NFTables Testing Framework

A comprehensive testing framework for NFTables firewall configurations using Docker containers to simulate realistic game server traffic.

🗀 Project Structure

```
nftables-testing/
├── config/                     # Configuration files
│   └── nftables.conf          # NFTables firewall rules
├── src/                       # Source code
│   ├── server/               # Server components
│   │   └── nftables_test_server.py
│   ├── client/               # Client components
│   │   └── game_client_simulator.py
│   └── utils/                # Utility scripts
│       └── test_nftables.sh
├── docker/                   # Docker configurations
│   ├── Dockerfile.server     # Server container
│   ├── Dockerfile.client     # Client container
│   ├── docker-compose.yml    # Basic setup
│   └── docker-compose-game.yml # Game simulation
├── scripts/                  # Execution scripts
│   ├── run-auto-tests.sh     # Automated testing
│   ├── run-game-simulation.sh # Full game simulation
│   └── run-direct.sh         # Direct Linux execution
├── results/                  # Output directory
│   ├── *.log                 # Test logs
│   └── *.json                # Report files
└── docs/                     # Documentation
    └── README.md             # This file
```

🚀 Quick Start

Option 1: Complete Game Simulation (Recommended)

Simulates 18 game players for 2 minutes with comprehensive reporting:

```
./scripts/run-game-simulation.sh
```

Option 2: Basic Testing

Tests firewall rules without client simulation:

```
docker-compose -f docker/docker-compose.yml up --build
```

Option 3: Direct Linux Execution

For systems with nftables installed:

```
sudo ./scripts/run-direct.sh
```

📊 What Gets Tested

Server Side

- ☑ NFTables rule loading and validation
- ☑ Port listeners on game server ports
- ☑ Connection handling and statistics
- ☑ Rate limiting effectiveness
- ☑ Rule performance counters

Client Side (Game Simulation)

- ☑ 18 concurrent game players
- ☑ Realistic traffic patterns:
  - Server queries and discovery
  - Game join attempts
  - Gameplay packets (movement, combat)
  - Heartbeat/keepalive packets
  - TCP service connections
- ☑ Comprehensive traffic statistics
- ☑ Success/failure rate analysis

Ports Tested (from nftables.conf)

- **TCP**: 20, 21, 990, 1194, 3467, 6560, 6567, 6671, 8095, 9075, etc.
- **UDP**: 6962, 6963, 7787, 7797, 9696, 9697, 5555, 5556, 7766, 7767, etc.
- **Special**: Rate limiting, connection states, blocked ports

📋 Reports Generated

After testing, check the `results/` folder for:

- `server-report-*.json` - Server performance metrics
- `client-report-*.json` - Client traffic statistics
- `nftables-test-results.log` - Detailed execution logs

🛠 Configuration

NFTables Rules

Edit `config/nftables.conf` to modify firewall rules.

Game Simulation Parameters

Edit simulation parameters in the scripts:

- Number of players (default: 18)
- Duration (default: 120 seconds)
- Server IP/hostname
- Traffic patterns

Docker Components

Server Container

- Ubuntu 22.04 base
- NFTables + network utilities
- Python test server
- Privileged mode for netfilter access

Client Container

- Python 3.11 slim base
- Game client simulator
- Network testing tools

Development

Adding New Tests

1. Modify `src/server/nftables_test_server.py` for server-side tests
2. Modify `src/client/game_client_simulator.py` for client-side tests
3. Update port lists and traffic patterns as needed

Custom NFTables Rules

1. Edit `config/nftables.conf`
2. Update port lists in test scripts to match your rules
3. Run tests to validate changes

Use Cases

- **Game Server Setup** - Test firewall rules for game servers
- **Load Testing** - Simulate realistic player traffic
- **Security Validation** - Verify rate limiting and blocked ports
- **Performance Analysis** - Measure rule performance under load
- **CI/CD Integration** - Automated firewall testing

⚠ Requirements

- Docker and Docker Compose
- Privileged container support (for netfilter access)
- Linux kernel with nftables support (in container)

🚨 Security Note

This framework runs containers in privileged mode to access netfilter. Only use in testing environments, not production.

📞 Support

The framework provides detailed logging and error reporting. Check the `results/` directory for troubleshooting information.

NFTables Testing Framework - Professional firewall rule validation with realistic traffic simulation.