



# NetF1

for VxWorks

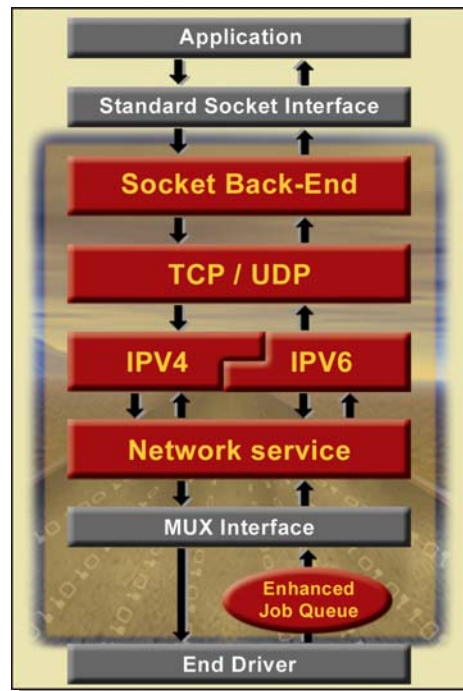
High Octane TCP/IP

**NetF1** is a robust, lean, flexible and high performance hardware-acceleration capable network stack implementation for VxWorks®. It includes a complete implementation of TCP, UDP, IP, IPv6, ICMP, and IGMP. **NetF1** can be configured to work in end-host mode or as an IPv4 or IPv6 router. In addition, it also provides a virtual routing framework, which allows the creation of multiple isolated and managed virtual "routers" in a single physical system. Given its ability to scale out optional features and its special emphasis on low resource environments, **NetF1** provides the essential functionality required by an embedded device to work either as a high performance connected end station or router.

## Seamless OS integration

**NetF1** requires no special VxWorks source code modifications and works seamlessly within the VxWorks environment while leveraging all native OS facilities. On the application side of the stack, **NetF1** installs itself as a socket backend, and at the bottom, it glues to the device-independent MUX

layer. Network applications continue to interact with VxWorks' standard socket front-end, allowing them to transparently access **NetF1**'s core high-performance enhanced functionality instead of the native networking stack via standard BSD sockets. Further, this architecture enables **NetF1** to work with any existing



## Features

- ❖ Lean, High Performance IPv4 and IPv6.
- ❖ Faster than native OS IPv4 stack and popular IPv6 add-ons.
- ❖ Highly concurrent and multi-instance capable.
- ❖ Includes standard TCP/IP protocols.
- ❖ Tested for protocol compliance and interoperability.
- ❖ Full IPv4 and IPv6 Routing Support.
- ❖ Virtual Routing Capabilities.
- ❖ Binary compatible with existing END drivers.
- ❖ Configurable via sysctl facility.
- ❖ Built-in Debugging Support.
- ❖ Support for big and little endian CPUs including x86, MIPS, ARM, XScale, PPC, and others.
- ❖ Royalty-free, source distribution.

data-link layer implementation and is binary compatible with standard END drivers without the need for driver or application source recompilation.

## VxWorks Edition Features

- ❖ Highly multi-threaded stack designed exclusively for VxWorks
- ❖ Requires no special VxWorks OS or networking source code for integration
- ❖ Binary compatible with standard VxWorks END drivers, including NPT ones
- ❖ Accessed through native VxWorks socket interface
- ❖ Support for Bootrom integration, including DHCP
- ❖ Support for driver buffer loaning to reduce data copies
- ❖ Enhanced memory management and partition support
- ❖ Integrated with Tornado / VxWorks Build and Install methodology
- ❖ Stack components can be configured using Tornado Project facility
- ❖ Easy integration with BSPs
- ❖ Native support for VxWorks 5.5.x, 5.4.x, 5.3.x and VxWorks AE

## Turbo V4

**NetF1** includes a high performance implementation of TCP/IP protocols and RFC implementations based on IPv4. This replaces the network stack provided with the OS and transparently provides a full-featured and highly concurrent alternative for a wide variety of connectivity applications for embedded devices. Included is a core set of functionality such as support for multicasting, IGMP v2, DHCP, and run-time configuration support through a BSD-style sysctl facility along with a

system-wide MIB tree for ease of management. Special attention to modularity within the core stack makes it possible to scale out any functionality not required by the application, as well as replace CPU-intensive functions with hardware equivalents (hardware acceleration) implemented in an ASIC, FPGA, or network processor.

### Hyper V6

With the IPv4 address space depleting rapidly, there is an accelerating move to adopt the new version of IP, IPv6, in embedded applications. Besides the 128-bit address space (as opposed to the 32-bit address space of IPv4), IPv6 also has important advantages, such as a simpler header format, better option and extension support, provisions for security and QoS management, and easier address auto-configuration. **NetF1**'s robust, fast and standards-based implementation of IPv6 enables embedded network applications to exploit the improvements in quality of service, security capabilities, and reliability inherent in this new technology with minimal integration effort. Overloaded routing systems can immediately leverage the routing memory space efficiency of IPv6 along with the extended address space available to it. **NetF1**'s IPv6 is as much as an order of magnitude faster than other embedded implementations. Full support for IPv6 security is available via TeamF1's V-IPSecure add-on IPsec/IKE

module.

### IPv6 Features

In addition to supporting standard V6 networking protocols such as ICMP6 and Multicasting, **NetF1** supports all the required features for an operational model as prescribed by the IPv6 related RFC standards.

**NetF1** hosts (non-routers) support the following features:

- ❖ Formation of interface identifier, link/site/global address generation from supplied prefix.
- ❖ Neighbor Discovery Algorithm.
- ❖ Site Renumbering/Link Address change processing.
- ❖ Multicast Scoping/Anycast addressing Support.
- ❖ Basic and Advanced Socket Extensions for new application development.
- ❖ ICMP Error message processing support.
- ❖ Router Solicitations.
- ❖ Security (IPsec) using V-IPSecure.

**NetF1** routers support:

- ❖ IPv6 packet forwarding.
- ❖ Static Route entries to be configured into routing table.
- ❖ Solicited/Unsolicited advertisements through 'radvd' module.

## Customization Flexibility

- ❖ Available in full-source format
- ❖ Run-time configurable through sysctl interface
- ❖ Easily hooked up to SNMP agents
- ❖ Customization hooks and callouts
- ❖ Unwanted components can be scaled out
- ❖ Choice of buffering schemes
- ❖ Choice of threading models
- ❖ Designed for future hardware acceleration

### IPv6 Extension Header Support

**NetF1** supports the following four IPv6 extension headers.

- ❖ Hop-by-Hop options header.
- ❖ Destination options header.
- ❖ Routing header.
- ❖ Fragment header.

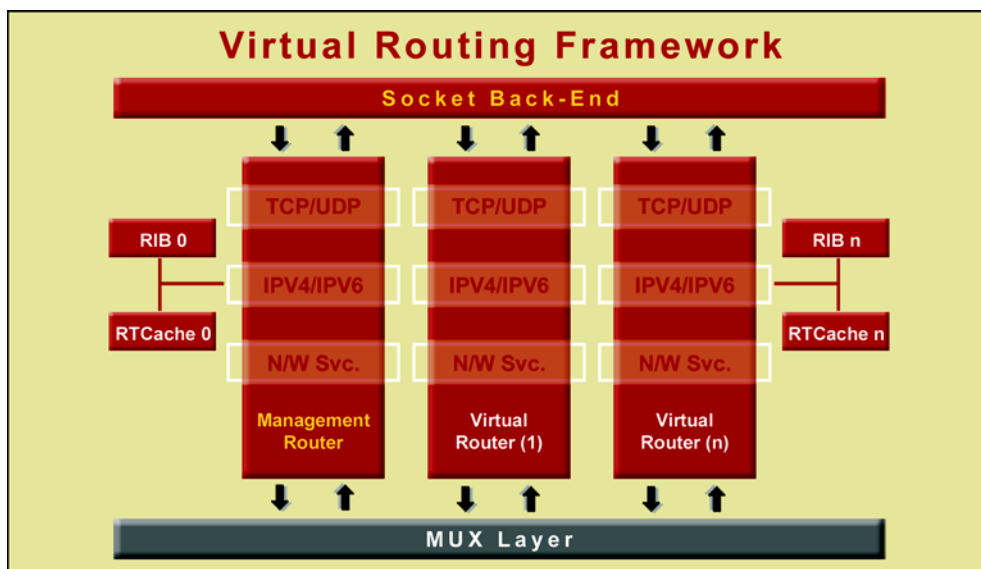
Support for security extensions are available by using **NetF1** with **TeamF1**'s V-IPSecure product.

### V4 to V6 Transition Support

Since most of the Internet and supporting routers are currently based on IPv4, **NetF1** provides built-in support for various transition mechanisms which allow the IPv6 stack to co-exist with IPv4 implementations. It enables the use of a dual-mode stack which handles both IPv4 and IPv6, as well as provides for a tunneling capability in which IPv6 packets can be tunneled through IPv4-unaware networks encapsulated in IPv4 packets. This gives embedded developers the ability to add support for the next version of Internet Protocol while maintaining backward compatibility with the current IPv4 standard.

### Virtual Routing Capabilities

"Virtual routers" are multiple logical router instances running on a single physical router system. By modularizing and isolating the different instances of the routing functionality, and dedicating separate resources of the physical



## IPV6 Tunneling



system to each logical instance, each virtual router can run independently as a full-fledged router with its own set of routing protocols. The **NetF1** virtual router framework replicates the networking stack up to the IP layer including data structures such as RTTables, Route Tables, interface lists and packet receive queues, to achieve complete segregation of user data. Each router instance is thus individually controlled and is isolated from the other instances. Further, the TCP/IP processing for each router instance is delegated to a specialized logical software module copy making it an ideal fit for custom hardware off-load applications, such as hardware-based packet forwarding in carrier-class applications. Virtual routing support within **NetF1** can be enabled for both IPv4 and IPv6, and combinations of the two can also co-exist. Third-party protocols including OSPF, BGP, and MPLS can run seamlessly within the virtual routing framework, attached to specific virtual router instances.

### Debugging / Trace Support

**NetF1** provides debugging support for application developers in the form of “show” routines and a trace library. The “show” routines keep track of, and display statistics for various protocols such as TCP, UDP, ICMP, IGMP, IPv6, etc. They can also be used to monitor the status of important tables such as multicast, neighbor discovery, and ARP caches. The trace library gives a view of function call sequence and execution time taken for each routine. Tracing is enabled through the use of special macros to instrument the code with

configurable resolution, so that the run-time view of the network stack execution can be captured in the form of a dynamic snapshot to a buffer that can be analyzed later. The flexible `sysctl` interface also allows for convenient control and observation of the stack characteristics.

### Binary Compatibility

All standard network applications such as Telnet, FTP, and HTTP servers that run on *VxWorks* using POSIX standards are binary compatible with **NetF1**, and can run without recompilation. Since the interface to the network stack is provided by means of standard sockets within *VxWorks*, all custom socket-based applications also run without modification. Enhanced features within **NetF1** can be activated through the use of special **NetF1** APIs. Further, **NetF1** is binary compatible with *VxWorks* binary END drivers and does not require driver source to be available. With minor modifications, older BSD-style drivers may also be used with **NetF1**.

### Management Framework

Configurable parameters in the various stack layers can be dynamically set using a `sysctl()` call, while their default values can be specified at compile time. **NetF1**'s internal viewable and controllable functions is organized into a Management Information Base (MIB) tree with hierarchically named `sysctl` variables corresponding to the various OIDs. The list of variables available through this interface can be configured in at compile time. `sysctl` handlers can be used to retrieve and set values of `sysctl` variables. Via this interface,

## Also Available

- ❖ **INSECTS Family**  
*NAT, Firewall and Queuing disciplines*
- ❖ **SSecure Family**  
*Security protocols (SSL, SSH, IPsec/IKE)*
- ❖ **AuthAgents Family**  
*Authentication agents: Kerberos, RADIUS*
- ❖ **Switchcraft Family**  
*802.n Switching components*

## Custom Solutions

**TeamF1**'s professional services can provide the resources and expertise to build customized implementations of **NetF1** including support for hardware acceleration within the network stack and implementation of application specific RFCs. **TeamF1** has specific experience in tying core hardware functionality such as checksumming, fast IP forwarding and segmentation support into the stack.

**NetF1** can be hooked up to any SNMP agent built into an embedded networking application.

## Built for VxWorks

**NetF1** is a drop-in component for *VxWorks 5.x* and *AE*. It has been extensively validated on a variety of CPU architectures, and this minimizes development and integration efforts. **NetF1** is especially optimized for *VxWorks* with support for multi-tasking, memory partitions, and abstractions that are lean, yet fast. It is compatible with all versions of *VxWorks 5.x*, with or without Network Protocol Toolkit (NPT) support, and works seamlessly with *WindNet*® routing protocols. For *VxWorks 5.4* and higher, **NetF1** can be configured in the Tornado® development environment as a set of components in Tornado's project facility and is designed to work with Tornado tools. For command-line builds independent of the Tornado facility, a build system that fits seamlessly into the *VxWorks* build system is also provided.

## Supported RFCs

RFC	Description
768	UDP
791	IP
792	ICMP
793	TCP
826	ARP
919	Broadcasting Internet Datagrams
922	Broadcasting Internet Datagrams with Subnets
950	IP Subnetting
1112	IP Multicast
1122	Requirements for Internet Hosts
1191	Path MTU
1323	TCP Extensions for High Performance (Large Windows)
1812	IPv4 Routers
1981	Path MTU for IPv6
2001	TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery
2018	TCP Selective Acknowledgements
2131	DHCP
2236	IGMPv2
2292	Advanced Sockets API for IPv6 (Partial)
2373	IPv6 Addressing Architecture
2374	IPv6 Aggregatable Global Unicast Address Format
2375	IPv6 Multicast Address Assignments
2452	IPv6 MIB for TCP
2454	IPv6 MIB for UDP
2460	IPv6
2461	IPv6 Neighbor Discovery
2462	IPv6 Stateless Address Autoconfiguration
2463	ICMPv6
2464	IPv6 over Ethernet
2465	IPv6 MIB general
2466	IPv6 MIB for ICMPv6 Group
2471	IPv6 Testing Address Allocation
2553	Basic Socket Interface Extensions for IPv6
2581	TCP Congestion Control
2582	New Reno TCP Fast Recovery

This product includes software from the BSD Kame project.

Email: [sales@TeamF1.com](mailto:sales@TeamF1.com)  
Web: [www.TeamF1.com](http://www.TeamF1.com)  
Ph: 510-505-9931 ext. 5  
Fax: (510) 505-9941



© 2002-2003 TeamF1, Inc.