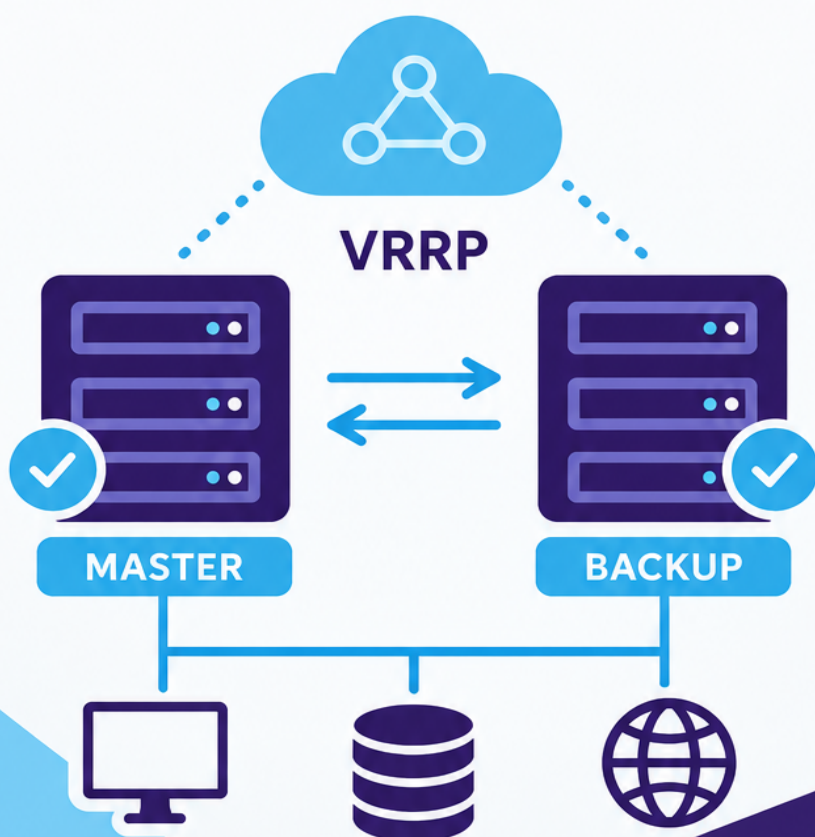


Mastering Keepalived

High Availability, VRRP, Failover
Architecture, Health Checking and
Production Operations



THOMAS HAYES

Mastering Keepalived

High Availability, VRRP, Failover Architecture, Health Checking and Production Operations

Steve Publications

This book is available at <https://leanpub.com/masteringkeepalived>

This version was published on 2026-10-03



This is a [Leanpub](#) book. Leanpub empowers authors and publishers with the Lean Publishing process. [Lean Publishing](#) is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

© 2026 Steve Publications

Contents

- High Availability, VRRP, Failover Architecture, Health Checking and Production Operations 1**
- Introduction 2**
 - What this book covers 3
 - Prerequisites and scope 4
 - Conventions and naming 5
- Chapter 1: High Availability Foundations 6**
 - The fundamental problem: the single address 6
 - What high availability actually means 6
 - The VIP pattern 7
 - Where Keepalived fits 7
 - The shared-nothing constraint 9
 - Understanding failure domains in HA pairs 9
 - The role of time in high availability 10
 - Summary 11
- Chapter 2: Linux Networking Prerequisites 12**
 - Interface addressing: primary, secondary, labels 12
 - The iproute2 suite 13
 - Routing tables and policy routing 13
 - ARP: the mechanism that makes VIPs work 14
 - IPv6 and Neighbor Discovery 15
 - Interface bonding and HA 15
 - VLANs and virtual interfaces 16
 - Network namespaces 16
 - Sysctl parameters that affect HA behavior 17
 - Multicast and IGMP 18
 - Summary 18

Chapter 3: VRRP Protocol Deep Dive 20

 Protocol history and RFC evolution 20

 The virtual router abstraction 20

 Master and Backup roles 20

 Priority and election 20

 Advertisements and heartbeat 20

 Master_Down_Timer and failure detection 20

 Preemption 21

 State machine and transitions 21

 Unicast mode 21

 Virtual MAC addresses 21

 Authentication and security 21

 Accept mode and packet reception 21

 Timer interactions and convergence 21

 Summary 22

Chapter 4: Keepalived Architecture and Design 23

 Project origins and evolution 23

 The process model 23

 Configuration parsing 23

 I/O multiplexing and event handling 23

 Netlink and kernel communication 23

 Signal handling 23

 Memory management 24

 Threading model and concurrency 24

 Interaction with iproute2 and other tools 24

 Debugging support 24

 Summary 24

Chapter 5: Installation and Deployment 25

 Version awareness 25

 Package installation on Debian and Ubuntu 25

 Package installation on RHEL, AlmaLinux, Rocky Linux 25

 Package installation on SUSE/openSUSE 25

 Package installation on Alpine Linux 25

 Building from source 25

 Service management with systemd 26

 SELinux and AppArmor considerations 26

 Initial configuration validation 26

CONTENTS

Upgrade procedures	26
Summary	26
Chapter 6: Configuration Fundamentals	27
Configuration file location and structure	27
The global_defs block	27
Including external files	27
Configuration validation	27
Reloading vs restarting	27
Common configuration mistakes	27
Best practices for maintainable configurations	28
Summary	28
Chapter 7: VRRP Instances: Core Configuration	29
The minimal vrrp_instance	29
state: MASTER or BACKUP	29
interface: binding to a network interface	29
virtual_router_id: the group identifier	29
priority: determining the master	29
advert_int: advertisement interval	29
nopreempt vs preempt	30
Authentication	30
Gratuitous ARP configuration	30
accept and no_accept	30
version: VRRP protocol version	30
A complete two-node example	30
Summary	30
Chapter 8: Virtual IP Addresses, Routes and Rules	32
virtual_ipaddress: basic VIP configuration	32
Multiple VIPs	32
virtual_ipaddress_excluded: the excluded pool	32
virtual_ipaddress_dev: binding VIPs to specific interfaces	32
Virtual routes	32
Virtual rules	32
How Keepalived programs the kernel	33
IPv6 virtual addresses	33
IP conflict behavior	33
Common mistakes	33

Summary	33
Chapter 9: Interface Tracking and Weighted Priorities	34
How interface tracking works	34
Weight semantics	34
Multiple tracked interfaces	34
Cumulative weight math	34
Interface tracking vs script tracking	34
Weight reverse	34
When NOT to use interface tracking	35
Multi-homed server example	35
Bonded interface tracking	35
Synchronization group tracking	35
Summary	35
Chapter 10: Health Checking: Scripts and Process Monitoring	36
vrrp_script: custom health checks	36
Designing effective health checks	36
Script example: checking a web server	36
Script example: checking HAProxy	36
Script permissions and security	36
Script execution performance	36
Script security: injection and environment	37
vrrp_track_process: process monitoring	37
Combining process and script tracking	37
Fall, rise and stability	37
Summary	37
Chapter 11: Notifications, Logging and State Changes	38
Notification scripts: when they fire	38
The notify catch-all	38
Notification scripts in sync groups	38
Logging with syslog	38
Debug levels	38
Structured JSON state dump	38
Log message taxonomy	39
Designing notification scripts	39
Integration with monitoring systems	39
Summary	39

Chapter 12: VRRP Synchronization and Grouping 40

 What synchronization groups do 40

 Configuring a sync group 40

 How synchronization works 40

 When to use synchronization groups 40

 Sync group notifications 40

 Weighted tracking in sync groups 40

 The global_tracking flag 41

 Example: load balancer pair 41

 Summary 41

Chapter 13: Multi-VIP and Active/Active Architectures 42

 True active/passive vs active/active 42

 Multiple VRRP instances on two nodes 42

 Service affinity with multiple VIPs 42

 Scaling: many VIPs across two nodes 42

 DNS round-robin with Keepalived 42

 Synchronization in active/active 42

 Weighted priorities for graceful degradation 43

 How many VIPs is too many? 43

 Summary 43

Chapter 14: Real-World HA Architectures: Common Patterns 44

 Pattern 1: simple two-node active/standby web server 44

 Pattern 2: highly available reverse proxy 44

 Pattern 3: highly available load balancer (HAProxy pair) 44

 Pattern 4: highly available API gateway 44

 Pattern 5: database connection router 44

 Common requirements across patterns 44

 Summary 45

Chapter 15: Advanced Architectures and Complex Environments 46

 Multi-tier architectures 46

 VLANs and Keepalived 46

 Bonded interfaces and Keepalived 46

 Virtual machines 46

 Network namespaces 46

 Containers and Keepalived 46

 Public cloud considerations 47

Summary	47
Chapter 16: Failure Modes, Pathologies and Split-Brain	48
Split brain: dual master	48
Asymmetric connectivity	48
Priority misconfiguration	48
Preemption problems	48
Health check false positives and false negatives	48
Interface flapping	48
Stale ARP and neighbor information	49
Duplicate VIPs from misconfiguration	49
Firewall blocking VRRP traffic	49
Script execution failures	49
Routing inconsistencies after failover	49
Virtualization-related issues	49
Cloud networking limitations	49
Summary	50
Chapter 17: Troubleshooting Methodology and Diagnosis	51
The troubleshooting mindset	51
Step 1: check Keepalived process status	51
Step 2: check VRRP state	51
Step 3: check VRRP advertisements with tcpdump	51
Step 4: check ARP and neighbor state	51
Step 5: check health check scripts	51
Step 6: check interface state	52
Step 7: check routing	52
Step 8: check firewall rules	52
Step 9: check for split brain	52
Step 10: check configuration syntax	52
Common problems and their diagnostic signatures	52
Distinguishing Keepalived issues from network or application issues	52
Summary	53
Chapter 18: Observability, Monitoring and Alerting	54
Key metrics to monitor	54
Using keepalived_exporter with Prometheus	54
Alerting on VRRP state changes	54
Monitoring VIP reachability	54

CONTENTS

Log-based monitoring	54
Grafana dashboards	54
Alerting philosophy	55
Integration with incident response	55
Summary	55
Chapter 19: Security Hardening	56
VRRP protocol security	56
Script execution security	56
Shell script injection risks	56
File permissions	56
SELinux and AppArmor	56
Firewall configuration	56
Network segmentation	57
Monitoring for anomalies	57
Secure upgrade procedures	57
Summary	57
Chapter 20: Performance, Scalability and Operational Limits	58
Resource usage	58
Many VIPs	58
Many VRRP instances	58
Health check overhead	58
Network traffic	58
Failover convergence time	58
System-level limits	59
Bottlenecks	59
Empirical guidance	59
Summary	59
Chapter 21: Production Operations: Runbooks, Upgrades, Maintenance	60
Configuration validation before changes	60
Controlled failover for maintenance	60
Upgrading Keepalived	60
Changing VIPs	60
Adding or removing nodes	60
Configuration drift detection	60
Backup and recovery	61
Post-incident review practices	61

Summary	61
Chapter 22: Production Reference	62
Quick command reference	62
Minimal two-node configuration template	62
Health check script template	62
Notification script template	62
Firewall rules template	62
Default values	62
Troubleshooting decision tree	63
Controlled failover procedure	63
Security checklist	63
Version compatibility notes	63
Useful log patterns	63
Conclusion	64
References	65

High Availability, VRRP, Failover Architecture, Health Checking and Production Operations

This book is a comprehensive, production-oriented guide to Keepalived, the Linux routing daemon that implements the Virtual Router Redundancy Protocol and provides high-availability failover for networked services. It is written for system administrators, DevOps engineers, network engineers and site reliability engineers who need to design, implement, operate and troubleshoot Keepalived-based HA architectures in real environments. The text takes you from the networking fundamentals through VRRP protocol mechanics, into detailed configuration of every major Keepalived feature and culminates in advanced architecture patterns, failure mode analysis, troubleshooting methodology, security hardening and production operations. All examples are complete and functional, all configuration directives are verified against current documentation and all architectural guidance is grounded in the realities of operating resilient infrastructure.

Introduction

A production web service has been running for seven months without incident when, at 03:14 on a Tuesday, a kernel panic takes down the primary load balancer. The VIP that 14 thousand users route through vanishes. The backup server exists, healthy and waiting, but the network does not know it. Three minutes pass before an engineer notices the alert, SSHes in and manually reassigns the IP. By then, 28 minutes of cumulative user downtime has occurred and the root cause is only half of the problem. The other half is that no one configured automatic failover correctly.

This is the gap Keepalived is designed to close. When a service depends on a single IP address - a load balancer front end, an API gateway, a database proxy, a reverse proxy - that IP becomes the single point of failure. Keepalived automates the transfer of that address to a backup node the moment the primary node fails, so clients never need to change their configuration and the service appears continuous.

Keepalived is not a simple heartbeat script. It is a mature routing daemon written in C, maintained since 2000 by Alexandre Cassen, that implements the Virtual Router Redundancy Protocol defined in IETF RFCs, integrates with the Linux kernel networking stack through netlink and extends VRRP with practical features like health-check scripts, interface tracking, synchronized failover groups and BFD-based fast detection. The daemon splits into child processes to isolate its responsibilities: one for VRRP state machines, one for health checks and LVS management and optionally one for BFD. It is deployed everywhere from bare-metal data centers to virtualized private clouds, from Kubernetes control plane proxies to traditional three-tier web architectures.

Yet Keepalived is frequently underused, misconfigured, or treated as a black box. Operators copy example configurations from blogs without understanding why priorities are set a certain way, what happens when both nodes think they are master, or how gratuitous ARP interacts with the ARP caches of downstream switches and clients. When failures occur, the resulting split-brain conditions, silent drops, or delayed convergence are blamed on the tool rather than on the gaps in understanding that led to the configuration.

This book takes a different approach. It treats Keepalived as a professional infrastructure tool that demands professional understanding. We begin with the networking and high-availability concepts that make Keepalived meaningful, then move through the VRRP protocol in depth so you understand the mechanics behind every state transition. From there we explore Keepalived's architecture, configuration model and every significant feature - not as isolated directives but as components of a coherent system. Each chapter builds on the previous ones, using consistent naming conventions and escalating complexity from a minimal two-node setup to multi-tier architectures with multiple VIPs, health checks, weighted tracking and automated notifications.

The central thesis of this book is straightforward: keeping services available is not about configuring the right syntax. It is about understanding how failures propagate through a system, how the Linux networking stack responds to address changes, how the VRRP protocol negotiates leadership between nodes and how Keepalived's extensions to VRRP allow you to express your operational requirements in configuration. When you understand these layers, the configuration becomes a direct expression of your design intent, rather than a collection of settings you hope work together.

What this book covers

The book is organized into a progressive journey:

Chapter 1, High Availability Foundations, establishes the problem Keepalived solves: why shared-nothing architectures need shared addresses, what VIPs actually do and where Keepalived fits among alternatives like Pacemaker, consul-template and cloud-native solutions.

Chapter 2, Linux Networking Prerequisites, covers the kernel networking concepts you need: interface addressing, routing tables, policy routing, ARP, Neighbor Discovery and the sysctl parameters that affect HA behavior.

Chapter 3, VRRP Protocol Deep Dive, is where most Keepalived guides fall short. We cover the protocol from first principles: the virtual router abstraction, master and backup roles, advertisements, priorities, preemption, the state machine, failure detection, multicast and unicast modes, virtual MAC addresses and the evolution of the protocol through RFC 2338, RFC 3768, RFC 5798 and RFC 9568. Understanding VRRP is essential because Keepalived's behavior is fundamentally defined by VRRP's rules.

Chapters 4 through 12 cover Keepalived's features in detail: its internal architecture and process model, installation across distributions, the configuration language and lifecycle, VRRP instances, virtual IPs and routes, interface tracking, health-check scripts, process monitoring, notifications and synchronization groups. For each feature we explain what it does, how it works internally, when it should and should not be used and what common mistakes look like.

Chapters 13 through 15 move into architecture. We explore multi-VIP and active/active patterns, then walk through common real-world topologies: highly available web servers, reverse proxies, load balancers, API gateways and database front ends. For each we describe the topology, traffic flow, failure domains, VIP ownership, election behavior, health-check strategy and operational trade-offs.

Chapters 16 and 17 focus on failure. We catalog failure modes and pathological scenarios - split brain, dual master, asymmetric connectivity, advertisement loss, flapping interfaces, stale ARP - and then develop a systematic troubleshooting methodology based on observable evidence.

Chapters 18 through 21 cover the operational layer: monitoring and alerting, security hardening, performance and scalability limits and production operations including runbooks, upgrades and maintenance procedures.

Chapter 22 is a production reference: directive summaries, command collections, configuration templates and troubleshooting decision trees organized for quick lookup.

Prerequisites and scope

This book assumes you are comfortable with Linux administration, basic networking and service management. You should understand what an IP address, subnet mask, default gateway and routing table are. You should be able to use `systemctl`, `ip` and text editors. Prior experience with VRRP or Keepalived is not required, but it will help.

The focus is on Keepalived as a high-availability and VRRP tool. Keepalived also provides a framework for Linux Virtual Server (LVS) load balancing through IPVS and we touch on that integration where relevant, but this book is not an LVS manual. Similarly, while we discuss Keepalived's use with HAProxy, Nginx

and other services, the emphasis is always on how Keepalived interacts with those services, not on configuring the services themselves.

Version awareness runs throughout. Keepalived is currently at version 2.4.3 (released July 2026), but distribution packages often ship older versions. We note version requirements for specific features and flag legacy syntax so you do not adopt practices that are deprecated. All configuration directives are verified against the official `keepalived.conf(5)` man page and project documentation.

Conventions and naming

Throughout the book we use a consistent set of names and addresses in examples:

- Primary node: `ha-primary.example.com` with management IP `192.168.1.10`
- Secondary node: `ha-secondary.example.com` with management IP `192.168.1.11`
- Virtual IP: `192.168.1.100`
- Interface: `eth0` (or `ens192` where distribution-specific)
- Virtual Router ID: `51`

Commands are shown as they should be typed, typically requiring root privileges. Where commands vary across distributions, we call out the differences explicitly. Inline citations in square brackets link to a reference list at the end of the book.

The rest of this book does not hold back on detail. Keepalived is a tool that operates at the boundary between networking and systems administration and mastery requires understanding both sides. Let us begin.

Chapter 1: High Availability Foundations

Before writing a single line of Keepalived configuration, you need to understand the problem it is solving, the constraints that make that problem hard and the alternatives that exist. This chapter establishes those foundations.

The fundamental problem: the single address

Modern applications are built with many redundant components: multiple web servers behind a load balancer, multiple database nodes, multiple caches, multiple queues. Individual nodes fail all the time and good architectures are designed so that one node dying does not bring down the service.

But somewhere in that architecture, clients need an entry point. A browser needs a URL. An API client needs an endpoint. A microservice needs a destination for its database connections. These entry points are expressed as IP addresses.

The problem is simple: if a single physical or virtual machine owns that IP address and that machine dies, the address dies with it. The load balancer is behind a VIP. The HAProxy cluster has a floating IP. The Kubernetes API server pair has a virtual address for control-plane clients. In every case, the high availability of the service depends on the high availability of that address.

What high availability actually means

High availability is often discussed in terms of “nines” - 99 percent uptime, 99.9 percent, 99.99 percent - but these numbers obscure more than they clarify. What actually matters in an HA design are three related concepts:

RTO (Recovery Time Objective) is the maximum acceptable time between a failure and service restoration. If your RTO is two minutes, your failover mechanism must detect the failure and restore service within that window.

RPO (Recovery Point Objective) is the maximum acceptable data loss measured in time. For most Keepalived use cases, RPO is irrelevant because Keepalived does not replicate application data – it only moves network addresses. RPO is a concern for the application layer, not the network redundancy layer.

Failure domains are the groupings of components that can fail together. If both nodes of your HA pair share the same power supply, the same network switch, or the same physical rack, they are in the same failure domain. When one fails due to an infrastructure event, the other likely will too. True high availability requires separating failure domains.

Keepalived addresses RTO and failure domains at the network level. It cannot make your architecture resilient to data center loss by itself, but it can reduce the RTO for node-level and network-path failures from human reaction time to seconds or milliseconds.

The VIP pattern

The solution to the single-address problem is the virtual IP, or VIP. A VIP is an IP address that does not belong permanently to any one machine. Instead, it is dynamically assigned to whichever node is currently designated as the leader for that address.

In a basic active/standby pair, the primary node holds the VIP. All client traffic flows to the VIP, which the primary receives and processes. If the primary fails, the secondary node takes ownership of the VIP. The secondary begins sending gratuitous ARP messages announcing that it now owns the VIP, switches update their MAC tables and client traffic starts flowing to the secondary.

From the client's perspective, nothing changed. They were talking to 192.168.1.100 and they are still talking to 192.168.1.100. The fact that the packet is now being handled by a different machine is transparent.

This transparency is the key insight. The VIP pattern decouples the service identity (the address clients use) from the physical or virtual machine that happens to be serving it at any given moment.

Where Keepalived fits

Keepalived is not the only tool that can move IPs around. The landscape of HA solutions includes:

Corosync and Pacemaker is a full-featured cluster stack. Corosync provides messaging and quorum, while Pacemaker provides resource management and fencing. It is powerful and flexible but complex. It is overkill if all you need is VIP failover with some health checking.

Heartbeat was an earlier cluster resource manager that predates Pacemaker. It is largely superseded but still appears in legacy environments.

Consul-template and similar service-mesh tools can template configuration files based on service discovery. They can drive VIP assignment but add operational complexity and dependency on Consul itself.

Cloud load balancers (ELB, ALB, Azure Load Balancer, GCP Load Balancer) abstract the entire VIP concept away. If you are fully cloud-native, this is often the right choice. But cloud LBs are expensive, add latency and do not work in bare-metal or private-cloud environments where networking is managed locally.

BGP-based solutions like MetalLB use BGP to announce VIPs to the network. They are popular in Kubernetes but require BGP-capable infrastructure.

Keepalived occupies a specific niche: it is a focused, low-overhead tool for environments where you manage your own networking and need reliable, automated VIP failover without the complexity of a full cluster stack. It is particularly suited to:

- Bare-metal and virtualized data centers with traditional Layer 2/Layer 3 networking
- Environments using HAProxy or Nginx as load balancers or reverse proxies
- Infrastructure where you need control over failover semantics
- Multi-tier architectures requiring multiple VIPs with coordinated failover

Keepalived is not a good choice when:

- You are fully committed to cloud-native managed services that provide built-in HA

- You need cluster-level resource management (starting/stopping multiple services, fencing nodes)
- You require geo-distributed HA with DNS-based failover at the Internet level
- Your network infrastructure does not support the required protocols (no multicast and no unicast routing for VRRP)

The shared-nothing constraint

Keepalived operates on a shared-nothing model. Neither node knows what the other is doing beyond what the VRRP protocol communicates. There is no shared storage, no quorum disk, no distributed consensus layer built into Keepalived itself.

This has implications:

1. **Configuration must be identical** on all nodes that participate in a VRRP group, except for values that are intentionally different (priority, initial state). If one node has a different `virtual_router_id` or VIP, the protocol breaks.
2. **Services must be independently running** on all nodes. If the VIP moves to the secondary, the application service must already be running there and ready to handle traffic. Keepalived does not start or stop services for you, though it can monitor whether they are running.
3. **Data consistency is your responsibility**. If the primary node has stateful data (session data, uploaded files, pending transactions), that state must be replicated or shared through some other mechanism before failover is safe.

This simplicity is also Keepalived's strength. There is no cluster manager to fail, no quorum to lose, no shared disk to become unavailable. If two nodes can reach each other on the network and they have the same configuration, they can perform failover.

Understanding failure domains in HA pairs

A two-node HA pair is the canonical Keepalived deployment. But its actual availability depends entirely on how independent the nodes really are.

Consider these failure scenarios:

- **Node-level failure:** The primary crashes, reboots, or has its OS hang. Keepalived fails over. This is the primary use case and works well.
- **Keepalived process failure:** Keepalived on the primary dies while the rest of the system is fine. The secondary detects the loss of VRRP advertisements and takes over. Good, unless the primary eventually comes back and causes a split-brain.
- **Application failure:** The web server crashes but Keepalived is running. By default, the primary will still hold the VIP, serving no traffic. This is why health checks are essential.
- **Network partition:** The two nodes can no longer see each other's VRRP advertisements, perhaps because of a switch failure or misconfiguration. Both may become master. This is split-brain and it is one of the hardest problems in HA design.
- **Shared infrastructure failure:** Both nodes share a power circuit, a cooling system, a switch, or a storage array. When that shared component fails, both nodes go down simultaneously. No amount of Keepalived configuration helps.

When designing a Keepalived deployment, you must identify all shared components and assess whether they constitute a risk. The best HA architecture in the world cannot protect you if both nodes share a single physical failure mode.

The role of time in high availability

High availability is ultimately about timing. Failover must happen fast enough that clients do not experience unacceptable disruption, but not so fast that transient glitches cause unnecessary failovers.

Consider the sequence when a master node fails:

1. The master stops sending VRRP advertisements.
2. Backups stop receiving advertisements and begin counting down their `Master_Down_Timer`.
3. When the timer expires, the highest-priority backup becomes master.
4. The new master assigns the VIP and sends gratuitous ARP.

5. Network devices update their MAC tables.
6. Client traffic begins flowing to the new master.

The total time from step 1 to step 6 is the failover time. VRRP timers, network conditions, ARP cache timeouts and client retry behavior all contribute. A well-tuned Keepalived deployment can achieve failover in three to ten seconds. With BFD integration, it can be sub-second.

On the flip side, if timers are too aggressive, a momentary network blip causes a failover that was not needed. The service is briefly disrupted on the backup even though the original master is fine. This is why understanding timer semantics and failure detection is central to Keepalived configuration.

Summary

Keepalived solves a specific problem: making a single IP address available across multiple nodes so that clients can reach a service even when individual nodes fail. It does this through the VRRP protocol, extended with practical features for health checking, tracking and synchronization.

To use Keepalived well, you need to understand that it operates on the shared-nothing model, that it does not manage application state and that its effectiveness is bounded by the independence of your nodes' failure domains. It is not a complete HA solution, but for the problem it addresses - automatic, protocol-based VIP failover - it is one of the most reliable and straightforward tools available in the Linux ecosystem.

The next chapter covers the Linux networking concepts that Keepalived depends on. Understanding how Linux manages IP addresses, routing, ARP and interfaces is essential for understanding how Keepalived manipulates these primitives to move VIPs between nodes.

Chapter 2: Linux Networking

Prerequisites

Keepalived is fundamentally a networking tool. It configures IP addresses on interfaces, manipulates routing tables and responds to ARP and Neighbor Discovery traffic. If you do not understand how Linux manages networking, Keepalived's behavior will seem opaque. This chapter covers the essential networking concepts, with attention to the details that matter for high-availability deployments.

Interface addressing: primary, secondary, labels

In Linux, an interface can have multiple IP addresses. Historically, the distinction between “primary” and “secondary” addresses was enforced. Modern Linux treats all addresses on an interface equally, but the terminology and some kernel behaviors persist.

When Keepalived assigns a VIP to an interface, it uses the same mechanisms the system uses for any IP address: the `ip` command from `iproute2` (or, in older versions, `ioctl` system calls). The VIP appears as another address on the interface, visible via `ip addr show`.

```
1 $ ip addr show eth0
2 2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
   ↪ group default qlen 1000
3     link/ether 52:54:00:12:34:56 brd ff:ff:ff:ff:ff:ff
4     inet 192.168.1.10/24 brd 192.168.1.255 scope global eth0
5         valid_lft forever preferred_lft forever
6     inet 192.168.1.100/32 scope global eth0
7         valid_lft forever preferred_lft forever
```

Here 192.168.1.10 is the interface's own address and 192.168.1.100 is the VIP assigned by Keepalived. Notice the `/32` prefix length on the VIP. This is deliberate: VIPs configured with `/32` are not considered part of any local

subnet for routing purposes. They only serve as destination addresses that the kernel will accept packets for.

Why does this matter? If a VIP is configured with the same prefix as the interface's own address (say /24 instead of /32), the kernel considers all addresses in that subnet as locally reachable. This can cause routing confusion and ARP behavior that is not what you want for a floating IP. The safest practice is to use /32 for IPv4 VIPs and /128 for IPv6 VIPs.

The iproute2 suite

Keepalived uses iproute2 to manage networking in modern Linux. The `ip` command is part of iproute2 and understanding its output is essential for diagnosing Keepalived issues.

Key commands:

- `ip addr show` - lists all interfaces and their addresses
- `ip route show` - lists the routing table
- `ip rule show` - lists policy routing rules
- `ip neigh show` - lists the ARP/ND neighbor cache
- `ip link show` - lists interfaces and their state

When Keepalived transitions to master and assigns a VIP, you will see the VIP appear in the output of `ip addr show`. When it transitions away from master, Keepalived removes the VIP. If the VIP does not appear or disappear as expected, the problem is either in Keepalived's configuration or in the kernel's networking state.

Routing tables and policy routing

Linux maintains routing tables that determine how packets are forwarded. The main routing table (table 254) is consulted for most packets, but Linux supports multiple routing tables and policy-based routing rules that can direct traffic based on source address, destination, interface, or other criteria.

This becomes important when Keepalived is configured with virtual routes or virtual rules. A virtual route is a static route that Keepalived installs only

when a VRRP instance is in the master state. When the instance becomes backup, the route is removed. This allows the master to advertise additional routes that backups should not.

Policy routing is more powerful. A rule can say “all traffic from VIP X should use routing table Y.” This is useful in multi-homed or multi-interface setups where traffic entering via the VIP must exit via a specific interface or through a specific gateway.

Keepalived’s `virtual_rules` configuration block installs these rules when the instance is master and removes them when it becomes backup. Without policy routing, you cannot guarantee that traffic received on the VIP will be routed correctly when the VIP moves to a different node.

ARP: the mechanism that makes VIPs work

The Address Resolution Protocol (ARP) maps IP addresses to MAC addresses on Layer 2 networks. When a client wants to send a packet to 192.168.1.100, it must first know the MAC address associated with that IP. It sends an ARP request (“Who has 192.168.1.100? Tell me your MAC”) and the owner responds with its MAC address.

Here is where Keepalived matters. When the primary node holds the VIP, clients have an ARP cache entry mapping 192.168.1.100 to the primary’s MAC address. When the VIP moves to the secondary node, clients’ ARP caches still point to the primary. Traffic continues to go to the primary’s MAC, but the primary is no longer the master and is no longer processing packets for that VIP. The traffic is lost.

The solution is gratuitous ARP. When the secondary becomes master and assigns the VIP to itself, it immediately sends gratuitous ARP messages announcing “192.168.1.100 is at MAC XX:XX:XX:XX:XX:XX” (its own MAC). These messages update the ARP caches of clients and switches so that subsequent traffic is sent to the correct node.

Gratuitous ARP is the bridge between Keepalived’s VIP assignment and network-wide convergence. If gratuitous ARP messages are lost, blocked by a firewall, or ignored by clients with long ARP cache timeouts, failover will be delayed or fail entirely.

Keepalived controls gratuitous ARP behavior through the `garp_master_*` parameters:

- `garp_master_repeat` - number of GARP messages to send on master transition
- `garp_master_delay` - delay between GARP messages, in seconds
- `garp_master_refresh` - periodic GARP refresh interval for master, in seconds (0 disables)
- `garp_master_refresh_repeat` - number of messages sent per refresh cycle

In production, it is common to increase these values to ensure GARP messages are not lost and reach all clients and switches. A typical production setting might send 3-5 GARP messages with a small delay between them.

IPv6 and Neighbor Discovery

IPv6 does not use ARP. Instead, it uses Neighbor Discovery Protocol (NDP), defined in RFC 4861. The equivalent of ARP request/reply is the Neighbor Solicitation/Neighbor Advertisement exchange. The equivalent of gratuitous ARP is the gratuitous Neighbor Advertisement.

Keepalived supports IPv6 VIPs and sends gratuitous Neighbor Advertisements automatically when taking over an IPv6 VIP. IPv6 addresses are configured in the same `virtual_ipaddress` block as IPv4 ones (in their own VRRPv3 instance) and there is no separate ND-delay directive - ND behavior follows the GARP timing of the instance.

If you are running a dual-stack environment with both IPv4 and IPv6 VIPs, you must configure and tune both sets of parameters. A common mistake is tuning the IPv4 GARP behavior and forgetting that IPv6 uses a different mechanism entirely.

Interface bonding and HA

Interface bonding (Linux bonding driver) combines multiple physical interfaces into a single logical interface for redundancy and/or throughput. When Keepalived is configured on a bonded interface, the bonding mode matters significantly.

Active-backup bonding (mode 1) is the most compatible with Keepalived. Only one physical link is active at a time and failover between physical links

happens independently of Keepalived. Keepalived sees the bonded interface as a single interface and operates normally.

Active-active modes (LACP/802.3ad, mode 4) can also work but introduce complexity. If the bond itself fails, Keepalived should detect it via interface tracking and adjust its priority or transition to backup. If the bond appears up but one member link has failed, Keepalived may not notice the degraded state unless you add health checks for the underlying connectivity.

The key principle: bonding provides link-level redundancy. Keepalived provides node-level redundancy. They address different failure domains and can coexist, but they are not substitutes for each other.

VLANs and virtual interfaces

When multiple VLANs traverse a server, each VLAN typically needs its own Keepalived VRRP instance with its own VIP. VLAN interfaces in Linux are created with a dot notation: `eth0.100` for VLAN 100 on interface `eth0`.

Each VLAN interface is a separate Layer 2 domain. VRRP advertisements sent on one VLAN are not visible on another VLAN. Therefore, you need separate VRRP instances, separate `virtual_router_id` values (or the same VRID if VLANs are completely isolated) and separate VIPs.

This becomes relevant in multi-VIP and multi-service architectures. A load balancer might have a management VIP on VLAN 10, an external-facing VIP on VLAN 100 and an internal-service VIP on VLAN 200, each with its own Keepalived instance.

Network namespaces

Network namespaces provide isolated networking stacks within a single Linux kernel. Each namespace has its own interfaces, addresses, routing tables and firewall rules. This is the foundation of container networking.

Keepalived supports running in a network namespace via the `-n` command line flag. This allows you to run multiple independent Keepalived instances on a single machine, each managing VIPs in a different namespace.

This is useful for:

- Multi-tenant environments where different tenants have isolated VIPs
- Container-based deployments where each service has its own namespace
- Testing and development setups

When running in a namespace, Keepalived manages only the interfaces and addresses within that namespace. It does not see or affect the host's networking directly.

Sysctl parameters that affect HA behavior

Several kernel parameters in `/proc/sys/net/` affect how Keepalived operates. Understanding these helps you diagnose and tune HA behavior.

arp_ignore controls how the kernel responds to ARP requests for local IP addresses:

- 0 (default): respond if the target IP is configured on any interface
- 1: respond only if the target IP is on the receiving interface

For VIP configurations where multiple nodes may momentarily share the same IP during failover, a non-zero value can prevent the wrong node from responding to ARP requests.

arp_announce controls how the kernel advertises its source IP in ARP requests:

- 0 (default): use any local address
- 1: try to avoid addresses not on the receiving interface
- 2: always use the best local address for the target

Higher values are more conservative and can help in complex multi-interface setups.

accept_local controls whether the kernel accepts packets destined for local addresses arriving on interfaces where those addresses are not configured:

- 0 (default): only accept on the correct interface
- 1: accept on any interface

For VIP failover where the VIP might be temporarily unreachable on one node while configured on another, this can matter.

proxy_arp enables proxy ARP, where a system answers ARP requests for addresses it does not own. This is generally not needed for Keepalived and can cause confusion. The Keepalived documentation recommends setting `proxy_arp` to 0 on interfaces hosting VIPs.

These parameters are typically set via `sysctl.conf` entries or runtime `sysctl` commands. For example:

```
1 sysctl -w net.ipv4.conf.all.arp_ignore=1
2 sysctl -w net.ipv4.conf.all.arp_announce=2
3 sysctl -w net.ipv4.conf.all.proxy_arp=0
```

Multicast and IGMP

VRRP advertisements are multicast packets. For IPv4, they use multicast address 224.0.0.18 and protocol number 112. For IPv6, they use ff02::12 with next header 112.

Multicast traffic requires support at multiple levels:

1. **The local system** must be allowed to join the multicast group. On Linux, this is handled by the kernel's IGMP implementation.
2. **The network switch** must allow multicast traffic to pass. Some switches have IGMP snooping enabled, which restricts multicast to ports that have explicitly joined the group. If IGMP snooping is misconfigured, VRRP advertisements may be dropped.
3. **The receiving node** must be listening on the multicast address. Keepalived sets this up automatically when configured.

When VRRP fails to work, multicast support is a common culprit. Testing involves verifying that multicast packets are actually being sent and received between nodes.

The alternative to multicast is unicast mode, which we will cover in depth in the VRRP chapter. Unicast mode sends VRRP advertisements as direct unicast packets to configured peers. It is required in cloud environments like AWS, Azure and GCP where multicast is blocked by default.

Summary

Linux networking is complex, but Keepalived's interaction with it is focused. It assigns and removes IP addresses from interfaces, optionally installs routes and rules, sends gratuitous ARP/ND messages to update network caches and listens for multicast or unicast VRRP advertisements. All of this is built on standard Linux primitives.

The key takeaways are:

- VIPs should be configured with /32 prefix to avoid routing confusion.
- Gratuitous ARP is essential for failover convergence and must be tuned for production.
- ARP, NDP, multicast and routing tables all influence how quickly and reliably failover happens.
- Kernel sysctl parameters can help or hinder HA behavior depending on configuration.
- Bonding, VLANs and namespaces add complexity but are fully supported.

With this foundation, we can now dive into the VRRP protocol itself - the mechanism that Keepalived uses to coordinate VIP ownership between nodes.

Chapter 3: VRRP Protocol Deep Dive

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Protocol history and RFC evolution

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

The virtual router abstraction

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Master and Backup roles

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Priority and election

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Advertisements and heartbeat

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Master_Down_Timer and failure detection

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Preemption

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

State machine and transitions

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Unicast mode

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Virtual MAC addresses

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Authentication and security

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Accept mode and packet reception

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Timer interactions and convergence

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 4: Keepalived Architecture and Design

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Project origins and evolution

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

The process model

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Configuration parsing

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

I/O multiplexing and event handling

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Netlink and kernel communication

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Signal handling

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Memory management

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Threading model and concurrency

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Interaction with iproute2 and other tools

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Debugging support

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 5: Installation and Deployment

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Version awareness

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Package installation on Debian and Ubuntu

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Package installation on RHEL, AlmaLinux, Rocky Linux

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Package installation on SUSE/openSUSE

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Package installation on Alpine Linux

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Building from source

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Service management with systemd

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

SELinux and AppArmor considerations

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Initial configuration validation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Upgrade procedures

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 6: Configuration

Fundamentals

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Configuration file location and structure

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

The global_defs block

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Including external files

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Configuration validation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Reloading vs restarting

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Common configuration mistakes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Best practices for maintainable configurations

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 7: VRRP Instances: Core Configuration

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

The minimal vrrp_instance

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

state: MASTER or BACKUP

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

interface: binding to a network interface

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

virtual_router_id: the group identifier

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

priority: determining the master

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

advert_int: advertisement interval

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

nopreempt vs preempt

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Authentication

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Gratuitous ARP configuration

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

accept and no_accept

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

version: VRRP protocol version

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

A complete two-node example

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 8: Virtual IP Addresses, Routes and Rules

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

virtual_ipaddress: basic VIP configuration

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Multiple VIPs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

virtual_ipaddress_excluded: the excluded pool

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

virtual_ipaddress_dev: binding VIPs to specific interfaces

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Virtual routes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Virtual rules

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

How Keepalived programs the kernel

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

IPv6 virtual addresses

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

IP conflict behavior

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Common mistakes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 9: Interface Tracking and Weighted Priorities

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

How interface tracking works

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Weight semantics

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Multiple tracked interfaces

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Cumulative weight math

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Interface tracking vs script tracking

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Weight reverse

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

When NOT to use interface tracking

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Multi-homed server example

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Bonded interface tracking

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Synchronization group tracking

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 10: Health Checking: Scripts and Process Monitoring

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

vrrp_script: custom health checks

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Designing effective health checks

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Script example: checking a web server

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Script example: checking HAProxy

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Script permissions and security

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Script execution performance

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Script security: injection and environment

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

vrrp_track_process: process monitoring

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Combining process and script tracking

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Fall, rise and stability

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 11: Notifications, Logging and State Changes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Notification scripts: when they fire

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

The notify catch-all

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Notification scripts in sync groups

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Logging with syslog

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Debug levels

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Structured JSON state dump

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Log message taxonomy

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Designing notification scripts

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Integration with monitoring systems

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 12: VRRP Synchronization and Grouping

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

What synchronization groups do

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Configuring a sync group

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

How synchronization works

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

When to use synchronization groups

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Sync group notifications

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Weighted tracking in sync groups

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

The global_tracking flag

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Example: load balancer pair

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 13: Multi-VIP and Active/Active Architectures

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

True active/passive vs active/active

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Multiple VRRP instances on two nodes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Service affinity with multiple VIPs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Scaling: many VIPs across two nodes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

DNS round-robin with Keepalived

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Synchronization in active/active

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Weighted priorities for graceful degradation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

How many VIPs is too many?

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 14: Real-World HA Architectures: Common Patterns

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Pattern 1: simple two-node active/standby web server

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Pattern 2: highly available reverse proxy

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Pattern 3: highly available load balancer (HAProxy pair)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Pattern 4: highly available API gateway

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Pattern 5: database connection router

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Common requirements across patterns

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 15: Advanced Architectures and Complex Environments

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Multi-tier architectures

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

VLANs and Keepalived

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Bonded interfaces and Keepalived

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Virtual machines

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Network namespaces

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Containers and Keepalived

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Public cloud considerations

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 16: Failure Modes, Pathologies and Split-Brain

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Split brain: dual master

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Asymmetric connectivity

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Priority misconfiguration

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Preemption problems

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Health check false positives and false negatives

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Interface flapping

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Stale ARP and neighbor information

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Duplicate VIPs from misconfiguration

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Firewall blocking VRRP traffic

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Script execution failures

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Routing inconsistencies after failover

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Virtualization-related issues

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Cloud networking limitations

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 17: Troubleshooting Methodology and Diagnosis

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

The troubleshooting mindset

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Step 1: check Keepalived process status

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Step 2: check VRRP state

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Step 3: check VRRP advertisements with tcpdump

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Step 4: check ARP and neighbor state

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Step 5: check health check scripts

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Step 6: check interface state

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Step 7: check routing

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Step 8: check firewall rules

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Step 9: check for split brain

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Step 10: check configuration syntax

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Common problems and their diagnostic signatures

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Distinguishing Keepalived issues from network or application issues

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 18: Observability, Monitoring and Alerting

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Key metrics to monitor

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Using keepalived_exporter with Prometheus

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Alerting on VRRP state changes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Monitoring VIP reachability

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Log-based monitoring

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Grafana dashboards

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Alerting philosophy

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Integration with incident response

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 19: Security Hardening

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

VRRP protocol security

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Script execution security

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Shell script injection risks

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

File permissions

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

SELinux and AppArmor

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Firewall configuration

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Network segmentation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Monitoring for anomalies

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Secure upgrade procedures

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 20: Performance, Scalability and Operational Limits

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Resource usage

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Many VIPs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Many VRRP instances

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Health check overhead

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Network traffic

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Failover convergence time

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

System-level limits

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Bottlenecks

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Empirical guidance

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 21: Production Operations: Runbooks, Upgrades, Maintenance

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Configuration validation before changes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Controlled failover for maintenance

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Upgrading Keepalived

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Changing VIPs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Adding or removing nodes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Configuration drift detection

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Backup and recovery

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Post-incident review practices

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Chapter 22: Production Reference

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Quick command reference

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Minimal two-node configuration template

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Health check script template

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Notification script template

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Firewall rules template

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Default values

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Troubleshooting decision tree

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Controlled failover procedure

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Security checklist

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Version compatibility notes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Useful log patterns

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

Conclusion

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.

References

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/masteringkeepalived>.