DISCLAIMER

Everything in this document shall not, under any circumstances, hold any legal liability whatsoever. Any usage of the data and information in this document shall be solely on the responsibility of the user. This document user has to take written consent from the author.

NETLINK

Netlink sockets in Linux provide a powerful and flexible mechanism for communication between user-space applications and the kernel. They enable efficient and structured data exchange, allowing user-space programs to interact with kernel modules, services, and subsystems. Netlink sockets are widely used in various use cases across different domains. Here's an overview of Netlink sockets and their common uses:

Netlink Socket Basics:

- Netlink sockets are created using the socket() system call with the address family set to AF_NETLINK.
- Each Netlink socket is associated with a specific Netlink protocol, represented by a protocol number.
- Netlink messages are sent and received using the send() and recv() system calls, respectively.
- Netlink messages consist of a Netlink message header (struct nlmsghdr) followed by optional attributes.
- Netlink attributes are used to carry data in a structured format within the message.

Use Cases of Netlink Sockets:

1. Network Configuration and Management:
    - Tools like iproute2 and ifconfig utilize Netlink sockets to manage network interfaces, configure IP addresses, manipulate routing tables, and set up network policies.
    - Netlink sockets are used for tasks such as adding and deleting routes, configuring network address translation (NAT) rules, and managing network namespaces.
2. Network Monitoring and Packet Analysis:
    - Network monitoring tools such as tcpdump and Wireshark leverage Netlink sockets for capturing and analyzing network packets.
    - By listening on a Netlink socket, these tools can receive real-time network events, monitor network interfaces, and retrieve traffic statistics.
3. Firewall Configuration and Management:
    - Netlink sockets are extensively used by firewall management tools like iptables and nftables.
    - They provide a means to interact with the kernel's netfilter subsystem, allowing user-space applications to configure firewall rules, manipulate packet filtering, and handle network address translation (NAT).
4. Kernel-User Communication:
    - Netlink sockets enable communication between user-space applications and kernel modules or

services.

- User–space applications can send requests to kernel modules using Netlink sockets, and kernel modules can respond or send events back to user–space.
- This functionality is utilized in various areas, such as device drivers, system monitoring, and control.

5. Virtualization and Containerization:

- Virtualization and containerization technologies, like Docker and Kubernetes, leverage Netlink sockets to manage virtual network interfaces, network namespaces, and network bridges.
- They use Netlink sockets to create and configure virtual network interfaces, set up network routing, and manage network isolation.

6. System Monitoring and Management:

- Netlink sockets are used by system monitoring tools and frameworks to gather information about system events, resource usage, and process management.
- Tools like systemd and sysstat utilize Netlink sockets to monitor system metrics, track system state changes, and perform system management tasks.

7. Security Frameworks and Access Control:

- Security frameworks such as SELinux and AppArmor rely on Netlink sockets to enforce security policies and handle security–related events.
- Netlink sockets facilitate communication between security modules and user–space applications, allowing for access control, privilege management, and intrusion detection.

Netlink sockets in Linux provide a versatile and efficient communication mechanism between user–space and the kernel. They are extensively used in network configuration, monitoring, kernel–user communication, virtualization, system monitoring, security frameworks, and more. Netlink sockets play a crucial role in enabling interaction and control of various kernel subsystems and services from user–space applications.

Types of netlinks

In Linux, Netlink supports different protocols or families, each designed for specific purposes. Here are some of the commonly used Netlink families and their use cases:

1. NETLINK_ROUTE:

- This Netlink family is primarily used for network configuration and management.
- It allows user–space applications to interact with the kernel's routing subsystem, manage network interfaces, manipulate routing tables, and configure network–related settings.
- Tools like iproute2, ifconfig, and network management daemons utilize NETLINK_ROUTE for network administration tasks.

2. NETLINK_NETFILTER:

- NETLINK_NETFILTER is used for communication with the kernel's netfilter subsystem, which handles packet filtering, network address translation (NAT), and connection tracking.
- User–space applications such as firewall management tools (iptables, nftables) and security frameworks utilize NETLINK_NETFILTER to configure firewall rules, manipulate packet filtering,

and handle NAT.

3. NETLINK_SELINUX:
    - This Netlink family is used for communication with the Security-Enhanced Linux (SELinux) subsystem.
    - It enables user-space applications to interact with SELinux policies, query security contexts, and enforce access control policies.
    - Tools and libraries related to SELinux, such as libselinux and semanage, use NETLINK_SELINUX for policy management and security-related tasks.
4. NETLINK_INET_DIAG:
    - NETLINK_INET_DIAG is used for network monitoring and diagnostics.
    - It provides information about active network connections, socket states, and network statistics.
    - Tools like ss (Socket Statistics) and network monitoring tools utilize NETLINK_INET_DIAG to retrieve information about network connections and perform network analysis.
5. NETLINK_XFRM:
    - NETLINK_XFRM is used for IPsec (Internet Protocol Security) key management and policy configuration.
    - It enables user-space applications to interact with the kernel's IPsec subsystem, set up security associations, manage cryptographic keys, and configure IPsec policies.
    - IPsec VPN clients, IPsec management tools, and security appliances use NETLINK_XFRM for IPsec-related operations.
6. NETLINK_GENERIC:
    - NETLINK_GENERIC is a generic Netlink family that can be used for various purposes.
    - It provides a flexible interface for user-space applications and kernel modules to communicate and exchange custom messages.
    - NETLINK_GENERIC is often used by specialized kernel modules or applications that require custom communication channels.

These are just a few examples of Netlink families commonly used in Linux. Each family has its specific purpose and allows user-space applications to interact with different kernel subsystems, configure network settings, manage security policies, perform network monitoring, and more.

Netlink NL80211

In Linux, the specific Netlink family used for communication with the 802.11 wireless subsystem is NL80211. NL80211 is a specialized Netlink family that is designed specifically for wireless networking configuration and management.

NL80211 provides a standardized interface for user-space applications to interact with the Linux kernel's wireless stack. It allows applications to control and monitor wireless network devices, configure wireless parameters, perform scanning for available networks, manage wireless connections, and retrieve wireless statistics.

NL80211 supports a wide range of operations related to 802.11 wireless networking, including:

1. Device Management:
    - Applications can use NL80211 to query information about available wireless network interfaces, enable or disable specific devices, and retrieve device capabilities and features.
2. Scanning:
    - NL80211 allows applications to initiate scanning for available wireless networks. It provides mechanisms to specify the scan parameters, retrieve the scan results, and filter the results based on criteria such as SSID, signal strength, and encryption.
3. Connection Management:
    - Applications can use NL80211 to configure wireless connections, initiate authentication and association with a specific network, set security parameters (e.g., encryption keys), and manage the connection state.
4. Regulatory and Channel Configuration:
    - NL80211 provides functions to query and configure wireless regulatory domains, set country codes, and manage channel configurations for regulatory compliance.
5. Monitoring and Statistics:
    - Applications can register for wireless events and receive notifications related to connection state changes, signal strength, channel switches, and other wireless-related events. NL80211 also allows retrieving statistics such as signal strength, noise level, and packet counts.

NL80211 is widely used by various user-space applications, libraries, and tools that require wireless network configuration and management. Some examples of tools that utilize NL80211 include iw, hostapd, wpa_supplicant, and network management utilities such as NetworkManager and wicd.

Using NL80211, developers can build custom applications or scripts to automate wireless network configuration, perform network diagnostics, conduct wireless surveys, and implement advanced wireless networking features.

To utilize NL80211 in your application, you would typically create a Netlink socket using the socket() system call with the family set to NETLINK_GENERIC, and then use Netlink messages with NL80211-specific attributes and operations to interact with the wireless subsystem.

Please note that working directly with NL80211 can be complex, and it is recommended to use higher-level libraries and tools that provide a more user-friendly interface for wireless network management, such as the libnl library or the iw command-line tool.

List of netlink protocols and numbers:

Here is a list of Netlink protocol numbers used in Linux along with their common uses:

1. NETLINK_ROUTE (0):
    - Used for network configuration, IP routing, and network device management.
2. NETLINK_UNUSED (1):

- Currently not used.

3. NETLINK_USERSOCK (2):
   - Used for communication between user-space and kernel-space sockets.
4. NETLINK_FIREWALL (3):
   - Used for communication with the kernel's firewall subsystem.
5. NETLINK_SOCK_DIAG (4):
   - Used for monitoring and diagnostics of socket information.
6. NETLINK_NFLOG (5):
   - Used for sending packets from the netfilter subsystem to user-space for logging or further processing.
7. NETLINK_XFRM (6):
   - Used for IPsec (IP Security) key management and policy configuration.
8. NETLINK_SELINUX (7):
   - Used for communication with the SELinux subsystem for enforcing security policies and managing access control.
9. NETLINK_ISCSI (8):
   - Used for communication with the iSCSI subsystem.
10. NETLINK_AUDIT (9):
    - Used for sending audit records to user-space for security auditing purposes.
11. NETLINK_FIB_LOOKUP (10):
    - Used for fast IP routing table lookups.
12. NETLINK_CONNECTOR (11):
    - Used for communication between the kernel and user-space connectors, such as udev.
13. NETLINK_NETFILTER (12):
    - Used for communication with the netfilter subsystem, which handles packet filtering, NAT, and connection tracking.
14. NETLINK_IP6_FW (13):
    - Used for communication with the IPv6 firewall subsystem.
15. NETLINK_DNRTMSG (14):
    - Currently not used.
16. NETLINK_KOBJECT_UEVENT (15):
    - Used for communication of kernel object events, such as device hotplug events.
17. NETLINK_GENERIC (16):
    - A generic protocol that provides a flexible interface for user-space applications and kernel modules to communicate using custom messages.
18. NETLINK_SCSITRANSPORT (18):

- Used for communication with the SCSI transport subsystem.

19.NETLINK_ECRYPTFS (19):

- Used for communication with the eCryptfs subsystem.

20.NETLINK_RDMA (20):

- Used for communication with the RDMA (Remote Direct Memory Access) subsystem.

21.NETLINK_CRYPTO (21):

- Used for communication with the kernel's cryptography subsystem.

These are the commonly used Netlink protocol numbers in Linux, each serving specific purposes within different subsystems of the kernel. It's important to note that not all protocols are widely used or have well-known user-space applications associated with them.