Kernel versions and sources	Linux 6.8-rc5 Mainline Source: https://github.com/torvalds/linux/blob/b401b621758e46812da61fa58a67c3fd8d91de0d, Feb 18 2024	Linux 5.4 Source: https://github.com/torvalds/linux/commit/219d54332a09e8d8741c1e1982f5eae56099de85, https://github.com/torvalds/linux/archive/refs/tags/v5.4.tar.gz, Nov 2019	UEKR6 5.4.17 https://yum.oracle.com/repo/OracleLinux/OL8/developer/UEKR6/aarch64/getPackage/kernel-uek-5.4.17-2136.327.2.el8uek.aarch64.rpm Source: https://yum.oracle.com/repo/OracleLinux/OL8/developer/UEKR6/aarch64/getPackageSource/kernel-uek-5.4.17-2136.327.2.el8uek.src.rpm, Jan 2024	UEKR7 5.15.0 https://yum.oracle.com/repo/OracleLinux/OL8/UEKR7/aarch64/getPackage/kernel-uek-5.15.0-105.125.6.2.1.el8uek.aarch64.rpm Source: https://yum.oracle.com/repo/OracleLinux/OL8/UEKR7/aarch64/getPackageSource/kernel-uek-5.15.0-105.125.6.2.1.el8uek.src.rpm, Sep 2023
Definitions of the syscall No.23	https://github.com/torvalds/linux/blob/b401b621758e46812da61fa58a67c3fd8d91de0d/include/uapi/asm-generic/mman-common.h#L76 #define MADV_WIPEONFORK 18 /* Zero memory on fork, child only */ #define MADV_KEEPONFORK 19 /* Undo MADV_WIPEONFORK */ #define MADV_COLD 20 /* deactivate these pages */ #define MADV_PAGEOUT 21 /* reclaim these pages */ #define MADV_POPULATE_READ 22 /* populate (prefault) page tables readable */ #define MADV_POPULATE_WRITE 23 /* populate (prefault) page tables writable */ #define MADV_DONTNEED_LOCKED 24 /* like DONTNEED, but drop locked pages too */ #define MADV_COLLAPSE 25 /* Synchronous hugepage collapse */ /* compatibility flags */ #define MAP_FILE 0	https://github.com/torvalds/linux/blob/219d54332a09e8d8741c1e1982f5eae56099de85/include/uapi/asm-generic/mman-common.h#L67 #define MADV_WIPEONFORK 18 /* Zero memory on fork, child only */ #define MADV_KEEPONFORK 19 /* Undo MADV_WIPEONFORK */ #define MADV_COLD 20 /* deactivate these pages */ #define MADV_PAGEOUT 21 /* reclaim these pages */ /* compatibility flags */ #define MAP_FILE 0	linux-5.4.17/include/uapi/asm-generic/mman-common.h#L75 #define MADV_WIPEONFORK 18 /* Zero memory on fork, child only */ #define MADV_KEEPONFORK 19 /* Undo MADV_WIPEONFORK */ #define MADV_COLD 20 /* deactivate these pages */ #define MADV_PAGEOUT 21 /* reclaim these pages */ #define MADV_DOEXEC 22 /* do inherit across exec */ #define MADV_DONTEXEC 23 /* don't inherit across exec */ /* compatibility flags */ #define MAP_FILE 0	Linux-5.15.0/include/uapi/asm-generic/mman-common.h#L77 #define MADV_WIPEONFORK 18 /* Zero memory on fork, child only */ #define MADV_KEEPONFORK 19 /* Undo MADV_WIPEONFORK */ #define MADV_COLD 20 /* deactivate these pages */ #define MADV_PAGEOUT 21 /* reclaim these pages */ #define MADV_POPULATE_READ 22 /* populate (prefault) page tables readable */ #define MADV_POPULATE_WRITE 23 /* populate (prefault) page tables writable */ #define MADV_DOEXEC 24 /* do inherit across exec */ #define MADV_DONTEXEC 25 /* don't inherit across exec */ /* compatibility flags */ #define MAP_FILE 0

Function madvise_behavior_valid	https://github.com/torvalds/linux/blob/b401b621758e46812da61fa58a67c3fd8d91de0d/mm/madvise.c#L1157
	
	madvise_behavior_valid(int behavior)
	{
	switch (behavior) {
	case MADV_DOFORK:
	case MADV_DONTFORK:
	case MADV_NORMAL:
	case MADV_SEQUENTIAL:
	case MADV_RANDOM:
	case MADV_REMOVE:
	case MADV_WILLNEED:
	case MADV_DONTNEED:
	case MADV_DONTNEED_LOCKED:
	case MADV_FREE:
	case MADV_COLD:
	case MADV_PAGEOUT:
	case MADV_POPULATE_READ:
	case MADV_POPULATE_WRITE:
	#ifdef CONFIG_KSM
	case MADV_MERGEABLE:
	case MADV_UNMERGEABLE:
	#endif
	#ifdef CONFIG_TRANSPARENT_HUGEPAGE
	case MADV_HUGEPAGE:
	case MADV_NOHUGEPAGE:
	case MADV_COLLAPSE:
	#endif
	case MADV_DONTDUMP:
	case MADV_DODUMP:
	case MADV_WIPEONFORK:
	case MADV_KEEPONFORK:
	#ifdef CONFIG_MEMORY_FAILURE
	case MADV_SOFT_OFFLINE:
	case MADV_HWPOISON:
	#endif
	return true;
	
	default:
	return false;
	}
	}
		https://github.com/torvalds/linux/blob/219d54332a09e8d8741c1e1982f5eae56099de85/mm/madvise.c#L948
		
		static bool
		madvise_behavior_valid(int behavior)
		{
		switch (behavior) {
		case MADV_DOFORK:
		case MADV_DONTFORK:
		case MADV_NORMAL:
		case MADV_SEQUENTIAL:
		case MADV_RANDOM:
		case MADV_REMOVE:
		case MADV_WILLNEED:
		case MADV_DONTNEED:
		case MADV_FREE:
		case MADV_COLD:
		case MADV_PAGEOUT:
		#ifdef CONFIG_KSM
		case MADV_MERGEABLE:
		case MADV_UNMERGEABLE:
		#endif
		#ifdef CONFIG_TRANSPARENT_HUGEPAGE
		case MADV_HUGEPAGE:
		case MADV_NOHUGEPAGE:
		#endif
		case MADV_DONTDUMP:
		case MADV_DODUMP:
		case MADV_WIPEONFORK:
		case MADV_KEEPONFORK:
		#ifdef CONFIG_MEMORY_FAILURE
		case MADV_SOFT_OFFLINE:
		case MADV_HWPOISON:
		#endif
		return true;
		
		default:
		return false;
		}
		}
			linux-5.4.17/mm/madvise.c#L976
			
			static bool
			madvise_behavior_valid(int behavior)
			{
			switch (behavior) {
			case MADV_DOFORK:
			case MADV_DONTFORK:
			case MADV_NORMAL:
			case MADV_SEQUENTIAL:
			case MADV_RANDOM:
			case MADV_REMOVE:
			case MADV_WILLNEED:
			case MADV_DONTNEED:
			case MADV_FREE:
			case MADV_COLD:
			case MADV_PAGEOUT:
			#ifdef CONFIG_KSM
			case MADV_MERGEABLE:
			case MADV_UNMERGEABLE:
			#endif
			#ifdef CONFIG_TRANSPARENT_HUGEPAGE
			case MADV_HUGEPAGE:
			case MADV_NOHUGEPAGE:
			#endif
			case MADV_DONTDUMP:
			case MADV_DODUMP:
			case MADV_WIPEONFORK:
			case MADV_KEEPONFORK:
			#ifdef CONFIG_MEMORY_FAILURE
			case MADV_SOFT_OFFLINE:
			case MADV_HWPOISON:
			#endif
			case MADV_DOEXEC:
			case MADV_DONTEXEC:
			return true;
			
			default:
			return false;
			}
			}
				linux-5.15.0/mm/madvise.c#L1032
				
				static bool
				madvise_behavior_valid(int behavior)
				{
				switch (behavior) {
				case MADV_DOFORK:
				case MADV_DONTFORK:
				case MADV_NORMAL:
				case MADV_SEQUENTIAL:
				case MADV_RANDOM:
				case MADV_REMOVE:
				case MADV_WILLNEED:
				case MADV_DONTNEED:
				case MADV_FREE:
				case MADV_COLD:
				case MADV_PAGEOUT:
				case MADV_POPULATE_READ:
				case MADV_POPULATE_WRITE:
				#ifdef CONFIG_KSM
				case MADV_MERGEABLE:
				case MADV_UNMERGEABLE:
				#endif
				#ifdef CONFIG_TRANSPARENT_HUGEPAGE
				case MADV_HUGEPAGE:
				case MADV_NOHUGEPAGE:
				#endif
				case MADV_DONTDUMP:
				case MADV_DODUMP:
				case MADV_WIPEONFORK:
				case MADV_KEEPONFORK:
				#ifdef CONFIG_MEMORY_FAILURE
				case MADV_SOFT_OFFLINE:
				case MADV_HWPOISON:
				#endif
				case MADV_DOEXEC:
				case MADV_DONTEXEC:
				return true;
				
				default:
				return false;
				}
				}

Function do_madvise, see the return value of madvise(0, 0, advice) call, advice == 23.	https://github.com/torvalds/linux/blob/b401b621758e46812da61fa58a67c3fd8d91de0d/mm/madvise.c#L1398
	
	int do_madvise(struct mm_struct *mm, unsigned long start, size_t len_in, int behavior)
	{
	unsigned long end;
	int error;
	int write;
	size_t len;
	struct blk_plug plug;
	
	if (!madvise_behavior_valid(behavior))
	return -EINVAL;
	
	if (!PAGE_ALIGNED(start))
	return -EINVAL;
	len = PAGE_ALIGN(len_in);
	
	/* Check to see whether len was rounded up from small -ve to zero */
	if (len_in && !len)
	return -EINVAL;
	
	end = start + len;
	if (end < start)
	return -EINVAL;
	
	if (end == start)
	return 0;
		https://github.com/torvalds/linux/blob/219d54332a09e8d8741c1e1982f5eae56099de85/mm/madvise.c#L1047
		
		SYSCALL_DEFINE3(madvise, unsigned long, start, size_t, len_in, int, behavior)
		{
		unsigned long end, tmp;
		struct vm_area_struct *vma, *prev;
		int unmapped_error = 0;
		int error = -EINVAL;
		int write;
		size_t len;
		struct blk_plug plug;
		
		start = untagged_addr(start);
		
		if (!madvise_behavior_valid(behavior))
		return error;
		
		if (start & ~PAGE_MASK)
		return error;
		len = (len_in + ~PAGE_MASK) & PAGE_MASK;
		
		/* Check to see whether len was rounded up from small -ve to zero */
		if (len_in && !len)
		return error;
		
		end = start + len;
		if (end < start)
		return error;
		
		error = 0;
		if (end == start)
		return error;
			linux-5.4.17/mm/madvise.c#L1080
			
			int do_madvise(unsigned long start, size_t len_in, int behavior)
			{
			unsigned long end, tmp;
			struct vm_area_struct *vma, *prev;
			int unmapped_error = 0;
			int error = -EINVAL;
			int write;
			size_t len;
			struct blk_plug plug;
			
			start = untagged_addr(start);
			
			if (!madvise_behavior_valid(behavior))
			return error;
			
			if (!PAGE_ALIGNED(start))
			return error;
			len = PAGE_ALIGN(len_in);
			
			/* Check to see whether len was rounded up from small -ve to zero */
			if (len_in && !len)
			return error;
			
			end = start + len;
			if (end < start)
			return error;
			
			error = 0;
			if (end == start)
			return error;
				linux-5.15.0/mm/madvise.c#L1160
				
				int do_madvise(struct mm_struct *mm, unsigned long start, size_t len_in, int behavior)
				{
				unsigned long end, tmp;
				struct vm_area_struct *vma, *prev;
				int unmapped_error = 0;
				int error = -EINVAL;
				int write;
				size_t len;
				struct blk_plug plug;
				
				start = untagged_addr(start);
				
				if (!madvise_behavior_valid(behavior))
				return error;
				
				if (!PAGE_ALIGNED(start))
				return error;
				len = PAGE_ALIGN(len_in);
				
				/* Check to see whether len was rounded up from small -ve to zero */
				if (len_in && !len)
				return error;
				
				end = start + len;
				if (end < start)
				return error;
				
				error = 0;
				if (end == start)
				return error;