

KEYSPACE

2025 / 北京

# Valkey over MPTCP

报告人：唐葛亮

日期：2025.12



# 目录

/ 个人简介

/ MPTCP技术简介

/ Valkey over MPTCP

/ MPTCP生态支持



# 01

## 个人简介

# 个人简介

## 唐葛亮

麒麟软件技术总监



Linux内核MPTCP  
Maintainer



openEuler技术委员会委员  
(2025-2026届)

</> Linux内核开发

咚 MPTCP协议栈



Linux内核上游贡献

2015年起向Linux内核主线贡献补丁  
**900+**

2019年起专注Linux内核MPTCP网络协  
议栈的开发，为该模块的**核心开发者**、  
主要贡献者、Reviewer及Maintainer



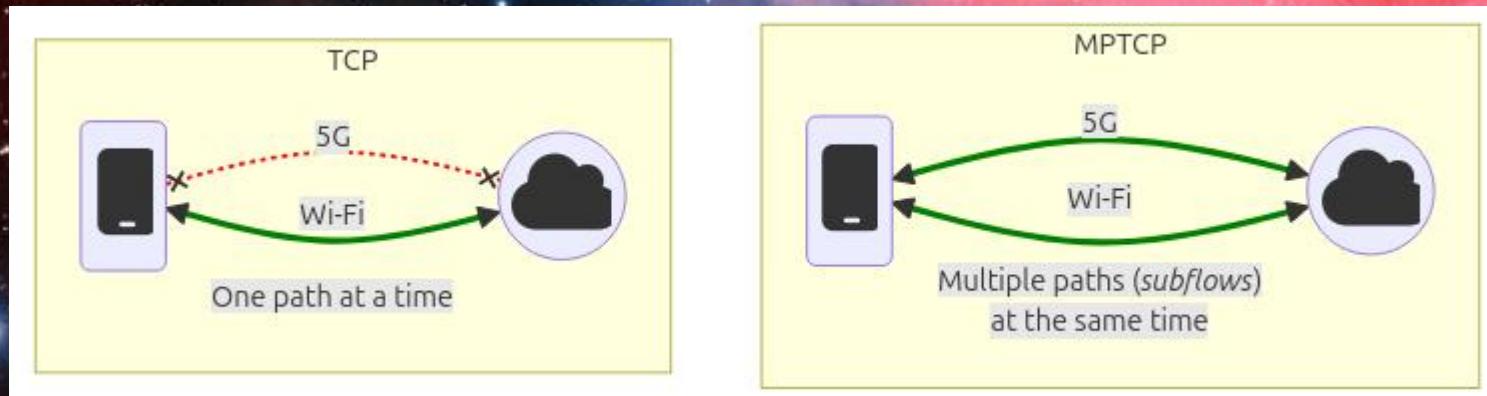
## 02

# MPTCP技术简介

# MPTCP技术简介

— 多路径TCP传输协议

- 多路径TCP或MPTCP是标准TCP的扩展，在RFC 8684 (TCP Extensions for Multipath Operation with Multiple Addresses) 中描述，通过TCP Option实现 (Kind Number 30)。
- 从Linux 5.6开始MPTCP进入主线内核，源码目录 net/mptcp/，网站：[www.mptcp.dev](http://www.mptcp.dev)
- 它允许设备同时使用多个接口通过一条MPTCP连接来发送和接收TCP数据包，其可以聚合多个接口的带宽或优先选择延迟最低的接口



## 01 带宽扩展

通过多个网络路径并行传输，MPTCP能够将数据传输的带宽提升至原有单条通道的n倍，提高数据传输的速度和效率。

## 02 故障转移

当某一条子流出现故障导致数据无法传输时，MPTCP能够将数据转移到其他可用的路径上，确保数据传输的连续性和完整性。

## 03 链路选择

MPTCP提供了路径管理器，可以根据实际需求来配置决定多张网卡的连接方式，支持用户态和内核态的路径管理器。

# MPTCP技术简介

——应用适配

## C语言

```
//需要把TCP socket创建的地方，第三个参数更换为IPPROTO_MPTCP。  
// 创建MPTCP socket  
#include <sys/socket.h>  
int s = socket(AF_INET, SOCK_STREAM, IPPROTO_MPTCP)  
if (s < 0) /* fallback to TCP */  
| s = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
```

## Golang

```
//SetMultipathTCP 在 GoLang 1.21 中支持:  
✓ import ("net" "context")  
// Client  
d := &net.Dialer{}  
d.SetMultipathTCP(true)  
c, err := d.Dial("tcp", *addr)  
// Serverlc := &ListenConfig{}  
lc.SetMultipathTCP(true)  
ln, err := lc.Listen(context.Background(), "tcp", *addr)  
defer ln.Close()
```

## Python

```
# socket.IPPROTO_MPTCP 在 CPython 3.10 中支持:  
import socket  
try:  
| s = socket.socket(socket.AF_INET, socket.SOCK_STREAM, socket.IPPROTO_MPTCP)  
except:  
| s = socket.socket(socket.AF_INET, socket.SOCK_STREAM, socket.IPPROTO_TCP)
```

# MPTCP技术简介

—— 系统配置（路径管理器）

## 命令行配置

iproute2包中已经集成了ip mptcp 命令，通过该命令来手动完成路径的配置。

- 1) 设置subflow 和 add\_addr\_accepted的数量限制，最大值为8

```
ip mptcp limits set add_addr_accepted 8
```

```
ip mptcp limits set subflows 8
```

- 2) 配置本地网卡的连接属性

```
ip mptcp endpoint add [ip_addr] [dev name] [id] [signal/subflow]
```

(服务端用signal标志，客户端用subflow标志)

如下是在一台机器上配置两张网卡ens3/ens4的方式：

```
ip mptcp endpoint add 10.44.76.151 id 1 dev ens3 signal
```

```
ip mptcp endpoint add 10.44.76.152 id 2 dev ens4 signal
```

## 自动化配置

mptcpd软件包可以监控当前网络环境的变化，自动化配置MPTCP的路径管理器。目前，主流的Linux发行版均已完成mptcpd适配。

- 1) 配置文件 /etc/mptcpd/mptcpd.conf，可以通过调整配置文件来自定义mptcpd行为 (subflow/signal)
- 2) systemctl restart mptcp.service

注：如配置mptcpd服务，则还需要将NetworkManager中的mptcp相关选项关闭，避免冲突。



A vibrant, multi-colored nebula with stars in the background.

# 03

## Valkey over MPTCP

# CLSF 2024

2024 · CLSF · 杭州



CCF 中国存储大会

Redis/Valkey Over RDMA  
极致的高性能 KV 存储

皮振伟 - 2024/11

 ByteDance 字节跳动

打造世界级操作系统中国品牌

kylinos.cn

KYLINSOFT  
麒麟软件

## MPTCP 2024 开发报告

唐葛亮 麒麟软件  
MPTCP maintainer  
geliang@kernel.org

 Valkey

# Valkey over MPTCP

——场景背景

Valkey，作为Redis的重要开源分支，是一款被广泛使用的高性能KV数据库。在多网卡环境中使用Valkey over MPTCP，可以提升数据库的并发量、降低延迟。

## Introduce MPTCP #1811

Merged

zuiderkwast merged 11 commits into valkey-io:unstable from pizhenwei:mptcp on Apr 15

Conversation 78

Commits 11

Checks 51

Files changed 7



pizhenwei commented on Mar 3 • edited

Contributor ...

Multipath TCP (MPTCP) is an extension of the standard TCP protocol that allows a single transport connection to use multiple network interfaces or paths. MPTCP is useful for applications like bandwidth aggregation, failover, and more resilient connections.

Linux kernel starts to support MPTCP since v5.6, it's time to support it.

The test report shows that MPTCP reduces latency by ~25% in a 1% networking packet drop environment.

Thanks to Matthieu Baerts [matttbe@kernel.org](mailto:matttbe@kernel.org) for lots of review suggestions.

Proposed-by: Geliang Tang [geliang@kernel.org](mailto:geliang@kernel.org)

Tested-by: Gang Yan [yangang@kylinos.cn](mailto:yangang@kylinos.cn)

Signed-off-by: zhenwei pi [zhenwei.pi@linux.dev](mailto:zhenwei.pi@linux.dev)

Signed-off-by: zhenwei pi [pizhenwei@bytedance.com](mailto:pizhenwei@bytedance.com)

Cc Linux kernel MPTCP maintainer [@matttbe](#)



6



# Valkey over MPTCP

——支持细节

## 代码展示

### 服务端

```
595 + static int _anetTcpServer(char *err, int port, char *bindaddr, int
596     af, int backlog, int mptcp) {
597     int s = -1, rv;
598     char _port[6]; /* strlen("65535") */
599     struct addrinfo hints, *servinfo, *p;
600
601     ndaddr, int af, int backl
602
603     return ANET_ERR;
604 }
605
606     for (p = servinfo; p != NULL; p = p->ai_next) {
607
608     +     rv = anetTcpSetMptcp(err, p->ai_protocol, mptcp);
609
610     +     if (rv == ANET_ERR) goto error;
611
612     +     if ((s = socket(p->ai_family, p->ai_socktype, rv)) == -1)
613         continue;
```

### 客户端

```
463     static int anetTcpGenericConnect(char *err, const char *addr, int
464         port, const char *source_addr, int flags) {
465
466     int s = ANET_ERR, rv;
467     char portstr[6]; /* strlen("65535") + 1; */
468
469     addr, int port, const ch
470
471     * Make sure connection-intensive things like the benchmark
472     tool
473
474     * will be able to close/open sockets a zillion of times.
475     */
476
477     +     int ai_protocol = anetTcpGetProtocol(flags &
478         ANET_CONNECT_MULTIPATH);
479
480     int sockFlags = ANET_SOCKET_CLOEXEC | ANET_SOCKET_REUSEADDR;
481
482     if (flags & ANET_CONNECT_NONBLOCK) sockFlags |=
483         ANET_SOCKET_NONBLOCK;
484
485     +     if ((s = anetCreateSocket(err, p->ai_family, p->ai_socktype,
486         ai_protocol, sockFlags)) == ANET_ERR) continue;
487
488     if (source_addr) {
489
490         int bound = 0;
491
492         /* Using getaddrinfo saves us from self-determining IPv4
493          vs IPv6 */
494
495     }
```

```
static int anetCreateSocket(char *err, int domain, int type, int protocol,
    int s;

#ifndef SOCK_CLOEXEC
    if (flags & ANET_SOCKET_CLOEXEC) {
        type |= SOCK_CLOEXEC;
        flags &= ~ANET_SOCKET_CLOEXEC;
    }
#endif

#ifndef SOCK_NONBLOCK
    if (flags & ANET_SOCKET_NONBLOCK) {
        type |= SOCK_NONBLOCK;
        flags &= ~ANET_SOCKET_NONBLOCK;
    }
#endif

    if ((s = socket(domain, type, protocol)) == -1) {
        anetSetError(err, "creating socket: %s", strerror(errno));
        return ANET_ERR;
    }
}
```

## 使用方法

自Valkey 9.0.0版本以后，上游版本原生集成MPTCP能力，启动时添加--mptcp参数即可启用。

eg:

valkey-cli --mptcp

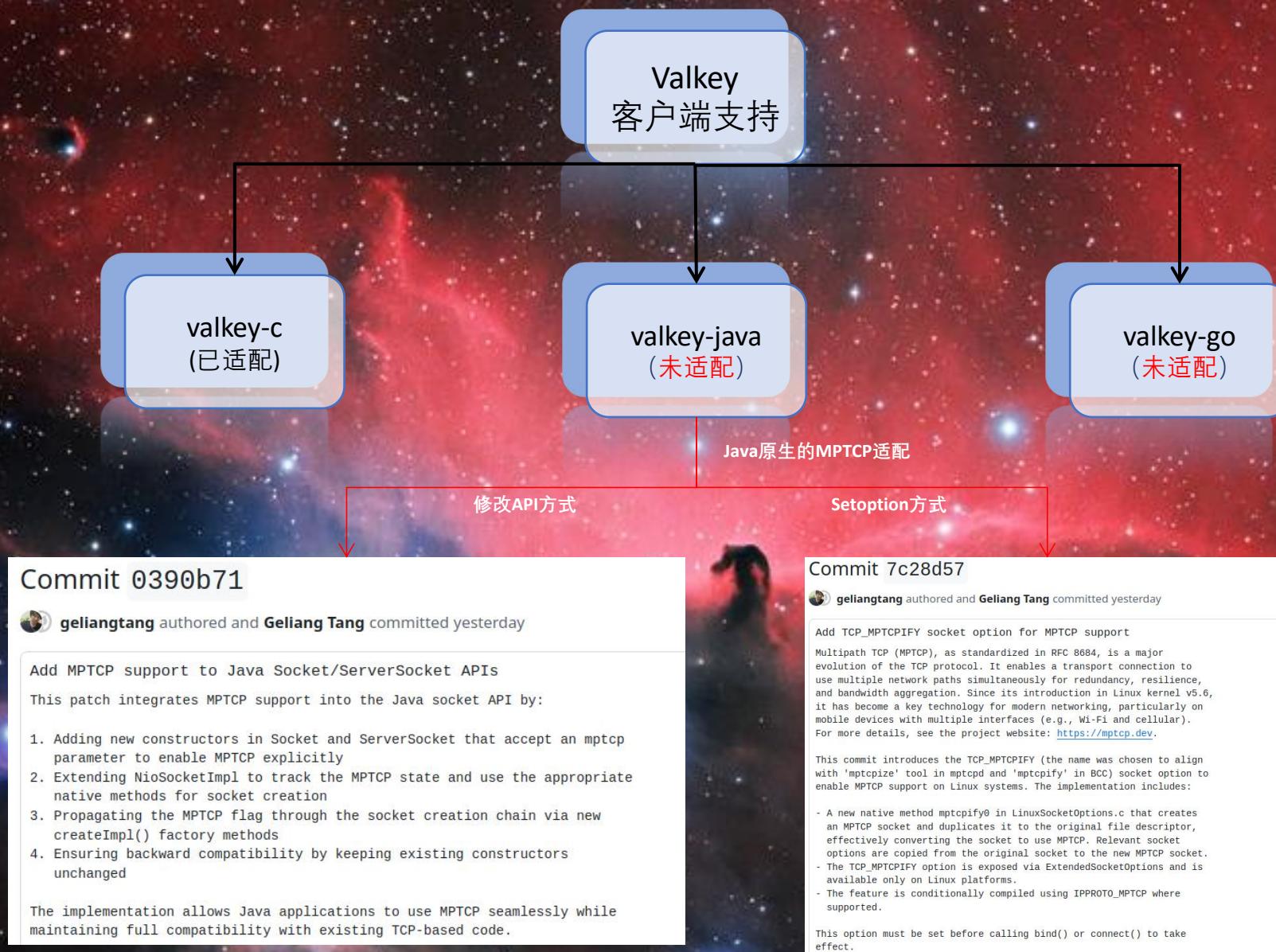
valkey-benchmark --mptcp

valkey-server --mptcp yes



# Valkey over MPTCP

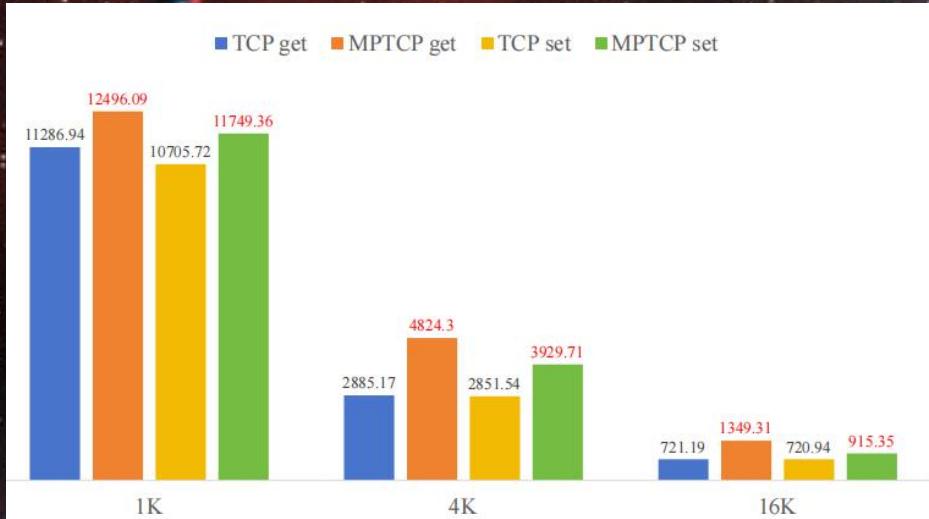
--TO DO



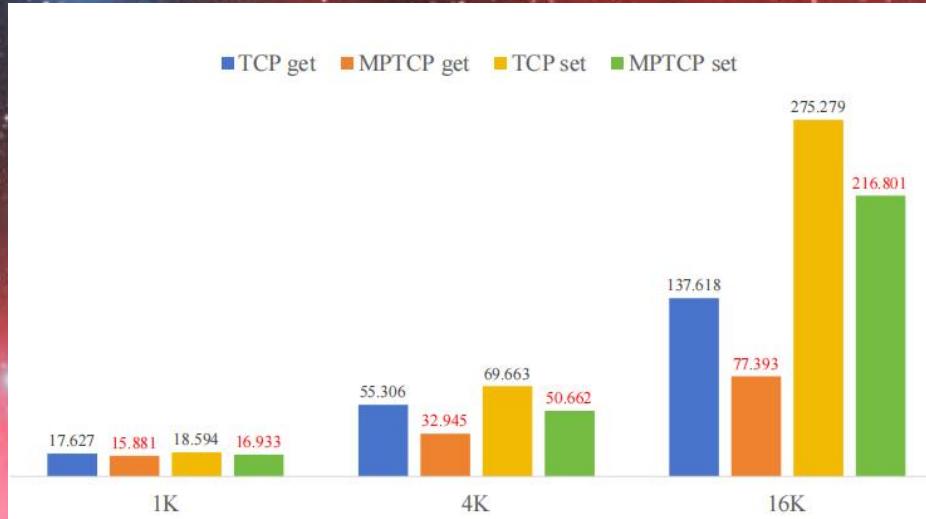
# Valkey over MPTCP

——性能展示 (2条100Mbit丢包链路)

并发量 (rpc)



延迟 (latency)



基于官方性能测试工具 valkey-benchmark，对 Valkey over TCP 以及 MPTCP 进行测试。

测试涵盖了传输 1K、4K 以及 16K 数据量下的场景，重点测量了 get 和 set 操作的并发量 rpc 以及延迟时间 latency。

# iperf3的MPTCP支持

## iperf3——快速验证

iperf3是开源网络性能测试工具，支持带宽、延迟、丢包率等指标测试。

3.19+版本支持MPTCP环境测速，用于MPTCP网络环境测试，**快速直观验证环境部署是否正确**。

eg:

iperf3 -s --mptcp

iperf3 -c <ip> -m

**iperf-3.19 2025-05-16**

**iperf-3.19 2025-05-16**

- Notable user-visible changes

- iperf3 now supports the use of Multi-Path TCP (MPTCPV1) on Linux with the use of the `-m` or `--mptcp` flag. (PR #1661)
- iperf3 now supports a `--cntl-ka` option to enable TCP keepalives on the control connection. (#812, #835, PR #1423)

## add MPTCPv1 support #1661

Merged

bmah888 merged 1 commit into esnet:master from geliangtang:master on Jan 18

Conversation 5

Commits 1

Checks 3

Files changed 8



geliangtang commented on Mar 12, 2024

Contributor ...

The Multipath TCP (MPTCP) protocol (v1 / RFC 8684) has been added in the upstream Linux kernel since v5.6.

MPTCP is strongly tied to TCP, and the kernel APIs are almost the same. The only required dependency is the 'IPPROTO\_MPTCP' protocol number definition, which should be provided by the netinet/in.h header if it is recent enough.

This patch adds a new flag '`-m`' or '`--mptcp`' to support MPTCPv1. It can be used like this:

```
iperf3 -m -s
```

```
iperf3 -m -c 127.0.0.1
```

There is no need to check for IPPROTO\_MPTCP support in configure.ac at build time, it is at runtime we will see if the kernel being use supports or not MPTCP.

If IPPROTO\_MPTCP is not supported by the kernel being tested, it is normal to fail because the feature is not available and the user explicitly asked to use MPTCP.

Closes: #1659

Co-developed-by: Paolo Abeni [paben@redhat.com](mailto:paben@redhat.com)

# BCC工具mptcpify

## BCC——快速部署

基于BPF程序拦截TCP连接建立过程，动态替换传输层为MPTCP，实现无感升级与灵活可控。

0.35.0+版本后添加了mptcpify工具，可以拦截TCP套接字创建过程，动态替换传输层为MPTCP，实现无感替换与灵活可控。

eg:

mptcpify -t redis,iperf3

目前已验证rsync、iperf3、curl、redis、valkey可以通过该方式快速完成MPTCP的适配。

Update debian changelog for release v0.35.

```
* Support for kernel up to 6.14
* New bcc tools: mptcp: enable mptcp for tcp traffic
* tools/biosnoop: Fix biosnoop pattern option
* Allow cmake run out of the source tree
* Fix for test bpf_stack_id when running on custom image in yocto
* libbpf-tools/map_helpers: Add bpf_map_lookup_and_delete_batch t
* libbpf-tools/biotop: Use dump_hash for map processing
* uninstall: use execute_process() instead of exec_program()
* clang: Fix pointer dereference on big-endian machines
* cii: Upgrade to ubuntu 24.04
* doc update, other bug fixes and example improvement.
```

```
+++ b/net/socket.c
@@ -1657,12 +1657,36 @@ struct file *__sys_socket_file(int family, int type, int proto)
     return sock_alloc_file(sock, flags, NULL);
 }

+/* A hook for bpf progs to attach to and update socket protocol.
+ *
+ * A static noinline declaration here could cause the compiler to
+ * optimize away the function. A global noinline declaration will
+ * keep the definition, but may optimize away the callsite.
+ * Therefore, __weak is needed to ensure that the call is still
+ * emitted, by telling the compiler that we don't know what the
+ * function might eventually be.
+ *
+ * __diag_* below are needed to dismiss the missing prototype warning.
+ */
+
+__diag_push();
+__diag_ignore_all("-Wmissing-prototypes",
+                  "A fmod_ret entry point for BPF programs");
+
+__weak noinline int update_socket_protocol(int family, int type, int protocol)
+{
+    return protocol;
+}
+
+__diag_pop();
+
int __sys_socket(int family, int type, int protocol)
{
    struct socket *sock;
    int flags;

-    sock = __sys_socket_create(family, type, protocol);
+    sock = __sys_socket_create(family, type,
+                               update_socket_protocol(family, type, protocol));
    if (IS_ERR(sock))
        return PTR_ERR(sock);

2023-08-16          Merge branch 'bpf: Force to MPTCP'
2023-08-16          selftests/bpf: Add mptcpify test
2023-08-16          selftests/bpf: Fix error checks of mptcp open_and_load
2023-08-16          selftests/bpf: Add two mptcp netns helpers
2023-08-16          bpf: Add update_socket_protocol hook
```

Python-BCC: support fmod\_ret attaching m  
add a useful tool for mptcp #5274

Merged chenhengqi merged 3 commits into iovisor:master from Dwyane-Yan:fmod\_ret

Conversation 30 Commits 3 Checks 12 Files changed 6

Dwyane-Yan commented on Apr 9 • edited Contributor

Multipath TCP (MPTCP) serves as an enhancement to the conventional TCP protocol, enabling a single transport-layer connection to leverage multiple network interfaces. This capability makes MPTCP advantageous for applications requiring bandwidth consolidation, seamless failover mechanisms, and more robust connectivity solutions.

Linux kernel starts to support MPTCP since v5.6, and it provides a fmod\_ret interface 'update\_socket\_protocol' to force applications using MPTCP instead of TCP without modifying its code.

So these patches provide a tool named 'mptcpify' which can achieve this. Using python-BCC is a more easy way for future development, so it is so important to support 'fmod\_ret' in python-BCC.

The first patch is suggested by Yonghong:

On Sun, 2024-08-25 at 21:05 -0700, Yonghong Song wrote:

> Gang Yan, could you explore to add fmod\_ret support in bcc? It should  
> be similar to kfunc/kretfunc support. I am happy to review your patches.

when I try to make a commit to kernel. (Link attached below)  
[https://patchwork.kernel.org/project/netdevbpf/patch/tencent\\_1E619C9E44C8C4B2B713A0D6DD45B92BF70A@qq.com/](https://patchwork.kernel.org/project/netdevbpf/patch/tencent_1E619C9E44C8C4B2B713A0D6DD45B92BF70A@qq.com/)

@matttbe from the MPTCP kernel team had a look at the new tool.



# 04

## MPTCP 生态支持

# 生态支持

——应用生态

Name	Version	How to use
Angie	1.10.0	<code>listen &lt;address port&gt; multipath</code>
Apache HTTP Server	2.5.1	<code>Listen 80 options=multipathtcp</code> OR <code>ProxyPass multipathtcp=On</code>
Apache Traffic Server	9.2.4	<code>server_ports: &lt;port&gt; mptcp</code>
Caddy	v2.7.4	Enabled by default since v2.10.0 (Go 1.24), controllable with <code>GODEBUG=multipath=</code> since v2.7.4 (Go 1.21)
cURL	8.9.0	<code>--mptcp</code>
dae	v0.8.0	<code>global { mptcp: true }</code>
Envoy	1.21.0	<code>"enable_mptcp": "true"</code> (Listening config)
freenginx	1.27.5	<code>listen &lt;address port&gt; multipath</code>
GoLang	1.21	Apps written in Go can easily enable MPTCP support. Since 1.24, it is enabled on the server side by default
HAProxy	3.1.0	<code>mptcp{,4,6}@&lt;address&gt;</code>

Lighttpd 1.4	1.4.76	<code>server.feature-flags = ( "server.network-mptcp" =&gt; "enable" )</code>
QEmu	6.1	<code>&lt;ip&gt;:&lt;port&gt;,mptcp</code>
Shadowsocks libev	v3.3.6	<code>--mptcp</code>
Shadowsocks Rust	v1.16.0	<code>--mptcp</code>
Shadowsocks Go	v1.9.0	<code>servers.tcpListeners.multipath</code> , <code>clients.multipathTCP</code>
sing-box	v1.4.0	<code>"tcp_multi_path": true</code>
SystemD	v257	<code>SocketProtocol=mptcp</code> ([Socket] section)
Traefik	v2.10.5	Enabled by default since v2.11.26 / v3.4.2 / v3.5.0-rc1 (Go 1.24), controllable with <code>GODEBUG=multipath=</code> since v2.10.5 (Go 1.21)
v2ray-core	v5.17.0	<code>"mptcp": true</code>
Valkey	v8.2.0	<code>mptcp yes</code> , <code>repl-mptcp yes</code> and <code>--mptcp</code>

# 生态支持

—— 操作系统生态

Name	Version	mptcpd	Comments	NixOS	✓	✓	<code>mptcpd</code> has been accepted in Jan. 2025
ArchLinux	✓	✓	Rolling release: MPTCP support is enabled since kernel v5.6. <code>mptcpd</code> is packaged in AUR.	Open MPTCP Router	v0.60	✓	MPTCP-specific, where everything is automatically configured.
Alpine Linux	3.20	✓	<code>mptcpd</code> is available since v3.22.	openEuler	24.03	✓	<code>mptcpd</code> 0.13 will be included in 24.03 SP3.
Amazon Linux	2023	✗	<code>mptcpd</code> / <code>mptcpize</code> can be used from containers.	OpenWrt	24.10	✓	<code>ip-full</code> package is required to use <code>ip mptcp</code> command.
Azure Linux	3.0	✗	<code>mptcpd</code> / <code>mptcpize</code> can be used from containers.	Raspberry Pi OS	20231004	✓	
Debian	12	✓		RHEL	9	✓	MPTCP is available as a tech-preview since v8.3.
Fedora	36	✓		Slackware	✓	✓	Version: to be completed. <code>mptcpd</code> is packaged in SlackBuilds
Gentoo	✓	✓	Rolling release.	SUSE	✓	✗	Version: to be completed.
Home Assistant OS	12.2	✗	<code>mptcpd</code> / <code>mptcpize</code> can be used from containers.	Ubuntu	22.04	✓	
Kylin OS	V11	✓		Void	✓	✗	
NixOS	✓	✓	<code>mptcpd</code> has been accepted in Jan. 2025				Rolling release: MPTCP support is enabled since kernel v5.10.

## 联系方式

GitHub:

@geliangtang;

邮箱:

geliang@kernel.org;

MPTCP技术社区:

官网: <https://www.mptcp.dev/>

讨论: [https://github.com/multipath-tcp/mptcp\\_net-next](https://github.com/multipath-tcp/mptcp_net-next)

邮件列表: [mptcp@lists.linux.dev](mailto:mptcp@lists.linux.dev)