Lies, Damn Lies and ARP Replies

Dirty things to do with the IPv4 Address Resolution Protocol (ARP)

Don Stokes
Knossos Networks Ltd

Protocol Walk-through

- Host wants to know MAC address of target IP:
  - Host sends request to network
  - Target replies
  - Done!

Y'think?

- ARP caching?
  - How long to you keep an IP/MAC pair around?
  - How and when do you refresh it?

Unicast ARP

- Once we have a MAC address for a target, we can maintain that by regularly polling, using a unicast ARP request, e.g. every 15 seconds
  - No-one else sees the request (or its reply);
  - Host failures can be quickly identified.
  - Fall back to broadcast ARP request when host becomes unreachable.
    - IP may have moved to a new host or interface
  - Described in RFC 1122 (1989)

Unsolicited ARP

- Gratuitous ARP request
  - Used at interface initialisation time
    Target IP = interface IP or 0.0.0.0
    - Identifies duplicate IP addresses prior to configuration
    - May prime some ARP caches
- Unsolicited ARP replies
  - Used to update ARP caches
    - “Fakes” an ARP responses
    - Unicast fairly reliable; broadcast less so
    - Most ARP implementations don't check source

Proxy ARP

- Answer ARP requests for IP addresses that are not local.
- Mostly a Bad Idea.
  - Cisco turns it on by default (!)
  - But we can use this ...

Point-to-Point Ethernet

- If there are only two devices on a segment, why do they need four unique IP addresses?
  - Network
  - Concentrator
  - Client
  - Broadcast
- Surely we only need the client address?
- PPP (and SLIP) folks figured this out ages ago.

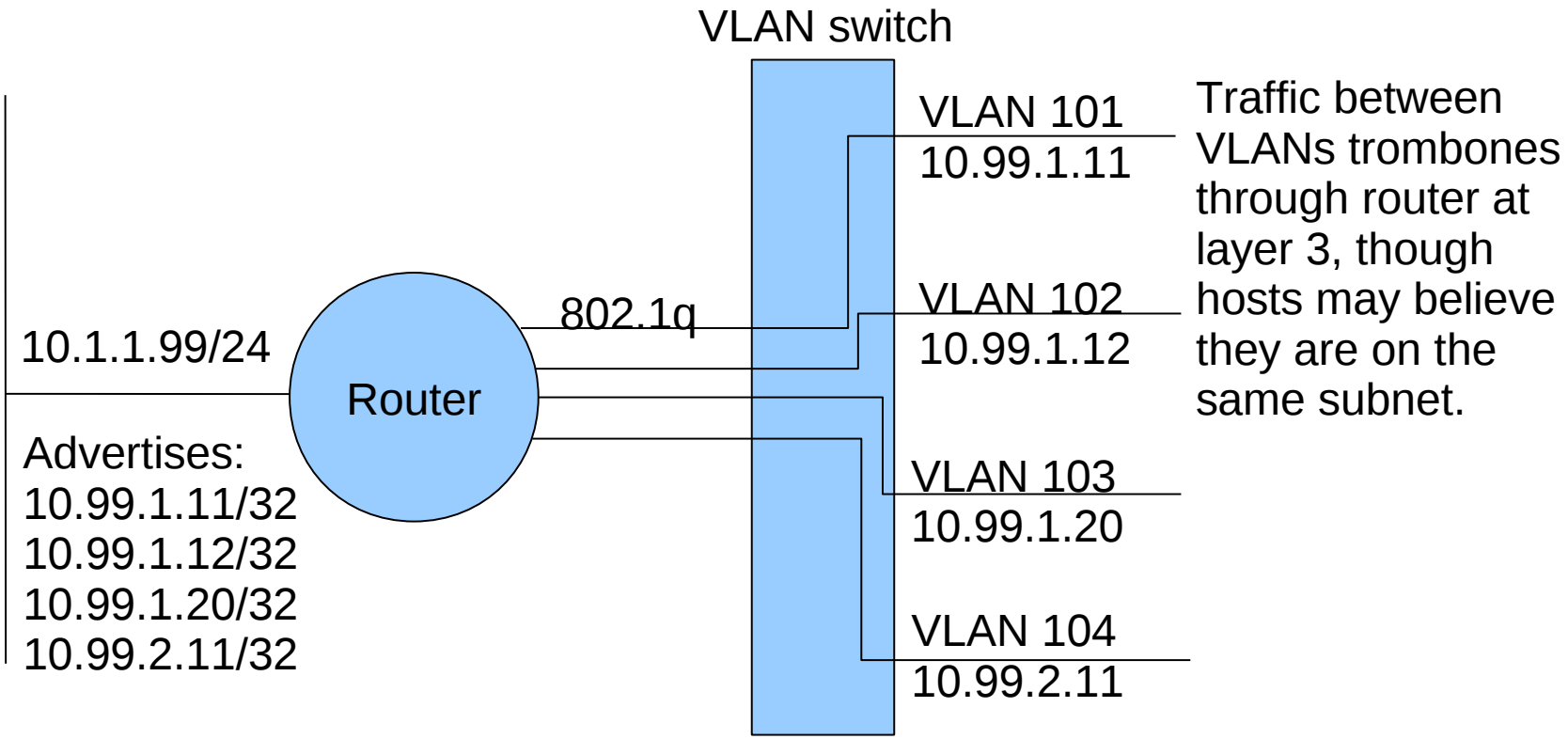
Co-operating hosts

- Don't even need ARP
  - Every packet received is destined for its target
  - Hard code destination MAC addresses
    - or send to the broadcast address.
  - Or packets can be sent to broadcast address
  - Operating systems don't expect this!
- Route client traffic to destination via interface, just like a point-to-point link.
  - Concentrator does not need link-specific IP address

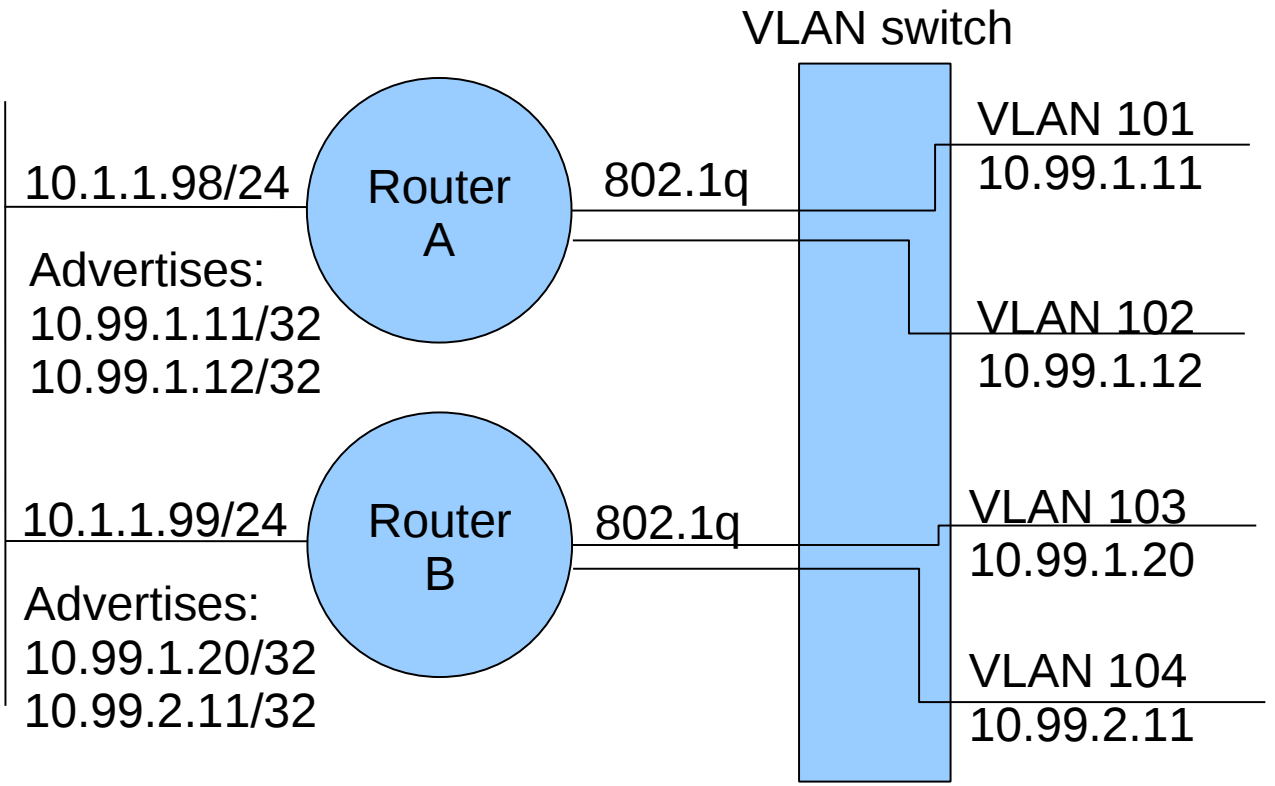
Uncooperative Hosts

- Tell client host a story:
  - IP address 10.99.1.11/24
  - Gateway 10.99.1.1
- Concentrator advertises only 10.99.1.11/32 to routing protocol. Other addresses in /24 could be *anywhere* ...
- Concentrator ARPs for 10.99.1.11
- But answers *all* ARP requests for 10.99.1.0/24
  - Only if they're *from* 10.99.1.11.

Concentrator example



Multiple accesses



Fail-over

- Each router can answer for gateway address (e.g. 10.99.1.1)
  - But each router uses a different address for its ARP requests to clients.
    - Router A: 10.99.1.2
    - Router B: 10.99.1.3
- Routers agree which host (the active router) will advertise a specific address.
- Routers may be active for some IP addresses, inactive for others.

Fail-over

Active router:

- Routes traffic inbound from the client.
- Announces client addresses to backbone.
- Answers all ARP requests from client for subnet addresses, except itself, client and other router.
- Maintains a regular ARP poll of client host.
- Relinquishes active status if poll fails and other router claims successful poll.

Fail-over

Inactive router:

- Routes traffic inbound from the client.
- Does not advertise client addresses to backbone.
- Does not answer any ARP request from client.
- Maintains a regular ARP poll of client host.
- Takes over active status if primary router indicates its ARP poll has failed, and polling is functioning.

Fail-over

- Inactive and active router are selected on an IP client by IP client basis
  - Router may be active for some clients and inactive for others
- Clients may share the same layer-2 subnet
  - Clients are identified by the source IP address in ARP requests.
  - We don't have to tell every client on a subnet the same story!

Fail-over

- No magic MAC addresses, unlike VRRP and friends.
  - No changes to MAC address.
  - No changes to layer 2 MAC table entries.
- No extraneous protocols on client links
  - Only “normal looking” ARP on client subnet.
- Regular polling maintains MAC table entries in switched infrastructure.

Load balancing

- Since clients can be distributed between routers, we can apply appropriate rules as to which hosts are active for which clients.
- Static load-balance based on knowledge of L2 topology.
  - Primary active router is closest to client.
- L2 topology generally not easily available. Could be done dynamically.
  - Could potentially be done by timing ARP responses: active host is fastest response.
  - Or by load.

Implementation

- FreeBSD
- NetGraph node, creates “point-to-point” interface (like PPP/SLIP/tunnel).
- Clustering and information distribution protocol.
  - Experimental “Potato” protocol.
- Could be used in software defined network.
  - Controller ARP daemon performs ARP protocol
  - ARP daemon instructs network via OpenFlow

IPv6

- No need for “point-to-point” semantics.
  - No shortage of addresses!
- IPv6 more difficult to identify end addresses
  - May be fixed address, or EIU-64.
- Fail-over technique can apply of we assume that one active device represents an entire subnet when determining active and inactive hosts.
  - True on genuine point-to-point link.
  - Mostly true if all client hosts are co-located.

Summary

- ARP not a routing protocol.
- But can be flexible.
  - If you are prepared to wrap some extra brains around it.