



The Flock Internet Routing Engine (FIRE)

14th January 2026

Flock Networks Ltd

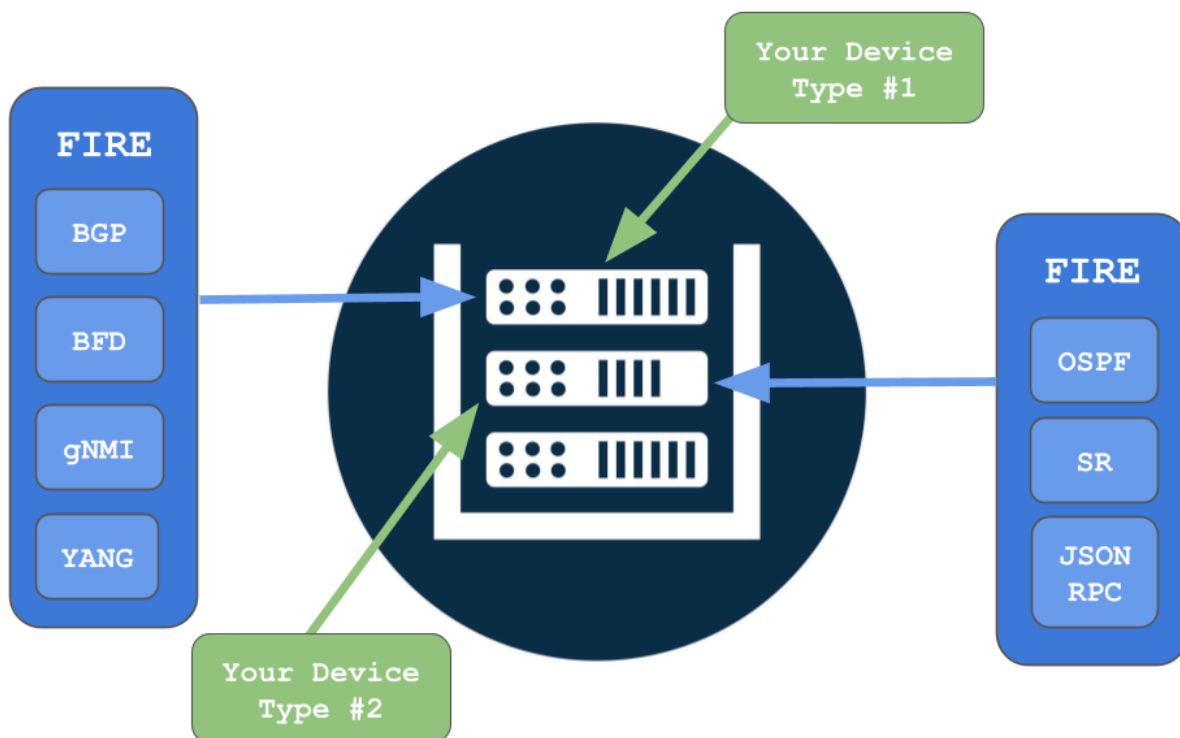
160 Kemp House,
London, EC1V 2NX

info@flocknetworks.com

Product Overview

The Flock Internet Routing Engine (FIRE) allows customers to quickly create a secure, performant and highly customizable IP routing suite that is easy to embed in any network device. With FIRE you get to choose what is in your device and just as importantly what is not.

The FIRE source code is available to license which enables network device manufacturers and network operators to rapidly create and deploy customized internet devices such as IP routers, IP gateways and “Internet of Things” (IOT) devices. Thousands of internet devices developed using FIRE have been deployed in production since 2022, and have proved to be reliable and efficient.



The FIRE source code is a platform agnostic library which implements a comprehensive set of internet RFC standards, a generic configuration and operation API, and a generic route programming API. Also included is a reference application that uses FIRE to implement a fully featured IP routing suite that runs on the Linux or SONiC platforms.

The FIRE source code is designed to be highly flexible. FIRE enables easy compile time customization and extension of the included protocol implementations, by providing embedded “modify hook” points in the source code.

FIRE Licensing

FIRE Source Code License

A FIRE source code license includes:

- A license to use the current source code in perpetuity
- Includes all source code for all the components in the FIRE library
- Source code for a fully functional routing daemon implementation for Linux
- Ability to create custom routing daemons
- Ability to implement proprietary protocol extensions
 - Customers keep the copyright to this additional code
- All documentation
 - The FIRE Developer Guide
 - The Linux routing daemon User Guide
- All test code
 - Unit Tests
 - Integration Tests
 - Platform mocking code
 - Protocol mocking code

FIRE Source Code Subscription

After a FIRE source code license has been purchased the source code subscription includes:

- Access to the latest source code via the github repository
- All bug fixes
- Support via github issues

FIRE is secure, fast, scalable and written in Rust

FIRE is compiled to match the hardware capabilities of the product that is being developed, from the lowest end consumer devices to the highest end internet route servers and Path Computation Element (PCE) servers.

FIRE is designed to scale at run time to match the available CPU cores. On a router with four logical CPU cores it can signal one billion BGP route updates across one thousand BGP neighbors in under 90s from startup. If that is too slow for your network requirements then just choose a router with more CPU cores.

Current IP routing suites were implemented decades ago. They no longer have a clean architecture so new feature development is very slow. They are written in unsafe languages such as C or C++, meaning a single incorrect line of code can crash the entire network device, or worse leave the network device running in an unknown and possibly insecure state.

FIRE is written exclusively in the Rust programming language. We are not alone in believing all performance critical infrastructure will be migrated to Rust.

*"We adopted Rust for its security and are seeing **a 1000x reduction in memory safety vulnerability density compared to Android's C and C++ code.**"*

Jeff Vander Stoep, Android (November 2025)

"Speaking of languages, it's time to halt starting any new projects in C/C++ and use Rust for those scenarios where a non-GC language is required. For the sake of security and reliability, the industry should declare those languages as deprecated."

Mark Russinovich, CTO Microsoft Azure (September 2022)

"roughly 70% of the security issues that the MSRC assigns a CVE to are memory safety issues. This means that if that software had been written in Rust, 70% of these security issues would most likely have been eliminated."

Ryan Levick, Microsoft Cloud Developer (July 2019)

The FIRE Development Team

FIRE has been developed by Flock Networks ® which was founded in 2019 by Nick Carter.

Nick has over 10 years of experience designing, implementing and operating some of the largest IP networks in the world. Nick also has over 25 years experience developing IP networking equipment where he specialized in the design and implementation of IP routing protocols and IP forwarding engines.

During the initial 18 months Nick designed and implemented the first version of FIRE, which included OSPF, BGP, the external API's and a comprehensive test suite. The Linux target platform implementation was also written. This became the first ever IP routing suite implemented in the Rust language.

From January 2021 onwards Nick was joined by a team of highly skilled developers to continue the development of FIRE. These developers were a mix of IP protocol implementation veterans and experts in the Rust language. The IP protocol developers had a combined 160 years of industry experience developing IP routing protocols. To date around 30 developer years of effort have been invested in creating FIRE.

FIRE system testing was taken over by a team of highly skilled QA engineers. With the growth in team members FIRE development was moved to a CI / CD pipeline with comprehensive unit test, integration test and system test coverage.

From January 2022 onwards an operations team was formed to roll out network devices built using FIRE. The FIRE telemetry and alarms infrastructure was finalized and put into production.

Thousands of IP network devices built using FIRE are in production, deployed across America, EMEA and Asia. FIRE has enabled these devices to be developed and deployed quickly. FIRE has proved to be reliable, fast and secure in production.

FIRE Contents in detail

FIRE contains the complete source code for the components shown in blue below. Customers are able to customize the implementation by implementing the green components. Some of the advantages of using FIRE are numbered, with explanations below.

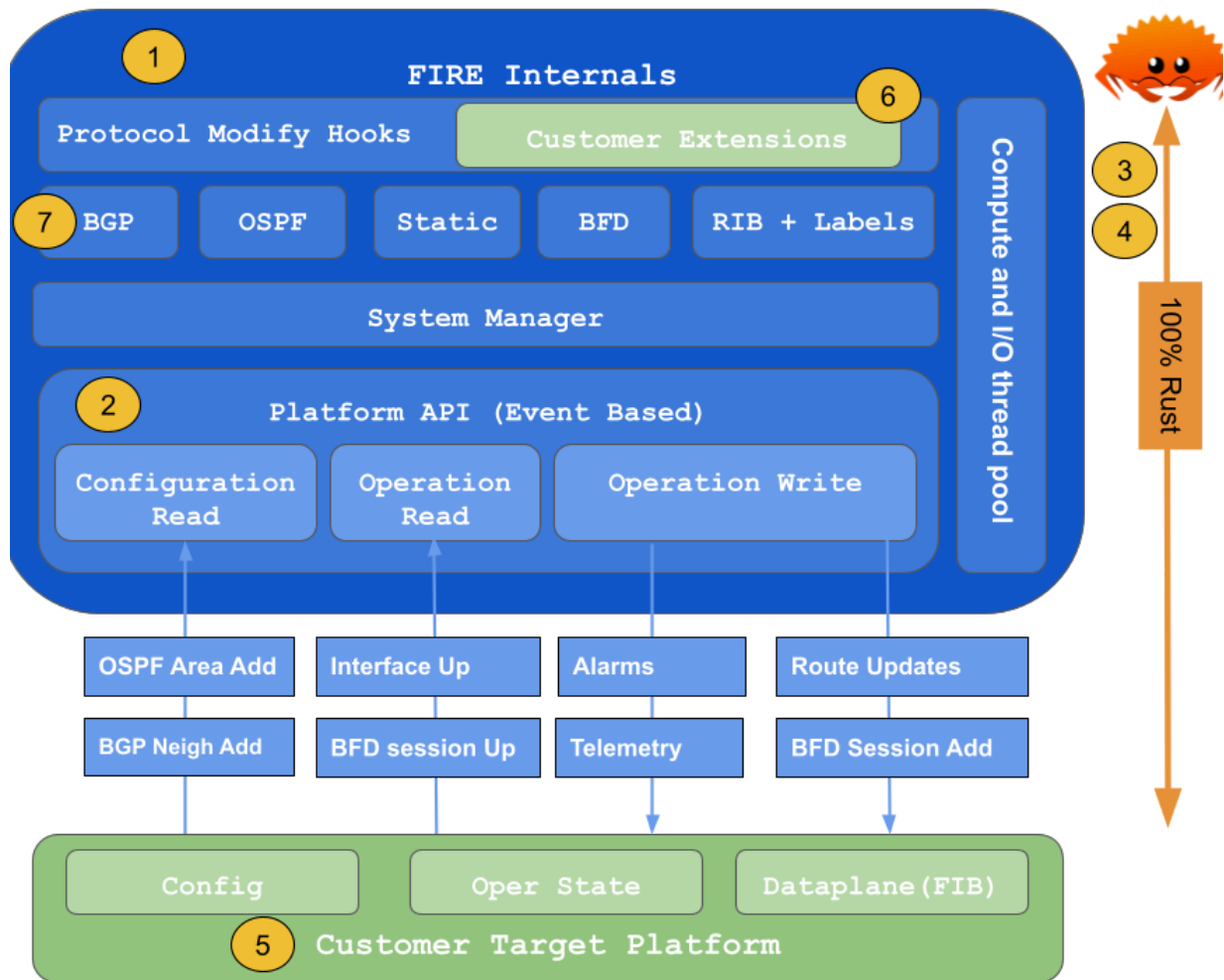


Figure 1: FIRE Contents

FIRE Advantages

1. **Clean architecture:** FIRE has been designed and developed from scratch by a team of highly experienced IP protocol experts, developers and testers. The best practices from the past 30 years of IP router design have been used, and designs that have historically proved to be unreliable and inflexible have been avoided.
2. **Easy customization:** FIRE is highly modular and presents strong flexible API's, making it trivial to customize. The generic configuration and operation event based API's allow FIRE to be controlled and monitored by any external model (YANG, NETCONF, REST, JSON, CLI, Prometheus...). The generic route programming event based API can easily be customized to program any kernel / data plane. The provided Linux reference implementation supports programming via Netlink and FPM (SONiC).
3. **Highly secure:** FIRE is entirely implemented in the Rust language. Rust is the only available memory safe systems programming language. Implementing applications in Rust has been shown to reduce security vulnerabilities by 70%. This security guarantee is provided at compile time so these bugs cannot exist in the deployed device. FIRE is the first and most widely deployed IP routing suite written in Rust.
4. **Highly performant:** The Rust language compile time memory safety guarantees means FIRE can run in a single operating system process. This means all the code has safe access to the same shared heap memory. FIRE does not need to perform any inter-process communication (IPC) making it far faster than other implementations. Rust also guarantees the absence of inter-thread "data races" at compile time. This eliminates the hardest problems when writing multithreaded code. FIRE has been designed to scale linearly vs the number of CPU cores available. FIRE is unique in that it can reliably utilize many cores in parallel whilst the network is converging, and then return to an idle state once converged. The FIRE implementation is lock free which means each thread can progress without being blocked by any other threads. This also means FIRE has a compile time guarantee that it cannot experience "dead lock", which can cause other implementations to fail at any time when running in production.
5. **Flexible to match the target platform:** FIRE can be compiled to execute as:
 - a. A single operating system thread, for use in low end devices
 - b. A thread per protocol, for use in mid range devices
 - c. A thread per protocol and a thread pool, for use in high end devices
6. **Extensible protocol behavior:** The FIRE API also contains "modify hooks" embedded at key points in each protocol's implementation. The standard IETF protocol behavior can be modified or extended at compile time. For example, to implement a customized BGP "best path decision process", a single Rust function can be written to override the default best path function. Because this is done at compile time there is no run time performance cost to achieve the new behavior.

7. **Extensive Protocol Support:** FIRE supports all the commonly used IETF RFC's for BGP, OSPFv2/v3, MPLS VPN, MPLS SR and BFD. The combination of the clean architecture and the expressiveness of the Rust language makes new feature development on FIRE incredibly fast. RFC's that would take months to implement on legacy IP routing suites, can be implemented in weeks.

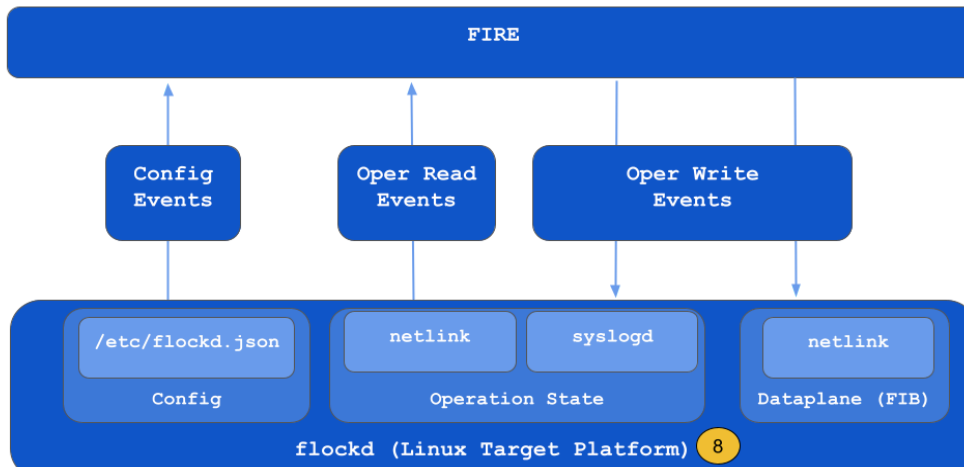


Figure 2: FIRE Reference Linux implementation

8. **Complete:** FIRE includes a production ready reference Linux implementation that works “Out of the box”. If your product requires standard Linux IP Routing Suite behavior, similar to Free Range Routing (FRR) or BIRD, then you don’t need to write any customization code. Just compile the reference implementation and deploy it. Alternatively this same implementation can be run on the SONiC network operating system, allowing a hardware forwarding / ASIC enabled network device to be quickly developed.

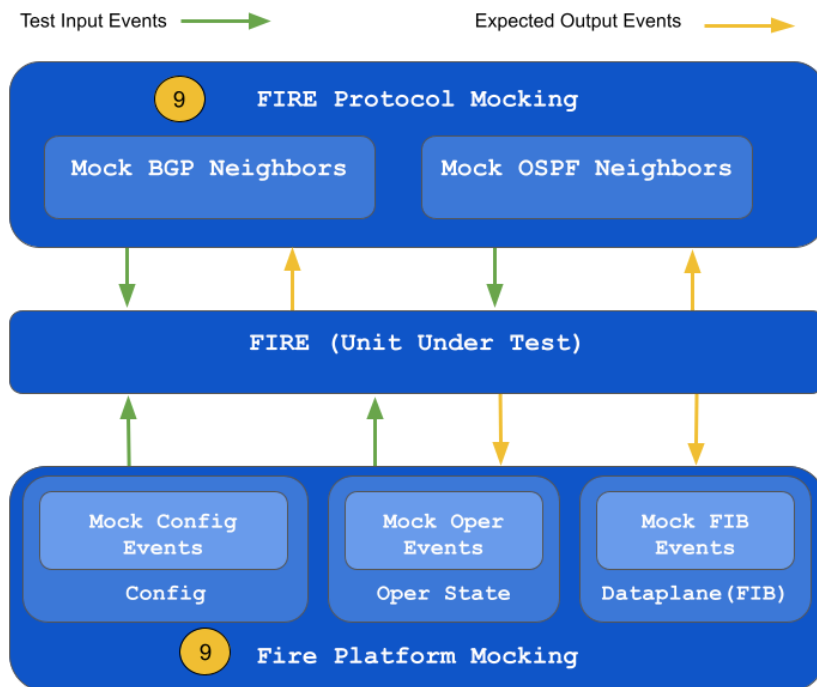


Figure 3: FIRE Testing Framework

9. **Highly reliable:** The FIRE source code includes a testing framework with hundreds of unit tests that test the internal behavior, and hundreds of integration tests that check the compliance of the external API and its conformance to RFC behavior. All these tests are run every time any single line of code is changed. This also provides a very agile development environment where changes can be comprehensively tested in seconds, rather than waiting hours for system test results. The integration tests also “fuzz test” the external API to check that malformed inputs are correctly handled and the tests track changes in memory usage.
10. **Clean code:** The FIRE source code is written to be easily read by other developers. This speeds up development and also allows the compiler to implement many optimizations. The code compiles with zero warnings. Static analysis of the code also produces zero warnings.
11. **Documented code:** The FIRE source code also includes a “developer guide” and a “user guide”. The “developer guide” explains the FIRE architecture, how to interface with and extend the architecture, and a detailed description of the implementation of each internal component. The “user guide” is aimed at end users that are using the reference Linux IP routing suite implemented by customizing FIRE. The “user guide” covers installation, configuration and operation. The contents of these guides are written in the markdown format and are included alongside the rest of the source code.

Summary

Creating new network devices using out-of-date IP routing suites is a mistake. The existing implementations are insecure and do not scale well on modern multi-core hardware. The complexity involved in adding new protocols has caused the internet infrastructure to stagnate.

The internet is now a utility and the networking industry needs to advance so that the deployed network devices are so stable, secure and efficient that they become transparent to end user applications.

If you have any questions or want to enquire about licensing FIRE then please email info@flocknetworks.com.