

Debugger

White-box debugger

IDE가 가

Black-box debugger

Black-box debugger 가 Reversing Engineering 가

User mode

가

Kernel mode

low-level component

가

- Breakpoint hits
- Memory violations
- exceptions

가

가

Breakpoint

Breakpoint

Breakpoint

가

Software breakpoint

CPU가 opcode(Operation Code) interrupt 3 (INT 3) instruction

Breakpoint opcode

0x44332211: 8BC3 MOV EAX, EBX

Breakpoint opcode

0x44332211: CCC3 MOV EAX, EBX

CRC

Hardware breakpoint

CPU debug register(DR0 ~ DR7) breakpoint
breakpoint가 . Debug register

Debug register	
DR0 ~ DR3	Breakpoint
DR4, DR5	Reserved
DR6	type
DR7	Breakpoint on/off, flag, length

4 , 4 break point

Memory breakpoint

()

Page execution	가 .	가 read/write	access violation
Page read	Page	가 .	Write/execution access violation
Page write	Page	가 .	
Guard page	Page		page

gaurd page가 reversing engineering

API

```

BOOL WINAPI CreateProcessA(
    LPCSTR lpApplicationName,
    LPTSTR lpCommandLine,
    LPSECURITY_ATTRIBUTES lpProcessAttributes,
    LPSECURITY_ATTRIBUTES lpThreadAttributes,
    BOOL bInheritHandles,
    DWORD dwCreationFlags,
    LPVOID lpEnvironment,
    LPCTSTR lpCurrentDirectory,
    LPSTARTUPINFO lpStartupInfo,
    LPPROCESS_INFORMATION lpProcessInformation
);

```

lpApplicationName, lpCommandLine, dwCreationFlags, lpStartupInfo, and lpProcessInformation 가 NULL . MSDN . STARTUPINFO, PROCESS_INFORMATION

CreateProcess : <http://msdn.microsoft.com/en-us/library/ms682425.aspx>
 STARTUPINFO : <http://msdn.microsoft.com/en-us/library/ms686331.aspx>
 PROCESS_INFORMATION : <http://msdn.microsoft.com/en-us/library/ms686331.aspx>

```

HANDLE WINAPI OpenProcess(
    DWORD dwDesiredAccess,
    BOOL bInheritHandle
    DWORD dwProcessId
);

```

dwDesiredAccess PROCESS_ALL_ACCESS, bInheritHandle FALSE 가 . dwProcessId PID .

OpenProcess : <http://msdn.microsoft.com/en-us/library/ms684320.aspx>

```

BOOL WINAPI DebugActiveProcess(
    DWORD dwProcessId
);

```

attach . Attach가 가 .

DebugActiveProcess : <http://msdn.microsoft.com/en-us/library/ms679295.aspx>

```

BOOL WINAPI WaitForDebugEvent(

```

```
LPDEBUG_EVENT lpDebugEvent,  
DWORD dwMilliseconds  
);
```

```
typedef struct DEBUG_EVENT {  
    DWORD dwDebugEventCode;  
    DWORD dwProcessId;  
    DWORD dwThreadId;  
    union {  
        EXCEPTION_DEBUG_INFO Exception;  
        CREATE_THREAD_DEBUG_INFO CreateThread;  
        CREATE_PROCESS_DEBUG_INFO CreateProcessInfo;  
        EXIT_THREAD_DEBUG_INFO ExitThread;  
        EXIT_PROCESS_DEBUG_INFO ExitProcess;  
        LOAD_DLL_DEBUG_INFO LoadDll;  
        UNLOAD_DLL_DEBUG_INFO UnloadDll;  
        OUTPUT_DEBUG_STRING_INFO DebugString;  
        RIP_INFO RipInfo;  
    } u;  
};
```

```
        . dwMilliseconds    INFINITE  
    . lpDebugEvent
```

WaitForDebugEvent : <http://msdn.microsoft.com/en-us/library/ms681423.aspx>

DEBUG_EVENT : <http://msdn.microsoft.com/en-us/library/ms679308.aspx>

```
BOOL WINAPI ContinueDebugEvent(  
    DWORD dwProcessId,  
    DWORD dwThreadId,  
    DWORD dwContinueStatus  
);
```

```
        WaitForDebugEvent    가  
dwThreadId가                . dwContinueStatus    DBG_CONTINUE  
DBG_EXCEPTION_NOT_HANDLED가    exception
```

ContinueDebugEvent : <http://msdn.microsoft.com/en-us/library/ms679285.aspx>

```
HANDLE WINAPI OpenThread(  
    DWORD dwDesiredAccess,  
    BOOL bInheritHandle,  
    DWORD dwThreadId
```

```
);
```

OpenThread : <http://msdn.microsoft.com/en-us/library/ms684335.aspx>

```
HANDLE WINAPI CreateToolhelp32Snapshot(  
    DWORD dwFlags,  
    DWORD th32ProcessID  
);
```

CreateToolhelp32Snapshot : <http://msdn.microsoft.com/en-us/library/ms682489.aspx>

```
BOOL WINAPI Thread32First(  
    HANDLE hSnapshot,  
    LPTHREADENTRY32 lpte  
);
```

```
typedef struct THREADENTRY32{  
    DWORD dwSize;  
    DWORD cntUsage;  
    DWORD th32ThreadID;  
    DWORD th32OwnerProcessID;  
    LONG tpBasePri;  
    LONG tpDeltaPri;  
    DWORD dwFlags;  
};
```

Thread32First : <http://msdn.microsoft.com/en-us/library/ms686728.aspx>

THREADENTRY32 : <http://msdn.microsoft.com/en-us/library/ms686735.aspx>

```
BOOL WINAPI GetThreadContext(  
    HANDLE hThread,  
    LPCONTEXT lpContext  
);
```

```
BOOL WINAPI SetThreadContext(  
    HANDLE hThread,  
    LPCONTEXT lpContext  
);
```

GetThreadContext : <http://msdn.microsoft.com/en-us/library/ms679362.aspx>

SetThreadContext : <http://msdn.microsoft.com/en-us/library/ms680632.aspx>

```
typedef struct CONTEXT {  
    DWORD ContextFlags;  
    DWORD Dr0;  
    DWORD Dr1;  
    DWORD Dr2;  
    DWORD Dr3;  
    DWORD Dr6;  
    DWORD Dr7;
```

```
FLOATING_SAVE_AREA FloatSave;  
DWORD SegGs;  
DWORD SegFs;  
DWORD SegEs;  
DWORD SegDs;  
DWORD Edi;  
DWORD Esi;  
DWORD Ebx;  
DWORD Edx;  
DWORD Ecx;  
DWORD Eax;  
DWORD Ebp;  
DWORD Eip;  
DWORD SegCs;  
DWORD EFlags;  
DWORD Esp;  
DWORD SegSs;  
BYTE ExtendedRegisters[MAXIMUM_SUPPORTED_EXTENSION];  
};
```

CONTEXT : [http://msdn.microsoft.com/en-us/library/windows/desktop/ms679284\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/windows/desktop/ms679284(v=vs.85).aspx)

Breakpoint

Soft Breakpoints

```
BOOL WINAPI ReadProcessMemory(  
    HANDLE hProcess,  
    LPCVOID lpBaseAddress,  
    LPVOID lpBuffer,  
    SIZE_T nSize,  
    SIZE_T* lpNumberOfBytesRead  
);  
  
BOOL WINAPI WriteProcessMemory(  
    HANDLE hProcess,  
    LPCVOID lpBaseAddress,  
    LPCVOID lpBuffer,  
    SIZE_T nSize,  
    SIZE_T* lpNumberOfBytesWritten  
);
```

ReadProcessMemory : <http://msdn.microsoft.com/en-us/library/ms680553.aspx>

WriteProcessMemory : <http://msdn.microsoft.com/en-us/library/ms681674.aspx>

```
FARPROC WINAPI GetProcAddress(  
    HMODULE hModule,
```

```
LPCSTR lpProcName
);

HMODULE WINAPI GetModuleHandle(
    LPCSTR lpModuleName
);
```

GetProcAddress : <http://msdn.microsoft.com/en-us/library/ms683212.aspx>

GetModuleHandle : <http://msdn.microsoft.com/en-us/library/ms683199.aspx>

Memory Breakpoints

```
void WINAPI GetSystemInfo(
    _Out_ LPSYSTEM_INFO lpSystemInfo
);

typedef struct _SYSTEM_INFO {
    union {
        DWORD dwOemId;
        struct {
            WORD wProcessorArchitecture;
            WORD wReserved;
        };
    };
    DWORD dwPageSize;
    LPVOID lpMinimumApplicationAddress;
    LPVOID lpMaximumApplicationAddress;
    DWORD_PTR dwActiveProcessorMask;
    DWORD dwNumberOfProcessors;
    DWORD dwProcessorType;
    DWORD dwAllocationGranularity;
    WORD wProcessorLevel;
    WORD wProcessorRevision;
} SYSTEM_INFO;
```

GetSystemInfo : <http://msdn.microsoft.com/en-us/library/ms724381.aspx>

SYSTEM_INFO : <http://msdn.microsoft.com/en-us/library/ms724958.aspx>

```
SIZE_T WINAPI VirtualQuery(
    HANDLE hProcess,
    LPCVOID lpAddress,
    PMEMORY_BASIC_INFORMATION lpBuffer,
    SIZE_T dwLength
);

typedef struct MEMORY_BASIC_INFORMATION{
    PVOID BaseAddress;
    PVOID AllocationBase;
    DWORD AllocationProtect;
```

```
SIZE_T RegionSize;  
DWORD State;  
DWORD Protect;  
DWORD Type;  
}
```

VirtualQueryEx : <http://msdn.microsoft.com/en-us/library/aa366907.aspx>

MEMORY_BASIC_INFORMATION : <http://msdn.microsoft.com/en-us/library/aa366775.aspx>

```
BOOL WINAPI VirtualProtectEx(  
    HANDLE hProcess,  
    LPVOID lpAddress,  
    SIZE_T dwSize,  
    DWORD flNewProtect,  
    PDWORD lpflOldProtect  
);
```

VirtualProtectEx : <http://msdn.microsoft.com/en-us/library/aa366899>

From:
<http://www.obg.co.kr/doku/> - **OBG Wiki**

Permanent link:
http://www.obg.co.kr/doku/doku.php?id=programming:reverse_engineering:debugger

Last update: **2020/11/29 14:09**

