WebRTC IP Address Privacy

Justin Uberti, Peter Thatcher, Eric Rescorla

Background: ICE

- RFC 5245 ICE gathers all the clients IP addresses for each interface
  - Host (local IP)
  - Server reflexive (apparent IP from STUN server)
  - Relayed (IP assigned by TURN server)
- So the Web server learns:
  - Local (RFC 1918) address
  - Public addresses even behind a VPN or proxy

Concerns

1. Increased fingerprinting surface
   - Distinguish people behind the same NAT
   - With multiple addresses, could be a unique identifier
   - Possible input to firewall bypass attacks
2. Identifies IP addresses “hidden” by VPNs
   - Should only happen with “split” VPNs -- but lots of people run them
3. Identifies IP addresses "hidden" by proxies

Considerations

1. Local address
    a. Threat of firewall bypass overstated; local info can be learned via XHR
    b. Info contained in addresses limited (e.g. 192.168.0.x) or ephemeral (RFC 4941 IPv6)
    c. Some WebRTC apps require local addresses to work; hairpin not dependable
2. VPN
    a. User may or may not want WebRTC traffic to traverse VPN
    (e.g. home user connected to corporate VPN)
3. Proxy
    a. User probably does not want WebRTC traffic to traverse proxy
    b. Enterprise admins can enforce this with firewall policy if desired

Goals

- Primary goal is to prevent drive-by harvesting of addresses
- VPN situation (#2) is the most critical thing to address
- Avoid degrading user experience or quality by default
- Provide options to cover #1/#3 for specific cases

Restricted gathering

- **General idea: only publicly visible addresses + single host candidate**
- Implementation:
  - Bind host candidate to 0.0.0.0 (Chrome) or default interface (FF)
  - Publish only srflx/relay candidates (stomping raddr as needed)
- Optionally send default interface as host candidate

Why send host candidate?

- Used telemetry to measure frequency of host-host connections
- Incidence was low (~5%), but some apps **expect** this to work
  - e.g. LAN-focused data channel apps; no TURN server
- Experiment with no host candidates resulted in high failure rate
  - ICE failures increased by ~10x
  - NAT hairpin apparently unreliable (or no STUN server)
- Basically, needed to avoid breaking apps; benefit outweighs cost

Restricted gathering, summary

- Minimal effect on media quality
  - Works for standard and LAN scenarios
  - Does not consider multiple routes (e.g. wifi vs cellular)
- Limits fingerprinting from use of local addresses
- Solves VPN issue
  - All media goes through VPN
  - Local address exposed is VPN adapter (tun0, typically nonroutable)
- Does not address proxy issue

Consent

- **General idea: give out all candidates if audio/video permission granted**
  - Otherwise, fall back to another solution
- No effect on media quality
  - Allows multiple routes to be used
- Exposes all addresses, but consent prevents drive-by gathering

Force proxy

- **General idea: TCP only; use proxy if configured**
    - Bind to 0.0.0.0; only make TCP connections
    - Only local candidate is TURN/TCP (if available); ICE-TCP also usable in some cases
    - No host candidates
- Significant effect on media quality (and proxy load)
- Exposes only the proxy's address

Proposal: 4 Modes

1. **Everything**
   (default, with consent)
2. **Restricted gathering, single host candidate**
   (default, no consent)
3. **Restricted gathering, no host candidates**
   (via prefs or extension)
4. **Force proxy**
   (via prefs or extension)