```
struct s0 {
signed char[4] pad4;
int32_t f4;
};

/* ?cout@std@@3V?$basic_ostream@DU?$char_traits@D@std@@@1@A */
struct s0** cout_std_3V_basic_ostream_DU_char_traits_D_std_1_A = reinterpret_cast<struct
s0**>(0x37f8);

struct s0** fun_4010e0(struct s0** ecx);

/*
* ??6?$basic_ostream@DU?$char_traits@D@std@@@std@@QAEAAV01@H@Z
* public: class std::basic_ostream<char,struct std::char_traits<char> > & __thiscall
std::basic_ostream<char,struct std::char_traits<char> >::operator<<(int)
*/
int32_t _6_basic_ostream_DU_char_traits_D_std_std_QAEAAV01_H_Z = 0x399c;

int32_t fun_401000(void** ecx) {
struct s0** ecx2;
struct s0** eax3;

ecx2 = cout_std_3V_basic_ostream_DU_char_traits_D_std_1_A;
eax3 = fun_4010e0(ecx2);
_6_basic_ostream_DU_char_traits_D_std_std_QAEAAV01_H_Z(eax3);
goto 0x8000;
}

void* g0;

struct s2 {
signed char[4] pad4;
int32_t f4;
};

struct s1 {
signed char[32] pad32;
uint32_t f32;
uint32_t f36;
signed char[16] pad56;
struct s2** f56;
};

struct s3 {
signed char[12] pad12;
```

```
int32_t f12;
signed char[44] pad60;
struct s0** f60;
};

/*
* ?flush@?$basic_ostream@DU?$char_traits@D@std@@@std@@QAEAAV12@XZ
* public: class std::basic_ostream<char,struct std::char_traits<char> > & __thiscall
std::basic_ostream<char,struct std::char_traits<char> >::flush(void)
*/
int32_t flush_basic_ostream_DU_char_traits_D_std_std_QAEAAV12_XZ = 0x391a;

/*
* ?sputn@?$basic_streambuf@DU?$char_traits@D@std@@@std@@QAE_JPBD_J@Z
* public: __int64 __thiscall std::basic_streambuf<char,struct std::char_traits<char> >::sputn(char
const *,__int64)
*/
int32_t sputn_basic_streambuf_DU_char_traits_D_std_std_QAE_JPBD_J_Z = 0x3856;

/*
* ?sputc@?$basic_streambuf@DU?$char_traits@D@std@@@std@@QAEHD@Z
* public: int __thiscall std::basic_streambuf<char,struct std::char_traits<char> >::sputc(char)
*/
int32_t sputc_basic_streambuf_DU_char_traits_D_std_std_QAEHD_Z = 0x389c;

/*
* ?setstate@?$basic_ios@DU?$char_traits@D@std@@@std@@QAEXH_N@Z
* public: void __thiscall std::basic_ios<char,struct std::char_traits<char> >::setstate(int,bool)
*/
int32_t setstate_basic_ios_DU_char_traits_D_std_std_QAEXH_N_Z = 0x395c;

/*
* ?uncaught_exception@std@@YA_NXZ
* bool __cdecl std::uncaught_exception(void)
*/
int32_t uncaught_exception_std_YA_NXZ = 0x3834;

/*
* ?_Osfx@?$basic_ostream@DU?$char_traits@D@std@@@std@@QAEXXZ
* public: void __thiscall std::basic_ostream<char,struct std::char_traits<char> >::_Osfx(void)
*/
int32_t _Osfx_basic_ostream_DU_char_traits_D_std_std_QAEXXZ = 0x38dc;

struct s4 {
signed char[8] pad8;
int32_t f8;
```

```
};

struct s0** fun_4010e0(struct s0** ecx) {
void* eax2;
void* v3;
struct s0** ebx4;
struct s0* ecx5;
struct s1* eax6;
struct s1* v7;
uint1_t less8;
uint1_t less9;
uint32_t edi10;
uint32_t v11;
uint32_t esi12;
uint32_t v13;
struct s2** eax14;
struct s0* edx15;
struct s0** v16;
struct s2* eax17;
struct s2** ecx18;
struct s3* eax19;
unsigned char al20;
struct s0** ecx21;
int32_t eax22;
int32_t ecx23;
int32_t eax24;
uint1_t less25;
int32_t ecx26;
int32_t eax27;
uint32_t tmp32_28;
uint1_t cf29;
uint1_t less30;
int32_t ecx31;
int32_t eax32;
uint32_t tmp32_33;
uint1_t cf34;
int32_t eax35;
signed char al36;
struct s4* eax37;

eax2 = g0;
v3 = eax2;
g0 = reinterpret_cast<void*>(reinterpret_cast<int32_t>(__zero_stack_offset()) - 4 - 12);
ebx4 = ecx;
ecx5 = *ebx4;
eax6 = reinterpret_cast<struct s1*>(ecx5->f4 + reinterpret_cast<int32_t>(ebx4));
```

```
v7 = eax6;
less8 = reinterpret_cast<uint1_t>(reinterpret_cast<int32_t>(eax6->f36) <
reinterpret_cast<int32_t>(0));
if (less8 || reinterpret_cast<uint1_t>(less8 | reinterpret_cast<uint1_t>(eax6->f36 == 0)) &&
(!eax6->f32 || ((less9 = reinterpret_cast<uint1_t>(reinterpret_cast<int32_t>(eax6->f36) <
reinterpret_cast<int32_t>(0)), less9) || reinterpret_cast<uint1_t>(less9 |
reinterpret_cast<uint1_t>(eax6->f36 == 0)) && eax6->f32 <= 25))) {
__asm__("xorps xmm0, xmm0");
__asm__("movlpd [ebp-0x24], xmm0");
edi10 = v11;
esi12 = v13;
} else {
esi12 = eax6->f32 - 25;
edi10 = eax6->f36 - reinterpret_cast<uint1_t>(eax6->f36 <
static_cast<uint32_t>(reinterpret_cast<uint1_t>(eax6->f32 < 25)));
}
eax14 = eax6->f56;
edx15 = ecx5;
v16 = ebx4;
if (eax14) {
eax17 = *eax14;
ecx18 = v7->f56;
eax17->f4(ecx18);
ecx5 = *ebx4;
edx15 = ecx5;
}
eax19 = reinterpret_cast<struct s3*>(ecx5->f4 + reinterpret_cast<int32_t>(ebx4));
if (!eax19->f12) {
if (!eax19->f60 || eax19->f60 == ebx4) {
al20 = 1;
} else {
ecx21 = eax19->f60;
flush_basic_ostream_DU_char_traits_D_std_std_QAEAAV12_XZ(ecx21);
ecx5 = *ebx4;
edx15 = ecx5;
al20 = reinterpret_cast<uint1_t>(*reinterpret_cast<int32_t*>(ecx5->f4 +
reinterpret_cast<int32_t>(ebx4) + 12) == 0);
}
} else {
al20 = 0;
}
if (al20) {
if ((*reinterpret_cast<uint32_t*>(ecx5->f4 + reinterpret_cast<int32_t>(ebx4) + 20) & 0x1c0) ==
64) {
addr_40120d_13:
eax22 = edx15->f4;
```

```
ecx23 = *reinterpret_cast<int32_t*>(eax22 + reinterpret_cast<int32_t>(ebx4) + 56);
eax24 =
reinterpret_cast<int32_t>(sputn_basic_streambuf_DU_char_traits_D_std_std_QAE_JPBD_J_Z(
ecx23, "Sum of the two number is ", 25, 0));
if (eax24 != 25 || edx15) {
addr_401258_14:
} else {
while ((less25 = reinterpret_cast<uint1_t>(reinterpret_cast<int32_t>(edi10) <
reinterpret_cast<int32_t>(0)), !less25) && (!reinterpret_cast<uint1_t>(less25 |
reinterpret_cast<uint1_t>(edi10 == 0)) || esi12)) {
ecx26 = *reinterpret_cast<int32_t*>((*ebx4)->f4 + reinterpret_cast<int32_t>(ebx4) + 56);
*reinterpret_cast<signed char*>(&v7) = *reinterpret_cast<signed char*>((*ebx4)->f4 +
reinterpret_cast<int32_t>(ebx4) + 64);
eax27 =
reinterpret_cast<int32_t>(sputc_basic_streambuf_DU_char_traits_D_std_std_QAEHD_Z(ecx26,
v7, "Sum of the two number is ", 25, 0));
if (eax27 == -1)
goto addr_401258_14;
tmp32_28 = esi12 - 1;
cf29 = reinterpret_cast<uint1_t>(tmp32_28 < esi12);
esi12 = tmp32_28;
edi10 = edi10 + 0xffffffff + cf29;
}
goto addr_401267_19;
}
} else {
while ((less30 = reinterpret_cast<uint1_t>(reinterpret_cast<int32_t>(edi10) <
reinterpret_cast<int32_t>(0)), !less30) && (!reinterpret_cast<uint1_t>(less30 |
reinterpret_cast<uint1_t>(edi10 == 0)) || esi12)) {
ecx31 = *reinterpret_cast<int32_t*>((*ebx4)->f4 + reinterpret_cast<int32_t>(ebx4) + 56);
*reinterpret_cast<signed char*>(&v7) = *reinterpret_cast<signed char*>((*ebx4)->f4 +
reinterpret_cast<int32_t>(ebx4) + 64);
eax32 =
reinterpret_cast<int32_t>(sputc_basic_streambuf_DU_char_traits_D_std_std_QAEHD_Z(ecx31,
v7));
if (eax32 == -1)
goto addr_4011fe_22;
tmp32_33 = esi12 - 1;
cf34 = reinterpret_cast<uint1_t>(tmp32_33 < esi12);
esi12 = tmp32_33;
edi10 = edi10 + 0xffffffff + cf34;
}
goto addr_40120b_24;
}
} else {
goto addr_4012a7_26;
```

```
}
addr_401269_27:
eax35 = (*ebx4)->f4;
*reinterpret_cast<int32_t*>(eax35 + reinterpret_cast<int32_t>(ebx4) + 32) = 0;
*reinterpret_cast<int32_t*>(eax35 + reinterpret_cast<int32_t>(ebx4) + 36) = 0;
addr_4012a7_26:
setstate_basic_ios_DU_char_traits_D_std_std_QAEXH_N_Z();
al36 = reinterpret_cast<signed char>(uncaught_exception_std_YA_NXZ());
if (!al36) {
_Osfx_basic_ostream_DU_char_traits_D_std_std_QAEXXZ(v16);
}
if (*reinterpret_cast<struct s4***>((*v16)->f4 + reinterpret_cast<int32_t>(v16) + 56)) {
eax37 = **reinterpret_cast<struct s4***>((*v16)->f4 + reinterpret_cast<int32_t>(v16) + 56);
eax37->f8();
}
g0 = v3;
return ebx4;
addr_401267_19:
goto addr_401269_27;
addr_40120b_24:
edx15 = *ebx4;
goto addr_40120d_13;
addr_4011fe_22:
goto addr_401269_27;
}

/* (image base) */
int32_t image_base_ = 0x4019a6;

void fun_401bb4() {
int32_t* esi1;
int32_t edi2;

esi1 = reinterpret_cast<int32_t*>(0x40354c);
if (!1) {
do {
edi2 = *esi1;
if (edi2) {
image_base_(edi2);
edi2(edi2);
}
++esi1;
} while (reinterpret_cast<uint32_t>(esi1) < 0x40354c);
}
return;
}
```

```
uint32_t fun_40188f(void** a1);

int32_t fun_4018bc(void** a1) {
uint32_t eax2;
uint32_t eax3;

eax2 = fun_40188f(a1);
eax3 = -eax2;
return reinterpret_cast<int32_t>(-(eax3 - (eax3 + reinterpret_cast<uint1_t>(eax3 < eax3 +
reinterpret_cast<uint1_t>(!!eax2))))) - 1;
}

int32_t fun_40196c() {
return 1;
}

int32_t _configure_narrow_argv = 0x3ab2;

int32_t fun_401eac() {
goto _configure_narrow_argv;
}

int32_t InitializeSListHead = 0x3dc0;

void fun_401976(void** ecx) {
InitializeSListHead();
goto 0x404358;
}

int32_t g40400c = 1;

int32_t fun_4019d0(void** ecx) {
int32_t eax2;
uint1_t zf3;

eax2 = 0;
zf3 = reinterpret_cast<uint1_t>(g40400c == 0);
*reinterpret_cast<unsigned char*>(&eax2) = zf3;
return eax2;
}

int32_t __setusermatherr = 0x3a9e;

void fun_401ea6(void** ecx) {
goto __setusermatherr;
```

```
}

int32_t _initialize_narrow_environment = 0x3acc;

void fun_401eb2(void** ecx) {
goto _initialize_narrow_environment;
}

void** fun_401969(void** ecx, int32_t a2, int32_t a3) {
return 0;
}

int32_t fun_401b02(void** ecx) {
return 0;
}

int32_t fun_401e6a();

struct s5 {
signed char[4] pad4;
int32_t f4;
};

struct s5* g18;

int32_t g404338;

signed char fun_4016c4(void** ecx) {
int32_t eax2;
struct s5* eax3;
int32_t edx4;
int1_t zf5;

eax2 = fun_401e6a();
if (!eax2) {
addr_4016ee_2:
return 0;
} else {
eax3 = g18;
edx4 = eax3->f4;
do {
zf5 = g404338 == edx4;
if (zf5) {
g404338 = edx4;
}
if (!0)
```

```
goto addr_4016ee_2;
} while (edx4);
}
return 1;
}

int32_t _initterm_e = 0x3b1c;

int32_t fun_401ec4() {
goto _initterm_e;
}

void fun_401c10(void** ecx, int32_t a2, int32_t a3);

/* (image base) */
int16_t image_base_;

int32_t g40003c;

struct s6 {
signed char[8] pad8;
int32_t f8;
};

struct s7 {
signed char[8] pad8;
int32_t f8;
uint32_t f12;
signed char[20] pad36;
int32_t f36;
};

struct s8 {
signed char[60] pad60;
int32_t f60;
};

struct s7* fun_401680(struct s8* a1, uint32_t a2);

struct s9 {
signed char[4] pad4;
int32_t f4;
};

signed char fun_4017b6(void** ecx, void** a2) {
int32_t ebp3;
```

```
int1_t zf4;
int32_t eax5;
uint32_t v6;
struct s6* ebp7;
struct s7* eax8;
int32_t ebp9;
int32_t ebp10;
int32_t ebp11;
struct s9* ebp12;

fun_401c10(ecx, 0x403640, 8);
*reinterpret_cast<uint32_t*>(ebp3 - 4) = 0;
zf4 = image_base_ == 0x5a4d;
if (!zf4 || ((eax5 = g40003c, *reinterpret_cast<int32_t*>(eax5 + 0x400000) != 0x4550) ||
(*reinterpret_cast<int16_t*>(eax5 + 0x400018) != 0x10b || ((v6 =
reinterpret_cast<uint32_t>(ebp7->f8 - 0x400000), eax8 = fun_401680(0x400000, v6), eax8 ==
0) || eax8->f36 < 0)))) {
*reinterpret_cast<int32_t*>(ebp9 - 4) = -2;
} else {
*reinterpret_cast<int32_t*>(ebp10 - 4) = -2;
}
g0 = *reinterpret_cast<void**>(ebp11 - 16);
goto ebp12->f4;
}

int32_t _register_thread_local_exe_atexit_callback = 0x3b78;

void fun_401ef4(void** ecx) {
goto _register_thread_local_exe_atexit_callback;
}

int32_t _cexit = 0x3b64;

void fun_401ee8(void** ecx) {
goto _cexit;
}

int32_t _c_exit = 0x3b6e;

void fun_401eee() {
goto _c_exit;
}

int32_t SetUnhandledExceptionFilter = 0x3cfe;

int32_t UnhandledExceptionFilter = 0x3ce2;
```

```
int32_t GetCurrentProcess = 0x3d1c;

int32_t TerminateProcess = 0x3d30;

void fun_40155f(int32_t a1, void* a2) {
int32_t ebp3;

SetUnhandledExceptionFilter(0, ebp3, __return_address());
UnhandledExceptionFilter(a1, 0, ebp3, __return_address());
GetCurrentProcess();
TerminateProcess();
goto 0xc0000409;
}

struct s10 {
signed char[6] pad6;
uint16_t f6;
signed char[12] pad20;
uint16_t f20;
};

struct s7* fun_401680(struct s8* a1, uint32_t a2) {
struct s10* ecx3;
struct s7* edx4;
struct s7* esi5;
struct s7* eax6;
uint32_t ecx7;

ecx3 = reinterpret_cast<struct s10*>(a1->f60 + reinterpret_cast<int32_t>(a1));
edx4 = reinterpret_cast<struct s7*>(reinterpret_cast<int32_t>(ecx3) + 24 + ecx3->f20);
esi5 = edx4 + ecx3->f6;
if (edx4 == esi5) {
addr_4016bb_2:
eax6 = reinterpret_cast<struct s7*>(0);
} else {
ecx7 = a2;
do {
if (ecx7 < edx4->f12)
continue;
if (ecx7 < edx4->f8 + edx4->f12)
goto addr_4016c0_6;
++edx4;
} while (edx4 != esi5);
goto addr_4016bb_2;
}
```

```
addr_4016bd_8:
return eax6;
addr_4016c0_6:
eax6 = edx4;
goto addr_4016bd_8;
}

uint32_t g404374;

uint32_t g404010 = 1;

int32_t fun_401f2a(void* a1, int32_t a2);

uint32_t g404378;

int32_t fun_401c9a() {
int32_t v1;
int32_t eax2;
uint32_t eax3;
uint32_t edi4;
uint32_t edi5;
uint32_t ecx6;
uint32_t eax7;
uint32_t eax8;

g404374 = 0;
g404010 = g404010 | 1;
eax2 = fun_401f2a(10, v1);
if (eax2) {
if (__intrinsic() ^ 0x756e6547 | __intrinsic() ^ 0x6c65746e | __intrinsic() ^ 0x49656e69 || (eax3
= __intrinsic() & 0xfff3ff0, eax3 != 0x106c0) && (eax3 != 0x20660 && (eax3 != 0x20670 &&
(eax3 != 0x30650 && (eax3 != 0x30660 && eax3 != 0x30670))))) {
edi4 = g404378;
} else {
edi5 = g404378;
edi4 = edi5 | 1;
g404378 = edi4;
}
ecx6 = __intrinsic();
if (__intrinsic() >= 7) {
ecx6 = ecx6;
if (__intrinsic() & 0x200) {
g404378 = edi4 | 2;
}
}
eax7 = g404010;
```

```
eax8 = eax7 | 2;
g404374 = 1;
g404010 = eax8;
if (!(ecx6 & 0x100000) || ((g404374 = 2, g404010 = eax8 | 4, (ecx6 & 0x8000000) == 0) || !(ecx6
& 0x10000000))) {
}
}
return 0;
}

signed char fun_401982(void** ecx) {
return 1;
}

int32_t g404014 = 1;

int32_t fun_401e6a() {
int32_t eax1;
int1_t zf2;

eax1 = 0;
zf2 = g404014 == 0;
*reinterpret_cast<unsigned char*>(&eax1) = reinterpret_cast<uint1_t>(!zf2);
return eax1;
}

int32_t _initialize_onexit_table = 0x3bdc;

int32_t fun_401f0c(void** ecx) {
goto _initialize_onexit_table;
}

int32_t _crt_atexit = 0x3c14;

int32_t fun_401f18() {
goto _crt_atexit;
}

int32_t GetSystemTimeAsFileTime = 0x3da6;

int32_t GetCurrentThreadId = 0x3d90;

int32_t GetCurrentProcessId = 0x3d7a;

int32_t QueryPerformanceCounter = 0x3d60;
```

```
uint32_t fun_4018d1() {
void* ebp1;
void* v2;
uint32_t eax3;
uint32_t eax4;
uint32_t v5;
uint32_t v6;

ebp1 = reinterpret_cast<void*>(reinterpret_cast<int32_t>(__zero_stack_offset()) - 4);
v2 = reinterpret_cast<void*>(reinterpret_cast<int32_t>(ebp1) - 12);
GetSystemTimeAsFileTime(v2);
eax3 = reinterpret_cast<uint32_t>(GetCurrentThreadId(v2));
eax4 = reinterpret_cast<uint32_t>(GetCurrentProcessId(v2));
QueryPerformanceCounter(reinterpret_cast<int32_t>(ebp1) - 20, v2);
return v5 ^ v6 ^ (eax3 ^ eax4) ^ reinterpret_cast<uint32_t>(reinterpret_cast<int32_t>(ebp1) - 4);
}

uint32_t g404370;

void fun_401bac(int32_t a1, void* a2, int32_t a3, void* a4, int32_t a5, int32_t a6) {
g404370 = 0;
return;
}

int32_t memset = 0x3a46;

int32_t fun_401e8e(void* a1, int32_t a2, int32_t a3, void* a4, int32_t a5, int32_t a6) {
goto memset;
}

uint32_t g404004 = 0xbb40e64e;

int32_t g404118 = 0;

uint32_t g404114 = 0;

int32_t g404110 = 0;

int32_t g40410c = 0;

int32_t g404108 = 0;

int32_t g404104 = 0;

int16_t g404130 = 0;
```

```
int16_t g404124 = 0;

int16_t g404100 = 0;

int16_t g4040fc = 0;

int16_t g4040f8 = 0;

int16_t g4040f4 = 0;

uint32_t g404128 = 0;

int32_t g40411c = 0;

int32_t g404120 = 0;

void* g40412c = reinterpret_cast<void*>(0);

int32_t g404068 = 0;

int32_t g404024 = 0;

int32_t g404018 = 0;

int32_t g40401c = 0;

int32_t g404028 = 0;

int32_t g40402c = 0;

void fun_4012fd(uint32_t ecx, int32_t a2) {
uint32_t eflags3;
int1_t zf4;
int32_t v5;
int32_t eax6;
int32_t edx7;
int32_t ebx8;
int32_t esi9;
int32_t edi10;
int16_t ss11;
int16_t cs12;
int16_t ds13;
int16_t es14;
int16_t fs15;
int16_t gs16;
int32_t ebp17;
```

```
int32_t eax18;

*reinterpret_cast<int1_t*>(reinterpret_cast<int32_t>(&eflags3) + 1) = 0;
zf4 = ecx == g404004;
if (zf4) {
return;
}
eax6 = fun_401f2a(23, v5);
*reinterpret_cast<int1_t*>(&eflags3) = 0;
*reinterpret_cast<int1_t*>(&eflags3) = __intrinsic();
*reinterpret_cast<int1_t*>(&eflags3) = eax6 == 0;
*reinterpret_cast<int1_t*>(&eflags3) = eax6 < 0;
*reinterpret_cast<int1_t*>(reinterpret_cast<int32_t>(&eflags3) + 1) = 0;
*reinterpret_cast<int1_t*>(&eflags3) = __undefined();
if (!*reinterpret_cast<int1_t*>(&eflags3))
goto addr_40159b_5;
addr_4015a0_6:
g404118 = eax6;
g404114 = ecx;
g404110 = edx7;
g40410c = ebx8;
g404108 = esi9;
g404104 = edi10;
g404130 = ss11;
g404124 = cs12;
g404100 = ds13;
g4040fc = es14;
g4040f8 = fs15;
g4040f4 = gs16;
g404128 = eflags3 & 0xfcffff;
g40411c = ebp17;
g404120 = reinterpret_cast<int32_t>(__return_address());
g40412c = reinterpret_cast<void*>(reinterpret_cast<int32_t>(__zero_stack_offset()) - 4 + 8);
g404068 = 0x10001;
eax18 = g404120;
g404024 = eax18;
g404018 = 0xc0000409;
g40401c = 1;
g404028 = 1;
g40402c = 2;
fun_40155f(0x403118, 23);
return;
addr_40159b_5:
ecx = 2;
__asm__("int 0x29");
goto addr_4015a0_6;
```

```
}

int32_t g404340;

int32_t fun_401f12();

uint32_t fun_40188f(void** a1) {
int1_t zf2;
int32_t eax3;
uint32_t eax4;

zf2 = g404340 == -1;
if (!zf2) {
eax3 = fun_401f12();
} else {
eax3 = fun_401f18();
}
eax4 = reinterpret_cast<uint32_t>(-eax3);
return ~(eax4 - (eax4 + reinterpret_cast<uint1_t>(eax4 < eax4 +
reinterpret_cast<uint1_t>(!!eax3)))) & reinterpret_cast<unsigned char>(a1);
}

int32_t fun_401970(int32_t a1) {
return 0x4000;
}

int32_t IsDebuggerPresent = 0x3dd6;

void fun_4019e8(void** ecx, void** a2) {
void* ebp3;
int32_t ebx4;
int32_t eax5;
void* v6;
int32_t v7;
int32_t v8;
int32_t v9;
int32_t eax10;
int32_t ebx11;
int32_t ebx12;
int32_t eax13;

ebp3 = reinterpret_cast<void*>(reinterpret_cast<int32_t>(__zero_stack_offset()) - 4);
eax5 = fun_401f2a(23, ebx4);
if (eax5) {
__asm__("int 0x29");
}
```

```
fun_401bac(3, 23, ebx4, v6, v7, v8);
fun_401e8e(reinterpret_cast<int32_t>(ebp3) - 0x324, 0, 0x2cc, 23, ebx4, v9);
fun_401e8e(reinterpret_cast<int32_t>(ebp3) - 88, 0, 80, 23, ebx4, 0x10001);
eax10 = reinterpret_cast<int32_t>(IsDebuggerPresent(23, ebx4, 0x10001));
ebx11 = eax10 - 1;
ebx12 = -ebx11;
SetUnhandledExceptionFilter(0, 23, ebx4, 0x10001);
eax13 = reinterpret_cast<int32_t>(UnhandledExceptionFilter());
if (!eax13 && !(reinterpret_cast<unsigned char>(*reinterpret_cast<unsigned char*>(&ebx12) -
reinterpret_cast<unsigned char>(*reinterpret_cast<unsigned char*>(&ebx12) +
(*reinterpret_cast<unsigned char*>(&ebx12) < reinterpret_cast<unsigned
char>(*reinterpret_cast<unsigned char*>(&ebx12) + reinterpret_cast<uint1_t>(!!ebx11))))) +
1)) {
fun_401bac(3, reinterpret_cast<int32_t>(ebp3) - 8, 0, 23, ebx4, 0x10001);
}
return;
}

struct s11 {
uint32_t f0;
int32_t f4;
};

struct s11* fun_4019a7();

struct s12 {
uint32_t f0;
int32_t f4;
};

struct s12* fun_4019ad();

void fun_4019b3(void** ecx) {
struct s11* eax2;
int32_t ecx3;
struct s12* eax4;
int32_t ecx5;

eax2 = fun_4019a7();
ecx3 = eax2->f4;
eax2->f0 = eax2->f0 | 36;
eax2->f4 = ecx3;
eax4 = fun_4019ad();
ecx5 = eax4->f4;
eax4->f0 = eax4->f0 | 2;
eax4->f4 = ecx5;
```

```
return;
}

void** fun_4019dc(void** ecx) {
return 0x404380;
}

void fun_4019a6(void** ecx) {
return;
}

int32_t fun_401f1e(int32_t a1, int32_t a2, int32_t a3);

void fun_401985(void** ecx) {
int32_t eax2;

eax2 = fun_401f1e(0, 0x10000, 0x30000);
if (eax2) {
fun_4019e8(ecx, 7);
goto 7;
} else {
return;
}
}

int32_t _configthreadlocale = 0x3ba6;

void fun_401efa(void** ecx) {
goto _configthreadlocale;
}

int32_t _get_initial_narrow_environment = 0x3aee;

int32_t fun_401eb8(void** ecx) {
goto _get_initial_narrow_environment;
}

int32_t GetModuleHandleW = 0x3dea;

struct s13 {
int16_t f0;
signed char[58] pad60;
int32_t f60;
};

struct s14 {
```

```
int32_t f0;
signed char[20] pad24;
int16_t f24;
signed char[90] pad116;
uint32_t f116;
signed char[112] pad232;
int32_t f232;
};

signed char fun_401b07(void** ecx) {
struct s13* eax2;
struct s14* ecx3;

eax2 = reinterpret_cast<struct s13*>(GetModuleHandleW());
if (!eax2 || (eax2->f0 != 0x5a4d || ((ecx3 = reinterpret_cast<struct s14*>(eax2->f60 +
reinterpret_cast<int32_t>(eax2)), ecx3->f0 != 0x4550) || (ecx3->f24 != 0x10b || (ecx3->f116 <=
14 || !ecx3->f232))))) {
goto 0;
} else {
goto 0;
}
}

int32_t exit = 0x3b2a;

void fun_401eca(void** ecx) {
goto exit;
}

int32_t _exit = 0x3b32;

void fun_401ed0(void** ecx) {
goto _exit;
}

int32_t g404000 = 0x44bf19b1;

void fun_40191e(void** ecx) {
uint32_t ecx2;
uint32_t eax3;

ecx2 = g404004;
if (ecx2 == 0xbb40e64e || !(0xffff0000 & ecx2)) {
eax3 = fun_4018d1();
ecx2 = eax3;
if (ecx2 != 0xbb40e64e) {
```

```
if (!(0xffff0000 & ecx2)) {
ecx2 = ecx2 | (eax3 | 0x4711) << 16;
}
} else {
ecx2 = 0xbb40e64f;
}
g404004 = ecx2;
}
g404000 = reinterpret_cast<int32_t>(~ecx2);
return;
}

int32_t fun_40184a(void** ecx, void** a2) {
int32_t eax3;
int32_t tmp32_4;

eax3 = fun_401e6a();
if (eax3 && !a2) {
tmp32_4 = g404338;
g404338 = 0;
eax3 = tmp32_4;
}
return eax3;
}

int32_t _initterm = 0x3b10;

void fun_401ebe(int32_t ecx) {
goto _initterm;
}

void fun_401c10(void** ecx, int32_t a2, int32_t a3) {
void* esp4;
void* ebp5;
int32_t* esp6;
int32_t ebx7;
int32_t* esp8;
int32_t esi9;
int32_t* esp10;
int32_t edi11;
uint32_t eax12;
uint32_t* esp13;
int32_t* esp14;

esp4 = reinterpret_cast<void*>(reinterpret_cast<int32_t>(__zero_stack_offset()) - 4 - 4);
ebp5 = reinterpret_cast<void*>(reinterpret_cast<int32_t>(esp4) + 16);
```

```
esp6 = reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(esp4) - a3 - 4);
*esp6 = ebx7;
esp8 = esp6 - 1;
*esp8 = esi9;
esp10 = esp8 - 1;
*esp10 = edi11;
eax12 = g404004;
esp13 = reinterpret_cast<uint32_t*>(esp10 - 1);
*esp13 = eax12 ^ reinterpret_cast<uint32_t>(ebp5);
esp14 = reinterpret_cast<int32_t*>(esp13 - 1);
*esp14 = reinterpret_cast<int32_t>(__return_address());
g0 = reinterpret_cast<void*>(reinterpret_cast<uint32_t>(ebp5) - 16);
goto *esp14;
}

void** fun_4019e2(void** ecx) {
return 0x40437c;
}

int32_t __p___argv = 0x3b56;

int32_t* fun_401ee2(void** ecx) {
goto __p___argv;
}

int32_t __p___argc = 0x3b48;

int32_t* fun_401edc(void** ecx) {
goto __p___argc;
}

signed char g40433c;

signed char fun_401867(void** ecx, int32_t a2, signed char a3) {
int1_t zf4;

zf4 = g40433c == 0;
if (zf4 || !a3) {
fun_401982(ecx);
fun_401982(ecx);
}
return 1;
}

signed char fun_4016f6(void** ecx, void** a2) {
signed char al3;
```

```
signed char al4;

if (!a2) {
g40433c = 1;
}
fun_401c9a();
al3 = fun_401982(ecx);
if (al3) {
al4 = fun_401982(ecx);
if (al4) {
return 1;
} else {
fun_401982(ecx);
}
}
return 0;
}

int32_t IsProcessorFeaturePresent = 0x3d44;

int32_t fun_401f2a(void* a1, int32_t a2) {
goto IsProcessorFeaturePresent;
}

signed char g40433d;

int32_t g404344;

int32_t g404348;

int32_t g40434c;

int32_t g404350;

int32_t g404354;

signed char fun_40172f(void** ecx, int32_t a2, int32_t a3, int32_t a4) {
int1_t zf5;
int32_t eax6;
int32_t eax7;
int32_t eax8;
signed char al9;
int1_t zf10;
int32_t eax11;
struct s7* eax12;
signed char al13;
```

```
zf5 = g40433d == 0;
if (!zf5) {
return 1;
}
if (!a2 || a2 == 1) {
eax6 = fun_401e6a();
if (!eax6 || a2) {
g404340 = -1;
g404344 = -1;
g404348 = -1;
g40434c = -1;
g404350 = -1;
g404354 = -1;
goto addr_4017a2_6;
} else {
eax7 = fun_401f0c(ecx);
if (eax7 || (eax8 = fun_401f0c(0x404340), !!eax8)) {
al9 = 0;
} else {
addr_4017a2_6:
g40433d = 1;
al9 = 1;
}
return al9;
}
}
fun_4019e8(ecx, 5);
fun_401c10(ecx, 0x403640, 8);
zf10 = image_base_ == 0x5a4d;
if (!zf10)
goto addr_401831_12;
eax11 = g40003c;
if (*reinterpret_cast<int32_t*>(eax11 + 0x400000) != 0x4550)
goto addr_401831_12;
if (*reinterpret_cast<int16_t*>(eax11 + 0x400018) != 0x10b)
goto addr_401831_12;
eax12 = fun_401680(0x400000, a2 - 0x400000);
if (!eax12)
goto addr_401831_12;
if (eax12->f36 >= 0)
goto addr_401810_17;
addr_401831_12:
al13 = 0;
addr_40183a_18:
g0 = reinterpret_cast<void*>(0x403640);
```

```
return al13;
addr_401810_17:
al13 = 1;
goto addr_40183a_18;
}

int32_t _register_onexit_function = 0x3bf8;

int32_t fun_401f12() {
goto _register_onexit_function;
}

int32_t _controlfp_s = 0x3c22;

int32_t fun_401f1e(int32_t a1, int32_t a2, int32_t a3) {
goto _controlfp_s;
}

int32_t _except_handler4_common = 0x3a50;

void fun_401e94(int32_t a1, int32_t a2, int32_t* a3, int32_t a4, int32_t a5, int32_t a6, int32_t
a7) {
goto _except_handler4_common;
}

struct s16 {
int32_t f0;
signed char[12] pad16;
int32_t f16;
int32_t f20;
};

struct s15 {
struct s16* f0;
int32_t f4;
};

int32_t fun_401b56(struct s15* a1);

void fun_401b4a() {
SetUnhandledExceptionFilter();
goto fun_401b56;
}

int32_t __current_exception = 0x3a12;
```

```
struct s16** fun_401e82() {
goto __current_exception;
}

int32_t __current_exception_context = 0x3a28;

int32_t* fun_401e88() {
goto __current_exception_context;
}

int32_t terminate = 0x3c32;

void fun_401f24() {
goto terminate;
}

int32_t _set_app_type = 0x3a8e;

void fun_401ea0(int32_t a1) {
goto _set_app_type;
}

int32_t _set_fmode = 0x3b3a;

void fun_401ed6(int32_t a1, int32_t a2) {
goto _set_fmode;
}

int32_t __p__commode = 0x3bcc;

void*** fun_401f06(int32_t a1, int32_t a2) {
goto __p__commode;
}

struct s11* fun_4019a7() {
return 0x404360;
}

struct s12* fun_4019ad() {
return 0x404368;
}

int32_t _set_new_mode = 0x3bbc;

void fun_401f00() {
goto _set_new_mode;
```

```
}

int32_t _seh_filter_exe = 0x3a7c;

void fun_401e9a() {
goto _seh_filter_exe;
}

int32_t fun_401f30(int32_t ecx, int32_t a2) {
int1_t less3;

less3 = reinterpret_cast<int32_t>(g404374) < reinterpret_cast<int32_t>(1);
if (less3 || a2 != 0xc00002b4 && a2 != 0xc00002b5) {
return a2;
}
}

void fun_401df5() {
int32_t ebp1;
int32_t eax2;
int32_t edx3;
int32_t ebp4;
int32_t edx5;
uint32_t eax6;
int32_t ebp7;
uint32_t eax8;
uint32_t eax9;
unsigned char bl10;
uint32_t ebx11;
int32_t ebp12;

*reinterpret_cast<int32_t*>(ebp1 - 20) = eax2 + edx3;
*reinterpret_cast<int32_t*>(ebp4 - 16) = edx5;
eax6 = *reinterpret_cast<uint32_t*>(ebp7 - 20);
if ((eax6 & 6) != 6)
goto 0x401e63;
eax8 = g404010;
eax9 = eax8 | 8;
g404374 = 3;
g404010 = eax9;
if (!(bl10 & 32))
goto 0x401e63;
g404374 = 5;
g404010 = eax9 | 32;
if ((ebx11 & 0xd0030000) != 0xd0030000)
goto 0x401e63;
```

```
if ((*reinterpret_cast<uint32_t*>(ebp12 - 20) & 0xe0) != 0xe0)
goto 0x401e63;
g404010 = g404010 | 64;
g404374 = 6;
}

void fun_401f50() {
uint32_t eax1;

eax1 = *reinterpret_cast<uint32_t*>(reinterpret_cast<int32_t>(__return_address()) - 4) ^ 63;
if (*reinterpret_cast<unsigned char*>(&eax1) & 0x81) {
if (eax1 & 0x204) {
if (eax1 & 0x102) {
if (eax1 & 0x408) {
if (eax1 & 0x810) {
if (eax1 & 0x1020)
goto 0x401fa3;
goto *reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(__return_address()) + 4);
} else {
goto *reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(__return_address()) + 4);
}
} else {
goto *reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(__return_address()) + 4);
}
}
} else {
goto *reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(__return_address()) + 4);
}
}
goto *reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(__return_address()) + 4);
}

struct s17 {
signed char[4] pad4;
int32_t f4;
};

struct s18 {
signed char[8] pad8;
int32_t f8;
};

void fun_40101b(struct s17*** ecx) {
void* eax2;
signed char al3;
struct s17** ecx4;
```

```
struct s18* eax5;

eax2 = g0;
g0 = reinterpret_cast<void*>(reinterpret_cast<int32_t>(__zero_stack_offset()) - 4 - 12);
al3 = reinterpret_cast<signed char>(uncaught_exception_std_YA_NXZ());
if (!al3) {
ecx4 = *ecx;
_Osfx_basic_ostream_DU_char_traits_D_std_std_QAEXXZ(ecx4);
}
if (*reinterpret_cast<struct s18***>((**ecx)->f4 + reinterpret_cast<int32_t>(*ecx) + 56)) {
eax5 = **reinterpret_cast<struct s18***>((**ecx)->f4 + reinterpret_cast<int32_t>(*ecx) + 56);
eax5->f8();
}
g0 = eax2;
return;
}

int32_t fun_40181b() {
int32_t ecx1;
int32_t ebp2;

ecx1 = 0;
*reinterpret_cast<unsigned char*>(&ecx1) =
reinterpret_cast<uint1_t>(***reinterpret_cast<int32_t***>(ebp2 - 20) == 0xc0000005);
return ecx1;
}

void fun_401be0() {
int32_t* esi1;
int32_t edi2;

esi1 = reinterpret_cast<int32_t*>(0x403554);
if (!1) {
do {
edi2 = *esi1;
if (edi2) {
image_base_(edi2);
edi2(edi2);
}
++esi1;
} while (reinterpret_cast<uint32_t>(esi1) < 0x403554);
}
return;
}

struct s19 {
```

```
signed char[4] pad4;
int32_t f4;
};

struct s20 {
signed char[8] pad8;
int32_t f8;
};

void fun_401082(struct s19*** ecx) {
void* eax2;
struct s20* eax3;

eax2 = g0;
g0 = reinterpret_cast<void*>(reinterpret_cast<int32_t>(__zero_stack_offset()) - 4 - 12);
if (*reinterpret_cast<struct s20***>((**ecx)->f4 + reinterpret_cast<int32_t>(*ecx) + 56)) {
eax3 = **reinterpret_cast<struct s20***>((**ecx)->f4 + reinterpret_cast<int32_t>(*ecx) + 56);
eax3->f8();
}
g0 = eax2;
return;
}

void fun_401280() {
setstate_basic_ios_DU_char_traits_D_std_std_QAEXH_N_Z();
goto 4;
}

void fun_40130e(void** ecx, int32_t a2) {
int32_t eax3;
void** eax4;
void*** eax5;
signed char al6;
int32_t eax7;
int32_t eax8;
void** eax9;
signed char al10;
int32_t eax11;

fun_401ea0(1);
eax3 = fun_401970(1);
fun_401ed6(eax3, 1);
eax4 = fun_401969(ecx, eax3, 1);
eax5 = fun_401f06(eax3, 1);
*eax5 = eax4;
al6 = fun_40172f(ecx, 1, eax3, 1);
```

```
if (al6 && (fun_401bb4(), fun_4018bc(fun_401be0), fun_40196c(), eax7 = fun_401eac(), ecx =
reinterpret_cast<void**>(fun_401be0), !eax7)) {
fun_401976(fun_401be0);
eax8 = fun_4019d0(fun_401be0);
if (eax8) {
fun_401ea6(fun_401be0);
ecx = reinterpret_cast<void**>(fun_401969);
}
fun_4019a6(ecx);
fun_4019a6(ecx);
fun_401985(ecx);
eax9 = fun_401969(ecx, __return_address(), a2);
fun_401efa(ecx);
ecx = eax9;
al10 = fun_401982(ecx);
if (al10) {
fun_401eb2(ecx);
}
fun_401969(ecx, __return_address(), a2);
eax11 = fun_401b02(ecx);
if (!eax11) {
return;
}
}
fun_4019e8(ecx, 7);
fun_4019b3(ecx);
goto 7;
}

void fun_4013c1(void** ecx, int32_t a2) {
fun_401b4a();
fun_401969(ecx, __return_address(), a2);
fun_401f00();
return;
}

void fun_4014fa() {
int32_t ebp1;
int32_t ebp2;

*reinterpret_cast<int32_t*>(ebp1 - 32) = ***reinterpret_cast<int32_t***>(ebp2 - 20);
fun_401e9a();
return;
}

void fun_40182e() {
```

```
}

int32_t __CxxFrameHandler3 = 0x39ea;

void fun_401fb0() {
uint32_t ecx1;
int32_t v2;

ecx1 = *reinterpret_cast<uint32_t*>(v2 - 8) ^ reinterpret_cast<uint32_t>(v2 + 12);
fun_4012fd(ecx1, __return_address());
goto __CxxFrameHandler3;
}

void fun_40129a() {
}

void fun_401c56(int32_t a1, int32_t a2, int32_t a3, int32_t a4) {
int32_t ebp5;
int32_t* esp6;
void* ebp7;

g0 = *reinterpret_cast<void**>(ebp5 - 16);
esp6 = reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(ebp7) + 4 - 4);
*esp6 = reinterpret_cast<int32_t>(__return_address());
goto *esp6;
}

struct s21 {
int32_t f0;
int32_t f4;
int32_t f8;
};

struct s22 {
int32_t f0;
void** f4;
};

void** g404334;

struct s23 {
int32_t f0;
void** f4;
};

struct s24 {
```

```
int32_t f0;
void** f4;
};

struct s25 {
int32_t f0;
void** f4;
};

struct s26 {
int32_t f0;
void** f4;
};

struct s27 {
int32_t f0;
void** f4;
};

struct s28 {
int32_t f0;
int32_t f4;
signed char f8;
};

struct s29 {
signed char[4] pad4;
int32_t f4;
};

void fun_40150e(void** ecx) {
int32_t* esp2;
int32_t ebp3;
signed char al4;
void* esp5;
int32_t* esp6;
int32_t ebp7;
int32_t* esp8;
int32_t* esp9;
int32_t* esp10;
int32_t* esp11;
struct s21* esp12;
int32_t* esp13;
struct s22* esp14;
signed char al15;
void*** esp16;
```

```
void* esp17;
signed char bl18;
int32_t ebp19;
int32_t ebp20;
int32_t* esp21;
signed char al22;
int32_t ebp23;
void** eax24;
int32_t* esp25;
struct s23* esp26;
void*** esp27;
int32_t ebp28;
int32_t* esp29;
struct s23* esp30;
int32_t* esp31;
int32_t eax32;
int32_t* esp33;
int32_t ecx34;
int32_t* esp35;
int32_t* esp36;
int32_t* esp37;
void*** esp38;
int32_t* esp39;
int32_t ebp40;
struct s23* esp41;
void*** esp42;
int32_t* esp43;
void** eax44;
void* esp45;
void*** esp46;
struct s24* esp47;
signed char al48;
void*** esp49;
void** esi50;
void*** esp51;
struct s24* esp52;
void*** esp53;
int32_t* esp54;
int32_t* esp55;
void*** esp56;
void** eax57;
void* esp58;
void*** esp59;
struct s25* esp60;
signed char al61;
void*** esp62;
```

```
void*** esp63;
struct s25* esp64;
void*** esp65;
void*** esp66;
int32_t eax67;
int32_t* esp68;
int32_t* eax69;
int32_t esi70;
int32_t* esp71;
int32_t* eax72;
int32_t* esp73;
int32_t* esp74;
int32_t* esp75;
int32_t* esp76;
int32_t eax77;
int32_t esi78;
struct s26* esp79;
signed char al80;
struct s27* esp81;
int32_t* esp82;
int32_t ebp83;
int32_t ebp84;
struct s27* esp85;
struct s27* esp86;
int32_t* esp87;
struct s28* esp88;
int32_t ebp89;
int32_t ebp90;
struct s29* ebp91;
int32_t ebp92;

esp2 = reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(*reinterpret_cast<void**>(ebp3 -
24)) - 4);
*esp2 = 0x401516;
al4 = fun_401b07(ecx);
esp5 = reinterpret_cast<void*>(esp2 + 1);
if (!al4) {
while (1) {
esp6 = reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(esp5) - 4);
*esp6 = *reinterpret_cast<int32_t*>(ebp7 - 32);
esp8 = esp6 - 1;
*esp8 = 0x401554;
fun_401ed0(ecx);
esp9 = esp8 + 1 - 1;
*esp9 = 0x40155a;
fun_40191e(ecx);
```

```
esp10 = esp9 + 1 - 1;
*esp10 = 20;
esp11 = esp10 - 1;
*esp11 = 0x403620;
esp12 = reinterpret_cast<struct s21*>(esp11 - 1);
esp12->f0 = 0x4013df;
fun_401c10(ecx, esp12->f4, esp12->f8);
esp13 = &esp12->f4 - 1;
*esp13 = 1;
esp14 = reinterpret_cast<struct s22*>(esp13 - 1);
esp14->f0 = 0x4013e6;
al15 = fun_4016f6(ecx, esp14->f4);
esp16 = &esp14->f4;
ecx = *esp16;
esp17 = reinterpret_cast<void*>(esp16 + 4);
if (!al15 || (bl18 = 0, *reinterpret_cast<signed char*>(ebp19 - 25) = 0,
*reinterpret_cast<uint32_t*>(ebp20 - 4) = 0, esp21 =
reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(esp17) - 4), *esp21 = 0x4013fd, al22 =
fun_4016c4(ecx), esp17 = reinterpret_cast<void*>(esp21 + 1), *reinterpret_cast<signed
char*>(ebp23 - 36) = al22, eax24 = g404334, ecx = reinterpret_cast<void**>(1), eax24 == 1)) {
esp25 = reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(esp17) - 4);
*esp25 = 7;
esp26 = reinterpret_cast<struct s23*>(esp25 - 1);
esp26->f0 = 0x401546;
fun_4019e8(ecx, esp26->f4);
esp27 = &esp26->f4;
} else {
if (eax24) {
bl18 = 1;
*reinterpret_cast<signed char*>(ebp28 - 25) = 1;
} else {
g404334 = reinterpret_cast<void**>(1);
esp29 = reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(esp17) - 4);
*esp29 = 0x403104;
esp30 = reinterpret_cast<struct s23*>(esp29 - 1);
esp30->f0 = 0x4030f8;
esp31 = reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(esp30) - 4);
*esp31 = 0x401429;
eax32 = fun_401ec4();
esp33 = esp31 + 1 + 1;
ecx34 = *esp33;
if (eax32)
goto addr_40142f_8;
esp35 = esp33 + 1 - 1;
*esp35 = 0x4030f4;
esp36 = esp35 - 1;
```

```
*esp36 = 0x4030ec;
esp37 = esp36 - 1;
*esp37 = 0x40144f;
fun_401ebe(ecx34);
esp38 = reinterpret_cast<void***>(esp37 + 1 + 1);
ecx = *esp38;
esp17 = reinterpret_cast<void*>(esp38 + 4);
g404334 = reinterpret_cast<void**>(2);
}
esp39 = reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(esp17) - 4);
*esp39 = *reinterpret_cast<int32_t*>(ebp40 - 36);
esp41 = reinterpret_cast<struct s23*>(esp39 - 1);
esp41->f0 = 0x40146a;
fun_40184a(ecx, esp41->f4);
esp42 = &esp41->f4;
ecx = *esp42;
esp43 = reinterpret_cast<int32_t*>(esp42 + 4 - 4);
*esp43 = 0x401470;
eax44 = fun_4019dc(ecx);
esp45 = reinterpret_cast<void*>(esp43 + 1);
if (*reinterpret_cast<void***>(eax44) && (esp46 =
reinterpret_cast<void***>(reinterpret_cast<int32_t>(esp45) - 4), *esp46 = eax44, esp47 =
reinterpret_cast<struct s24*>(esp46 - 4), esp47->f0 = 0x40147e, al48 = fun_4017b6(ecx, esp47-
>f4), esp49 = &esp47->f4, ecx = *esp49, esp45 = reinterpret_cast<void*>(esp49 + 4), !!al48)) {
esi50 = *reinterpret_cast<void***>(eax44);
esp51 = reinterpret_cast<void***>(reinterpret_cast<int32_t>(esp45) - 4);
*esp51 = reinterpret_cast<void**>(0);
esp52 = reinterpret_cast<struct s24*>(esp51 - 4);
esp52->f0 = 2;
esp53 = reinterpret_cast<void***>(reinterpret_cast<int32_t>(esp52) - 4);
*esp53 = reinterpret_cast<void**>(0);
ecx = esi50;
esp54 = reinterpret_cast<int32_t*>(esp53 - 4);
*esp54 = 0x401491;
image_base_(ecx);
esp55 = esp54 + 1 - 1;
*esp55 = 0x401493;
esi50(ecx);
esp45 = reinterpret_cast<void*>(esp55 + 1);
}
esp56 = reinterpret_cast<void***>(reinterpret_cast<int32_t>(esp45) - 4);
*esp56 = reinterpret_cast<void**>(0x401498);
eax57 = fun_4019e2(ecx);
esp58 = reinterpret_cast<void*>(esp56 + 4);
if (*reinterpret_cast<void***>(eax57) && (esp59 =
reinterpret_cast<void***>(reinterpret_cast<int32_t>(esp58) - 4), *esp59 = eax57, esp60 =
```

```
reinterpret_cast<struct s25*>(esp59 - 4), esp60->f0 = 0x4014a4, al61 = fun_4017b6(ecx, esp60->f4), esp62 = &esp60->f4, ecx = *esp62, esp58 = reinterpret_cast<void*>(esp62 + 4), !!al61)) {
esp63 = reinterpret_cast<void***>(reinterpret_cast<int32_t>(esp58) - 4);
*esp63 = *reinterpret_cast<void***>(eax57);
esp64 = reinterpret_cast<struct s25*>(esp63 - 4);
esp64->f0 = 0x4014b0;
fun_401ef4(ecx);
esp65 = &esp64->f4;
ecx = *esp65;
esp58 = reinterpret_cast<void*>(esp65 + 4);
}
esp66 = reinterpret_cast<void***>(reinterpret_cast<int32_t>(esp58) - 4);
*esp66 = reinterpret_cast<void**>(0x4014b6);
eax67 = fun_401eb8(ecx);
esp68 = reinterpret_cast<int32_t*>(esp66 + 4 - 4);
*esp68 = 0x4014bd;
eax69 = fun_401ee2(ecx);
esi70 = *eax69;
esp71 = esp68 + 1 - 1;
*esp71 = 0x4014c4;
eax72 = fun_401edc(ecx);
esp73 = esp71 + 1 - 1;
*esp73 = eax67;
esp74 = esp73 - 1;
*esp74 = esi70;
esp75 = esp74 - 1;
*esp75 = *eax72;
esp76 = esp75 - 1;
*esp76 = 0x4014cd;
eax77 = fun_401000(ecx);
esi78 = eax77;
esp79 = reinterpret_cast<struct s26*>(esp76 + 1 + 3 - 1);
esp79->f0 = 0x4014d7;
al80 = fun_401b07(ecx);
esp27 = &esp79->f4;
if (al80)
break;
}
esp81 = reinterpret_cast<struct s27*>(esp27 - 4);
esp81->f0 = esi78;
esp82 = reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(esp81) - 4);
*esp82 = 0x40154c;
fun_401eca(ecx);
esp5 = reinterpret_cast<void*>(esp82 + 1);
}
} else {
```

```
if (!*reinterpret_cast<signed char*>(ebp83 - 25)) {
*reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(esp5) - 4) = 0x401525;
fun_401eee();
}
*reinterpret_cast<int32_t*>(ebp84 - 4) = -2;
goto addr_40152f_19;
}
if (!bl18) {
esp85 = reinterpret_cast<struct s27*>(esp27 - 4);
esp85->f0 = reinterpret_cast<int32_t>("j");
fun_401ee8(ecx);
esp27 = &esp85->f4;
}
esp86 = reinterpret_cast<struct s27*>(esp27 - 4);
esp86->f0 = 0;
esp87 = reinterpret_cast<int32_t*>(reinterpret_cast<int32_t>(esp86) - 4);
*esp87 = 1;
esp88 = reinterpret_cast<struct s28*>(esp87 - 1);
esp88->f0 = 0x4014ed;
fun_401867(ecx, esp88->f4, esp88->f8);
*reinterpret_cast<int32_t*>(ebp89 - 4) = -2;
addr_40152f_19:
g0 = *reinterpret_cast<void**>(ebp90 - 16);
goto ebp91->f4;
addr_40142f_8:
*reinterpret_cast<int32_t*>(ebp92 - 4) = -2;
goto addr_40152f_19;
}

int32_t __std_terminate = 0x3a00;

void fun_401e7c() {
goto __std_terminate;
}

int32_t fun_401b56(struct s15* a1) {
struct s16* esi2;
struct s16** eax3;
int32_t esi4;
int32_t* eax5;

esi2 = a1->f0;
if (esi2->f0 != 0xe06d7363 || (esi2->f16 != 3 || esi2->f20 != 0x19930520 && (esi2->f20 !=
0x19930521 && (esi2->f20 != 0x19930522 && esi2->f20 != 0x1994000)))) {
return 0;
} else {
```

```
eax3 = fun_401e82();
*eax3 = esi2;
esi4 = a1->f4;
eax5 = fun_401e88();
*eax5 = esi4;
fun_401f24();
}
}

void fun_401fd0() {
uint32_t ecx1;
int32_t v2;

ecx1 = *reinterpret_cast<uint32_t*>(v2 - 4) ^ reinterpret_cast<uint32_t>(v2 + 12);
fun_4012fd(ecx1, __return_address());
goto 0x401e76;
}

void fun_401c6b(int32_t ecx, int32_t* a2, int32_t a3, int32_t a4, int32_t a5) {
int32_t v6;
int32_t eax7;

v6 = *a2;
eax7 = fun_401f30(ecx, v6);
*a2 = eax7;
fun_401e94(0x404004, fun_4012fd, a2, a3, a4, a5, v6);
return;
}

void fun_401ff0() {
goto 0x401090;
}

void fun_401ff8() {
goto 0x401020;
}

void fun_402000() {
uint32_t ecx1;
int32_t v2;

ecx1 = *reinterpret_cast<uint32_t*>(v2 - 40) ^ reinterpret_cast<uint32_t>(v2 + 12);
fun_4012fd(ecx1, __return_address());
goto 0x401e76;
}
```

```
void fun_40201b() {
signed char* eax1;
signed char* eax2;
signed char al3;
signed char* eax4;
signed char* eax5;
signed char al6;
signed char* eax7;
signed char* eax8;
signed char al9;
signed char* eax10;
signed char* eax11;
signed char al12;
signed char* eax13;
signed char* eax14;
signed char al15;
signed char* eax16;
signed char* eax17;
signed char al18;
signed char* eax19;
signed char* eax20;
signed char al21;
signed char* eax22;
signed char* eax23;
signed char al24;
signed char* eax25;
signed char* eax26;
signed char al27;
signed char* eax28;
signed char* eax29;
signed char al30;
signed char* eax31;
signed char* eax32;
signed char al33;
signed char* eax34;
signed char* eax35;
signed char al36;
signed char* eax37;
signed char* eax38;
signed char al39;
signed char* eax40;
signed char* eax41;
signed char al42;
signed char* eax43;
signed char* eax44;
signed char al45;
```

```
signed char* eax46;
signed char* eax47;
signed char al48;
signed char* eax49;
signed char* eax50;
signed char al51;
signed char* eax52;
signed char* eax53;
signed char al54;
signed char* eax55;
signed char* eax56;
signed char al57;
signed char* eax58;
signed char* eax59;
signed char al60;
signed char* eax61;
signed char* eax62;
signed char al63;
signed char* eax64;
signed char* eax65;
signed char al66;
signed char* eax67;
signed char* eax68;
signed char al69;
signed char* eax70;
signed char* eax71;
signed char al72;
signed char* eax73;
signed char* eax74;
signed char al75;
signed char* eax76;
signed char* eax77;
signed char al78;
signed char* eax79;
signed char* eax80;
signed char al81;
signed char* eax82;
signed char* eax83;
signed char al84;
signed char* eax85;
signed char* eax86;
signed char al87;
signed char* eax88;
signed char* eax89;
signed char al90;
signed char* eax91;
```

```
signed char* eax92;
signed char al93;
signed char* eax94;
signed char* eax95;
signed char al96;
signed char* eax97;
signed char* eax98;
signed char al99;
signed char* eax100;
signed char* eax101;
signed char al102;
signed char* eax103;
signed char* eax104;
signed char al105;
signed char* eax106;
signed char* eax107;
signed char al108;
signed char* eax109;
signed char* eax110;
signed char al111;
signed char* eax112;
signed char* eax113;
signed char al114;
signed char* eax115;
signed char* eax116;
signed char al117;
signed char* eax118;
signed char* eax119;
signed char al120;
signed char* eax121;
signed char* eax122;
signed char al123;
signed char* eax124;
signed char* eax125;
signed char al126;
signed char* eax127;
signed char* eax128;
signed char al129;
signed char* eax130;
signed char* eax131;
signed char al132;
signed char* eax133;
signed char* eax134;
signed char al135;
signed char* eax136;
signed char* eax137;
```

```
signed char al138;
signed char* eax139;
signed char* eax140;
signed char al141;
signed char* eax142;
signed char* eax143;
signed char al144;
signed char* eax145;
signed char* eax146;
signed char al147;
signed char* eax148;
signed char* eax149;
signed char al150;
signed char* eax151;
signed char* eax152;
signed char al153;
signed char* eax154;
signed char* eax155;
signed char al156;
signed char* eax157;
signed char* eax158;
signed char al159;
signed char* eax160;
signed char* eax161;
signed char al162;
signed char* eax163;
signed char* eax164;
signed char al165;
signed char* eax166;
signed char* eax167;
signed char al168;
signed char* eax169;
signed char* eax170;
signed char al171;
signed char* eax172;
signed char* eax173;
signed char al174;
signed char* eax175;
signed char* eax176;
signed char al177;
signed char* eax178;
signed char* eax179;
signed char al180;
signed char* eax181;
signed char* eax182;
signed char al183;
```

```
signed char* eax184;
signed char* eax185;
signed char al186;
signed char* eax187;
signed char* eax188;
signed char al189;
signed char* eax190;
signed char* eax191;
signed char al192;
signed char* eax193;
signed char* eax194;
signed char al195;
signed char* eax196;
signed char* eax197;
signed char al198;
signed char* eax199;
signed char* eax200;
signed char al201;
signed char* eax202;
signed char* eax203;
signed char al204;
signed char* eax205;
signed char* eax206;
signed char al207;
signed char* eax208;
signed char* eax209;
signed char al210;
signed char* eax211;
signed char* eax212;
signed char al213;
signed char* eax214;
signed char* eax215;
signed char al216;
signed char* eax217;
signed char* eax218;
signed char al219;
signed char* eax220;
signed char* eax221;
signed char al222;
signed char* eax223;
signed char* eax224;
signed char al225;
signed char* eax226;
signed char* eax227;
signed char al228;
signed char* eax229;
```

```
signed char* eax230;
signed char al231;
signed char* eax232;
signed char* eax233;
signed char al234;
signed char* eax235;
signed char* eax236;
signed char al237;
signed char* eax238;
signed char* eax239;
signed char al240;
signed char* eax241;
signed char* eax242;
signed char al243;
signed char* eax244;
signed char* eax245;
signed char al246;
signed char* eax247;
signed char* eax248;
signed char al249;
signed char* eax250;
signed char* eax251;
signed char al252;
signed char* eax253;
signed char* eax254;
signed char al255;
signed char* eax256;
signed char* eax257;
signed char al258;
signed char* eax259;
signed char* eax260;
signed char al261;
signed char* eax262;
signed char* eax263;
signed char al264;
signed char* eax265;
signed char* eax266;
signed char al267;
signed char* eax268;
signed char* eax269;
signed char al270;
signed char* eax271;
signed char* eax272;
signed char al273;
signed char* eax274;
signed char* eax275;
```

```
signed char al276;
signed char* eax277;
signed char* eax278;
signed char al279;
signed char* eax280;
signed char* eax281;
signed char al282;
signed char* eax283;
signed char* eax284;
signed char al285;
signed char* eax286;
signed char* eax287;
signed char al288;
signed char* eax289;
signed char* eax290;
signed char al291;
signed char* eax292;
signed char* eax293;
signed char al294;
signed char* eax295;
signed char* eax296;
signed char al297;
signed char* eax298;
signed char* eax299;
signed char al300;
signed char* eax301;
signed char* eax302;
signed char al303;
signed char* eax304;
signed char* eax305;
signed char al306;
signed char* eax307;
signed char* eax308;
signed char al309;
signed char* eax310;
signed char* eax311;
signed char al312;
signed char* eax313;
signed char* eax314;
signed char al315;
signed char* eax316;
signed char* eax317;
signed char al318;
signed char* eax319;
signed char* eax320;
signed char al321;
```

```
signed char* eax322;
signed char* eax323;
signed char al324;
signed char* eax325;
signed char* eax326;
signed char al327;
signed char* eax328;
signed char* eax329;
signed char al330;
signed char* eax331;
signed char* eax332;
signed char al333;
signed char* eax334;
signed char* eax335;
signed char al336;
signed char* eax337;
signed char* eax338;
signed char al339;
signed char* eax340;
signed char* eax341;
signed char al342;
signed char* eax343;
signed char* eax344;
signed char al345;
signed char* eax346;
signed char* eax347;
signed char al348;
signed char* eax349;
signed char* eax350;
signed char al351;
signed char* eax352;
signed char* eax353;
signed char al354;
signed char* eax355;
signed char* eax356;
signed char al357;
signed char* eax358;
signed char* eax359;
signed char al360;
signed char* eax361;
signed char* eax362;
signed char al363;
signed char* eax364;
signed char* eax365;
signed char al366;
signed char* eax367;
```

```
signed char* eax368;
signed char al369;
signed char* eax370;
signed char* eax371;
signed char al372;
signed char* eax373;
signed char* eax374;
signed char al375;
signed char* eax376;
signed char* eax377;
signed char al378;
signed char* eax379;
signed char* eax380;
signed char al381;
signed char* eax382;
signed char* eax383;
signed char al384;
signed char* eax385;
signed char* eax386;
signed char al387;
signed char* eax388;
signed char* eax389;
signed char al390;
signed char* eax391;
signed char* eax392;
signed char al393;
signed char* eax394;
signed char* eax395;
signed char al396;
signed char* eax397;
signed char* eax398;
signed char al399;
signed char* eax400;
signed char* eax401;
signed char al402;
signed char* eax403;
signed char* eax404;
signed char al405;
signed char* eax406;
signed char* eax407;
signed char al408;
signed char* eax409;
signed char* eax410;
signed char al411;
signed char* eax412;
signed char* eax413;
```

```
signed char al414;
signed char* eax415;
signed char* eax416;
signed char al417;
signed char* eax418;
signed char* eax419;
signed char al420;
signed char* eax421;
signed char* eax422;
signed char al423;
signed char* eax424;
signed char* eax425;
signed char al426;
signed char* eax427;
signed char* eax428;
signed char al429;
signed char* eax430;
signed char* eax431;
signed char al432;
signed char* eax433;
signed char* eax434;
signed char al435;
signed char* eax436;
signed char* eax437;
signed char al438;
signed char* eax439;
signed char* eax440;
signed char al441;
signed char* eax442;
signed char* eax443;
signed char al444;
signed char* eax445;
signed char* eax446;
signed char al447;
signed char* eax448;
signed char* eax449;
signed char al450;
signed char* eax451;
signed char* eax452;
signed char al453;
signed char* eax454;
signed char* eax455;
signed char al456;
signed char* eax457;
signed char* eax458;
signed char al459;
```

```
signed char* eax460;
signed char* eax461;
signed char al462;
signed char* eax463;
signed char* eax464;
signed char al465;
signed char* eax466;
signed char* eax467;
signed char al468;
signed char* eax469;
signed char* eax470;
signed char al471;
signed char* eax472;
signed char* eax473;
signed char al474;
signed char* eax475;
signed char* eax476;
signed char al477;
signed char* eax478;
signed char* eax479;
signed char al480;
signed char* eax481;
signed char* eax482;
signed char al483;
signed char* eax484;
signed char* eax485;
signed char al486;
signed char* eax487;
signed char* eax488;
signed char al489;
signed char* eax490;
signed char* eax491;
signed char al492;
signed char* eax493;
signed char* eax494;
signed char al495;
signed char* eax496;
signed char* eax497;
signed char al498;
signed char* eax499;
signed char* eax500;
signed char al501;
signed char* eax502;
signed char* eax503;
signed char al504;
signed char* eax505;
```

```
signed char* eax506;
signed char al507;
signed char* eax508;
signed char* eax509;
signed char al510;
signed char* eax511;
signed char* eax512;
signed char al513;
signed char* eax514;
signed char* eax515;
signed char al516;
signed char* eax517;
signed char* eax518;
signed char al519;
signed char* eax520;
signed char* eax521;
signed char al522;
signed char* eax523;
signed char* eax524;
signed char al525;
signed char* eax526;
signed char* eax527;
signed char al528;
signed char* eax529;
signed char* eax530;
signed char al531;
signed char* eax532;
signed char* eax533;
signed char al534;
signed char* eax535;
signed char* eax536;
signed char al537;
signed char* eax538;
signed char* eax539;
signed char al540;
signed char* eax541;
signed char* eax542;
signed char al543;
signed char* eax544;
signed char* eax545;
signed char al546;
signed char* eax547;
signed char* eax548;
signed char al549;
signed char* eax550;
signed char* eax551;
```

```
signed char al552;
signed char* eax553;
signed char* eax554;
signed char al555;
signed char* eax556;
signed char* eax557;
signed char al558;
signed char* eax559;
signed char* eax560;
signed char al561;
signed char* eax562;
signed char* eax563;
signed char al564;
signed char* eax565;
signed char* eax566;
signed char al567;
signed char* eax568;
signed char* eax569;
signed char al570;
signed char* eax571;
signed char* eax572;
signed char al573;
signed char* eax574;
signed char* eax575;
signed char al576;
signed char* eax577;
signed char* eax578;
signed char al579;
signed char* eax580;
signed char* eax581;
signed char al582;
signed char* eax583;
signed char* eax584;
signed char al585;
signed char* eax586;
signed char* eax587;
signed char al588;
signed char* eax589;
signed char* eax590;
signed char al591;
signed char* eax592;
signed char* eax593;
signed char al594;
signed char* eax595;
signed char* eax596;
signed char al597;
```

```
signed char* eax598;
signed char* eax599;
signed char al600;
signed char* eax601;
signed char* eax602;
signed char al603;
signed char* eax604;
signed char* eax605;
signed char al606;
signed char* eax607;
signed char* eax608;
signed char al609;
signed char* eax610;
signed char* eax611;
signed char al612;
signed char* eax613;
signed char* eax614;
signed char al615;
signed char* eax616;
signed char* eax617;
signed char al618;
signed char* eax619;
signed char* eax620;
signed char al621;
signed char* eax622;
signed char* eax623;
signed char al624;
signed char* eax625;
signed char* eax626;
signed char al627;
signed char* eax628;
signed char* eax629;
signed char al630;
signed char* eax631;
signed char* eax632;
signed char al633;
signed char* eax634;
signed char* eax635;
signed char al636;
signed char* eax637;
signed char* eax638;
signed char al639;
signed char* eax640;
signed char* eax641;
signed char al642;
signed char* eax643;
```

```
signed char* eax644;
signed char al645;
signed char* eax646;
signed char* eax647;
signed char al648;
signed char* eax649;
signed char* eax650;
signed char al651;
signed char* eax652;
signed char* eax653;
signed char al654;
signed char* eax655;
signed char* eax656;
signed char al657;
signed char* eax658;
signed char* eax659;
signed char al660;
signed char* eax661;
signed char* eax662;
signed char al663;
signed char* eax664;
signed char* eax665;
signed char al666;
signed char* eax667;
signed char* eax668;
signed char al669;
signed char* eax670;
signed char* eax671;
signed char al672;
signed char* eax673;
signed char* eax674;
signed char al675;
signed char* eax676;
signed char* eax677;
signed char al678;
signed char* eax679;
signed char* eax680;
signed char al681;
signed char* eax682;
signed char* eax683;
signed char al684;
signed char* eax685;
signed char* eax686;
signed char al687;
signed char* eax688;
signed char* eax689;
```

```
signed char al690;
signed char* eax691;
signed char* eax692;
signed char al693;
signed char* eax694;
signed char* eax695;
signed char al696;
signed char* eax697;
signed char* eax698;
signed char al699;
signed char* eax700;
signed char* eax701;
signed char al702;
signed char* eax703;
signed char* eax704;
signed char al705;
signed char* eax706;
signed char* eax707;
signed char al708;
signed char* eax709;
signed char* eax710;
signed char al711;
signed char* eax712;
signed char* eax713;
signed char al714;
signed char* eax715;
signed char* eax716;
signed char al717;
signed char* eax718;
signed char* eax719;
signed char al720;
signed char* eax721;
signed char* eax722;
signed char al723;
signed char* eax724;
signed char* eax725;
signed char al726;

*eax1 = reinterpret_cast<signed char>(*eax2 + al3);
*eax4 = reinterpret_cast<signed char>(*eax5 + al6);
*eax7 = reinterpret_cast<signed char>(*eax8 + al9);
*eax10 = reinterpret_cast<signed char>(*eax11 + al12);
*eax13 = reinterpret_cast<signed char>(*eax14 + al15);
*eax16 = reinterpret_cast<signed char>(*eax17 + al18);
*eax19 = reinterpret_cast<signed char>(*eax20 + al21);
*eax22 = reinterpret_cast<signed char>(*eax23 + al24);
```

```
*eax25 = reinterpret_cast<signed char>(*eax26 + al27);
*eax28 = reinterpret_cast<signed char>(*eax29 + al30);
*eax31 = reinterpret_cast<signed char>(*eax32 + al33);
*eax34 = reinterpret_cast<signed char>(*eax35 + al36);
*eax37 = reinterpret_cast<signed char>(*eax38 + al39);
*eax40 = reinterpret_cast<signed char>(*eax41 + al42);
*eax43 = reinterpret_cast<signed char>(*eax44 + al45);
*eax46 = reinterpret_cast<signed char>(*eax47 + al48);
*eax49 = reinterpret_cast<signed char>(*eax50 + al51);
*eax52 = reinterpret_cast<signed char>(*eax53 + al54);
*eax55 = reinterpret_cast<signed char>(*eax56 + al57);
*eax58 = reinterpret_cast<signed char>(*eax59 + al60);
*eax61 = reinterpret_cast<signed char>(*eax62 + al63);
*eax64 = reinterpret_cast<signed char>(*eax65 + al66);
*eax67 = reinterpret_cast<signed char>(*eax68 + al69);
*eax70 = reinterpret_cast<signed char>(*eax71 + al72);
*eax73 = reinterpret_cast<signed char>(*eax74 + al75);
*eax76 = reinterpret_cast<signed char>(*eax77 + al78);
*eax79 = reinterpret_cast<signed char>(*eax80 + al81);
*eax82 = reinterpret_cast<signed char>(*eax83 + al84);
*eax85 = reinterpret_cast<signed char>(*eax86 + al87);
*eax88 = reinterpret_cast<signed char>(*eax89 + al90);
*eax91 = reinterpret_cast<signed char>(*eax92 + al93);
*eax94 = reinterpret_cast<signed char>(*eax95 + al96);
*eax97 = reinterpret_cast<signed char>(*eax98 + al99);
*eax100 = reinterpret_cast<signed char>(*eax101 + al102);
*eax103 = reinterpret_cast<signed char>(*eax104 + al105);
*eax106 = reinterpret_cast<signed char>(*eax107 + al108);
*eax109 = reinterpret_cast<signed char>(*eax110 + al111);
*eax112 = reinterpret_cast<signed char>(*eax113 + al114);
*eax115 = reinterpret_cast<signed char>(*eax116 + al117);
*eax118 = reinterpret_cast<signed char>(*eax119 + al120);
*eax121 = reinterpret_cast<signed char>(*eax122 + al123);
*eax124 = reinterpret_cast<signed char>(*eax125 + al126);
*eax127 = reinterpret_cast<signed char>(*eax128 + al129);
*eax130 = reinterpret_cast<signed char>(*eax131 + al132);
*eax133 = reinterpret_cast<signed char>(*eax134 + al135);
*eax136 = reinterpret_cast<signed char>(*eax137 + al138);
*eax139 = reinterpret_cast<signed char>(*eax140 + al141);
*eax142 = reinterpret_cast<signed char>(*eax143 + al144);
*eax145 = reinterpret_cast<signed char>(*eax146 + al147);
*eax148 = reinterpret_cast<signed char>(*eax149 + al150);
*eax151 = reinterpret_cast<signed char>(*eax152 + al153);
*eax154 = reinterpret_cast<signed char>(*eax155 + al156);
*eax157 = reinterpret_cast<signed char>(*eax158 + al159);
*eax160 = reinterpret_cast<signed char>(*eax161 + al162);
```

```
*eax163 = reinterpret_cast<signed char>(*eax164 + al165);
*eax166 = reinterpret_cast<signed char>(*eax167 + al168);
*eax169 = reinterpret_cast<signed char>(*eax170 + al171);
*eax172 = reinterpret_cast<signed char>(*eax173 + al174);
*eax175 = reinterpret_cast<signed char>(*eax176 + al177);
*eax178 = reinterpret_cast<signed char>(*eax179 + al180);
*eax181 = reinterpret_cast<signed char>(*eax182 + al183);
*eax184 = reinterpret_cast<signed char>(*eax185 + al186);
*eax187 = reinterpret_cast<signed char>(*eax188 + al189);
*eax190 = reinterpret_cast<signed char>(*eax191 + al192);
*eax193 = reinterpret_cast<signed char>(*eax194 + al195);
*eax196 = reinterpret_cast<signed char>(*eax197 + al198);
*eax199 = reinterpret_cast<signed char>(*eax200 + al201);
*eax202 = reinterpret_cast<signed char>(*eax203 + al204);
*eax205 = reinterpret_cast<signed char>(*eax206 + al207);
*eax208 = reinterpret_cast<signed char>(*eax209 + al210);
*eax211 = reinterpret_cast<signed char>(*eax212 + al213);
*eax214 = reinterpret_cast<signed char>(*eax215 + al216);
*eax217 = reinterpret_cast<signed char>(*eax218 + al219);
*eax220 = reinterpret_cast<signed char>(*eax221 + al222);
*eax223 = reinterpret_cast<signed char>(*eax224 + al225);
*eax226 = reinterpret_cast<signed char>(*eax227 + al228);
*eax229 = reinterpret_cast<signed char>(*eax230 + al231);
*eax232 = reinterpret_cast<signed char>(*eax233 + al234);
*eax235 = reinterpret_cast<signed char>(*eax236 + al237);
*eax238 = reinterpret_cast<signed char>(*eax239 + al240);
*eax241 = reinterpret_cast<signed char>(*eax242 + al243);
*eax244 = reinterpret_cast<signed char>(*eax245 + al246);
*eax247 = reinterpret_cast<signed char>(*eax248 + al249);
*eax250 = reinterpret_cast<signed char>(*eax251 + al252);
*eax253 = reinterpret_cast<signed char>(*eax254 + al255);
*eax256 = reinterpret_cast<signed char>(*eax257 + al258);
*eax259 = reinterpret_cast<signed char>(*eax260 + al261);
*eax262 = reinterpret_cast<signed char>(*eax263 + al264);
*eax265 = reinterpret_cast<signed char>(*eax266 + al267);
*eax268 = reinterpret_cast<signed char>(*eax269 + al270);
*eax271 = reinterpret_cast<signed char>(*eax272 + al273);
*eax274 = reinterpret_cast<signed char>(*eax275 + al276);
*eax277 = reinterpret_cast<signed char>(*eax278 + al279);
*eax280 = reinterpret_cast<signed char>(*eax281 + al282);
*eax283 = reinterpret_cast<signed char>(*eax284 + al285);
*eax286 = reinterpret_cast<signed char>(*eax287 + al288);
*eax289 = reinterpret_cast<signed char>(*eax290 + al291);
*eax292 = reinterpret_cast<signed char>(*eax293 + al294);
*eax295 = reinterpret_cast<signed char>(*eax296 + al297);
*eax298 = reinterpret_cast<signed char>(*eax299 + al300);
```

```
*eax301 = reinterpret_cast<signed char>(*eax302 + al303);
*eax304 = reinterpret_cast<signed char>(*eax305 + al306);
*eax307 = reinterpret_cast<signed char>(*eax308 + al309);
*eax310 = reinterpret_cast<signed char>(*eax311 + al312);
*eax313 = reinterpret_cast<signed char>(*eax314 + al315);
*eax316 = reinterpret_cast<signed char>(*eax317 + al318);
*eax319 = reinterpret_cast<signed char>(*eax320 + al321);
*eax322 = reinterpret_cast<signed char>(*eax323 + al324);
*eax325 = reinterpret_cast<signed char>(*eax326 + al327);
*eax328 = reinterpret_cast<signed char>(*eax329 + al330);
*eax331 = reinterpret_cast<signed char>(*eax332 + al333);
*eax334 = reinterpret_cast<signed char>(*eax335 + al336);
*eax337 = reinterpret_cast<signed char>(*eax338 + al339);
*eax340 = reinterpret_cast<signed char>(*eax341 + al342);
*eax343 = reinterpret_cast<signed char>(*eax344 + al345);
*eax346 = reinterpret_cast<signed char>(*eax347 + al348);
*eax349 = reinterpret_cast<signed char>(*eax350 + al351);
*eax352 = reinterpret_cast<signed char>(*eax353 + al354);
*eax355 = reinterpret_cast<signed char>(*eax356 + al357);
*eax358 = reinterpret_cast<signed char>(*eax359 + al360);
*eax361 = reinterpret_cast<signed char>(*eax362 + al363);
*eax364 = reinterpret_cast<signed char>(*eax365 + al366);
*eax367 = reinterpret_cast<signed char>(*eax368 + al369);
*eax370 = reinterpret_cast<signed char>(*eax371 + al372);
*eax373 = reinterpret_cast<signed char>(*eax374 + al375);
*eax376 = reinterpret_cast<signed char>(*eax377 + al378);
*eax379 = reinterpret_cast<signed char>(*eax380 + al381);
*eax382 = reinterpret_cast<signed char>(*eax383 + al384);
*eax385 = reinterpret_cast<signed char>(*eax386 + al387);
*eax388 = reinterpret_cast<signed char>(*eax389 + al390);
*eax391 = reinterpret_cast<signed char>(*eax392 + al393);
*eax394 = reinterpret_cast<signed char>(*eax395 + al396);
*eax397 = reinterpret_cast<signed char>(*eax398 + al399);
*eax400 = reinterpret_cast<signed char>(*eax401 + al402);
*eax403 = reinterpret_cast<signed char>(*eax404 + al405);
*eax406 = reinterpret_cast<signed char>(*eax407 + al408);
*eax409 = reinterpret_cast<signed char>(*eax410 + al411);
*eax412 = reinterpret_cast<signed char>(*eax413 + al414);
*eax415 = reinterpret_cast<signed char>(*eax416 + al417);
*eax418 = reinterpret_cast<signed char>(*eax419 + al420);
*eax421 = reinterpret_cast<signed char>(*eax422 + al423);
*eax424 = reinterpret_cast<signed char>(*eax425 + al426);
*eax427 = reinterpret_cast<signed char>(*eax428 + al429);
*eax430 = reinterpret_cast<signed char>(*eax431 + al432);
*eax433 = reinterpret_cast<signed char>(*eax434 + al435);
*eax436 = reinterpret_cast<signed char>(*eax437 + al438);
```

```
*eax439 = reinterpret_cast<signed char>(*eax440 + al441);
*eax442 = reinterpret_cast<signed char>(*eax443 + al444);
*eax445 = reinterpret_cast<signed char>(*eax446 + al447);
*eax448 = reinterpret_cast<signed char>(*eax449 + al450);
*eax451 = reinterpret_cast<signed char>(*eax452 + al453);
*eax454 = reinterpret_cast<signed char>(*eax455 + al456);
*eax457 = reinterpret_cast<signed char>(*eax458 + al459);
*eax460 = reinterpret_cast<signed char>(*eax461 + al462);
*eax463 = reinterpret_cast<signed char>(*eax464 + al465);
*eax466 = reinterpret_cast<signed char>(*eax467 + al468);
*eax469 = reinterpret_cast<signed char>(*eax470 + al471);
*eax472 = reinterpret_cast<signed char>(*eax473 + al474);
*eax475 = reinterpret_cast<signed char>(*eax476 + al477);
*eax478 = reinterpret_cast<signed char>(*eax479 + al480);
*eax481 = reinterpret_cast<signed char>(*eax482 + al483);
*eax484 = reinterpret_cast<signed char>(*eax485 + al486);
*eax487 = reinterpret_cast<signed char>(*eax488 + al489);
*eax490 = reinterpret_cast<signed char>(*eax491 + al492);
*eax493 = reinterpret_cast<signed char>(*eax494 + al495);
*eax496 = reinterpret_cast<signed char>(*eax497 + al498);
*eax499 = reinterpret_cast<signed char>(*eax500 + al501);
*eax502 = reinterpret_cast<signed char>(*eax503 + al504);
*eax505 = reinterpret_cast<signed char>(*eax506 + al507);
*eax508 = reinterpret_cast<signed char>(*eax509 + al510);
*eax511 = reinterpret_cast<signed char>(*eax512 + al513);
*eax514 = reinterpret_cast<signed char>(*eax515 + al516);
*eax517 = reinterpret_cast<signed char>(*eax518 + al519);
*eax520 = reinterpret_cast<signed char>(*eax521 + al522);
*eax523 = reinterpret_cast<signed char>(*eax524 + al525);
*eax526 = reinterpret_cast<signed char>(*eax527 + al528);
*eax529 = reinterpret_cast<signed char>(*eax530 + al531);
*eax532 = reinterpret_cast<signed char>(*eax533 + al534);
*eax535 = reinterpret_cast<signed char>(*eax536 + al537);
*eax538 = reinterpret_cast<signed char>(*eax539 + al540);
*eax541 = reinterpret_cast<signed char>(*eax542 + al543);
*eax544 = reinterpret_cast<signed char>(*eax545 + al546);
*eax547 = reinterpret_cast<signed char>(*eax548 + al549);
*eax550 = reinterpret_cast<signed char>(*eax551 + al552);
*eax553 = reinterpret_cast<signed char>(*eax554 + al555);
*eax556 = reinterpret_cast<signed char>(*eax557 + al558);
*eax559 = reinterpret_cast<signed char>(*eax560 + al561);
*eax562 = reinterpret_cast<signed char>(*eax563 + al564);
*eax565 = reinterpret_cast<signed char>(*eax566 + al567);
*eax568 = reinterpret_cast<signed char>(*eax569 + al570);
*eax571 = reinterpret_cast<signed char>(*eax572 + al573);
*eax574 = reinterpret_cast<signed char>(*eax575 + al576);
```

```
*eax577 = reinterpret_cast<signed char>(*eax578 + al579);
*eax580 = reinterpret_cast<signed char>(*eax581 + al582);
*eax583 = reinterpret_cast<signed char>(*eax584 + al585);
*eax586 = reinterpret_cast<signed char>(*eax587 + al588);
*eax589 = reinterpret_cast<signed char>(*eax590 + al591);
*eax592 = reinterpret_cast<signed char>(*eax593 + al594);
*eax595 = reinterpret_cast<signed char>(*eax596 + al597);
*eax598 = reinterpret_cast<signed char>(*eax599 + al600);
*eax601 = reinterpret_cast<signed char>(*eax602 + al603);
*eax604 = reinterpret_cast<signed char>(*eax605 + al606);
*eax607 = reinterpret_cast<signed char>(*eax608 + al609);
*eax610 = reinterpret_cast<signed char>(*eax611 + al612);
*eax613 = reinterpret_cast<signed char>(*eax614 + al615);
*eax616 = reinterpret_cast<signed char>(*eax617 + al618);
*eax619 = reinterpret_cast<signed char>(*eax620 + al621);
*eax622 = reinterpret_cast<signed char>(*eax623 + al624);
*eax625 = reinterpret_cast<signed char>(*eax626 + al627);
*eax628 = reinterpret_cast<signed char>(*eax629 + al630);
*eax631 = reinterpret_cast<signed char>(*eax632 + al633);
*eax634 = reinterpret_cast<signed char>(*eax635 + al636);
*eax637 = reinterpret_cast<signed char>(*eax638 + al639);
*eax640 = reinterpret_cast<signed char>(*eax641 + al642);
*eax643 = reinterpret_cast<signed char>(*eax644 + al645);
*eax646 = reinterpret_cast<signed char>(*eax647 + al648);
*eax649 = reinterpret_cast<signed char>(*eax650 + al651);
*eax652 = reinterpret_cast<signed char>(*eax653 + al654);
*eax655 = reinterpret_cast<signed char>(*eax656 + al657);
*eax658 = reinterpret_cast<signed char>(*eax659 + al660);
*eax661 = reinterpret_cast<signed char>(*eax662 + al663);
*eax664 = reinterpret_cast<signed char>(*eax665 + al666);
*eax667 = reinterpret_cast<signed char>(*eax668 + al669);
*eax670 = reinterpret_cast<signed char>(*eax671 + al672);
*eax673 = reinterpret_cast<signed char>(*eax674 + al675);
*eax676 = reinterpret_cast<signed char>(*eax677 + al678);
*eax679 = reinterpret_cast<signed char>(*eax680 + al681);
*eax682 = reinterpret_cast<signed char>(*eax683 + al684);
*eax685 = reinterpret_cast<signed char>(*eax686 + al687);
*eax688 = reinterpret_cast<signed char>(*eax689 + al690);
*eax691 = reinterpret_cast<signed char>(*eax692 + al693);
*eax694 = reinterpret_cast<signed char>(*eax695 + al696);
*eax697 = reinterpret_cast<signed char>(*eax698 + al699);
*eax700 = reinterpret_cast<signed char>(*eax701 + al702);
*eax703 = reinterpret_cast<signed char>(*eax704 + al705);
*eax706 = reinterpret_cast<signed char>(*eax707 + al708);
*eax709 = reinterpret_cast<signed char>(*eax710 + al711);
*eax712 = reinterpret_cast<signed char>(*eax713 + al714);
```

```
*eax715 = reinterpret_cast<signed char>(*eax716 + al717);
*eax718 = reinterpret_cast<signed char>(*eax719 + al720);
*eax721 = reinterpret_cast<signed char>(*eax722 + al723);
*eax724 = reinterpret_cast<signed char>(*eax725 + al726);
}
```