FreeBSD Wireless Networking

Sam Leffler
Errno Consulting
sam@errno.com
Project Goals

- Device-independent 802.11 support
- Use full hardware functionality
- Production quality
- Reusable code:
  - Portable code but no portability layer
  - Native management API (e.g. Wireless Extensions)
- Dual BSD/GPL license
Background

- Original version by Atsushi Onoe
- Overhaul (1) for multi-mode devices
- Overhaul (2) for security protocols
- Overhaul (2.5) multimedia extensions
- Overhaul (3) for multi-BSS support
Background: Original Version

- Circa 2001 (NetBSD)
- Simple devices (e.g. only 11b)
- Mostly firmware-based devices
- Pre-shared key WEP for crypto
Background: Multi-mode Devices

- Summer 2003 (started Fall 2002)
- Multi-band: 2.4GHz, 5GHz, etc.
- Multi-mode: 11a, 11b, 11g, Turbo, etc.
- 11g protocol

BIG CHANGE...

All the world is not 11b
Background: Security Protocols

- Summer 2004
- WPA protocol
- 802.11i, aka WPA2, protocol
- TKIP, CCMP, etc.: cipher modules
- Hardware crypto acceleration

BIG CHANGE... All the world is not WEP
Background: Multimedia Protocols

- Fall 2004
- WME/WMM protocol
- QoS traffic handling
- Hardware acceleration

BIG CHANGE... All traffic is not equal
Background: Multi-BSS Support

- 2005
- Multiple BSS with one device
- WDS support
- Repeater/bridge applications
- Foundation for mesh support

BIG CHANGE...

Separation of BSS and device
Comparison to Other Projects

- Microsoft “Native WiFi”
- Various proprietary
- MultiNet
- Linux
Microsoft Native WiFi

- Windows-specific
- Device independent
- Single BSS
- Expected in Longhorn
- Code access not generally available
Proprietary Products

- Usually device specific
- Often OS-specific
- Single BSS (mostly)
- Code sometimes available for a price
MultiNet

- Research project
- Multiple BSS
- Windows only (NDIS)

MORE INFO... http://research.microsoft.com/~bahl/MS_Projects/MultiNet/default.htm
Linux

- “Generic 802.11 Stack”
- Recent development (March 2005)
- Derived from device-specific code
- Linux-specific
- Single BSS
- Early stage--limited usability

MORE INFO...
http://marc.theaimsgroup.com/?l=linux-netdev&m=111174142325384&w=2
Security Protocols: Standards

- Wi-Fi Protected Access (WPA)
  - April 2003
  - Based on IEEE 802.11i Draft 3.0
  - Authenticated key management
  - TKIP+Michael (WEP on ‘roids)
  - AES-CCMP (optional)

MISSING... Preauthentication and fast handoff
Security Protocols: Standards

- IEEE 802.11i (aka WPA2/RSN)
  - Approved July 2004
  - AES-CCMP required
  - Preauthentication and fast handoff
- Management frames still not encrypted

GOOD INFO…
http://www.drizzle.com/~aboba/IEEE/
http://www.wi-fi.org/OpenSection/
Security Protocols: Key Handling

AP translates between EAPOL Messages and AS messages (RADIUS or other AS)

- (2) 802.11 AUTH
- (2) 802.11 ASSOC
- (14) EAPOL START

WPA
4-way handshake
EAPOL KEY msgs

Group Keys (encrypted)
Security Protocols: How it Works

- **Kernel support:**
  - 802.11 protocol (e.g. mgt frames)
  - cipher support

- **User-mode support:**
  - supplicant (station operation)
  - authenticator (AP operation)
Security Protocols: Kernel Support

- 802.11 protocol: beacon, auth, etc.
- Extensible crypto framework
- Cipher modules
- Management ioctls
- Application control of scanning
- 802.11 events via routing socket

FULL PERFORMANCE... No degradation with hardware crypto
Security Protocols: Supplicant

- wpa_supplicant from Jouni Malinen:
  - WPA/802.11i protocol
  - EAP/802.1x support
  - scanning and AP selection
  - driver_bsd.c for net80211 glue

- BSD/GPL license

WHERE TO FIND IT...

http://hostap.epitest.fi/wpa_supplicant/
ports/security/wpa_supplicant
Security Protocols: Authenticator

- hostapd from Jouni Malinen:
  - WPA/802.11i protocol
  - EAP/802.1x support
  - some built-in AS support
  - driver_bsd.c for net80211 glue
- BSD/GPL license

WHERE TO FIND IT...

http://hostap.epitest.fi/hostapd/
ports/security/hostapd
Multimedia Protocols: Standards

- **Wireless Multimedia Enhancements (WME)**
  - July 2003
  - Based on IEEE 802.11e draft
  - Capabilities negotiation
  - Quality of Service (QoS)
  - Enhanced DCF (EDCF)

**APPLICATIONS...**

Streaming video and VoIP
Multimedia Protocols: How it Works

- **Kernel support:**
  - 802.11 protocol (e.g. beacon frames)
  - Traffic classification
  - Device support (no software fallback, hard)

- **User-mode support:**
  - `ifconfig` report/set parameters
Multi-BSS: Motivation

- Multiple BSS with a single radio
  - Multiple virtual AP’s (different security policies)
  - Multiple IBSS’s
  - Mesh networks
  - Special-purpose applications (e.g. Atheros XR mode)

- Combo applications:
  - Repeater (station + AP)
  - Extender (AP + WDS links)
Single-BSS: Previous Model

- One network (BSS) per device:
  - ath0 is the device and the network

- Device configuration/operation is modal:
  - `ifconfig wi0 mediaopt hostap`
  - `ifconfig awi0 mediaopt adhoc`

- Combination modes require special handling (repeater = station + AP)
Multi-BSS: New Model

- **Device is a blank substrate:**
  ```
  # ifconfig iwi0
  iwi0: flags=8802<BROADCAST,SIMPLEX,MULTICAST> mtu 2290
  ether 00:03:7f:04:a0:a4
  media: IEEE 802.11 Wireless Ethernet autoselect
  status: no carrier
  ```

- **Network devices are cloned:**
  ```
  # ifconfig wlan create wlandev wi0 wlanmode adhoc
  wlan0
  # ifconfig wlan0
  wlan0: flags=8802<BROADCAST,SIMPLEX,MULTICAST> mtu 1500
  ether 00:03:7f:04:a0:a4
  media: IEEE 802.11 Wireless Ethernet autoselect <adhoc>
  status: no carrier
  ssid ""
  authmode OPEN privacy OFF txpowmax 100 ff
  ```

**DEFINITION...**

wlanX is a *Virtual AP (VAP)*
Multi-BSS: New Model (2)

- **Multi-BSS = multiple vaps:**

  ```
  # ifconfig wlan create wlandev ath0 wlanmode ap
  # ifconfig wlan create wlandev ath0 wlanmode ap
  # ifconfig
  
  ath0:  flags=8802<BROADCAST,SIMPLEX,MULTICAST> mtu 2290
         ether 00:03:7f:04:a0:a4
         media: IEEE 802.11 Wireless Ethernet autoselect (autoselect <hostap>)
         status: associated
  
  wlan0: flags=8802<BROADCAST,SIMPLEX,MULTICAST> mtu 1500
         ether 00:03:7f:04:a0:a4
         media: IEEE 802.11 Wireless Ethernet autoselect <hostap>
         status: no carrier
         ssid ""
         authmode OPEN privacy OFF txpowmax 100 ff dtimperiod 1
  
  wlan1: flags=8802<BROADCAST,SIMPLEX,MULTICAST> mtu 1500
         ether 00:03:7f:04:a0:a4
         media: IEEE 802.11 Wireless Ethernet autoselect <hostap>
         status: no carrier
         ssid ""
         authmode OPEN privacy OFF txpowmax 100 ff dtimperiod 1
  ```
Multi-BSS: New Model (3)

- Multi-use = combined vaps:
  - ifconfig wlan create wlandev ath0 wlanmode ap
  - ifconfig wlan create wlandev ath0 wlanmode sta wds

[repeater = ap + sta in 4-address mode]
Multi-BSS: VAP Creation

- VAP create succeeds only if all info is provided:
  - Parent device
  - Operating mode
  - Mode-specific state (e.g. BSSID for WDS link)

- VAP mode is fixed at create; simplifies work:
  - Check if multiple instances are supported
  - Check if combination is supported
  - Check if too many instances

- Device is involved so it can impose policy
Multi-BSS: Fixed Operating Mode

- Fixing the operating mode enables the use of mode-specific code:
  - Reduced memory footprint (e.g. no AP support)
  - Simpler (optimized) code
  - Existing code can still be reused

- Devices can load mode-specific firmware
Multi-BSS: Multi-BSSID

- Desirable for VAP’s to have unique station address (AP’s can make do by hiding SSID)
- Some VAP’s want to share station address
- Requires device support (hardware ACKs)
- Use 802.3 Local Address Management for address provisioning

PER-VAP MAC ADDRESS... Depends on device capability
Multi-BSS: User Visible Changes

- **Clone device first:**
  
  ```bash
  ifconfig wlan create wlandev ath0
  ```

- **After that everything is as before:**
  
  ```bash
  dhclient wlan0
  ```

- **Parent device available via sysctl:**
  
  ```bash
  # sysctl net.wlan.0
  net.wlan.0.%parent: ath0
  net.wlan.0.debug: 0
  ...
  ```

- **Changing shared state affects all vap’s**
  
  ```bash
  ifconfig wlan0 channel 36
  ```
Multi-BSS: Kernel Changes

- **State is split:**
  
  \[
  \text{struct xxx_softc + struct ieee80211com} \rightarrow \\
  \text{struct xxx_softc + struct ieee80211com +} \\
  \text{struct ieee80211vap + struct ieee80211vap + ...}
  \]

- **Reference** `ieee80211vap` **instead of** `ieee80211com` *(mechanical changes)*

- **VAP create/destroy callbacks to driver (policy)*

- **Changing shared state requires more care:**
  - State may be created by another vap (e.g. scan cache)
  - Notify all vap’s on state change
  - Restructure data to eliminate recalc of per-vap state
Multi-BSS: Kernel Changes (more)

- Eliminate “current mode”: a channel uniquely defines mode/band
- Coordinate certain virtual state:
  - Multicast filtering
  - Promiscuous mode
  - WME
  - ACL’s
  - 11g
  - 11h
  - Power save
  - Crypto
Multi-BSS: Input Handling

- Common station/neighbor table
- RX frames find station/neighbor using sender MAC address and this identifies VAP
- Multicast/unknown senders are broadcast to all VAP’s (can optimize if frame is unicast)

OVERHEAD...
Typically the same as single-BSS design
Multi-BSS: Output Handling

- Per-VAP send queue
- 802.11 processing partly done before passing to device send queue
  - WME traffic classification
  - Traffic diversion for stations in power-save mode
- 802.11 encap still done in driver (required for fast frame aggregation)
- Separate transmit queues enable system traffic control (e.g. load balancing)

Additional handoff to net80211 layer
Multi-BSS: Beacons

- Each IBSS/HostAP VAP must transmit a beacon at a regular interval
- Beacon frames must have TSF that is a multiple of the beacon interval

- Two choices:
  - Burst frames together
  - Stagger frame transmission over beacon interval
Multi-BSS: Beacons (continue)

- Bursting makes beacon delivery jittery from the stations’ POV (can mitigate by permuting order)
  - Power save
  - VoIP
- Staggering is good but TSF must be adjusted for beacon interval (requires device support)

OVERHEAD... Additional beacon timer interrupts
Multi-BSS: Crypto

- Unicast keys are easy
- Global key table is the issue:
  - WPA/802.11i Group keys: proper device support can deal with this
  - WEP keys: can do this in software but typically not hardware

OVERHEAD... May need to fallback to software
Multi-BSS: Summary

- New user-visible device model
- Operating mode fixed for life of vap
- Multi-BSSID requires device support
- Staggered beacons require TSF adjust
- Group key requires multicast search support
- WEP is problematic

OVERHEAD... Minimal unless we fallback to software
Ongoing/Future Work

- **Atheros SuperG support:**
  - fast frames
  - dynamic turbo

- **Scanning rewrite:**
  - Modular policies (in-kernel and user-mode)
  - Background scanning
  - Roaming

- **Atheros eXtended Range (XR) support**

- **Mesh network protocols** (e.g. 802.11s)

- **Multi-channel support?**
Contributors include...

| Joerg Albert | Eric Lammerts | Michael Renzmann |
| Satish Balay | Stephane Laroché | Paul Stewart |
| John Bicket  | Divy Le Ray    | Dieter Stolte  |
| Vivien Chappelier | Tai-hwa Liang | Jonas Tarnstrom |
| Greg Chesson | Warner Losh    | Bindu Therthala |
| Tong Chia    | Georg Lukas    | Carl Thompson  |
| Jeffrey Chung| Jouni Malinen  | Jim Thompson   |
| Richard Dawe | Tom Marshall   | Thorsten von Eicken |
| Srinivasa Duvvuri | Nick Moss | Carl Thompson |
| Guy Erb      | Atsushi Onoe   | Sebastian Weitzel |
| Joachim Gleissner | Nick Petroni | Dale Whitfield |
| Raja Gobi    | Andy Patti     | Alexander Wirtz |
| Kristian Hoffmann | Henry Qian | Michael Wong |
| William Kish | Mark Rakes     | David Young    |
| Mathieu Lacage | Bruno Randolph | Kevin Yu      |

CORPORATE SPONSORS...

Atheros, Vivato, Video54, 5Bridge, Red-M, Rincon Networks, Pelco, Visidaq, SuSE, 2Wire
Availability

- FreeBSD -current has everything up to the multi-BSS support
- Madwifi project for Linux tracks FreeBSD -current code
- NetBSD planning to import security and multimedia work