WebRTC IP address information leakage

Current situation

ICE candidates exposure

- ICE candidates include host, reflexive and relay candidates.
  - If multihomed device, more info is made known.
  - Can circumvent VPN services in certain scenarios exposing host, public _and_ VPN IP addresses.
- Exposed to Web client JS.
- Exposed to STUN/TURN servers.
- Exposed to HTTP servers for peer-2-peer connection establishment, if separate from web site.

Example Security and privacy implications

- Host/reflexive addresses can be resolved into phone numbers in mobile networks and other kinds of ISP networks when user has a bundled telephony and Internet subscription service package.
- Local/reflexive addresses may be used for device fingerprinting.
- Users striving for anonymity using VPN can be exposed.
- Leveraging PeerConnection data channel statistics, the user network can be profiled.

nice.com and not-so-nice.com

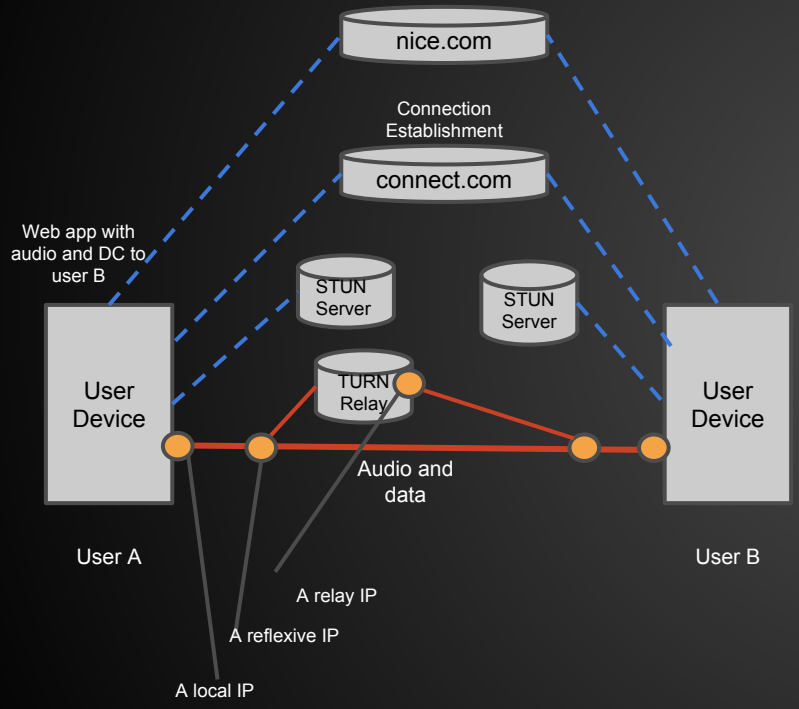
There are websites with clear malicious intent and web sites who do respect and concern themselves with user security and privacy. We denote these not-so-nice.com and nice.com respectively.

The user should get help (from the UA) in managing security and privacy but nice.com providers should also have tools to enable them to protect their users interests.

nice.com, basic case

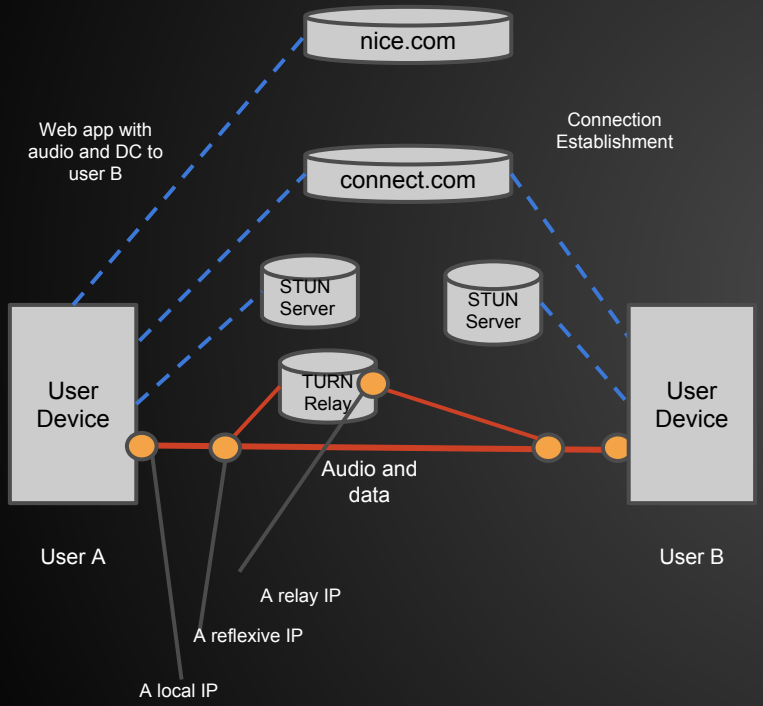
Do we need this?

Nice.com, delegated connect



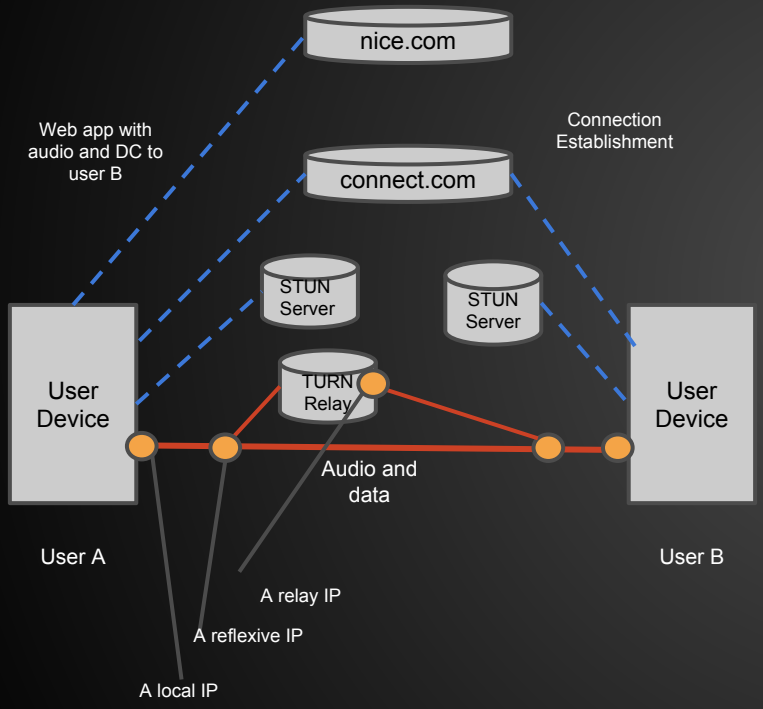
1. Web page from nice.com includes possibility to have an RTC connection to User B.
2. A's ICE's candidates exchanged via connect.com.
3. Data and audio exchanged peer-to-peer or via TURN relay server.

Potential leakage if no measures



1. nice.com connection:
   a. nice.com sees A's "source" IP address.
   b. HTTP MITMs sees the same as nice.com.
2. connect.com connection:
   a. connect.com can read SDP blobs of A and B.
   b. connect.com sees A's "source" IP address.
   c. HTTP MITMs can see either A or B source IP address as well as SDP blobs of A and B.
3. STUN server:
   a. Sees all ICE candidates as well as A's source IP address of ICE messages.
   b. MITMs sees same as STUN server.
4. TURN relay server:
   a. Sees A's and B's source and target addresses.
   b. MITMs sees same as relay server.
5. Data and audio exchanged peer-to-peer:
   a. MITMs sees source and IP addresses.
6. Web client leakage via cross site content injection, poorly written or malicious web app code or backends hacked.

Closing the lid #1: https:// and CSP.



1. https://nice.com
   a. nice.com still sees A's "source" IP address.
   b. MITMs still sees A's source address.

2. https://connect.com
   a. connect.com can read SDP blobs of A and B.
   b. connect.com sees A's and B's source IP address.
   c. *"HTTP MITMs can see either A or B source IP address as well as SDP blobs of A and B"* is partly blocked. MITMs can still see A's and B's source IP address.

3. STUN server:
   a. Sees all ICE candidates as well as source address of ICE messages.
   b. MITMs sees same as STUN server.

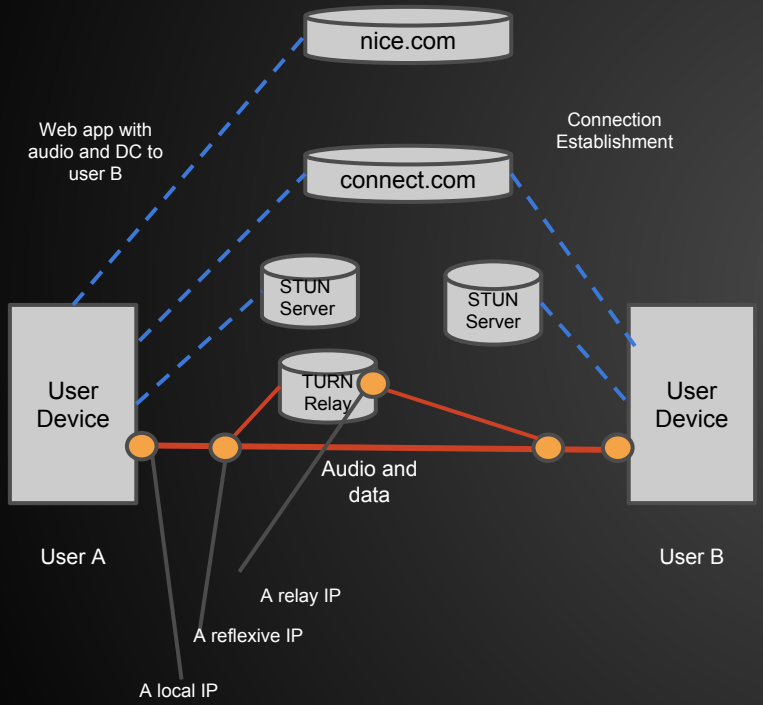
4. TURN relay server:
   a. Sees A's and B's source and target addresses
   b. MITMs sees same as relay server.

5. Data and audio exchanged peer-to-peer.
   a. MITMs sees source and IP addresses.

6. Web client leakage:
   Currently, some UA's will block block http to other origins if active content such as XHR. A hacked web client or JSL may however connect to other origins with https.

https prevents MIMTs but to lock down the client, more measure may be needed, e.g. use of CSP.

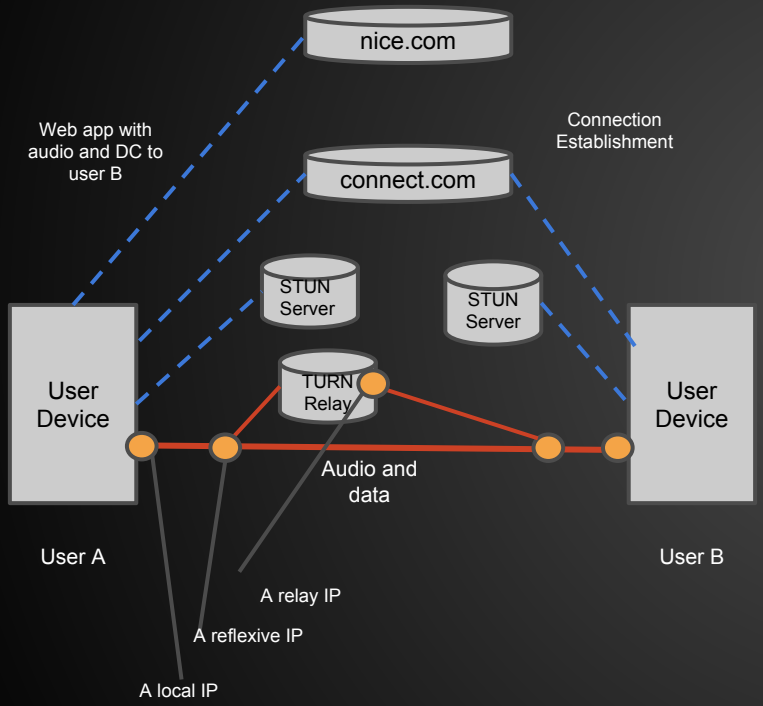
Closing the lid #2: stuns and turns



1. https://:nice.com
   a. Nice.com sees A's "source" IP address.
   b. MITM's still sees A's source address.
2. https://connect.com
   a. connect.com can read SDP blobs of A and B.
   b. *"HTTP MITMs can see either A or B source IP address as well as SDP blobs of A and B"* is blocked.
3. stuns to STUNS server
   a. STUN server sees all ICE candidates as well as source address of ICE messages.
   b. MITMs sees same as STUN server.
4. https:// or turns to TURN
   a. Sees A's and B's source and target addresses.
   b. MITMs sees same as relay server.
5. Data and audio exchanged peer-to-peer.
   a. MITMs sees source and IP addresses.
6. Web client leakage via content injection or poorly written or malicious web app code.
   a. Question: does using https:// to nice.com lock down the client for xross-site scripting? If not, then other actions required.

stuns and turns secures the communication to STUN/TURN server. Should be MUST if web app loaded over HTTPS (which any sensible Web IT-admin should ensure). Certificate for stuns and turns as well as the proposed origin concept is FFS.

Closing the lid #3: TURNs relay



1. https://:nice.com
   a. Nice.com sees A's "source" IP address.
   b. MITM's still sees A's source address.

2. https://connect.com
   a. connect.com can read SDP blobs of A and B.
   b. *"HTTP MITMs can see either A or B source IP address as well as SDP blobs of A and B"* is blocked.

3. Secure ICE to STUN server, use turns, stuns...
   a. Sees all ICE candidates as well as source address of ICE messages.
   b. "MITMs sees same as STUN server" is blocked.

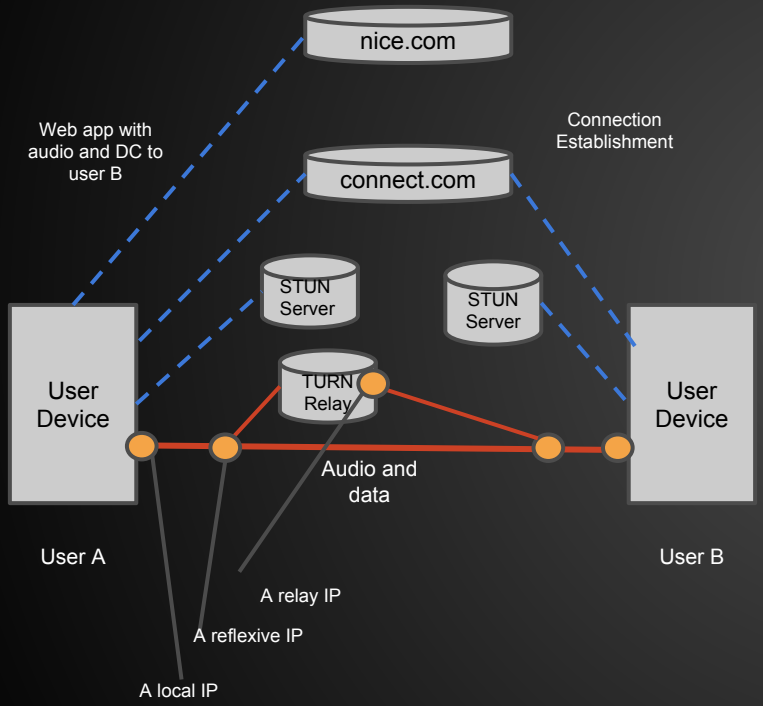
4. https:// to TURN relay server
   a. Sees A's and B's source and target addresses
   b. See's SDP blob.
   c. "MITM's sees same as relay server" is blocked.

5. TURN Relay: Data and audio exchanged peer-to-peer.
   a. MITM's sees source and IP addresses is blocked.

6. Web client leakage via content injection or poorly written or malicious web app code.
   a. Question: does using https:// to nice.com lock down the client for xross-site scripting? If not, then other actions required.

A TURN relay server block ICE MITMs but also IP leackage info to peer.

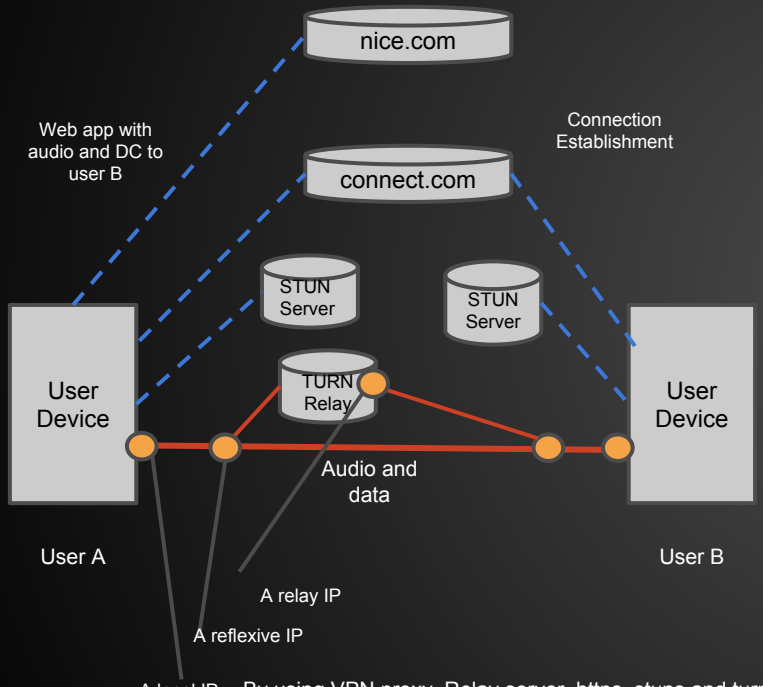
Closing the lid #4: VPN proxy



1. https:// proxy, https://:nice.com
   a. Nice.com sees A's "source" IP address.
   b. "MITM's still sees A's source address" is blocked.
2. https://connect.com
   a. connect.com can read SDP blobs of A and B.
   b. connect.com sees A's "source" IP address.
   c. *"HTTP MITMs can see either A or B source IP address as well as SDP blobs of A and B"* is blocked.
3. Secure ICE to STUN server
   a. Sees all ICE candidates as well as source address of ICE messages.
   b. "MITMs sees same as STUN server" is blocked.
4. https:// to TURN relay server
   a. Sees A's and B's source and target addresses
   b. See's SDP blob.
   c. "MITM's sees same as relay server" is blocked.
5. TURN Relay: Data and audio exchanged peer-to-peer.
   a. MITM's sees source and IP addresses is blocked.
6. Web client leakage via content injection or poorly written or malicious web app code.

Introduction of "Relay_only" mode on browsing context necessary. User setting and CSP- directive controlled.

Closing the lid #1-4, summing up

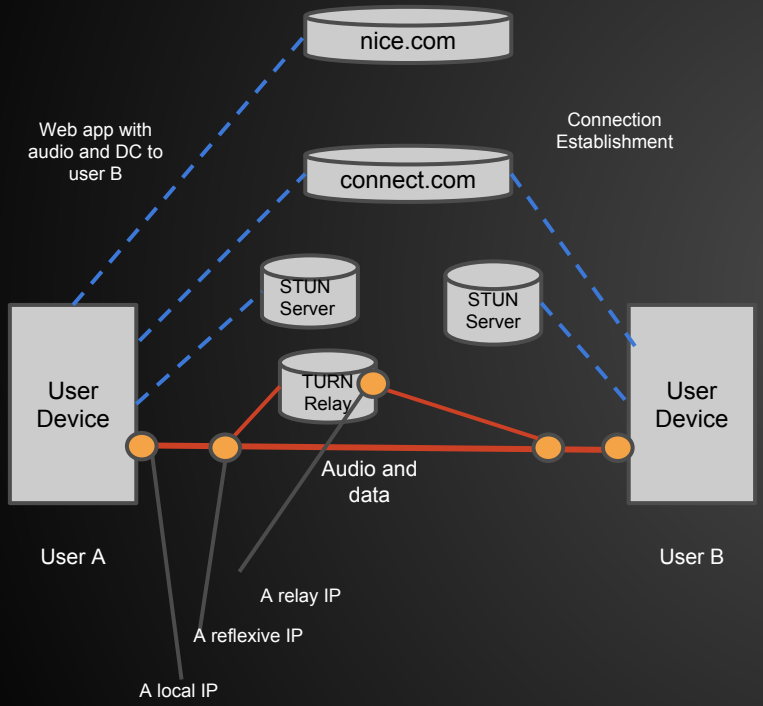


1. https:// proxy, https://:nice.com
   a. Nice.com sees A's "source" IP address.
   b. "MITM's still sees A's source address" is blocked.
2. https://connect.com
   a. connect.com can read SDP blobs of A and B.
   b. *"HTTP MITMs can see either A or B source IP address as well as SDP blobs of A and B"* is blocked.
3. Secure ICE to STUN server
   a. Sees all ICE candidates as well as source address of ICE messages.
   b. "MITMs sees same as STUN server" is blocked.
4. https:// to TURN relay server
   a. Sees A's and B's source and target addresses
   b. See's SDP blob.
   c. "MITM's sees same as relay server" is blocked.
5. TURN Relay: Data and audio exchanged peer-to-peer.
   a. MITM's sees source and IP addresses is blocked.
6. Web client leakage via content injection or poorly written or malicious web app code.
   a. Question: does using https:// to nice.com lock down the client for xross-site scripting? If not, then other actions required.

By using VPN proxy, Relay server, https, stuns and turns and CSP, some of the problems can be solved.
However, there seem to be a need to add a "Relay_mode" control via user settings and CSP in the UA. Also, UA enforcing stuns and turns when web app loaded via https seems like a good precaution. The UA should show more console information. CSP for controlling the STUN/TURN servers used should be considered. Same goes for user settings.

Furthermore, encrypting the SDP blobs seem like a good idea, but more of a BCP measure. Hashing the user identities is similarly a standard practice.

Closing the lid #5...the web client



1. Threats:
   a. Cross origin content injection used to have js send SDP blob to maliceous.com using networking services or services using network services.
   b. Content injection due to poor design or hack attack on connect.com assuming it has js to runf the PeerConnection..
   c. Content injection due to poor design or hack attack on nice.com.

2. Possible measures:
   a. Cross site attacks:
      i. POWER, MIX, CSP.
      ii. Certificate management.
   b. Poorly designed or hacked connect.com:
   c. Poorly designed or hacked nice.com

not-so-nice.com example

This case has been studied in the WebRTC security work to some extent. Compared to nice.com solutions, the following measure could be considered:

- WebRTC IdP.
- Web site blacklists.
- Always relay only.
- ...