Using Neighbor Graphs in support of fast and secure WLAN mobility

William Arbaugh

University of Maryland

College Park

Joint work with:

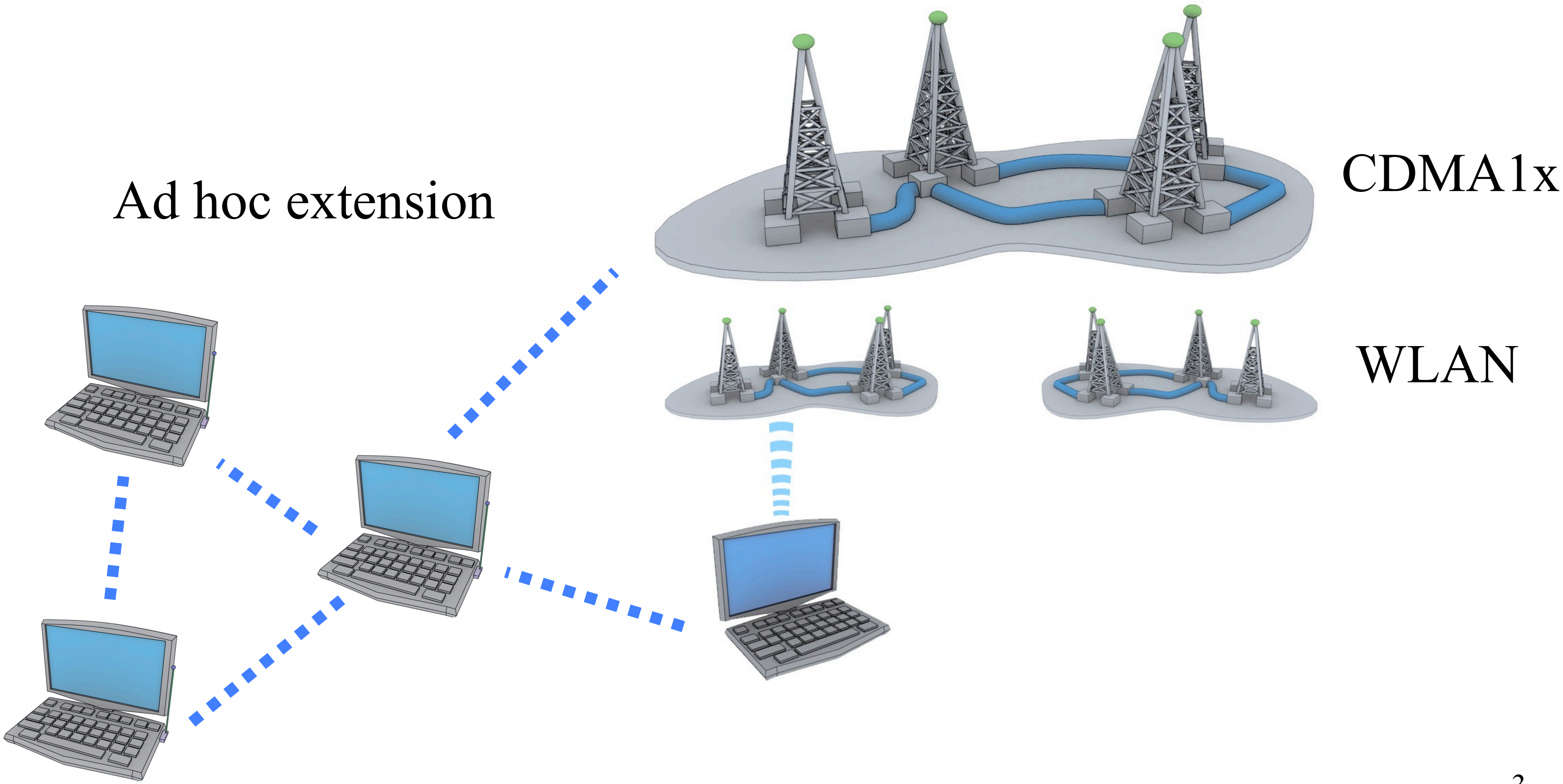
Arunesh Mishra, Min-ho Shin, Nick Petroni, T. Charles Clancy, Insun Lee, and Kyunghun Jang

This work funded by Samsung Electronics and grants from the U.S. National Institute of Standards Technology and the DOD Laboratory for Telecommunications Science

Outline of Talk

- Empirical Analysis of 802.11 hand-offs
- Neighbor graphs
  - Proactive caching
    - Experimental results
    - Simulation results
  - Proactive key distribution for LANs and Interworking
    - Experimental results
- Conclusions and Future Work

One View of the Future



Mobility Definitions

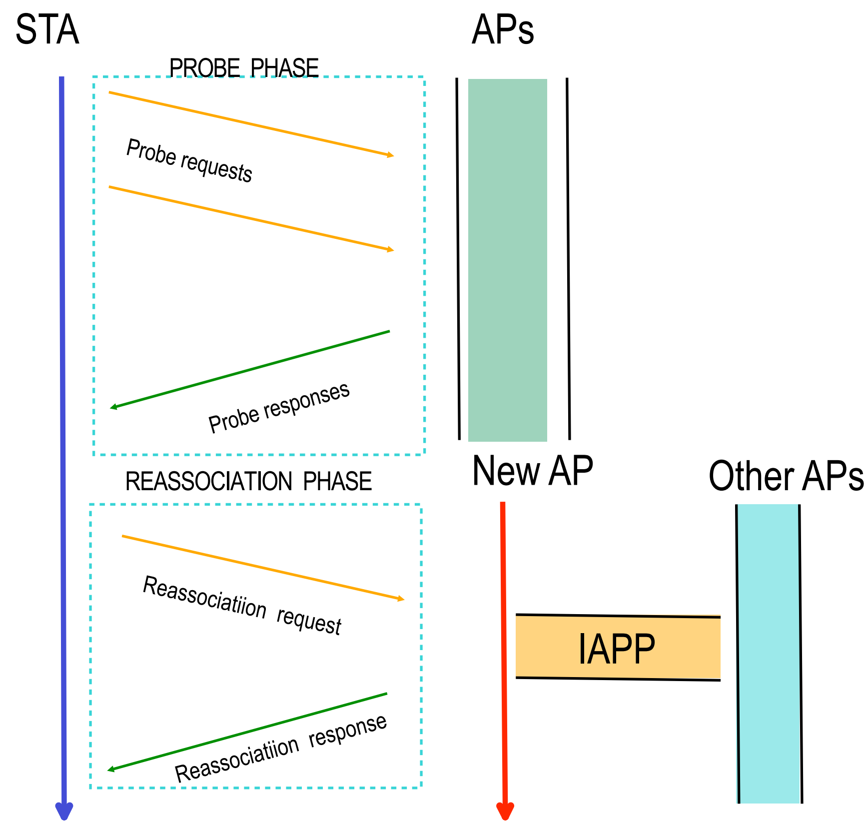
- Discrete Mobility: A mobile station utilizes the network without movement. Prior to movement operation ceases and begins again associated to a new base station.
- Continuous Mobility: A mobile station moves and operates simultaneously.

Properties Required

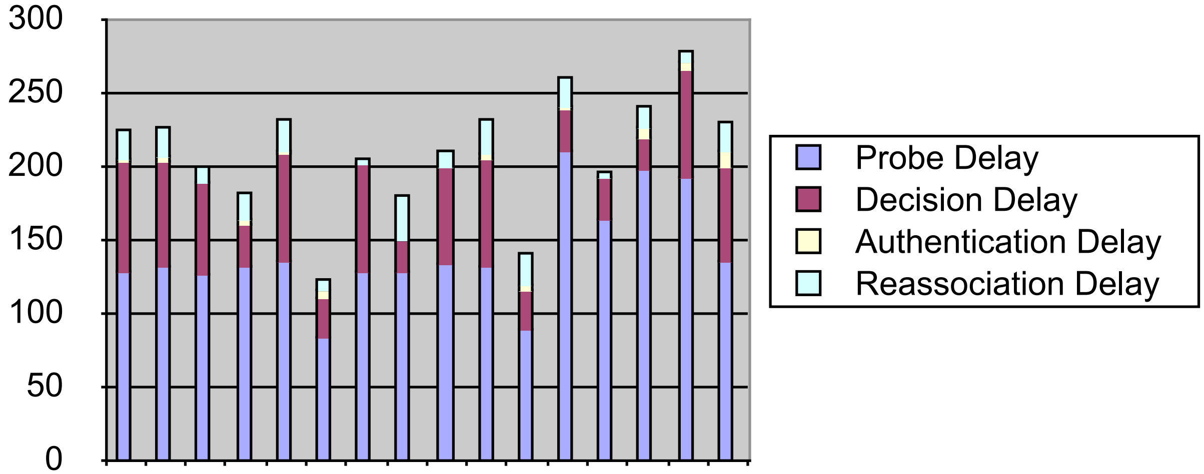
- Transparency
- Security
- Ubiquity
- Performance

The Handoff Procedure

- Probe Phase
  - STA scans for APs
- Reassociation Phase
  - STA attempts to associate to preferred AP



Prism2 (Zoomair)



Data from an “Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process” to appear in ACM CCR

Why is this important?

- Hand-off times MUST be efficient to support synchronous connnections, e.g. VoIP
- ITU guidance on TOTAL hand-off latency is that it should be less than 50 ms. Cellular networks try to keep it less than 35 ms.

Improving the Reassociation latency

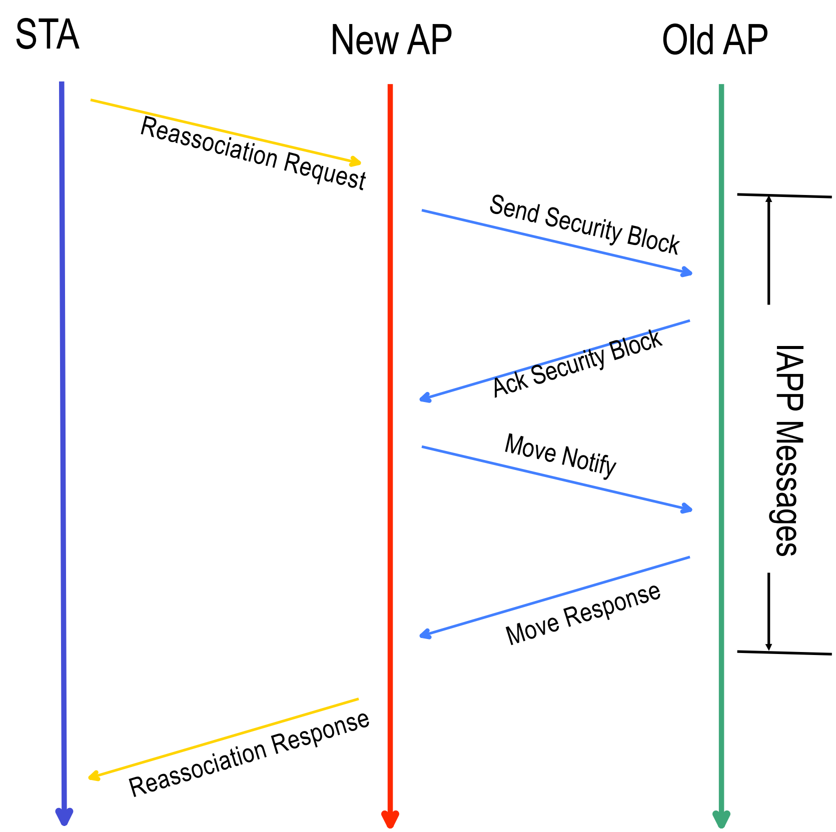
- Review Previous work
- Introduce Neighbor Graphs
- Introduce Proactive Caching
  - Experimental results
  - Simulation results
- Introduce Proactive Key Distribution
  - Experimental results

Related Work

- Context Caching
  - SEAMOBY (IETF) - Generic context algorithm
  - Koodli and Perkins 2001- Layer 3 reactive algorithm
  - IEEE IAPP draft 4 (January 2003)
- Network Topology
  - Pack et. al. weighted matrix for pre-authentication
  - Learning bridge

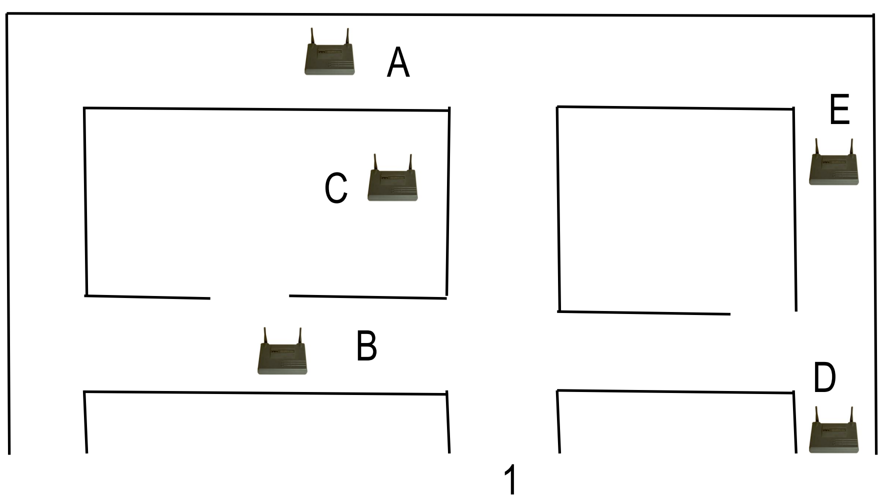
The Handoff Procedure- Reassociation Phase - Draft version of IAPP

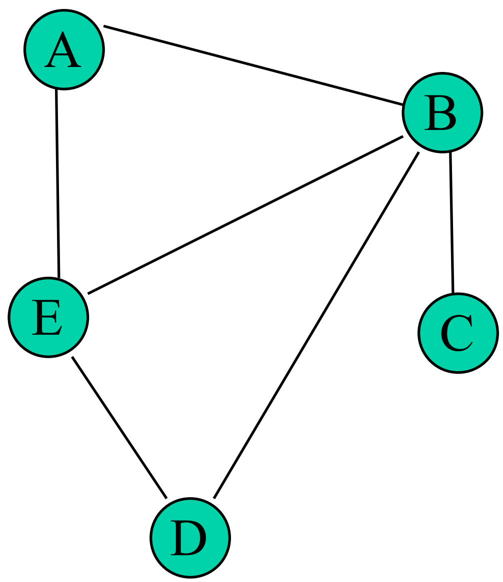
- Four IAPP Messages
  - IAPP Latency > 4 * RTT
- Move Request and Move Response messages over TCP
- RADIUS interaction not shown (further delay)



Neighbor Definition and Graph

- Two APs i and j are neighbors iff
  - There exists a path of motion between i and j such that it is possible for a mobile STA to perform a reassociation
  - Captures the ‘potential next AP’ relationship
  - Distributed data-structure i.e. each AP can maintain a list of neighbors
  - Centralized if AAA server holds entire graph



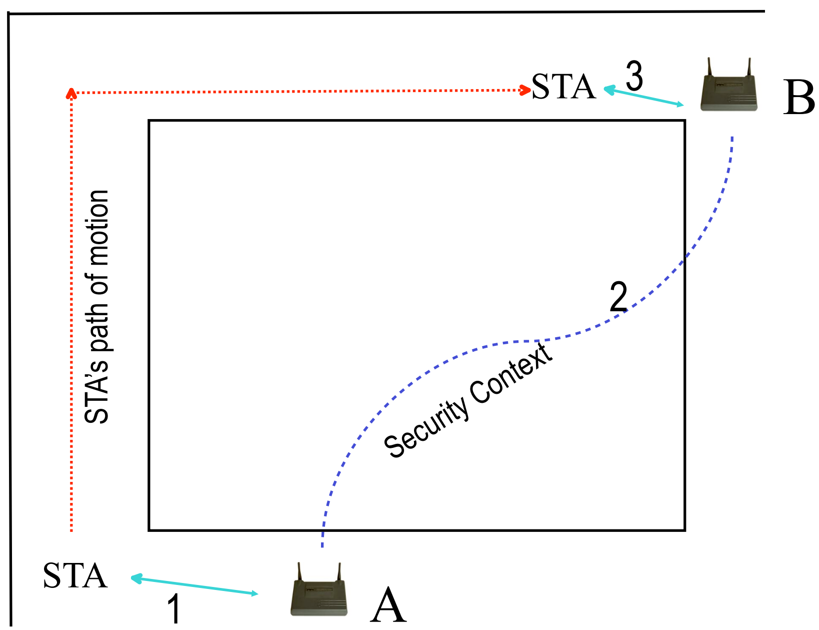


AP Neighborhood Graph – Automated Learning

- Construction
  - AP can learn:
    - If STA c sends Reassociate Request to AP i, with old-ap = AP j :
    - Create new neighbors (i,j) (i.e. an entry in AP i, for j and vice versa from move-notify message)
    - Learning costs only one 'high latency handoff' per edge in the graph
    - Enables mobility of APs, can be extended to wireless networks with an ad-hoc backbone infrastructure
    - Dynamic, i.e. stale entries time out
  - Easily extended to a AAA server

Proactive Caching Algorithm

- Key Idea :
  - Propagate context to potential 'next' APs to eliminate IAPP latency during reassociation
    1. STA associates to AP A
    2. AP A sends context to AP B proactively (new IAPP message)
    3. STA moves to AP B – does fast reassociation since B has context in cache

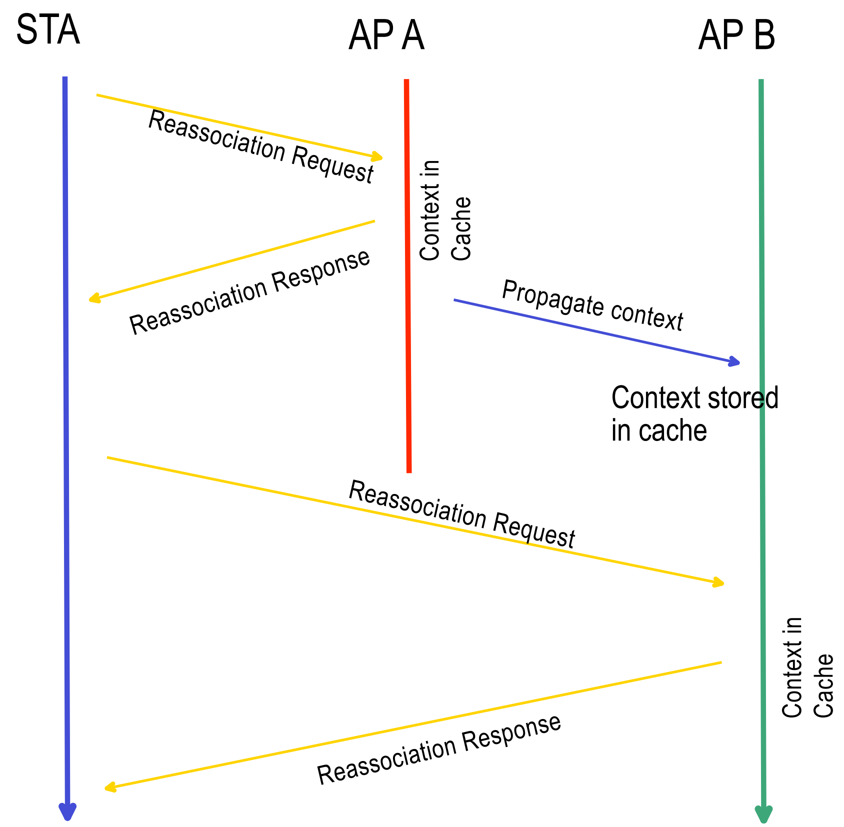


Proactive Caching – The Algorithm

- When STA c associates/reassociates to AP i
  - If context(c) in cache:
    - Send Reassociate Response to client
    - Send Move-Notify to Old-AP
    - Old-AP invalidates its neighbor caches
  - If context(c) not in cache, perform normal IAPP operation
  - Send security context to all Neighbours(i)
- Cache Replacement : Least Recently Used
- Cache size vendor dependent

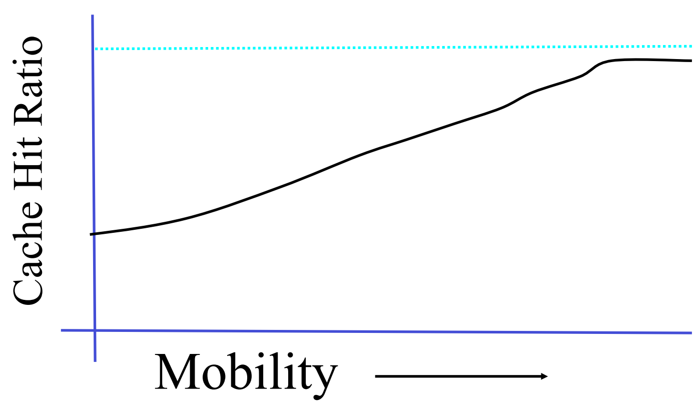
IAPP Messages with Proactive Caching

1. STA reassociates to AP A
2. AP A has security context in cache
3. AP A propagates context to AP B (all neighbors of A)
4. STA reassociates to AP B which again has security context in cache



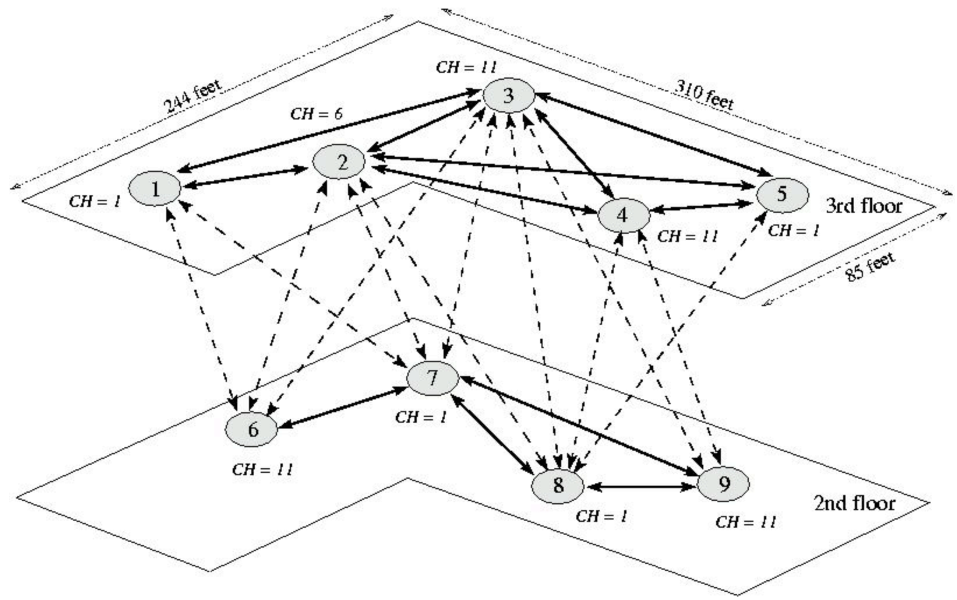
Proactive Caching – Expected Performance

- Handoff latencies play a role in performance when mobility is high
- With an LRU cache, higher the mobility, higher the cache-hit ratio (on average), implies larger number of fast-handoffs

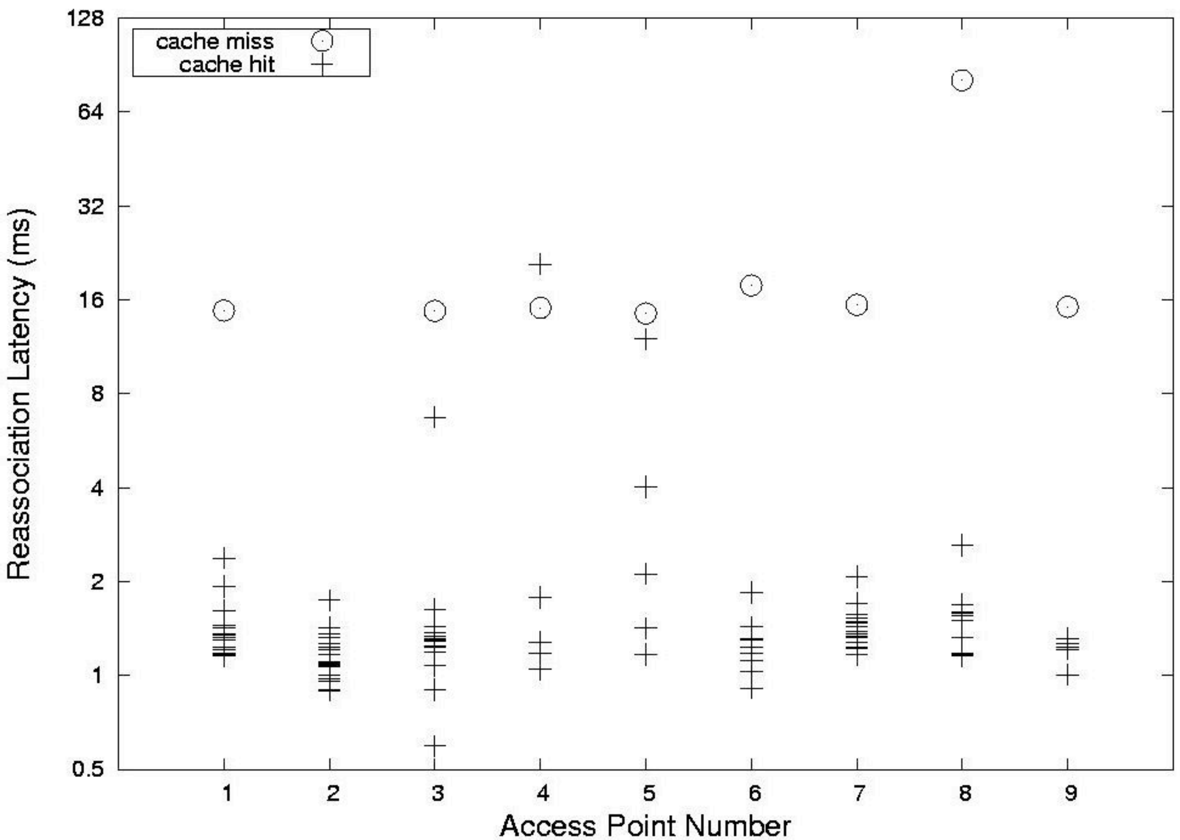


Test Bed

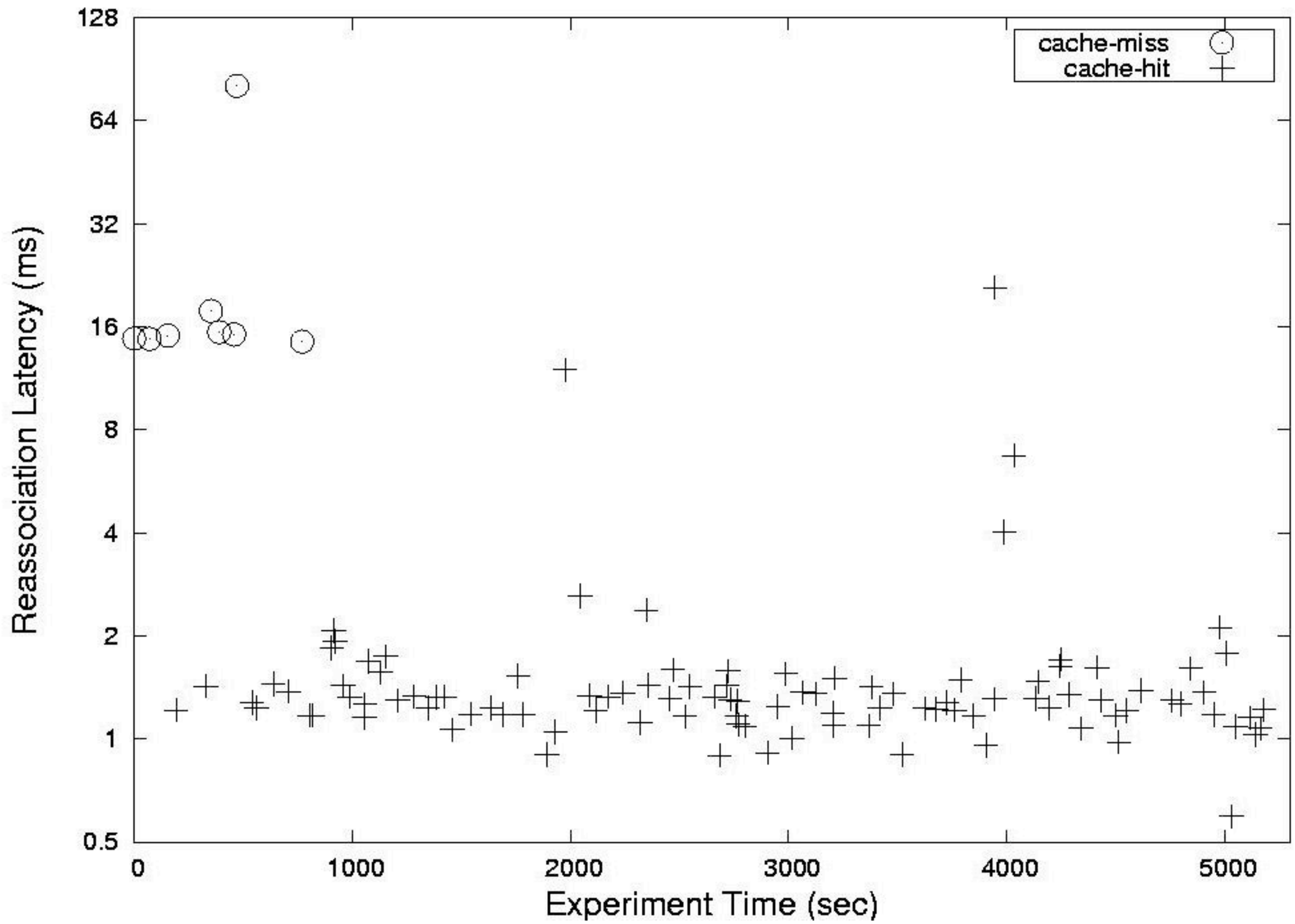
- Custom built access points using Soekris 4521 boards, OpenBSD, and Prism2 chipsets
- Custom IAPP implementation



Experimental Results by AP



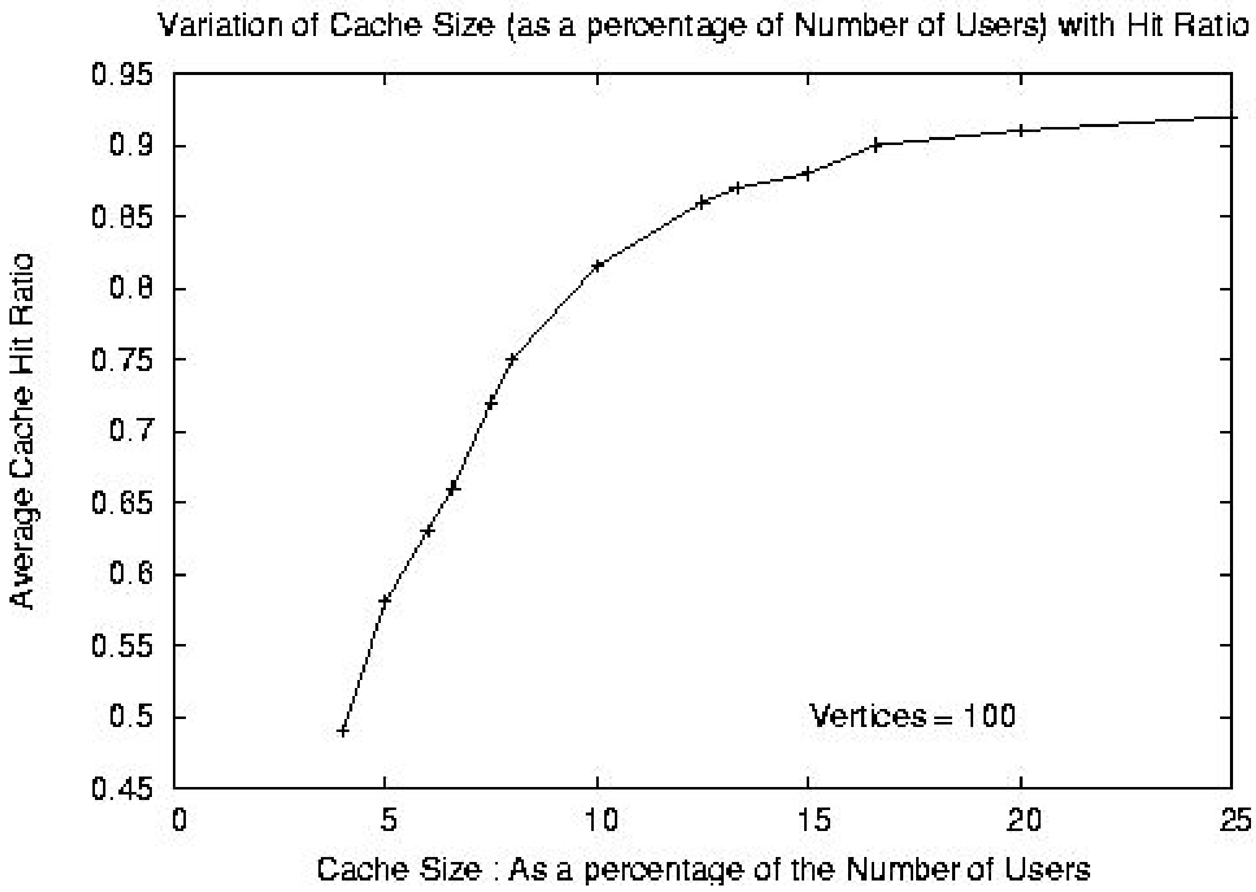
Experimental Results by Time



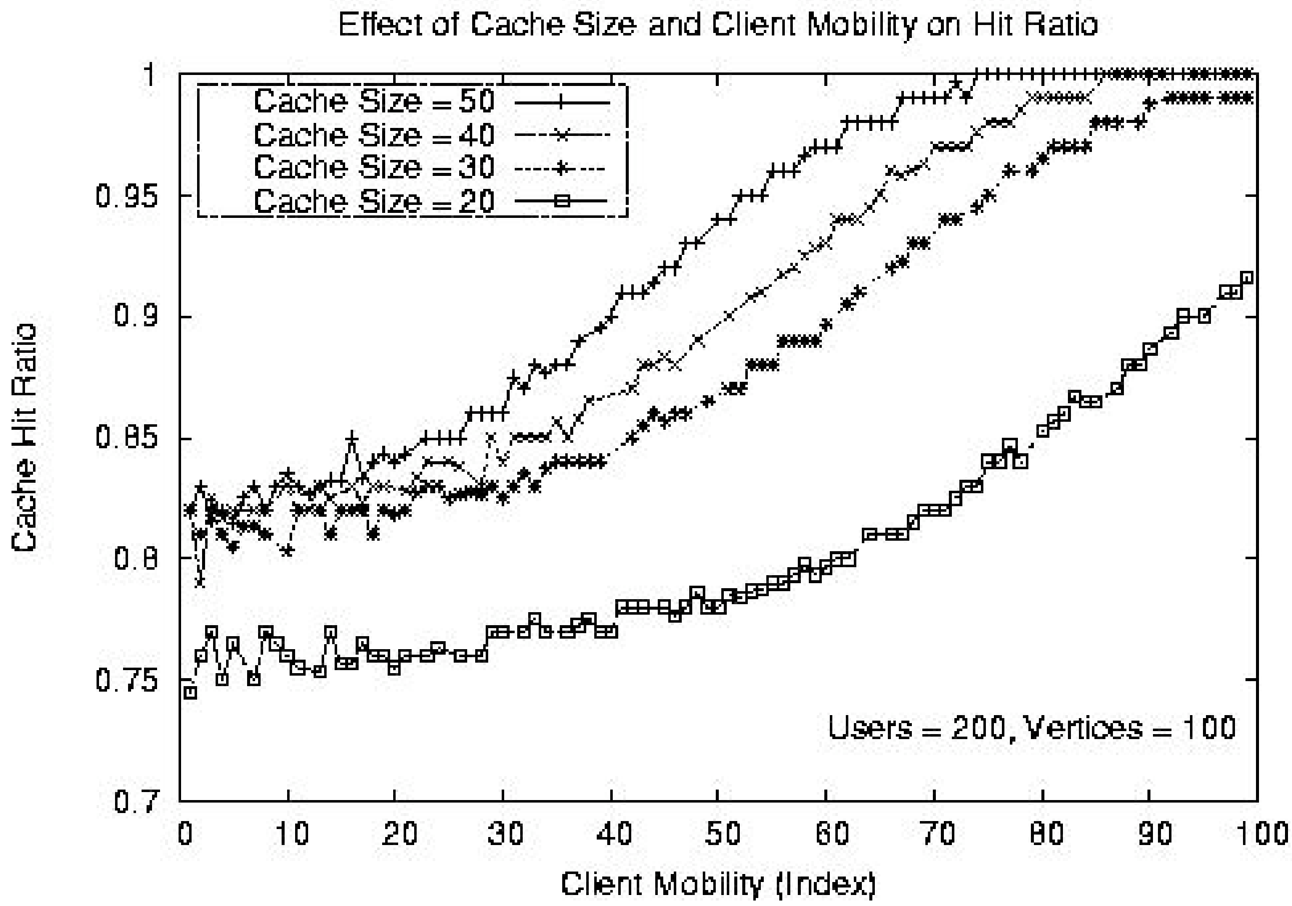
Simulation Details

- Stations follow a random association pattern and can move to a neighbor AP with equal probability
- Stations have a mobility index assigned uniformly:
  - $Mobility_{index}$ = (time moving / total time) * 100
  - Continuous mobility has a mobility index of 100.

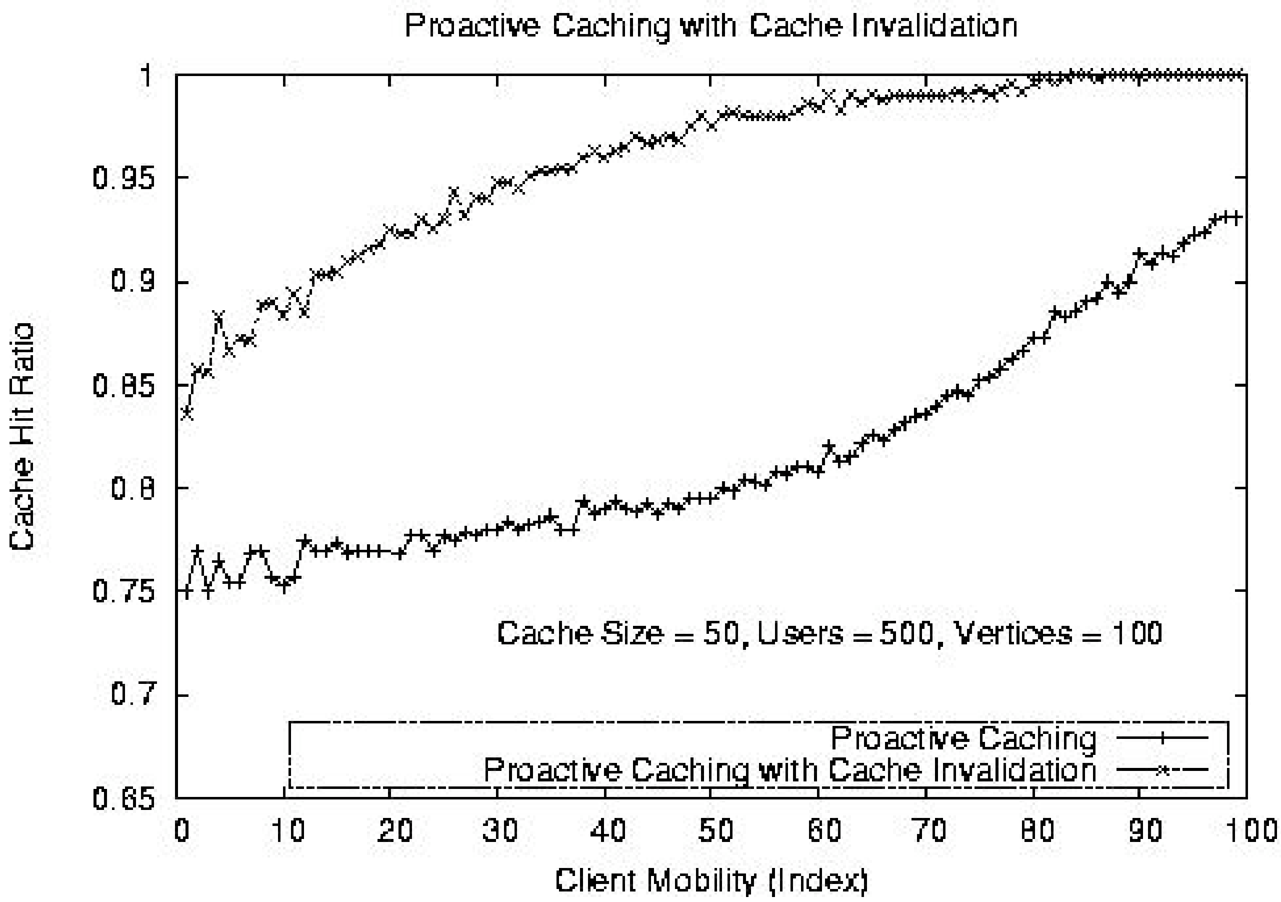
Simulation Results: Cache Size



Simulation Results: Client Mobility



Simulation: Cache Eviction Invariant



TGi Fast Roaming Goals

- Handoff to next AP <u>SHOULD NOT</u> require a complete 802.1x re-authentication.

- Compromise of one AP <u>MUST NOT</u> compromise past or future key material.

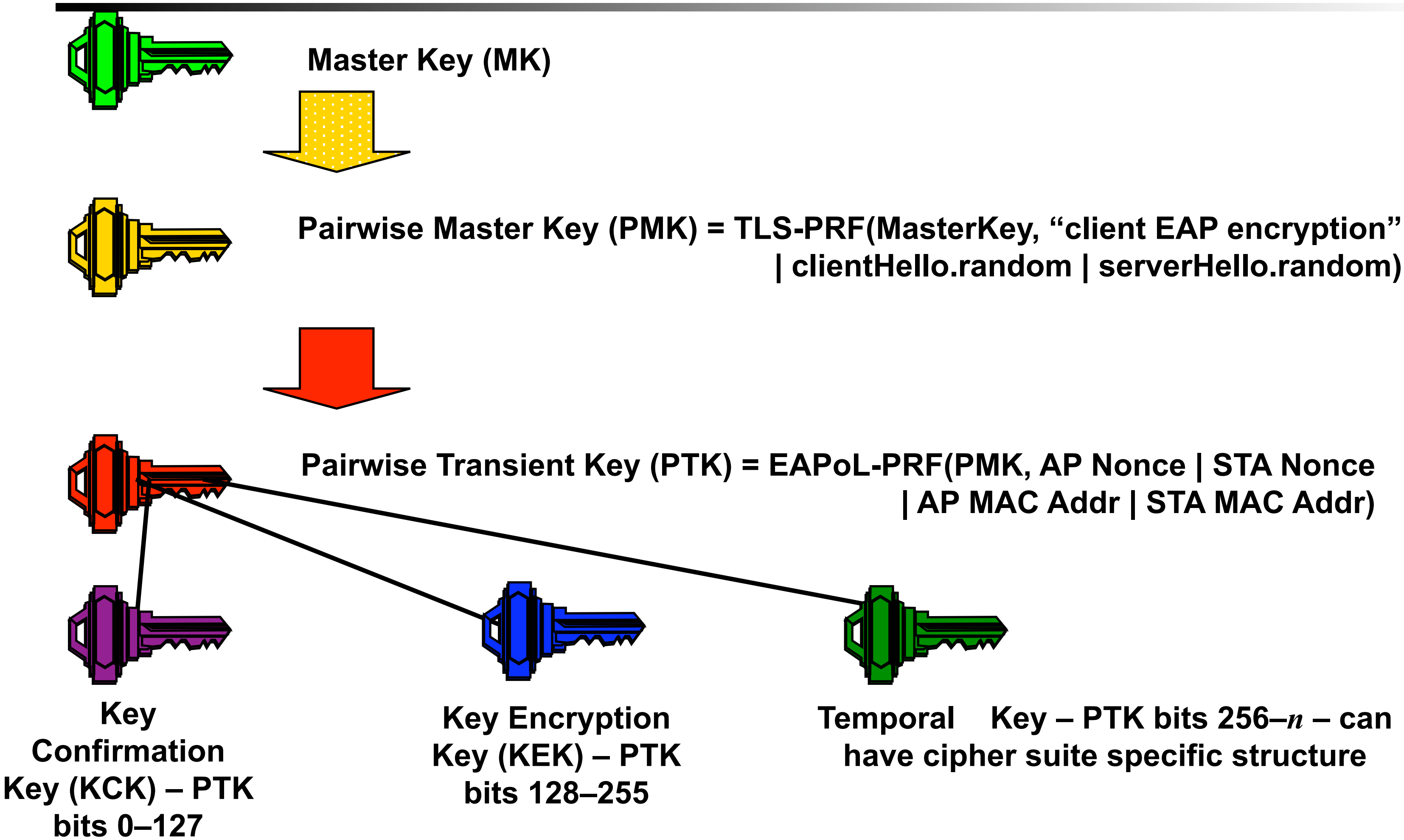
Only Two Ways

- Exponentiation support for assymmetric cryptographic operations at AP, or
- Trusted Third Party, i.e. Roaming or AAA Server

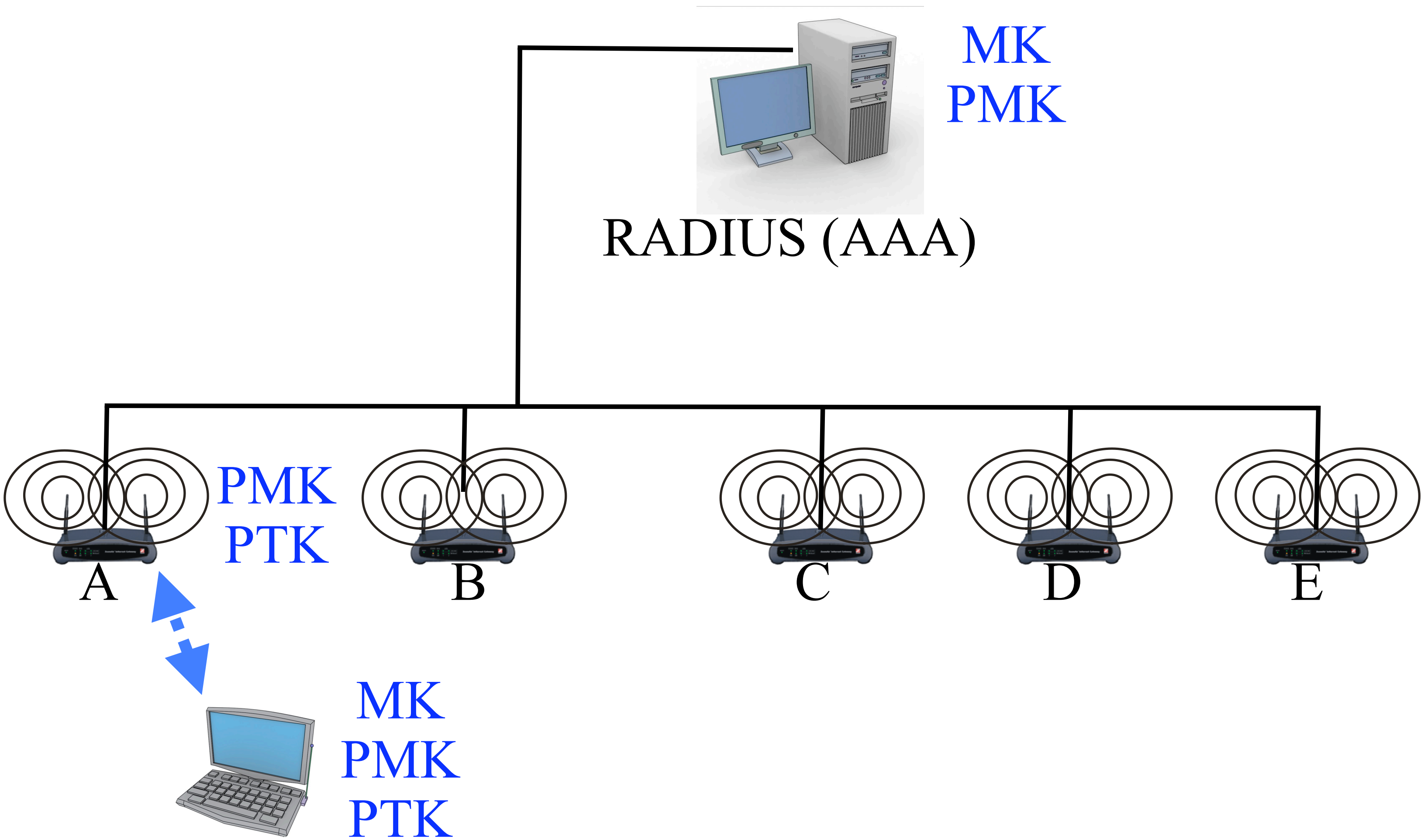
Proactive Key Distribution (TGi)

- Extend Neighbor Graphs and Proactive Caching to a Roaming Server
  - Eliminates problems with sharing key material amongst multiple APs
  - Easily extended to support WAN roaming
  - Extendable to support Interworking

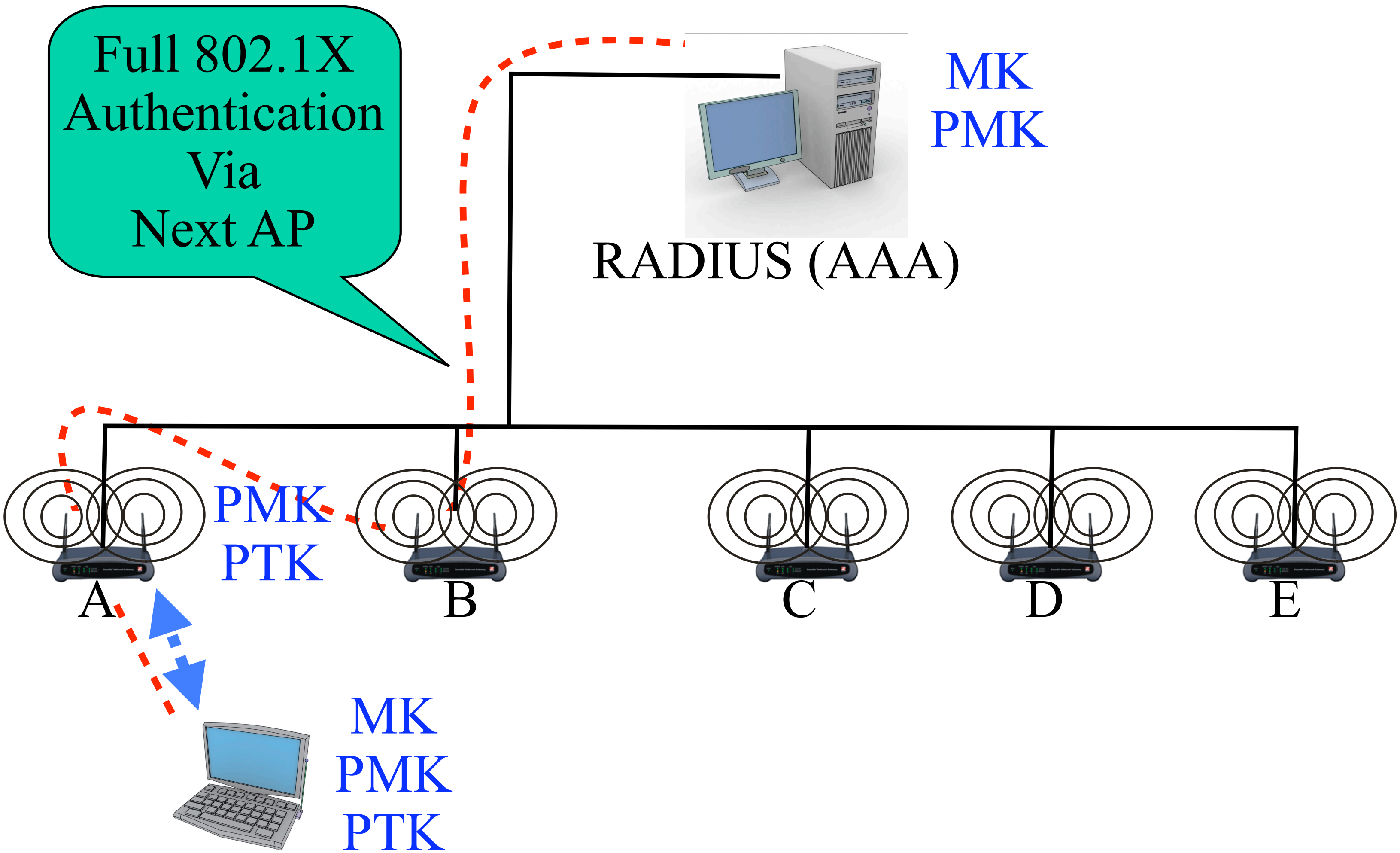
TGi Pairwise Key Hierarchy



Post Authentication and 4-handshake



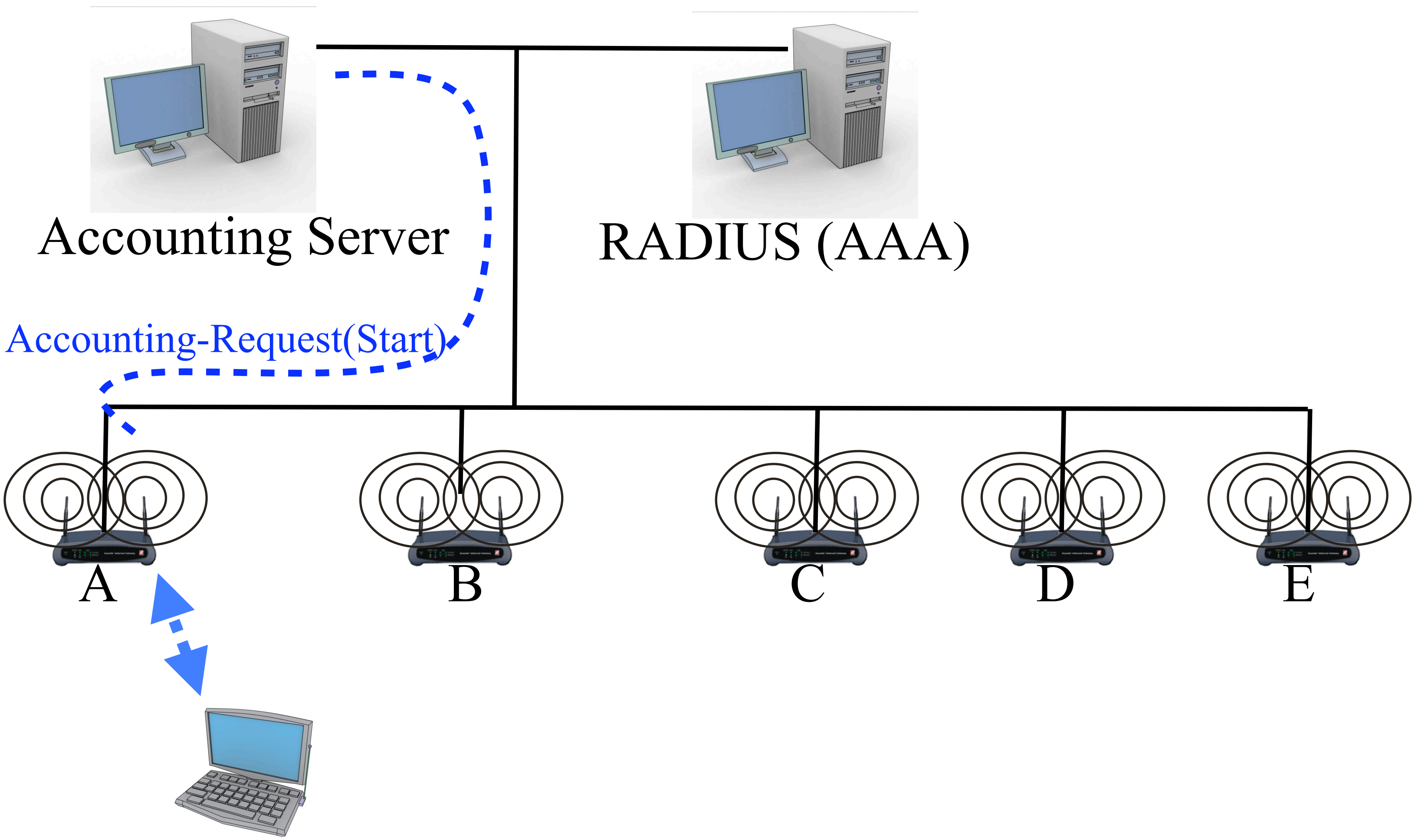
Pre-authentication



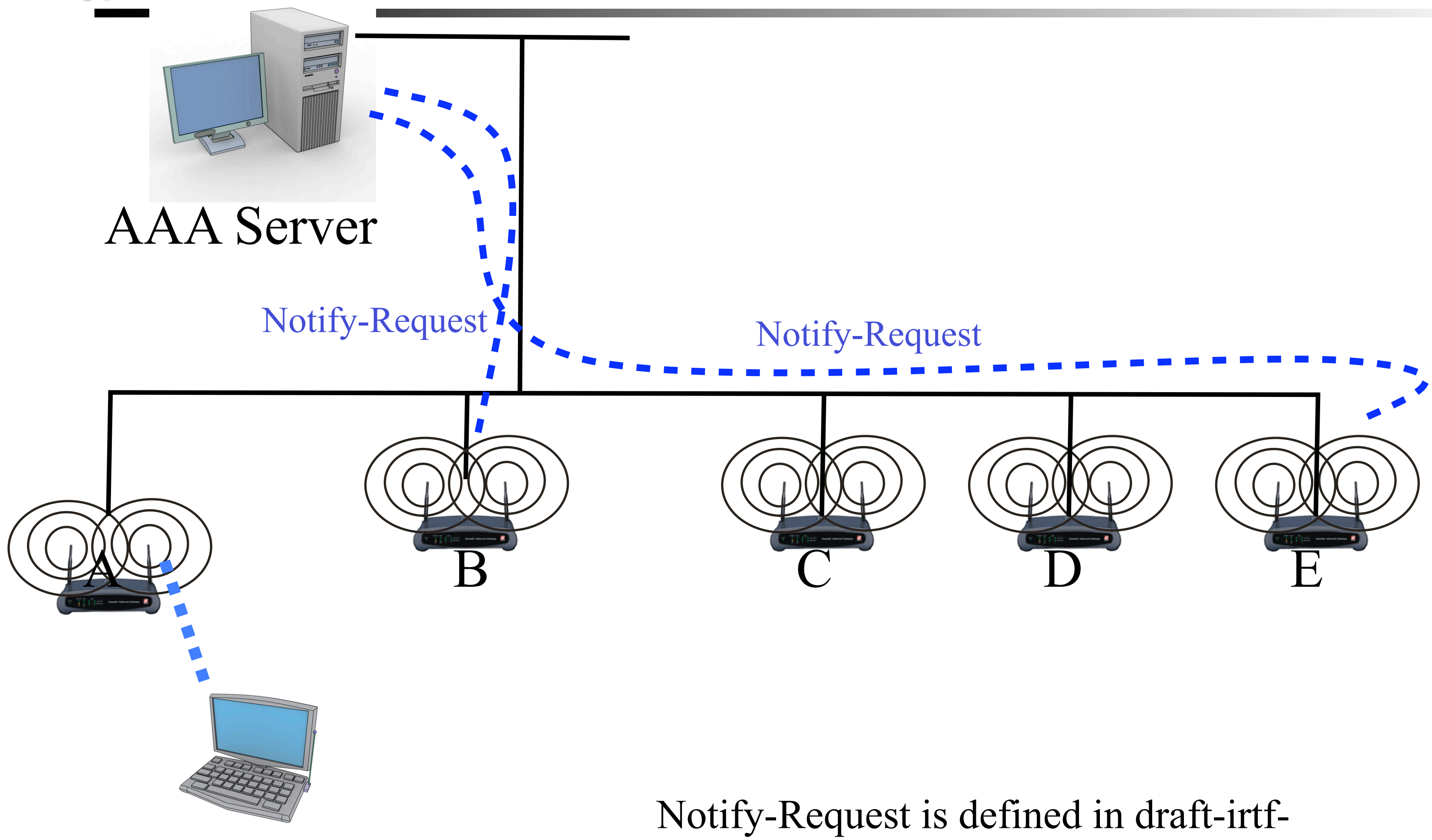
Problems with Pre-Auth

- Expensive in terms of computational power for client, and time (Full EAP-TLS takes ~800ms).
- Limited to the same LAN or VLAN
- Requires well designed and overlapping coverage areas
- Edge cases

Post Authentication and 4-handshake

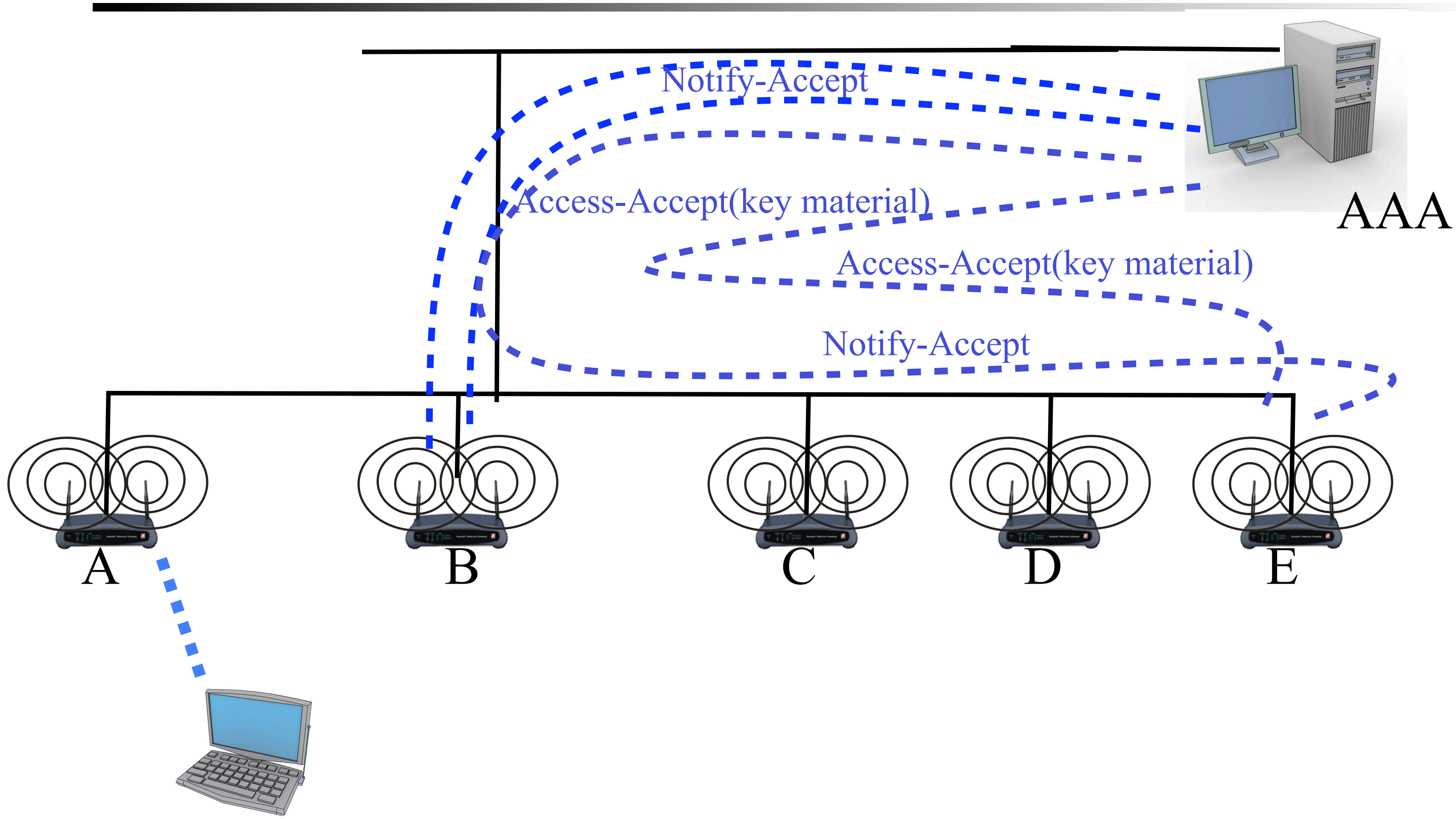


Post Authentication



Notify-Request is defined in draft-irtf-aaaarch-handoff-01.txt

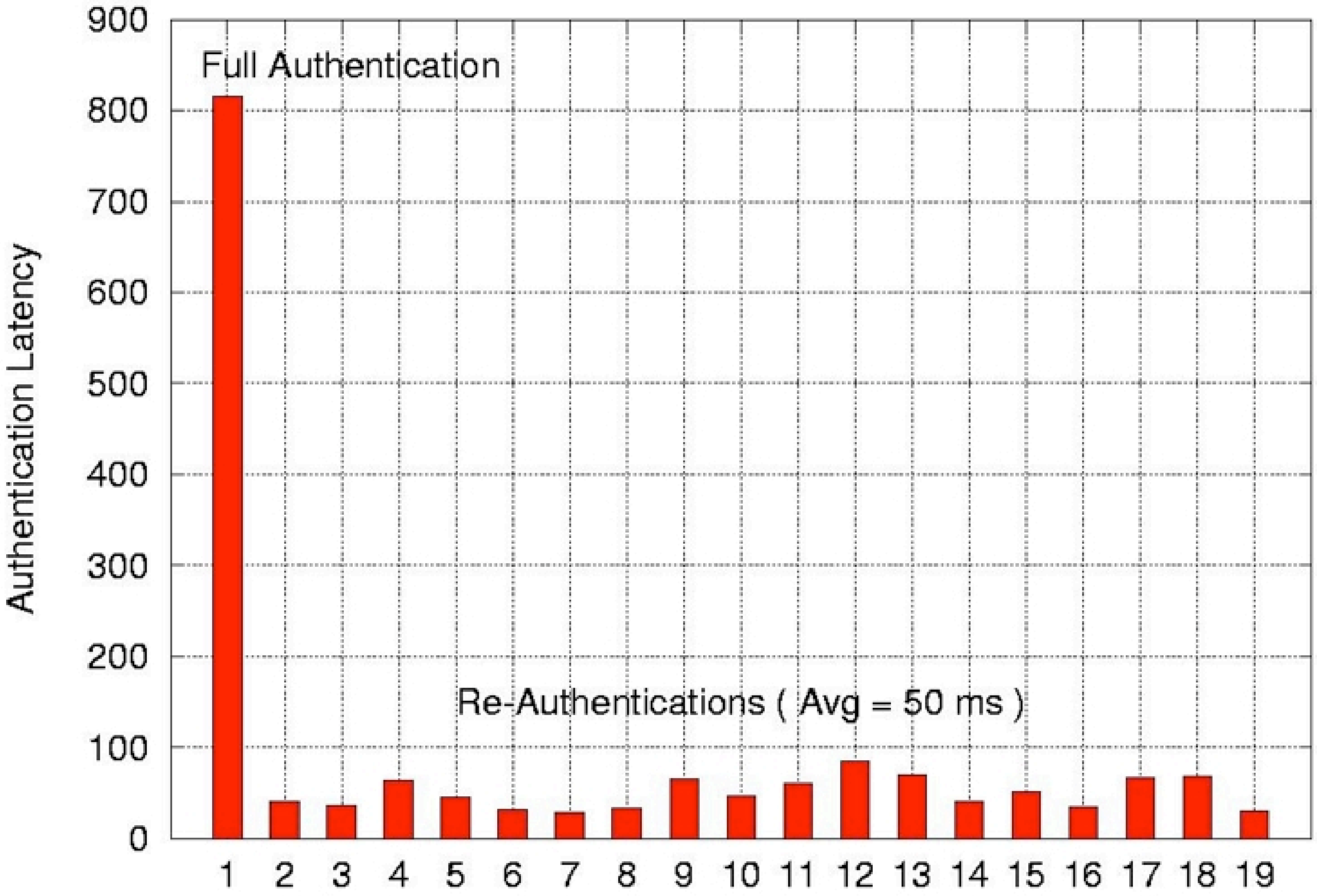
Post Authentication



AP Actions on Notify Request

- Dynamic Keys, i.e. PMK changes per roam.
  - AP MAY send an ACCESS-REQUEST to AS
- Static Key, i.e. PMK is unique per AP but never changes.
  - Nothing unless authorization is required.

Experimental Results



Maximum STA Velocity

For the Notify and PMK install to occur in time, we need:

2 RTT + handshake < D/v

Where:

D = coverage diameter

v = STA velocity

RTT = round-trip time from AP to AAA server, including processing.

Assuming D=100 ft, handshake = 10 ms, and RTT = 100ms, we get:

v = 100 ft/ (200ms + 10 ms) ~ 500 ft/sec = **Mach 0.5!!**

Analysis thanks to Bernard Aboba

Future Work

- Investigate use of eviction invariant
- Use of NG to reduce probe delay
  - To appear in Mobisys '04
- Use of NG to assist in load balancing
- Pulling keys rather than pushing
  - Wireless Communications Magazine, Feb 04.

Conclusions

- Neighbor graphs dramatically improve handoff speeds by an order of magnitude.
- And may have other potential uses within wireless networking.

Impact and Status

- Major Wi-Fi vendor improved firmware as a result of measurements.
- NG and Proactive Caching included in IEEE 802.11 recommended practice document for Inter Access Point Protocol (IAPP).
- Proactive Key distribution under consideration by IETF:

  draft-irtf-aaaarch-handoff-00.txt

Questions