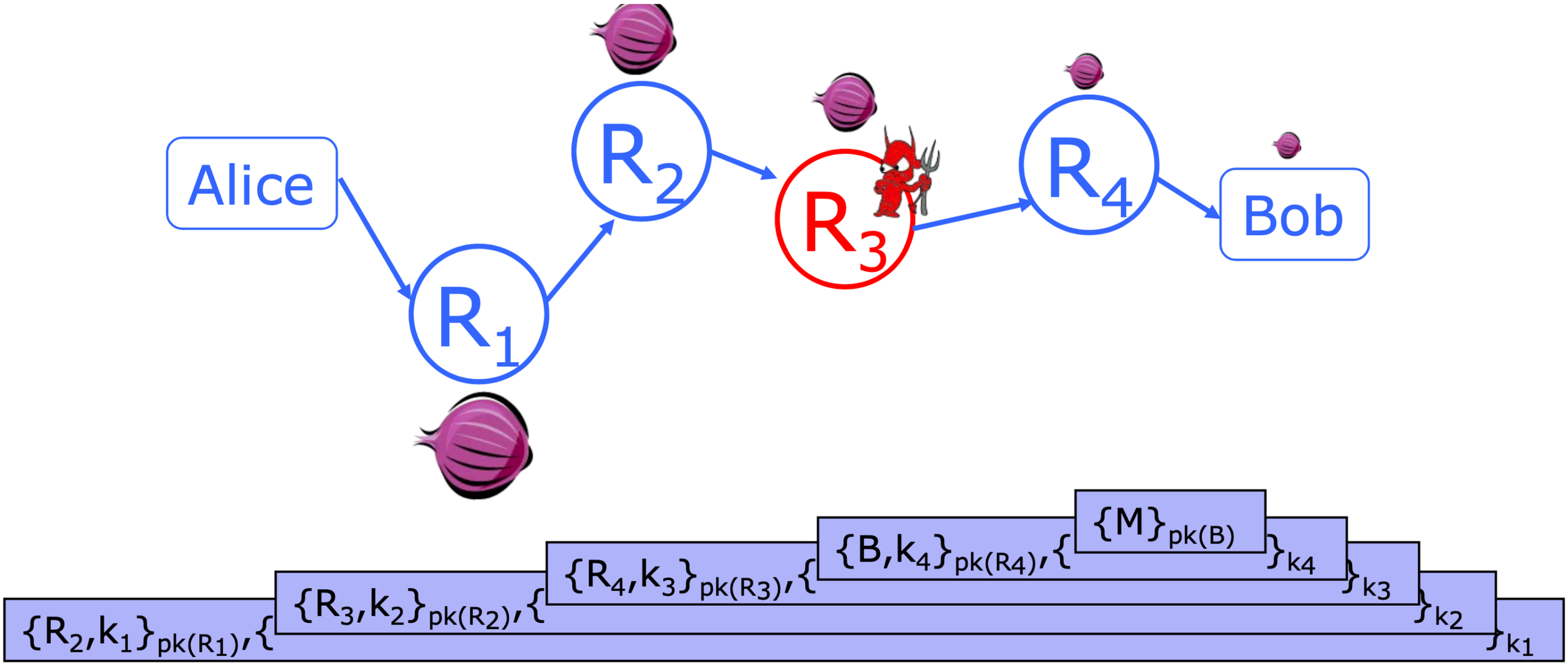
- Hide message source by routing it randomly
  - Popular technique: Crowds, Freenet, Onion routing.
  - Routers don’t know for sure if the apparent source message is the true sender or another router.

Onion routing



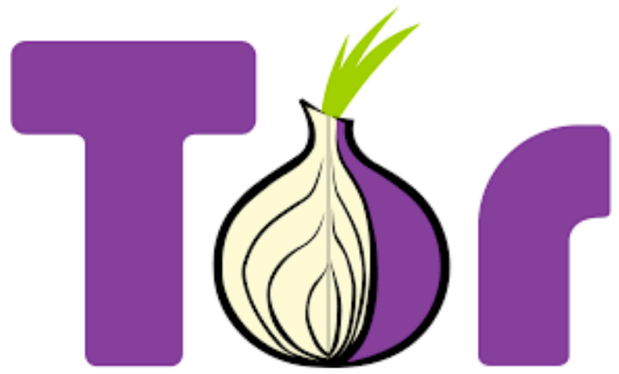
- Sender chooses a random sequence of routers.
  - Some routers are honest, some controlled by attacker
  - Sender controls the length of the path.

Route establishment



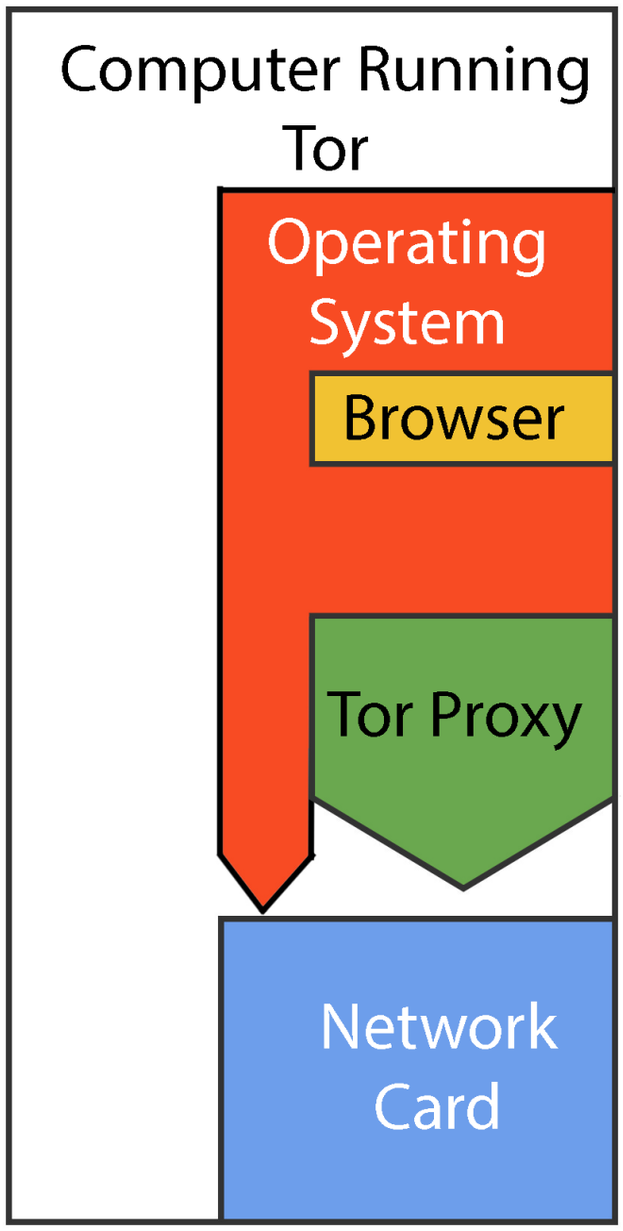
- Routing info for each link encrypted with router's public key.
- Each router learns only the identity of the next router

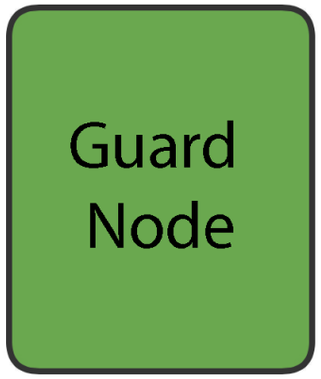
Tor (The Onion Router)

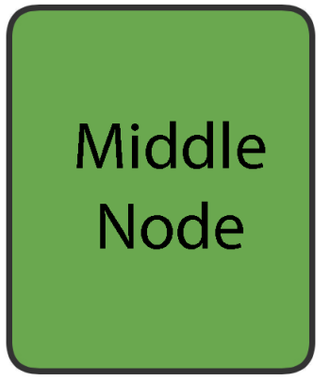


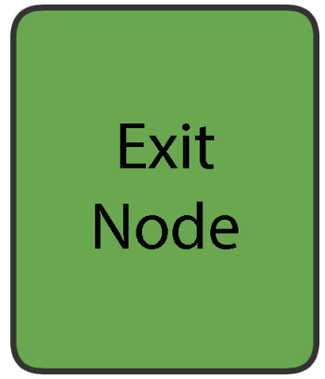
- Second-generation onion routing network.
  - http://tor.eff.org
  - Specifically designed for low-latency anonymous Internet communications (e.g., Web browsing).
    - Running since October 2003.
- Hundreds of nodes on all continents.
- Approximately 300,000 users in 2009, about 2 million in 2015.
- “Easy-to-use” client proxy.
  - Freely available, can use it for anonymous browsing.

Tor circuit setup

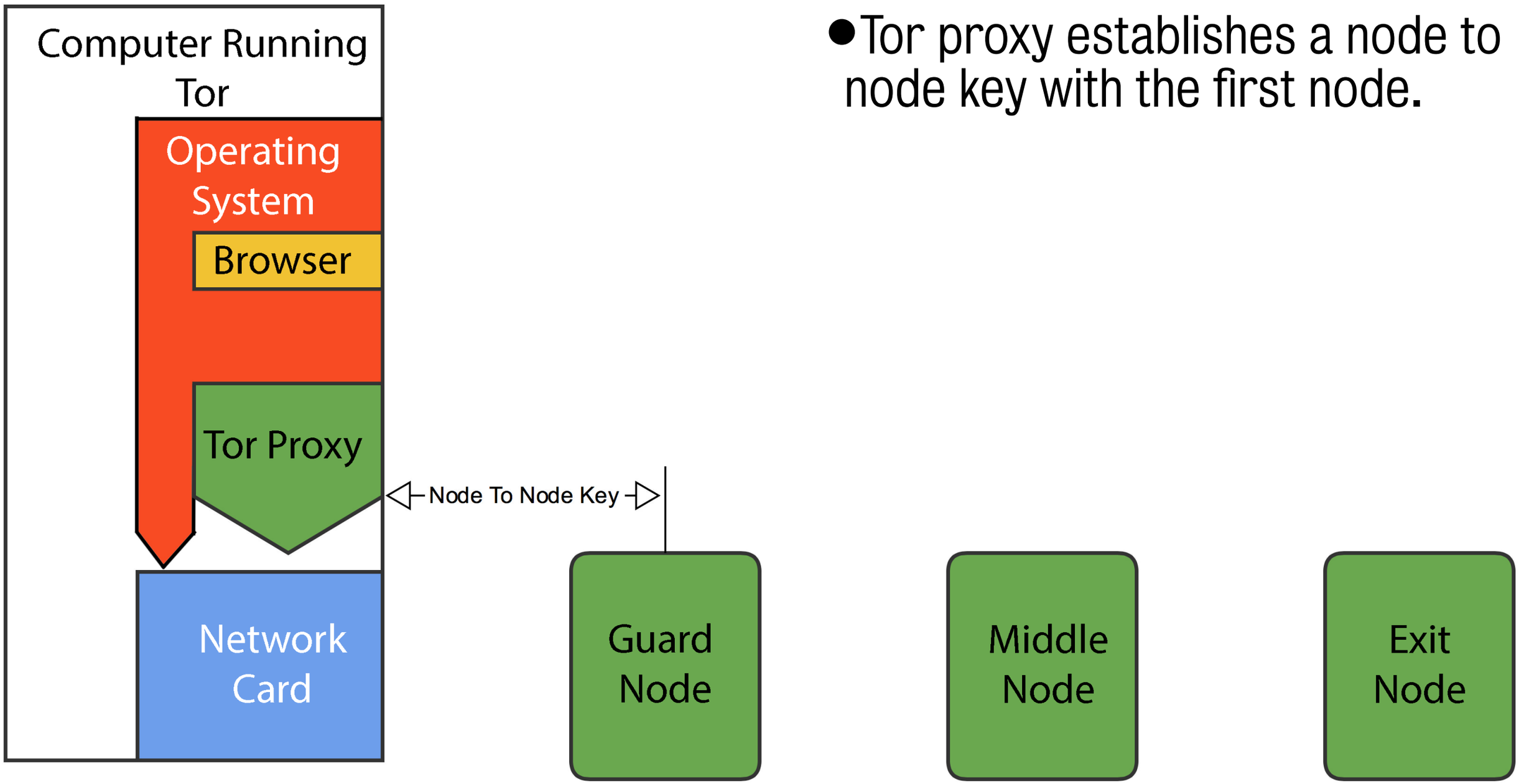






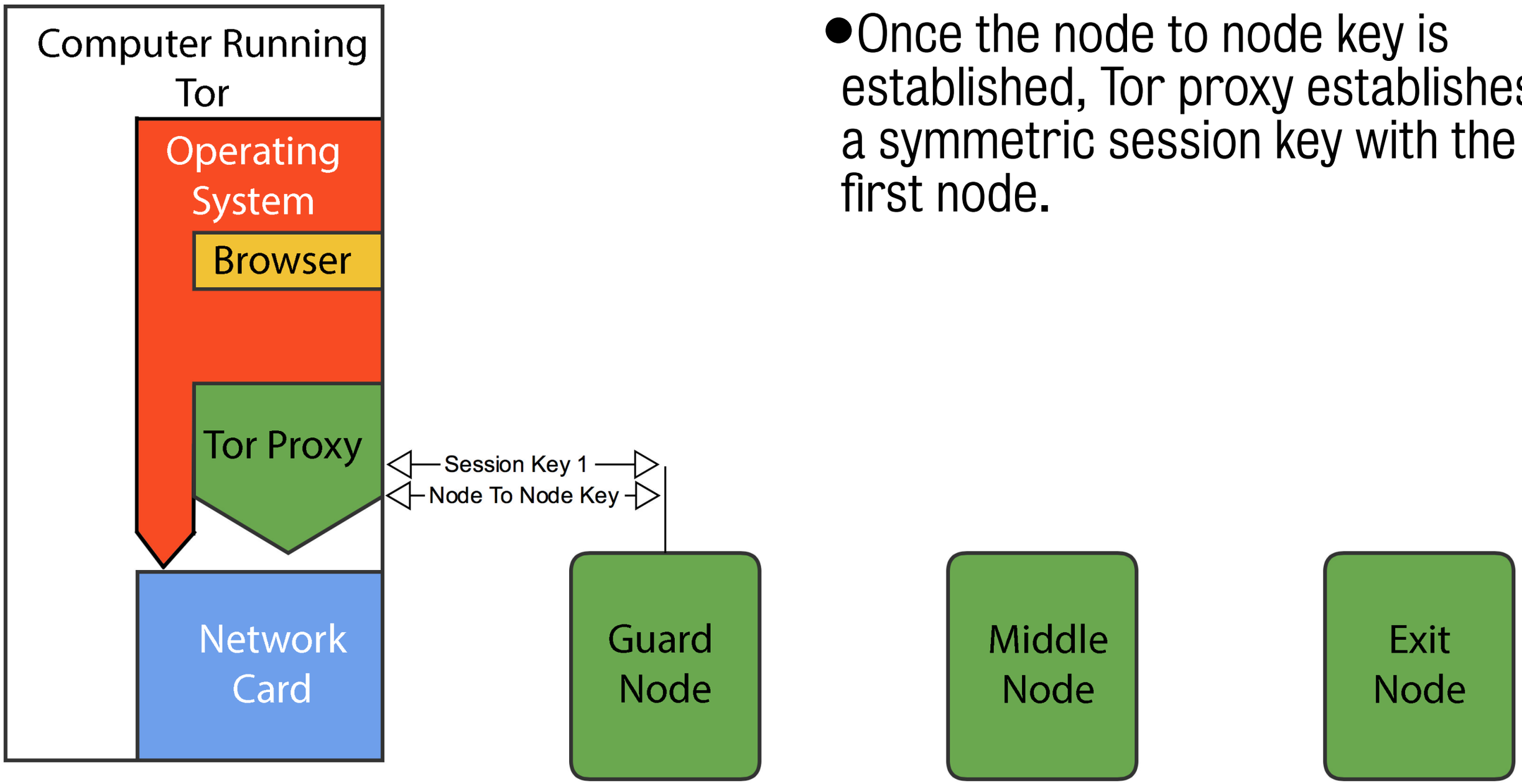


Tor circuit setup



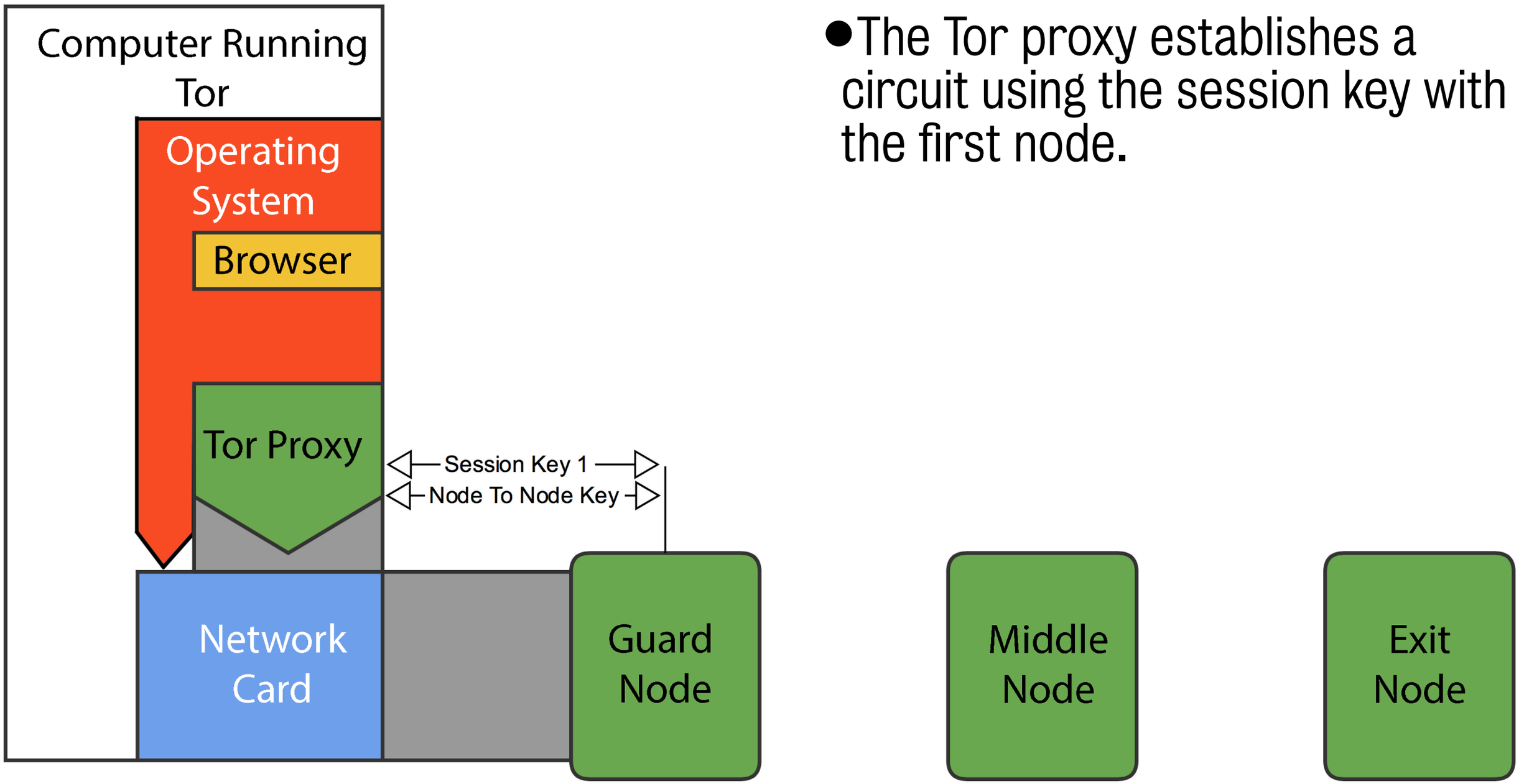
- Tor proxy establishes a node to node key with the first node.

Tor circuit setup



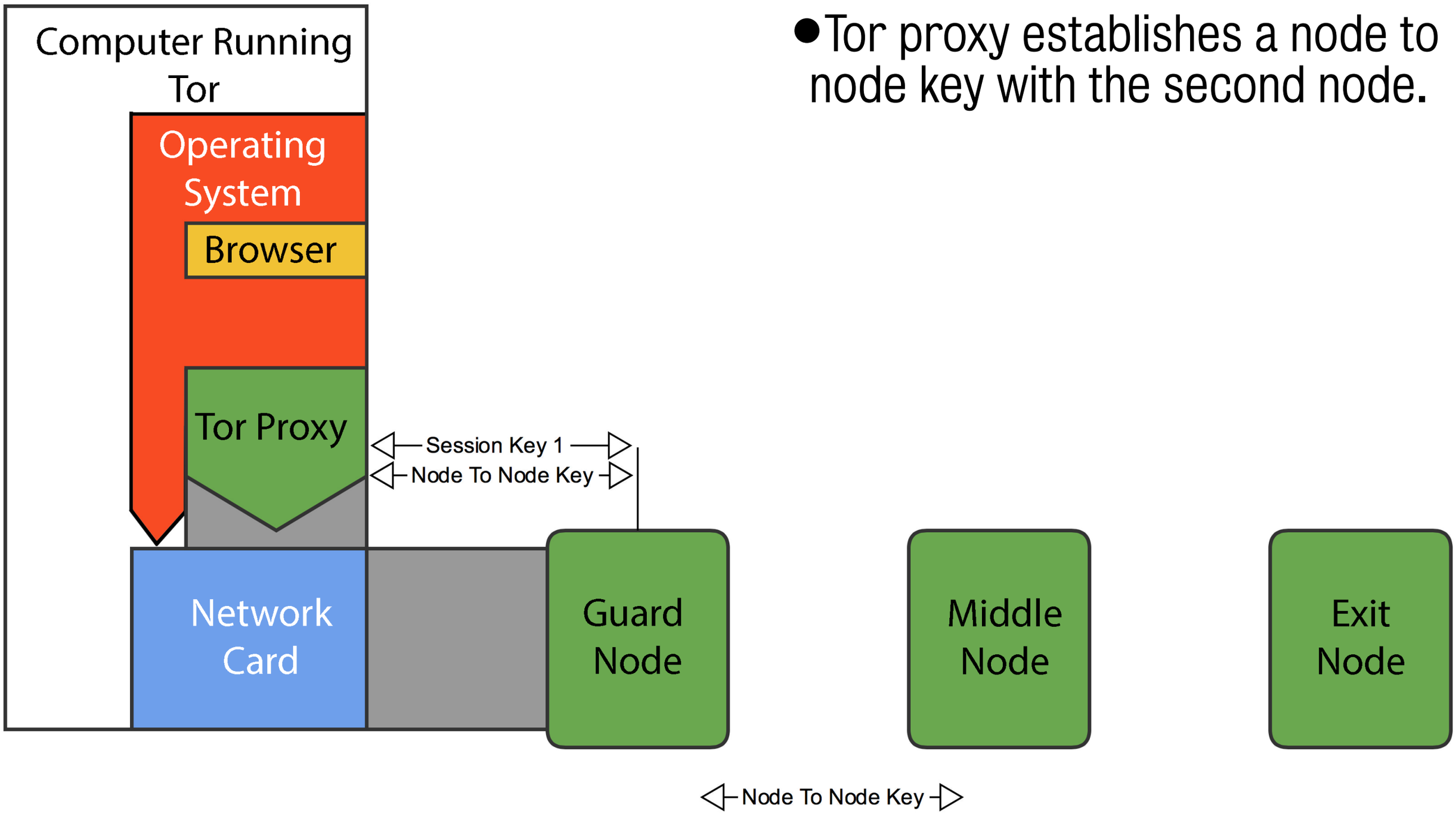
- Once the node to node key is established, Tor proxy establishes a symmetric session key with the first node.

Tor circuit setup



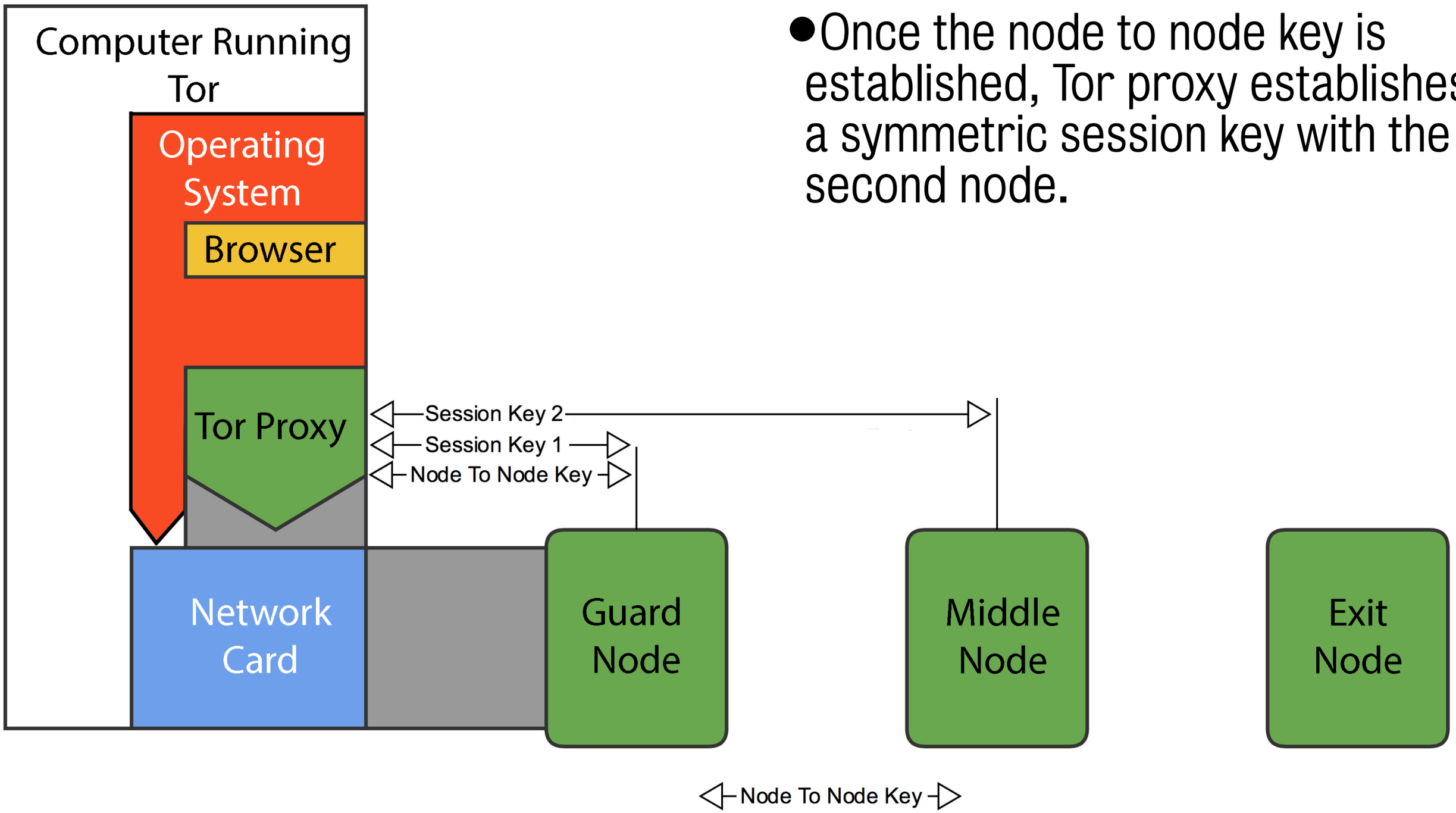
- The Tor proxy establishes a circuit using the session key with the first node.

Tor circuit setup



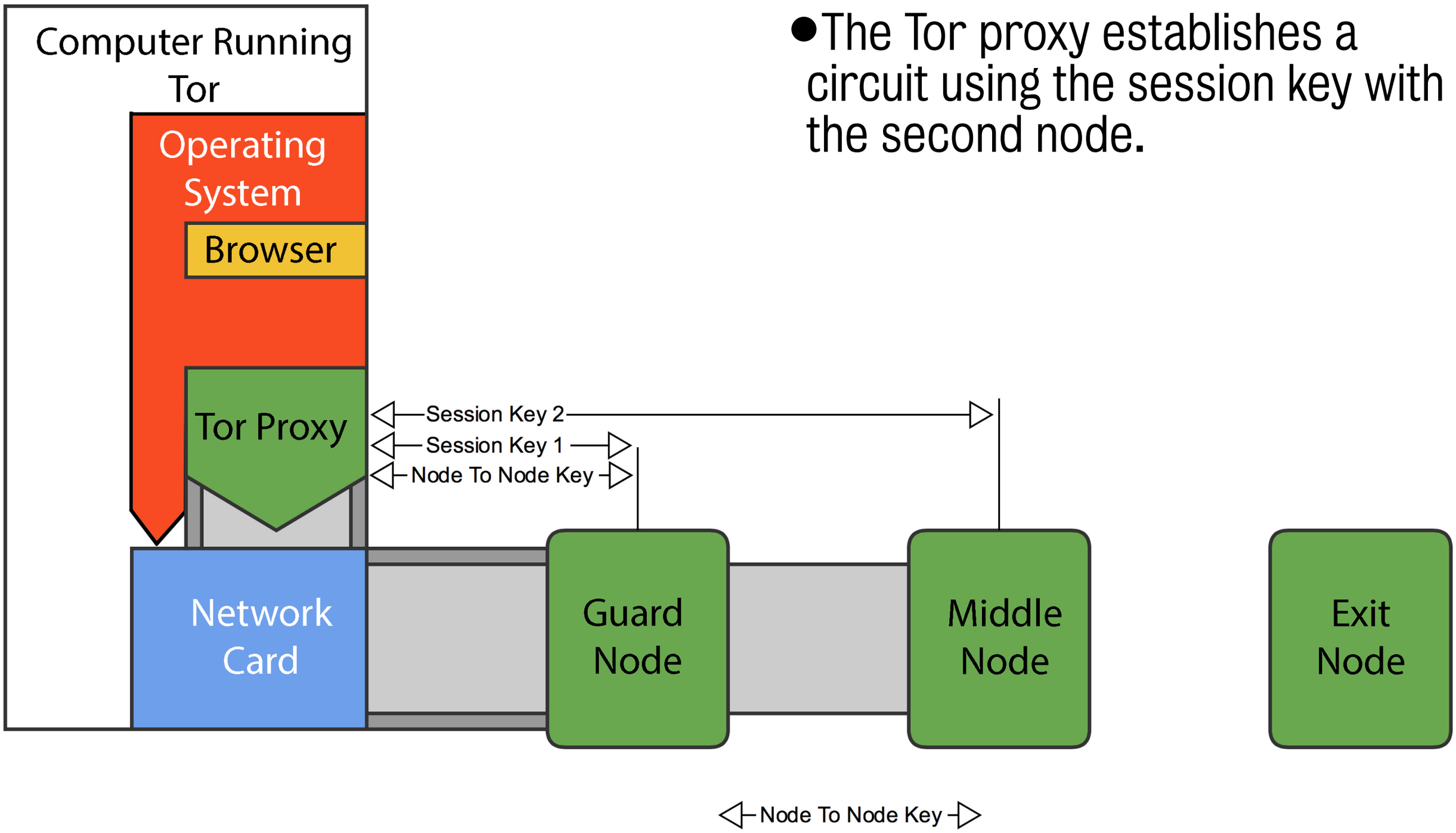
- Tor proxy establishes a node to node key with the second node.

Tor circuit setup



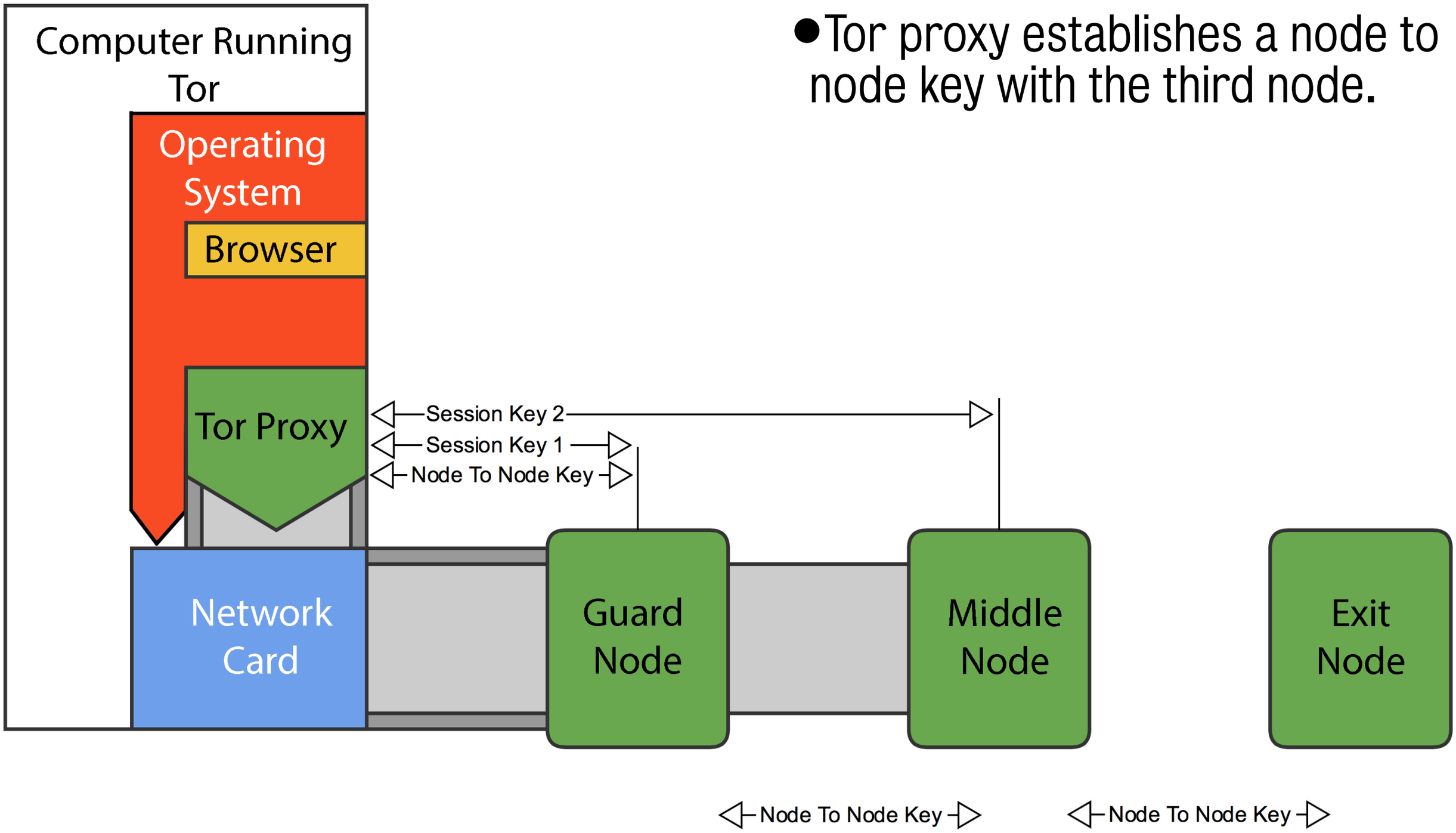
- Once the node to node key is established, Tor proxy establishes a symmetric session key with the second node.

Tor circuit setup



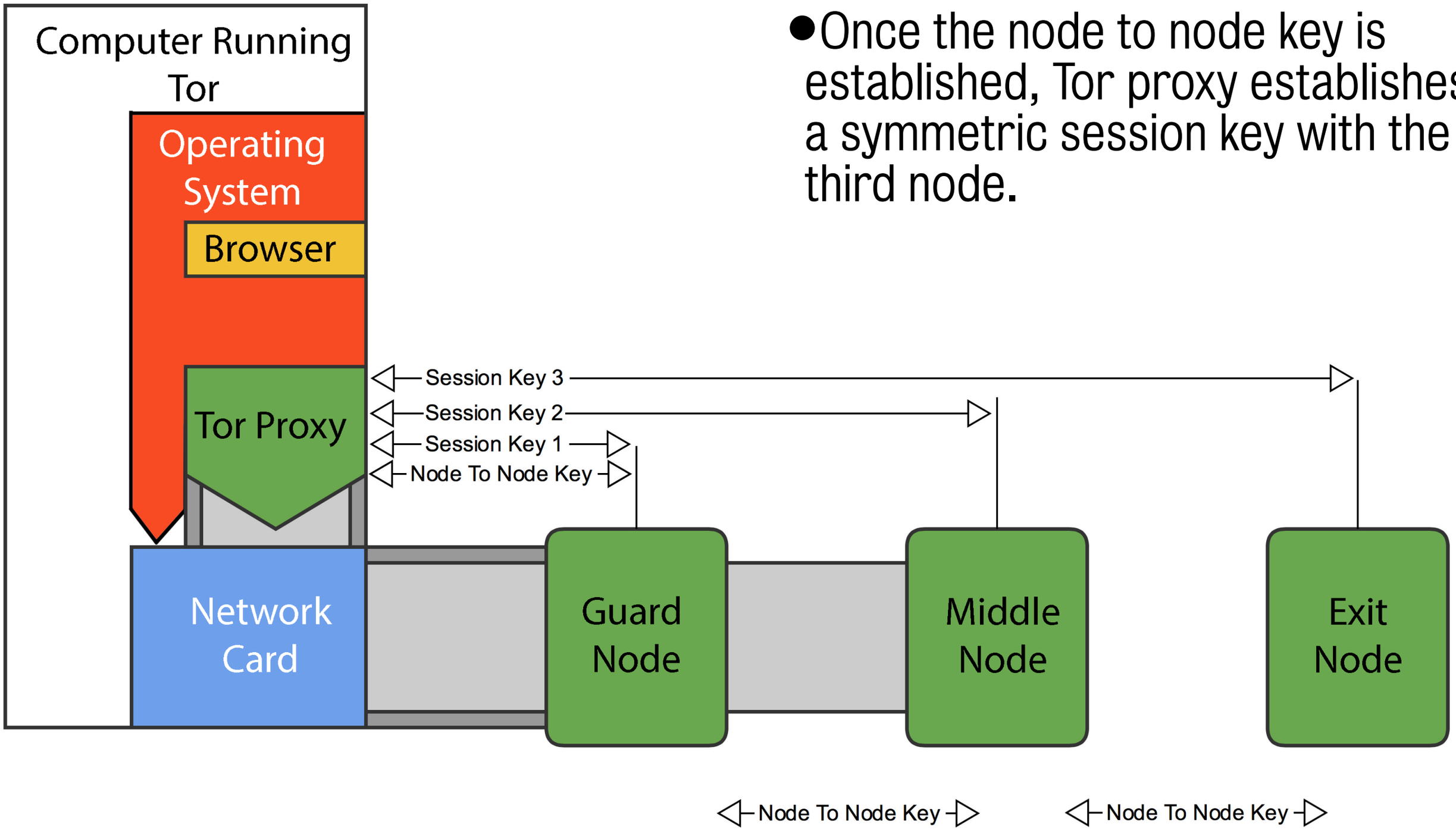
- The Tor proxy establishes a circuit using the session key with the second node.

Tor circuit setup



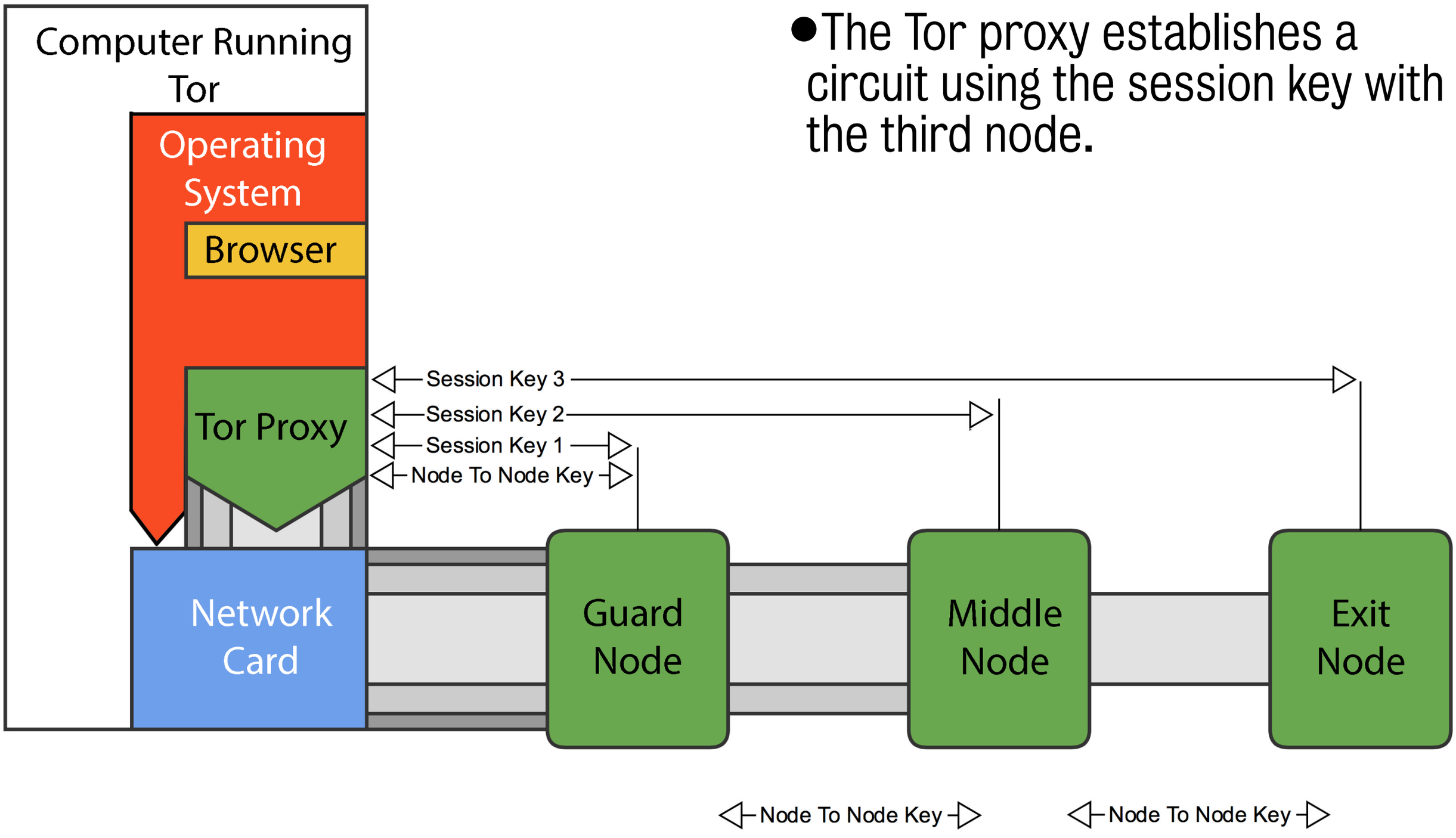
- Tor proxy establishes a node to node key with the third node.

Tor circuit setup



- Once the node to node key is established, Tor proxy establishes a symmetric session key with the third node.

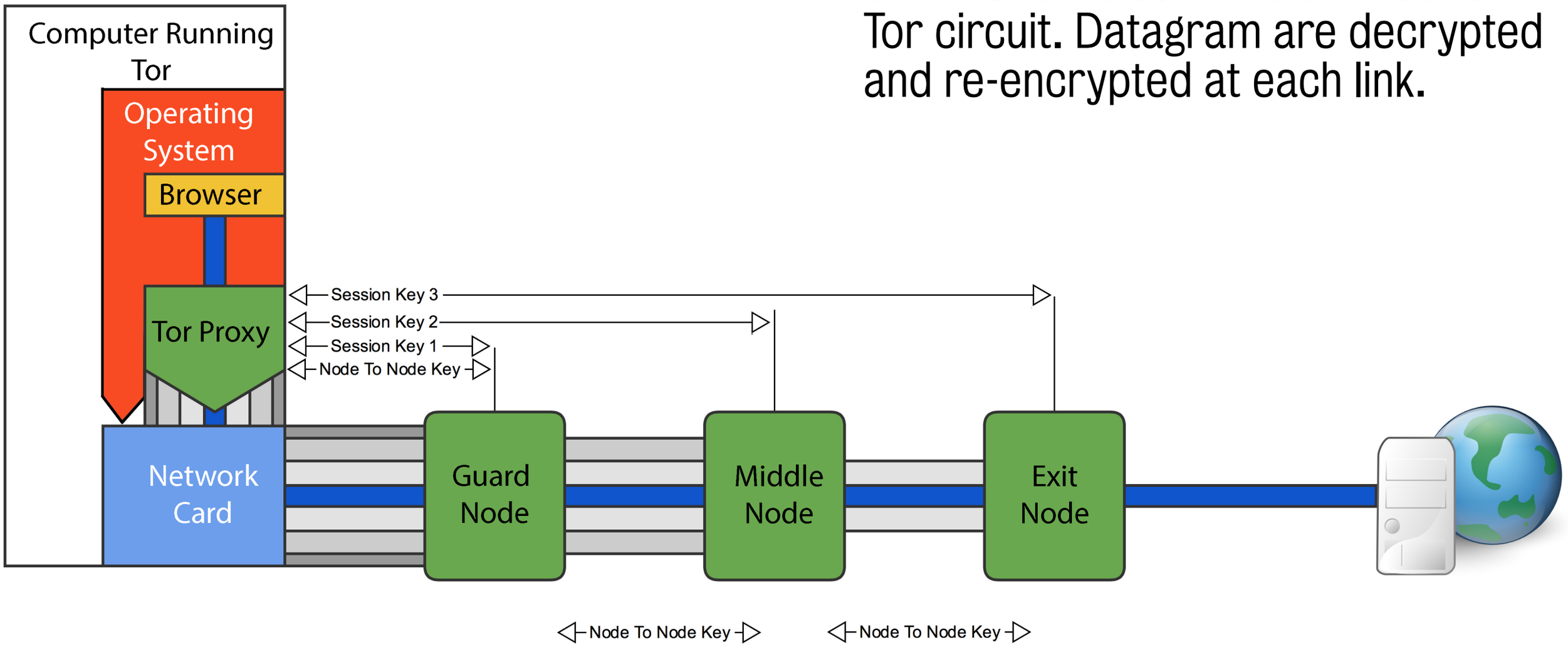
Tor circuit setup



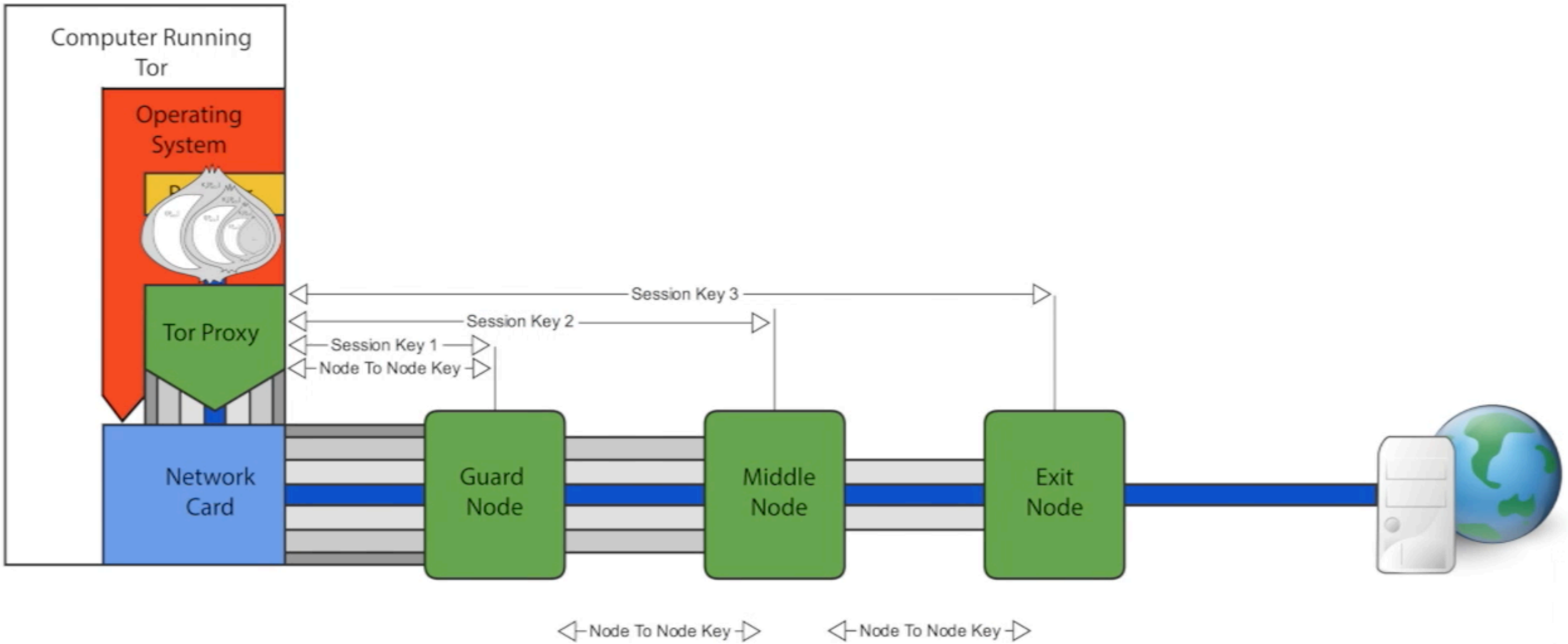
- The Tor proxy establishes a circuit using the session key with the third node.

Using a Tor circuit

- Client applications connect and communicate over the established Tor circuit. Datagram are decrypted and re-encrypted at each link.



Using a Tor circuit



Tor management issues

- **Many applications can share one circuit.**
  - Multiple TCP streams over one anonymous connection.
- **Tor router doesn't need root privileges.**
  - Encourages people to set up their own routers.
  - More participants = better anonymity for everyone.
- **Directory servers.**
  - Maintain lists of active onion routers, their locations, current public keys, etc.
  - Control how new routers join the network.
    - Sybil attack : attacker creates a large number of routers.
  - Directory servers' keys ship with Tor code.

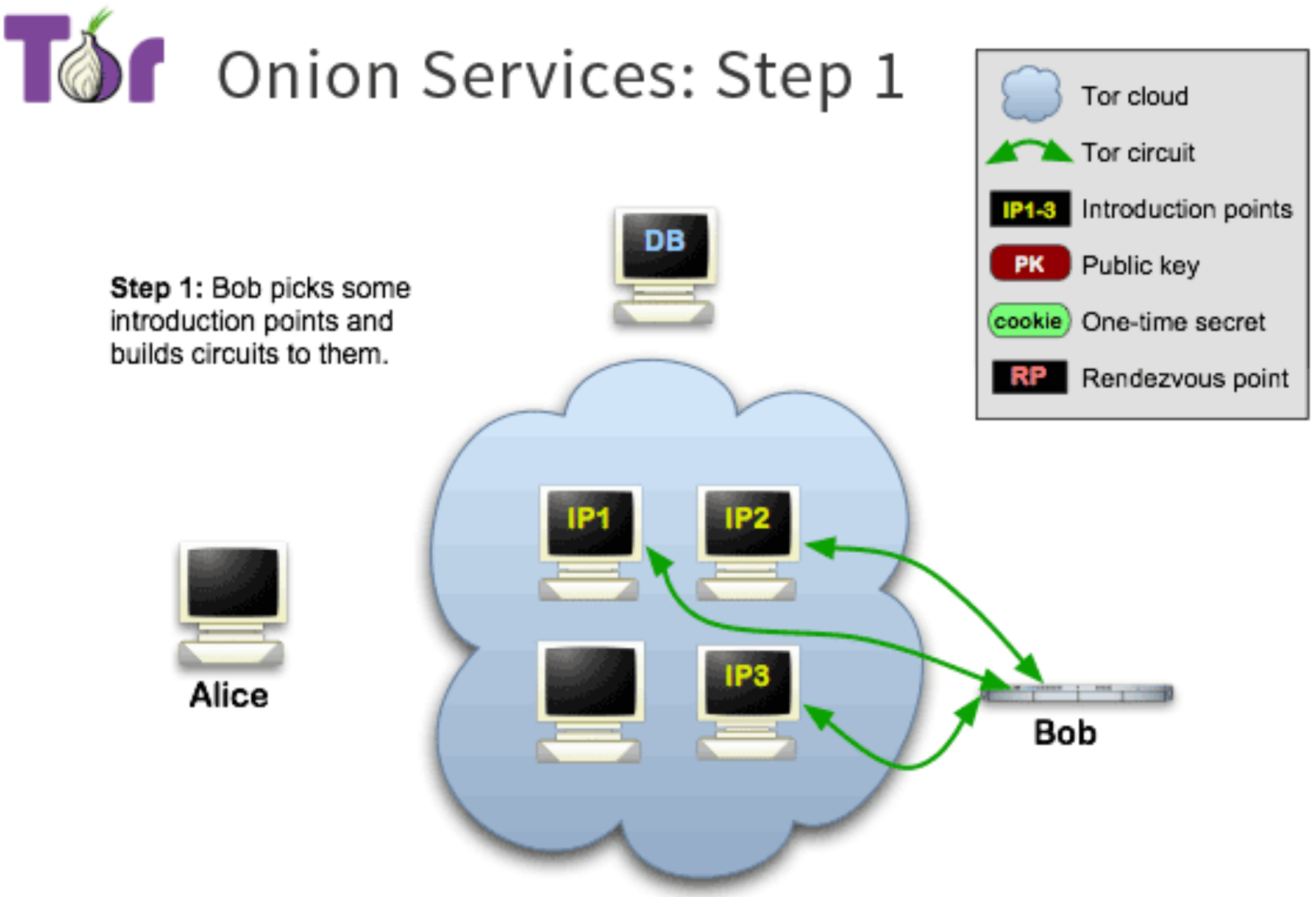
Location hidden servers

- Goal: deploy a server on the Internet that anyone can connect to without knowing where it is or who runs it.
- Accessible from anywhere.
- Resistant to censorship.
- Can survive a full-blown DoS attack.
- Resistant to physical attack.
  - Can't find the physical server!

Read up on: Silk Road, Darknet, Deep Web, etc.

Mr Robot: TOR

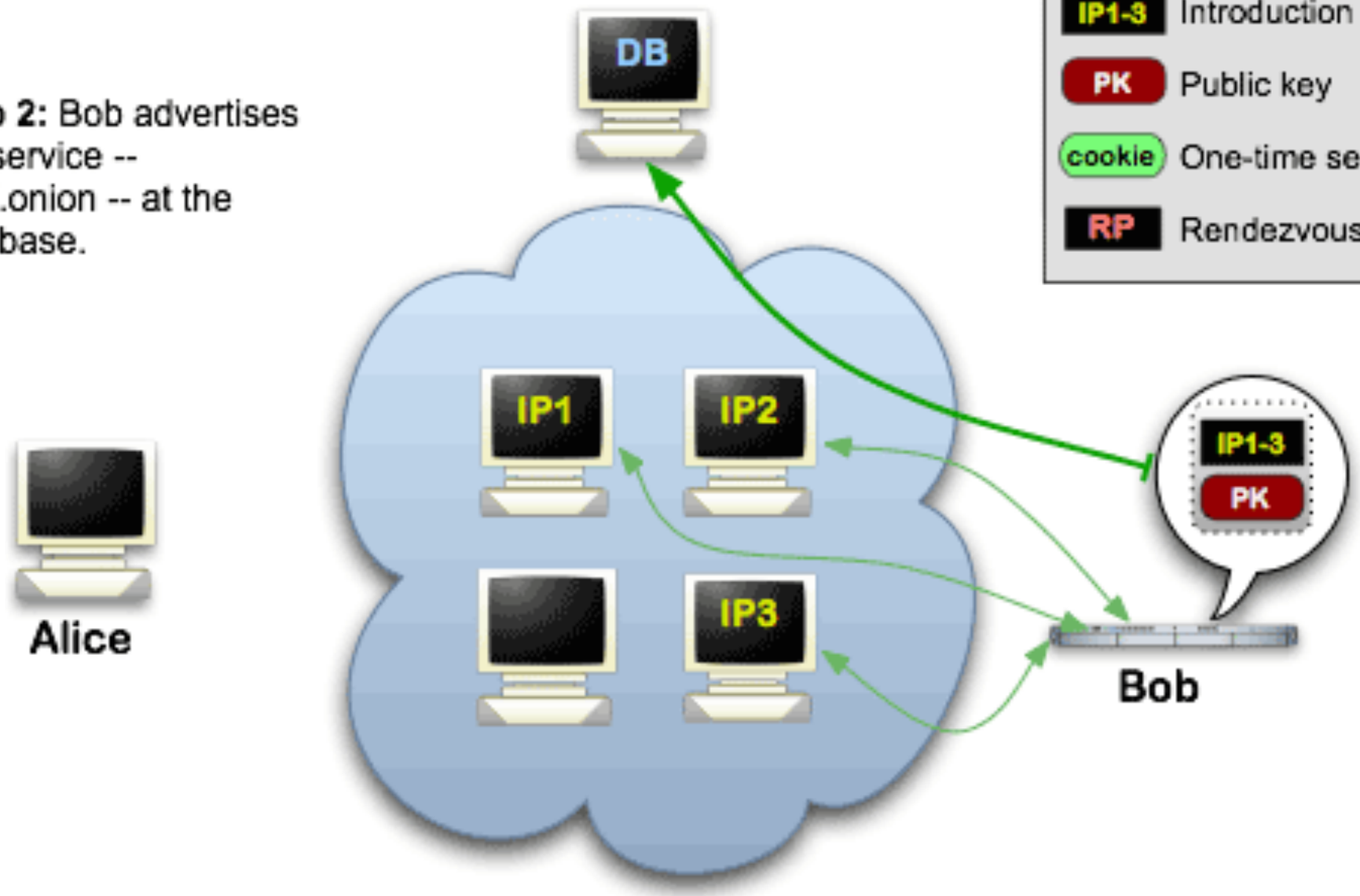
Creating a location hidden server



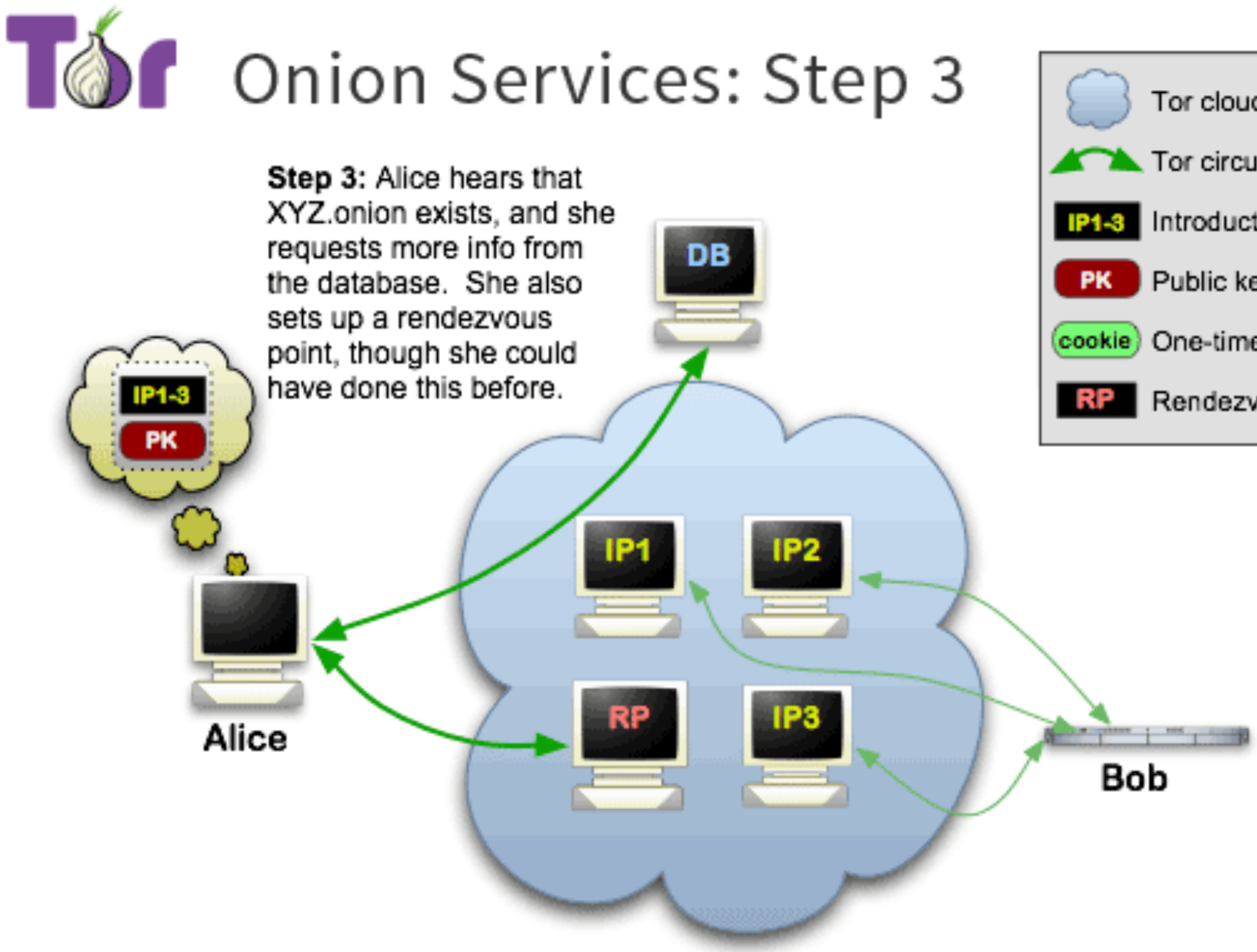
Creating a location hidden server

Onion Services: Step 2

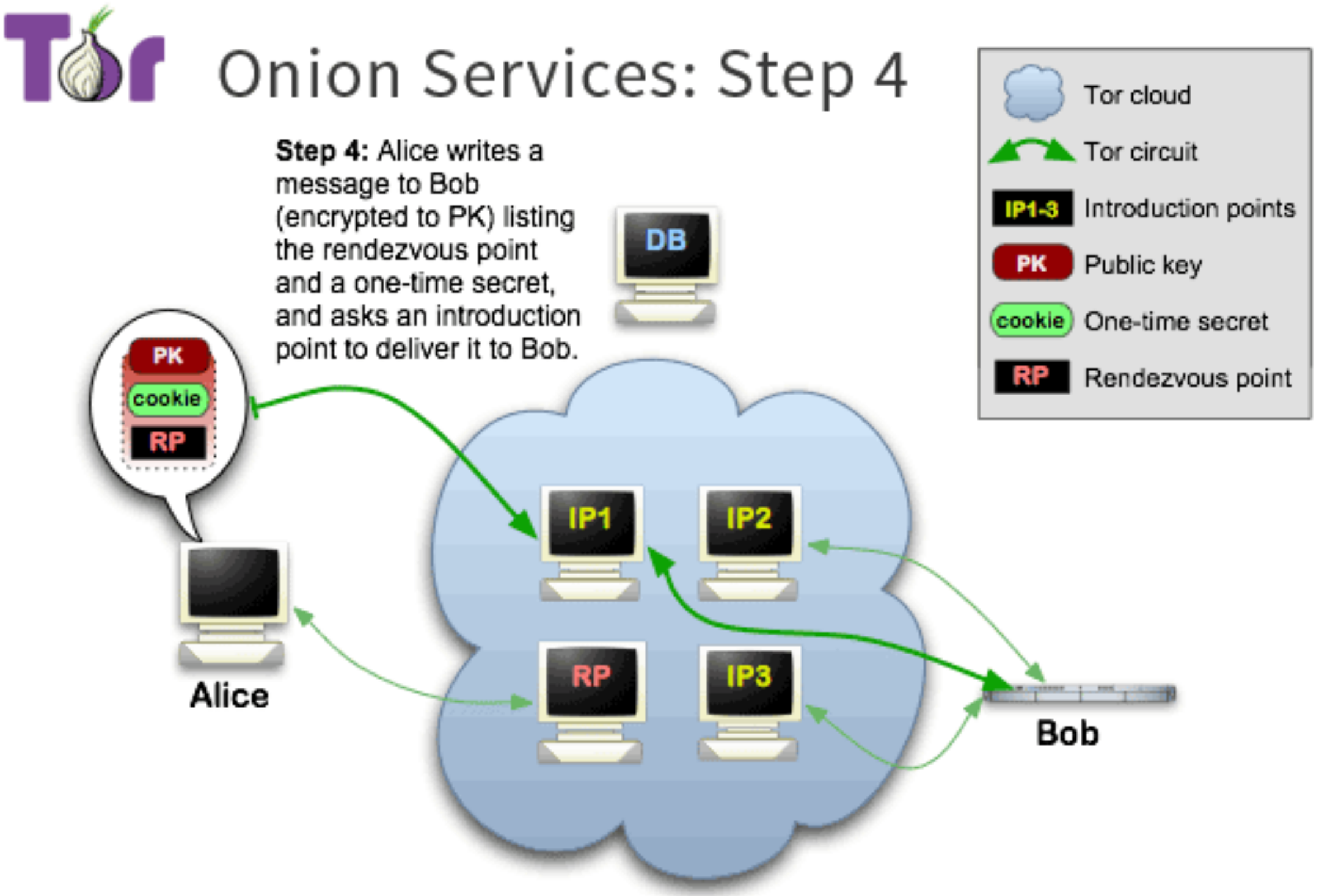
Step 2: Bob advertises his service -- XYZ.onion -- at the database.



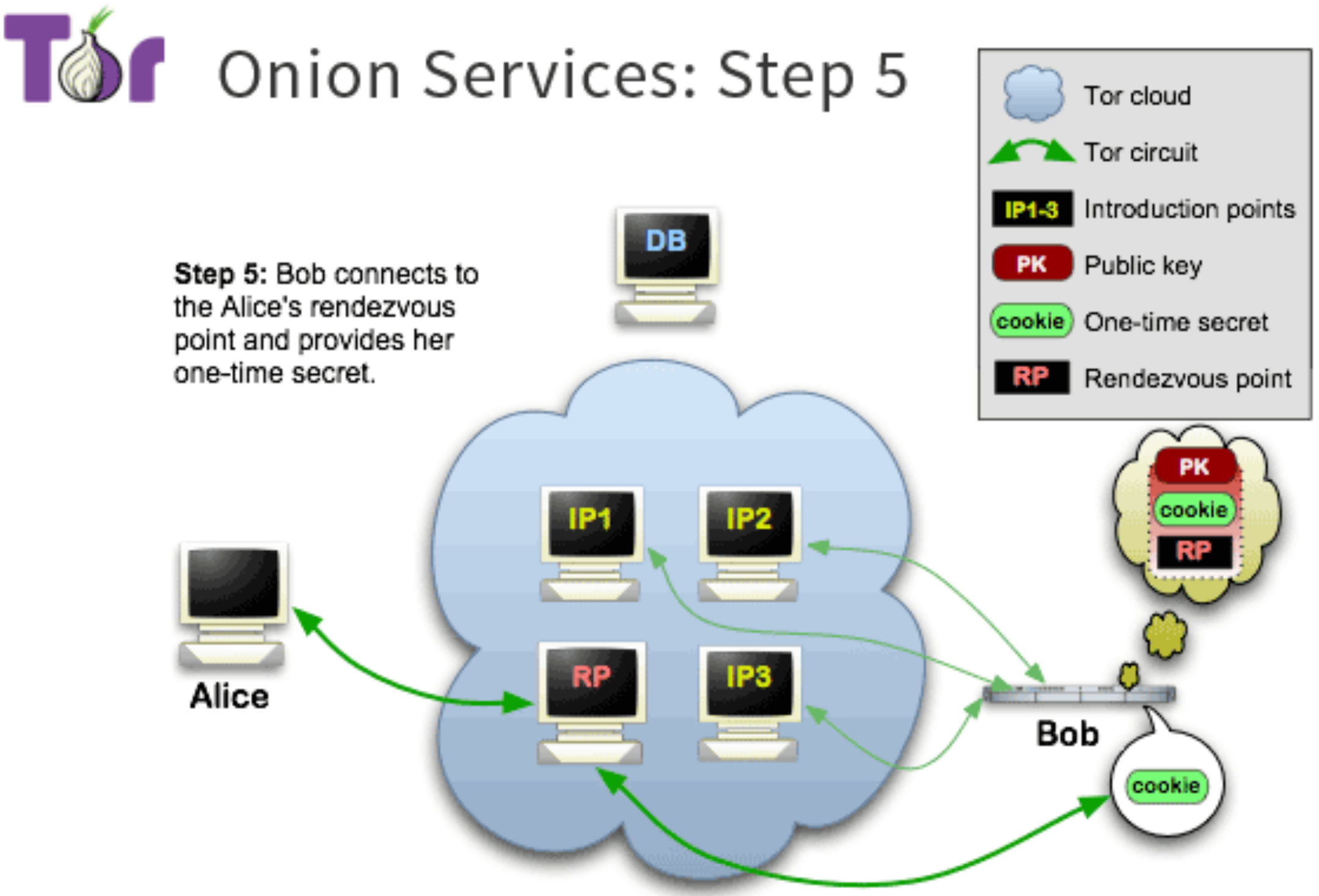
Creating a location hidden server



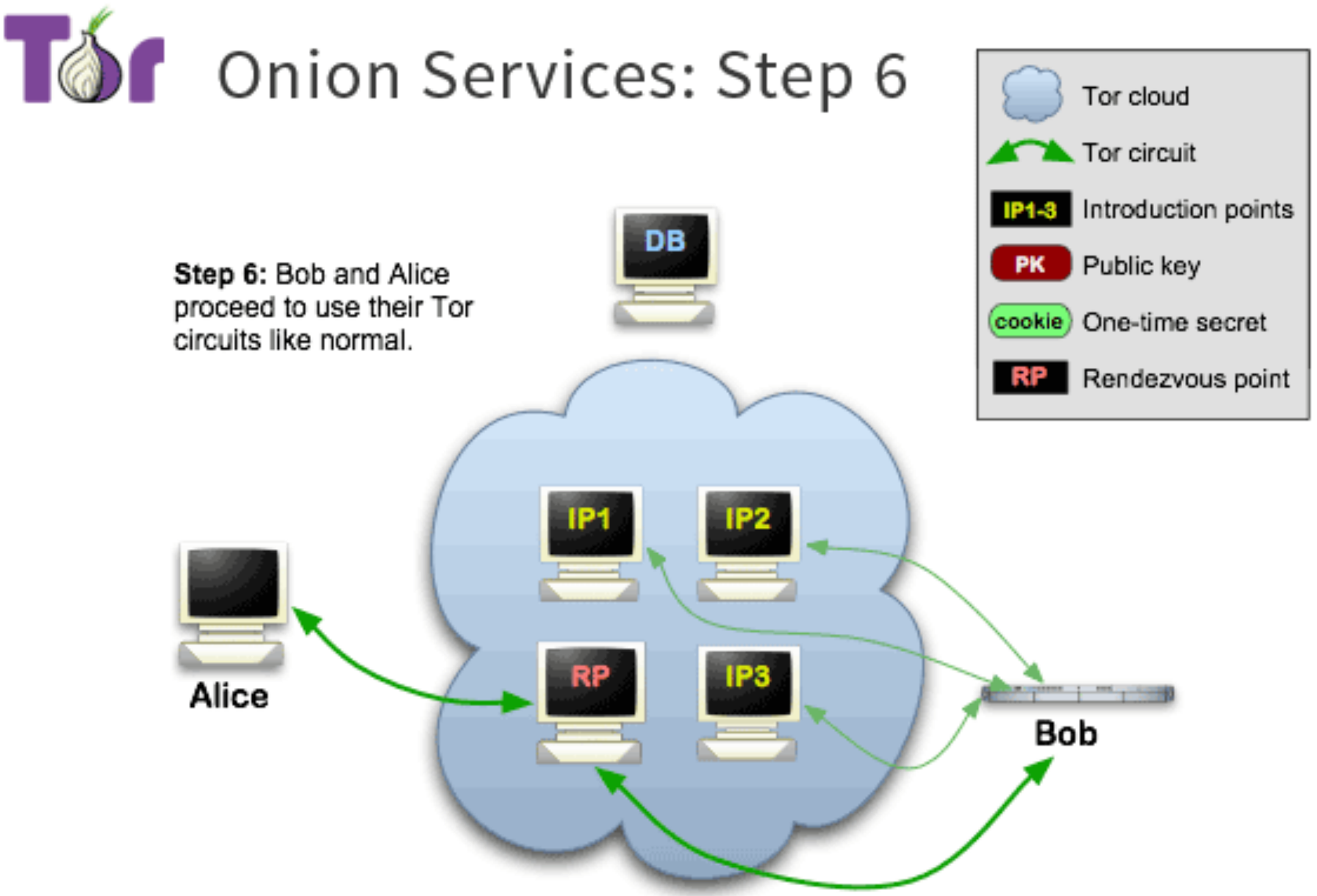
Creating a location hidden server



Creating a location hidden server



Creating a location hidden server



Deployed anonymity systems

- Tor (http://tor.eff.org)
  - Overlay circuit-based anonymity network.
  - Best for low-latency applications such as anonymous Web browsing.
- Mixminion (http://www.mixminion.net)
  - Network of mixes.
  - Best for high-latency applications such as anonymous email.
- JonDo (https: //anonymous-proxy-servers.net/en/jondo.html)
  - IP changer proxy tool for anonymous surfing, anonymous e-mail, chats and other purposes.
- Free Haven project has an excellent bibliography on anonymity.
  - http://www.freehaven.net/anonbib

Dining cryptographers

Dining Cryptographers (DC)

- Clever idea how to make a message public in a perfectly untraceable manner
  (David Chaum. “The dining cryptographers problem: unconditional sender and recipient untraceability”. Journal of Cryptology, 1988).

- Guarantees information-theoretic anonymity for message senders.
  - This is an unusually strong form of security: defeats adversary who has unlimited computational power.
- Difficult to make practical.
  In group of size N, need N random bits to send 1 bit.

Three-person DC Protocol

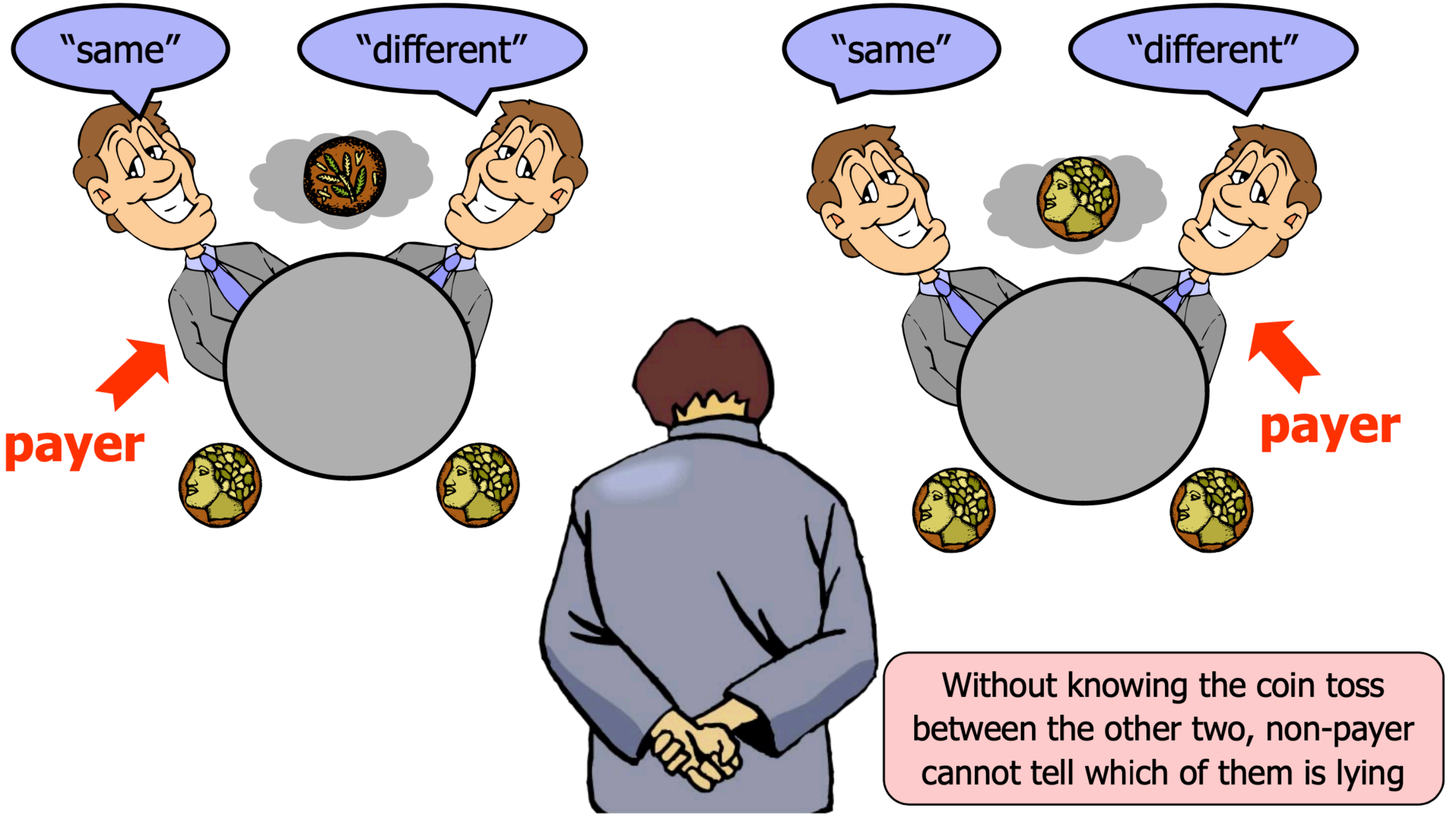
- Three cryptographers are having dinner.
- Either NSA is paying for the dinner, or one of them is paying, but wishes to remain anonymous.

1. Each diner flips a coin and shows it to his left neighbour.
   - Every diner will see two coins: his own and his right neighbour's.
2. Each diner announces whether the two coins are the same.

   If he is the payer, he lies (says the opposite).
3. Odd number of "same" $\Rightarrow$ NSA is paying;
   even number of "same" $\Rightarrow$ one of them is paying.

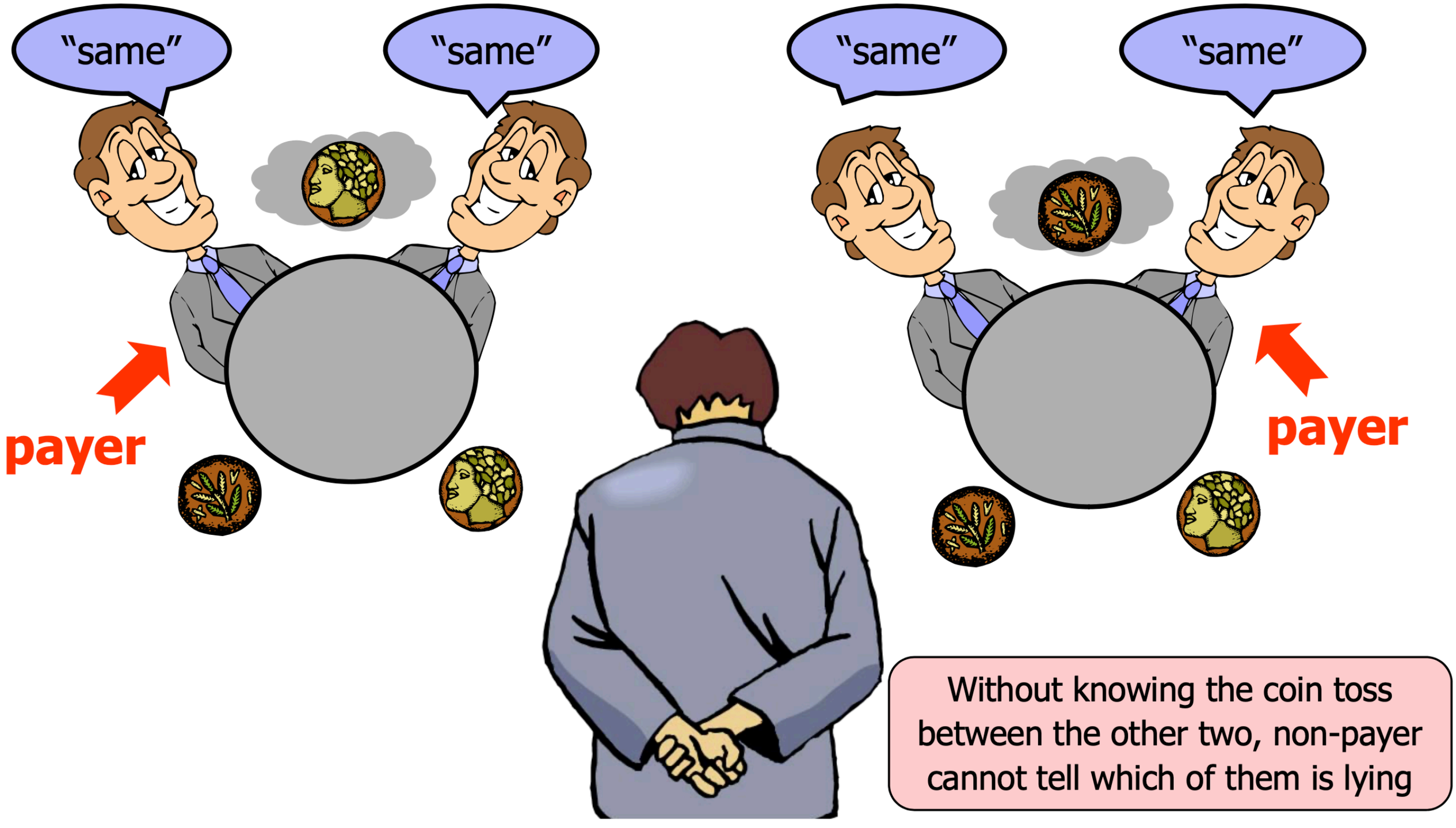
- But a non-payer cannot tell which of the other two is paying!

Dining Cryptographers (DC)



Without knowing the coin toss between the other two, non-payer cannot tell which of them is lying

Dining Cryptographers (DC)



Without knowing the coin toss between the other two, non-payer cannot tell which of them is lying

The Three-person DC Protocol in full

- So, summarising, the Three-person DC Protocol is as follows:

1. Each of the three cryptographers C_1, C_2 and C_3 flips an unbiased coin keeping the result b_i secret ($i \in \{1, 2, 3\}$).

2. Each cryptographer whispers the result b_i in the ear of the person to their immediate left.

3. Each cryptographer computes $d_i = b_i \oplus b_{i-1}$ where
   - b_i is the cryptographer's own coin flip and
   - b_{i-1} is the coin flip of the person on the right.

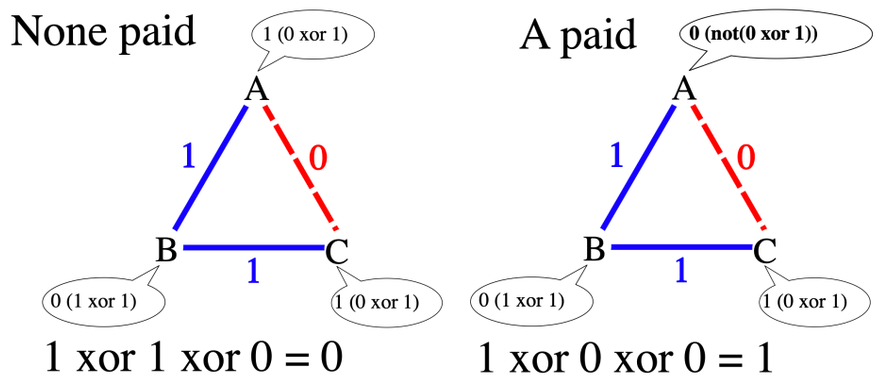
- Note that $d_0 = d_3$ and $b_0 = b_3$.

4. A cryptographer that did not pay for the meal announces her own d_i.
   A cryptographer that did pay for the meal lies by announcing the negation of d_i , i.e. $d_i \oplus 1$.

Note that the protocol would work as well if they all sent the bit to their right neighbour.

Proofs

Let A_1, A_2 and A_3 be the three values announced by the three cryptographers, where $A_j = d_j$ if cryptographer i did not pay for the meal and $A_j = d_j \oplus 1$ if cryptographer i did pay for the meal.



- We can show that if the XOR of the values announced by the three cryptographers is 0, then someone else paid for the meal, else one of the cryptographers paid for it.
  - If none of the cryptographers paid for the meal, then the XOR of the values they announced is

$$\begin{aligned} A_1 \oplus A_2 \oplus A_3 &= d_1 \oplus d_2 \oplus d_3 \\ &= (b_1 \oplus b_0) \oplus (b_2 \oplus b_1) \oplus (b_3 \oplus b_2) \\ &= (b_1 \oplus b_3) \oplus (b_2 \oplus b_1) \oplus (b_3 \oplus b_2) \\ &= 0 \end{aligned}$$

If cryptographer i paid for the meal, then i announces $d_j \oplus 1$. Say, for instance, that i = 3. Hence, the XOR of the values announced by the cryptographers is

$$\begin{aligned} A_1 \oplus A_2 \oplus A_3 &= d_1 \oplus d_2 \oplus (d_3 \oplus 1) \\ &= (b_1 \oplus b_3) \oplus (b_2 \oplus b_1) \oplus (b_3 \oplus b_2 \oplus 1) \\ &= 1 \end{aligned}$$

Proofs

- We can argue that if the answer (i.e., the XOR of the values announced by the three cryptographers) is 1, then any of the cryptographers who did not pay for the meal has no idea which of the other two did.
  - Let cryptographer 3 be the one who paid. Then cryptographer 1 sees that $A_1 \oplus A_2 \oplus A_3 = 1$ (as in the previous subquestion), and knows that
    - either cryptographer 2 announced $(d_2 \oplus 1) = (b_2 \oplus b_1 \oplus 1)$
    - or cryptographer 3 announced $(d_3 \oplus 1) = (b_3 \oplus b_2 \oplus 1)$

    but has absolutely no way to distinguish the two cases as the hidden coin flips are fair — both possibilities are equally likely..

The same holds of cryptographer 2.

Superposed sending

- This idea generalises to any group of size N.
  - For each bit of the message, every user generates 1 random bit and sends it to 1 neighbour.
- Every user learns 2 bits (his own and his neighbour's).
- Each user announces own bit XOR neighbour's bit.
- Sender announces own bit XOR neighbour's bit XOR message bit.
- XOR of all announcements = message bit.
  - Every randomly generated bit occurs in this sum twice (and is canceled by XOR), message bit occurs once.

PRIVACY

Privacy

- According to Oxford English Dictionary
  A state in which one is not observed or disturbed by others.

- In IT context this can be specialised as follows:
  Anonymity: unobservability of our actions when they occur.
  Data Protection: ensuring that our collected data is not distributed and used in undesired ways.

- As unobservability/anonymity is more-or-less understood, let us now focus on data protection: controlling the dissemination and usage of sensitive personal data.

Acceptable usage? How do we formulate this? Enforcement?

Privacy requirements and policies

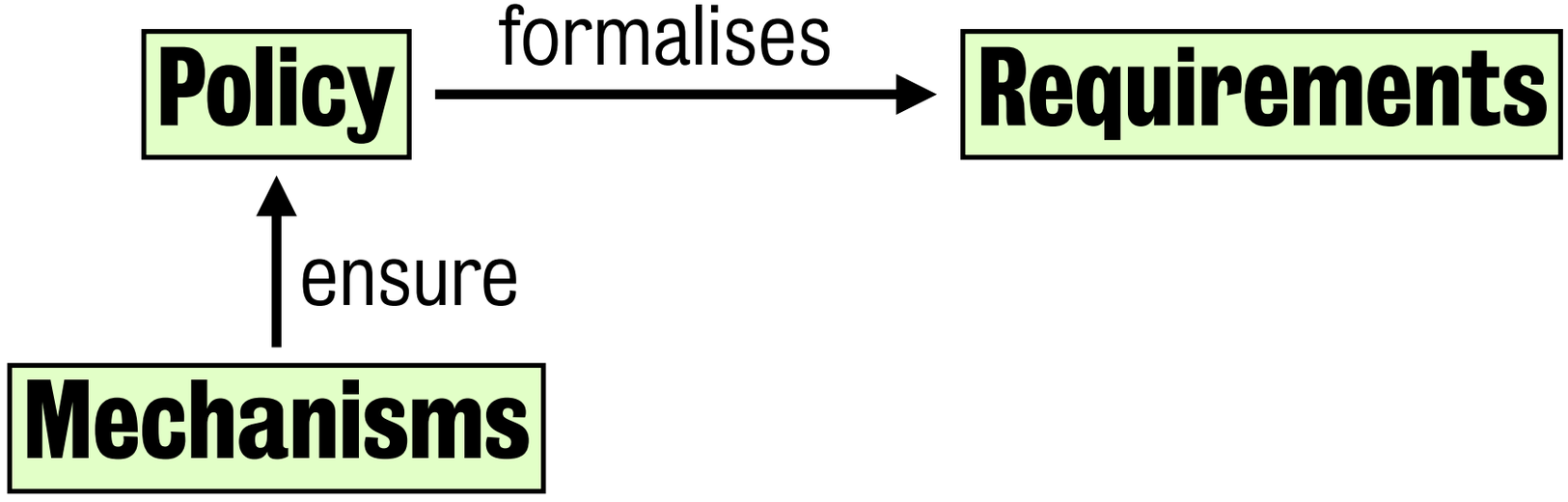
- Privacy requires that data usage respects purpose.
- A privacy policy specifies how data may be used, under which conditions, and what obligations this entails

- We use personal information that you provide to register you in programs, create and maintain accounts, ship products, ...
- We share your personal information with certain business affiliates.
- We and our subsidiaries may send you email with offers about products unless you indicate you do not want to receive them.
- Before processing a minor's order, parental consent must be given.
- Data associated with inactive accounts will be deleted after one year.

Examples of privacy requirements

- Access control requirements.
  Only X may access Y.
- Actions required before the access.
  Gathering owner
- Actions that must be performed within a certain time period.
  Informing data owner whenever the data is used.
- Restrictions on the further distribution of the data.
  Own use only.
- Restriction of purposes for which data may be used.
  Statistical purposes only.
- Limitations on retention time.
  Delete after 7 days.
- Mandatory use of protection mechanisms.
  Encrypt backups.
- Duties of keeping the data up-to-date.
  Update every 30 days.

What are the problems?



- How do we formalise requirements?
- How do we enforce them?

- Let's briefly consider current standards and limitations.

What are the problems?

- Normally just unstructured text.
- Machine-friendly forms being standardised.
  Example: Platform for Privacy Preferences (P3P) of W3C.
- P3P allows web sites to communicate their privacy policies in computer readable format (XML).
  Data collected, purpose, how long it is retained, etc.
- Enables tools (in browsers or stand-alone) to be developed that:
  - summarise privacy policies,
  - compare policies with user preferences, and
  - advise and alert users.

An example

- We do not collect any information from site visitors except the information contained in standard web server logs (your IP address, browser information, etc.). This log information will be used only by us for web site administration. It will not be disclosed unless required by law. We may retain these log files indefinitely. Please direct questions about this privacy policy to privacy@company.com.

```
<POLICY discuri="http://company.com/privacy.html" name="policy">
  <ENTITY>
    <DATA-GROUP>
      <DATA ref="#business.contact-info.online.email">privacy@company.com</DATA>
      <DATA ref="#business.contact-info.online.uri">http://company.com/ </DATA>
    </DATA-GROUP>
  </ENTITY>
  <STATEMENT>
      <CONSEQUENCE>We keep standard web server logs.</CONSEQUENCE>
      <PURPOSE><admin/><current/><develop/></PURPOSE>
      <RECIPIENT><ours/></RECIPIENT>
      <RETENTION><indefinitely/></RETENTION>
  </STATEMENT>
</POLICY>
```

P3P and other languages

- There are other "privacy" languages. E.g.,
  - Security Assertion Markup Language (SAML).
  - Microsoft security metadata to SOAP.
  - IBM's Enterprise Privacy Authorisation Language (EPAL).
  - Permission-based attributes in Liberty Alliance.
- But relationship to privacy is often poorly understood.
  In most cases, privacy = anonymity (via pseudonyms).
- P3P attempts to do more here.

Semantics? Enforcement?

Privacy enforcement mechanisms

- Current mechanisms based on company/legal processes.
- Example: Online Privacy Alliance.
  - 50+ large ("global players") US companies.
  - Set of guidelines addressing collection, use and disclosure of individually identifiable information and data, as well as special measure to protect privacy of children on-line.
  - Self-enforcing, with third-party monitoring and audit.
- While a good first step, such best efforts have limited effectiveness.
  - Analogous to self restraint in conventional (CIA) security.
  - Policies prone to misunderstanding, misuse, and abuse.

Research on this is ongoing...

LAB time (EPISODE 9)

- TODO