

Forum:[Hack Forums](#)

Thread: [Exploring Syscalls on MacOS](#)

Hackers - October 5, 2025, 8:01 pm

Syscall numbers on Mac are obviously different than Windows and Linux, and there aren't that many resources available. In general, much of the tech enthusiast world focuses on Windows.

This is similar to Syswhispers, but for MacOS - the goal here is to run a program that can dynamically pull the information easily. This could be done per major version for static lists & headers, param types, etc...

sys/syscalls.h

example path:

Quote:

/Applications/Xcode.app/Contents/Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX.sdk/usr/include/sys/

Example...

delete 0x020000e2 226

Format:

0x02 << 24 = UNIX class tag

0x000000E2 = decimal 226

0x020000E2 == DARWIN_UNIX_SYSCALL(226)

We can verify this with the static header file in the path above

An example use, assuming this syscall1 function is defined for params:

Quote:long r = do_syscall1(0x020000E2, (long)path);

This instead of using a more public API from userland to kernel, like with the function unlink(path) instead.

We can also call unlinkat. Here is an example of those syscalls:

Quote:long r = do_syscall1(0x0200000A, (long)path); // unlink(path), unix_n=10

// or unlinkat(dirfd, path, 0) => unix_n=472 -> 0x020001D8 with 3 args

I am focused on dynamically finding this info from a program, but here is a clear definition from apple oss repo:

<https://github.com/apple-oss-distributions/xnu/blob/8d741a5de7ff4191bf97d57b9f54c2f6d4a15585/libsyscall/wrappers/unlinkat.c>

Once you have the internal numbers for a specific build/os version, you still need to handle param types, output handling, etc.

I am using some data from apple-oss-distributions for now, but would like to make this dynamic to

the compiler OR runtime host.

```
Code int sw_kill(int pid, int signum, int posix);  
// ...  
int sw_kill(int pid, int signum, int posix) { return (  
int)mos_syscall3(0x02000025,(long)pid,(long)signum,(long)posix); }  
This is what I came up with, and I confirmed it here, for kill function:
```

<https://github.com/apple-oss-distributions/xnu/blob/e3723e1f17661b24996789d8afc084c0c3303b26/libsyscall/wrappers/kill.c>

Quote:Class tag: 0x02 << 24 = 0x02000000

Number: 0x02000025 & 0x00FFFFFF = 0x25 = 37

I hope this thread is helpful for anyone diving into MacOS. Keep in mind that the numbers & info below may not match your specific Mac version!

Data dump:

Spoiler ([Click to View](#))

Quote:name	full_id	unix_n
_____	_____	_____
__channel_get_info	0x020001ff	511
__channel_get_opt	0x02000201	513
__channel_open	0x020001fe	510
__channel_set_opt	0x02000202	514
__channel_sync	0x02000200	512
__disable_threadsignal	0x0200014b	331
__mac_execve	0x0200017c	380
__mac_get_fd	0x02000184	388
__mac_get_file	0x0200017e	382
__mac_get_link	0x02000180	384
__mac_get_mount	0x020001a9	425
__mac_get_pid	0x02000186	390
__mac_get_proc	0x02000182	386
__mac_getfsstat	0x020001aa	426
__mac_mount	0x020001a8	424
__mac_set_fd	0x02000185	389
__mac_set_file	0x0200017f	383
__mac_set_link	0x02000181	385
__mac_set_proc	0x02000183	387
__mac_syscall	0x0200017d	381
__mach_bridge_remote_time	0x02000213	531
__nexus_create	0x020001fa	506
__nexus_deregister	0x020001f9	505
__nexus_destroy	0x020001fb	507
__nexus_get_opt	0x020001fc	508
__nexus_open	0x020001f7	503
__nexus_register	0x020001f8	504
__nexus_set_opt	0x020001fd	509
__pthread_canceled	0x0200014d	333
__pthread_chdir	0x0200015c	348
__pthread_fchdir	0x0200015d	349
__pthread_kill	0x02000148	328

__pthread_markcancel	0x0200014c	332
__pthread_sigmask	0x02000149	329
__semwait_signal	0x0200014e	334
__semwait_signal_nocancel	0x020001a7	423
__sigwait	0x0200014a	330
__sigwait_nocancel	0x020001a6	422
abort_with_payload	0x02000209	521
accept	0x0200001e	30
accept_nocancel	0x02000194	404
access	0x02000021	33
access_extended	0x0200011c	284
acct	0x02000033	51
adjtime	0x0200008c	140
aio_cancel	0x0200013c	316
aio_error	0x0200013d	317
aio_fsync	0x02000139	313
aio_read	0x0200013e	318
aio_return	0x0200013a	314
aio_suspend	0x0200013b	315
aio_suspend_nocancel	0x020001a5	421
aio_write	0x0200013f	319
audit	0x0200015e	350
audit_session_join	0x020001ad	429
audit_session_port	0x020001b0	432
audit_session_self	0x020001ac	428
auditctl	0x02000167	359
audition	0x0200015f	351
bind	0x02000068	104
bsdthread_create	0x02000168	360
bsdthread_ctl	0x020001de	478
bsdthread_register	0x0200016e	366
bsdthread_terminate	0x02000169	361
change_fdguard_np	0x020001bc	444
chdir	0x0200000c	12
chflags	0x02000022	34
chmod	0x0200000f	15
chmod_extended	0x0200011a	282
chown	0x02000010	16
chroot	0x0200003d	61
clonefileat	0x020001ce	462
close	0x02000006	6
close_nocancel	0x0200018f	399
coalition	0x020001ca	458
coalition_info	0x020001cb	459
coalition_ledger	0x02000214	532
coalition_policy_get	0x0200022d	557
coalition_policy_set	0x0200022c	556
connect	0x02000062	98
connect_nocancel	0x02000199	409
connectx	0x020001bf	447
copyfile	0x020000e3	227
crossarch_trap	0x02000026	38
csops	0x020000a9	169

csops_audittoken	0x020000aa	170
csrctl	0x020001e3	483
debug_syscall_reject	0x02000223	547
debug_syscall_reject_config	0x02000224	548
delete	0x020000e2	226
disconnectx	0x020001c0	448
dup	0x02000029	41
dup2	0x0200005a	90
exchangedata	0x020000df	223
execve	0x0200003b	59
exit	0x02000001	1
faccessat	0x020001d2	466
fchdir	0x0200000d	13
fchflags	0x02000023	35
fchmod	0x0200007c	124
fchmod_extended	0x0200011b	283
fchmodat	0x020001d3	467
fchown	0x0200007b	123
fchownat	0x020001d4	468
fclonefileat	0x02000205	517
fcntl	0x0200005c	92
fcntl_nocancel	0x02000196	406
fdatasync	0x020000bb	187
ffsctl	0x020000f5	245
fgetattrlist	0x020000e4	228
fgetxattr	0x020000eb	235
fhopen	0x020000f8	248
fileport_makefd	0x020001af	431
fileport_makeport	0x020001ae	430
flistxattr	0x020000f1	241
flock	0x02000083	131
fmount	0x0200020e	526
fork	0x02000002	2
fpathconf	0x020000c0	192
freadlink	0x02000227	551
fremovexattr	0x020000ef	239
fs_snapshot	0x02000206	518
fsctl	0x020000f2	242
fsetattrlist	0x020000e5	229
fsetxattr	0x020000ed	237
fsgetpath	0x020001ab	427
fsgetpath_ext	0x020000d9	217
fstat	0x020000bd	189
fstat64	0x02000153	339
fstat64_extended	0x02000157	343
fstat_extended	0x02000119	281
fstatat	0x020001d5	469
fstatat64	0x020001d6	470
fstatfs	0x0200009e	158
fstatfs64	0x0200015a	346
fsync	0x0200005f	95
fsync_nocancel	0x02000198	408
ftruncate	0x020000c9	201

futimes	0x0200008b 139
getattrlist	0x020000dc 220
getattrlistat	0x020001dc 476
getattrlistbulk	0x020001cd 461
getaudit_addr	0x02000165 357
getaudit	0x02000161 353
getdirentries	0x020000c4 196
getdirentries64	0x02000158 344
getdirentriesattr	0x020000de 222
getdtablesize	0x02000059 89
getegid	0x0200002b 43
getentropy	0x020001f4 500
geteuid	0x02000019 25
getfh	0x020000a1 161
getfsstat	0x02000012 18
getfsstat64	0x0200015b 347
getgid	0x0200002f 47
getgroups	0x0200004f 79
gethostuuid	0x0200008e 142
getitimer	0x02000056 86
getlogin	0x02000031 49
getpeername	0x0200001f 31
getpgid	0x02000097 151
getpgrp	0x02000051 81
getpid	0x02000014 20
getppid	0x02000027 39
getpriority	0x02000064 100
getrlimit	0x020000c2 194
getrusage	0x02000075 117
getsgroups	0x02000120 288
getsid	0x02000136 310
getsockname	0x02000020 32
getsockopt	0x02000076 118
gettid	0x0200011e 286
gettimeofday	0x02000074 116
getuid	0x02000018 24
getwgroups	0x02000122 290
getxattr	0x020000ea 234
grab_pgo_data	0x020001ed 493
graftdmg	0x02000225 549
guarded_close_np	0x020001ba 442
guarded_kqueue_np	0x020001bb 443
guarded_open_dprotected_np	0x020001e4 484
guarded_open_np	0x020001b9 441
guarded_pwrite_np	0x020001e6 486
guarded_write_np	0x020001e5 485
guarded_writev_np	0x020001e7 487
identitysvc	0x02000125 293
initgroups	0x020000f3 243
invalid	0x0200003f 63
ioctl	0x02000036 54
iopolicysys	0x02000142 322
issetugid	0x02000147 327

kas_info	0x020001b7	439
kdebug_trace	0x020000b4	180
kdebug_trace64	0x020000b3	179
kdebug_trace_string	0x020000b2	178
kdebug_typefilter	0x020000b1	177
kevent	0x0200016b	363
kevent64	0x02000171	369
kevent_id	0x02000177	375
kevent_qos	0x02000176	374
kill	0x02000025	37
kqueue	0x0200016a	362
kqueue_workloop_ctl	0x02000212	530
lchown	0x0200016c	364
ledger	0x02000175	373
link	0x02000009	9
linkat	0x020001d7	471
lio_listio	0x02000140	320
listen	0x0200006a	106
listxattr	0x020000f0	240
log_data	0x02000215	533
lseek	0x020000c7	199
lstat	0x020000be	190
lstat64	0x02000154	340
lstat64_extended	0x02000156	342
lstat_extended	0x02000118	280
mach_eventlink_signal	0x020001f0	496
mach_eventlink_signal_wait_until	0x020001f2	498
mach_eventlink_wait_until	0x020001f1	497
madvise	0x0200004b	75
map_with_linking_np	0x02000226	550
memorystatus_available_memory	0x02000216	534
memorystatus_control	0x020001b8	440
memorystatus_get_level	0x020001c5	453
microstackshot	0x020001ec	492
mincore	0x0200004e	78
minherit	0x020000fa	250
mkdir	0x02000088	136
mkdir_extended	0x02000124	292
mkdirat	0x020001db	475
mkfifo	0x02000084	132
mkfifo_extended	0x02000123	291
mkfifoat	0x02000229	553
mknod	0x0200000e	14
mknodat	0x0200022a	554
mlock	0x020000cb	203
mlockall	0x02000144	324
mmap	0x020000c5	197
mount	0x020000a7	167
mprotect	0x0200004a	74
mremap_encrypted	0x020001e9	489
msgctl	0x02000102	258
msgget	0x02000103	259
msgrcv	0x02000105	261

msgrcv_nocancel	0x020001a3	419
msgsnd	0x02000104	260
msgsnd_nocancel	0x020001a2	418
msgsys	0x020000fc	252
msync	0x02000041	65
msync_nocancel	0x02000195	405
munlock	0x020000cc	204
munlockall	0x02000145	325
munmap	0x02000049	73
necp_client_action	0x020001f6	502
necp_match_policy	0x020001cc	460
necp_open	0x020001f5	501
necp_session_action	0x0200020b	523
necp_session_open	0x0200020a	522
net_qos_guideline	0x0200020d	525
netagent_trigger	0x020001ea	490
nfssvc	0x0200009b	155
ntp_adjtime	0x0200020f	527
ntp_gettime	0x02000210	528
objc_bp_assist_cfg_np	0x02000217	535
open	0x02000005	5
open_dprotected_np	0x020000d8	216
open_extended	0x02000115	277
open_nocancel	0x0200018e	398
openat	0x020001cf	463
openat_dprotected_np	0x020000da	218
openat_nocancel	0x020001d0	464
openbyid_np	0x020001df	479
os_fault_with_payload	0x02000211	529
oslog_coproc	0x02000044	68
oslog_coproc_reg	0x02000043	67
panic_with_data	0x020000b9	185
pathconf	0x020000bf	191
peeloff	0x020001c1	449
persona	0x020001ee	494
pid_hibernate	0x020001b3	435
pid_resume	0x020001b2	434
pid_shutdown_sockets	0x020001b4	436
pid_suspend	0x020001b1	433
pipe	0x0200002a	42
pivot_root	0x02000219	537
poll	0x020000e6	230
poll_nocancel	0x020001a1	417
posix_spawn	0x020000f4	244
pread	0x02000099	153
pread_nocancel	0x0200019e	414
preadv	0x0200021c	540
preadv_nocancel	0x0200021e	542
proc_info	0x02000150	336
proc_info_extended_id	0x02000221	545
proc_rlimit_control	0x020001be	446
proc_trace_log	0x020001dd	477
proc_uuid_policy	0x020001c4	452

process_policy	0x02000143	323
pselect	0x0200018a	394
pselect_nocancel	0x0200018b	395
psynch_cvbroad	0x0200012f	303
psynch_cvclrprepost	0x02000138	312
psynch_cvsignal	0x02000130	304
psynch_cvwait	0x02000131	305
psynch_mutexdrop	0x0200012e	302
psynch_mutexwait	0x0200012d	301
psynch_rw_downgrade	0x0200012b	299
psynch_rw_longrdlock	0x02000129	297
psynch_rw_rdlock	0x02000132	306
psynch_rw_unlock	0x02000134	308
psynch_rw_unlock2	0x02000135	309
psynch_rw_upgrade	0x0200012c	300
psynch_rw_wrlock	0x02000133	307
psynch_rw_yieldwrlock	0x0200012a	298
ptrace	0x0200001a	26
pwrite	0x0200009a	154
pwrite_nocancel	0x0200019f	415
pwritev	0x0200021d	541
pwritev_nocancel	0x0200021f	543
quotactl	0x020000a5	165
read	0x02000003	3
read_nocancel	0x0200018c	396
readlink	0x0200003a	58
readlinkat	0x020001d9	473
readv	0x02000078	120
readv_nocancel	0x0200019b	411
reboot	0x02000037	55
record_system_event	0x02000228	552
recvfrom	0x0200001d	29
recvfrom_nocancel	0x02000193	403
recvmsg	0x0200001b	27
recvmsg_nocancel	0x02000191	401
recvmsg_x	0x020001e0	480
register_uexc_handler	0x02000207	519
removexattr	0x020000ee	238
rename	0x02000080	128
renameat	0x020001d1	465
renameatx_np	0x020001e8	488
revoke	0x02000038	56
rmdir	0x02000089	137
searchfs	0x020000e1	225
select	0x0200005d	93
select_nocancel	0x02000197	407
sem_close	0x0200010d	269
sem_open	0x0200010c	268
sem_post	0x02000111	273
sem_trywait	0x02000110	272
sem_unlink	0x0200010e	270
sem_wait	0x0200010f	271
sem_wait_nocancel	0x020001a4	420

semctl	0x020000fe 254
semget	0x020000ff 255
semop	0x02000100 256
semsys	0x020000fb 251
sendfile	0x02000151 337
sendmsg	0x0200001c 28
sendmsg_nocancel	0x02000192 402
sendmsg_x	0x020001e1 481
sendto	0x02000085 133
sendto_nocancel	0x0200019d 413
setattrlist	0x020000dd 221
setattrlistat	0x0200020c 524
setaudit_addr	0x02000166 358
setauid	0x02000162 354
setegid	0x020000b6 182
seteuid	0x020000b7 183
setgid	0x020000b5 181
setgroups	0x02000050 80
setitimer	0x02000053 83
setlogin	0x02000032 50
setpgid	0x02000052 82
setpriority	0x02000060 96
setprivexec	0x02000098 152
setregid	0x0200007f 127
setreuid	0x0200007e 126
setrlimit	0x020000c3 195
setsgroups	0x0200011f 287
setsid	0x02000093 147
setsockopt	0x02000069 105
settid	0x0200011d 285
settid_with_pid	0x02000137 311
settimeofday	0x0200007a 122
setuid	0x02000017 23
setwgroups	0x02000121 289
setxattr	0x020000ec 236
sfi_ctl	0x020001c8 456
sfi_pidctl	0x020001c9 457
shared_region_check_np	0x02000126 294
shared_region_map_and_slide_2_np	0x02000218 536
shm_open	0x0200010a 266
shm_unlink	0x0200010b 267
shmat	0x02000106 262
shmctl	0x02000107 263
shmdt	0x02000108 264
shmget	0x02000109 265
shmsys	0x020000fd 253
shutdown	0x02000086 134
sigaction	0x0200002e 46
sigaltstack	0x02000035 53
sigpending	0x02000034 52
sigprocmask	0x02000030 48
sigreturn	0x020000b8 184
sigsuspend	0x0200006f 111

sigsuspend_nocancel	0x0200019a	410
socket	0x02000061	97
socket_delegate	0x020001c2	450
socketpair	0x02000087	135
stack_snapshot_with_config	0x020001eb	491
stat	0x020000bc	188
stat64	0x02000152	338
stat64_extended	0x02000155	341
stat_extended	0x02000117	279
statfs	0x0200009d	157
statfs64	0x02000159	345
swapon	0x02000055	85
symlink	0x02000039	57
symlinkat	0x020001da	474
sync	0x02000024	36
syscall	0x02000000	0
sysctl	0x020000ca	202
sysctlbyname	0x02000112	274
system_override	0x020001c6	454
task_inspect_for_pid	0x0200021a	538
task_read_for_pid	0x0200021b	539
telemetry	0x020001c3	451
terminate_with_payload	0x02000208	520
thread_selfcounts	0x020000ba	186
thread_selfid	0x02000174	372
thread_selfusage	0x020001e2	482
tracker_action	0x02000222	546
truncate	0x020000c8	200
ulock_wait	0x02000203	515
ulock_wait2	0x02000220	544
ulock_wake	0x02000204	516
umask	0x0200003c	60
umask_extended	0x02000116	278
undelete	0x020000cd	205
ungraftdmg	0x0200022b	555
unlink	0x0200000a	10
unlinkat	0x020001d8	472
unmount	0x0200009f	159
usrctl	0x020001bd	445
utimes	0x0200008a	138
vfork	0x02000042	66
vfs_purge	0x020001c7	455
vm_pressure_monitor	0x02000128	296
wait4	0x02000007	7
wait4_nocancel	0x02000190	400
waitid	0x020000ad	173
waitid_nocancel	0x020001a0	416
work_interval_ctl	0x020001f3	499
workq_kernreturn	0x02000170	368
workq_open	0x0200016f	367
write	0x02000004	4
write_nocancel	0x0200018d	397
writew	0x02000079	121

There are also many definitions here:

<https://raw.githubusercontent.com/apple-oss-distributions/xnu/main/bsd/kern/syscalls.master>

Spoiler ([Click to View](#))

Quote:; System call name/number master file.

; This is file processed by .../xnu/bsd/kern/makesyscalls.sh and creates:

; .../xnu/bsd/kern/init_sysent.c

; .../xnu/bsd/kern/syscalls.c

; .../xnu/bsd/sys/syscall.h

; .../xnu/bsd/sys/sysproto.h

; .../xnu/bsd/security/audit_syscalls.c

; Columns -> | Number Audit Files | { Name and Args } | { Comments }

; Number: system call number, must be in order

; Audit: the audit event associated with the system call

; A value of AUE_NULL means no auditing, but it also means that

; there is no audit event for the call at this time. For the

; case where the event exists, but we don't want auditing, the

; event should be #defined to AUE_NULL in audit_kevents.h.

; Files: with files to generate - "ALL" or any combo of:

; "T" for syscall table (in init_sysent.c)

; "N" for syscall names (in syscalls.c)

; "H" for syscall headers (in syscall.h)

; "P" for syscall prototypes (in sysproto.h)

; Name and Args: function prototype, optionally followed by

; NO_SYSCALL_STUB (which mean no system call stub will

; be generated in libSystem) and ending with a semicolon.

; (Note: functions prefixed by double-underscore are

; automatically given the NO_SYSCALL_STUB attribute.)

; Comments: additional comments about the sys call copied to output files

; `#ifdef`s, `#include`s, `#if`s etc. are copied to all output files.

; N.B.: makesyscalls.sh and createsyscalls.pl must be updated to account

; for any new argument types.

*/

Another file that may be useful in the future is this script that constructs the syscall files for a distribution:

<https://github.com/apple-oss-distributions/xnu/blob/8d741a5de7ff4191bf97d57b9f54c2f6d4a15585/bsd/kern/makesyscalls.sh>

It was fun to learn a bit more MacOS.

I was able to create wrapper functions easily to avoid well-known & documented API / defined funcs, using direct syscalls instead.

I did not use this for any sort of hacking, as I am just learning the arch still & figured I would share

what I discovered for Mac specifically.

Specs for these tests / syscalls:

Mac mini, with Apple M4 / silicon, with Sequoia 15.3.1

- I will be testing shortly with Sequoia 15.7x, followed by Tahoe 26x