Аннотация – В документе представлен анализ уязвимостей, выявленных в Ivanti Secure Access VPN (Pulse Secure VPN) с их потенциальным воздействием на использующие ПО организации. В анализе рассматриваются различные аспекты этих уязвимостей, включая методы их использования, потенциальные последствия и проблемы, с которыми сталкиваются в процессе эксплуатации.

Документ предоставляет ценную информацию специалистам по кибербезопасности, ИТ-администраторам и другим заинтересованным сторонам в различных отраслях. Понимая технические нюансы, методы эксплуатации и стратегии смягчения последствий, становится возможно повысить уровень безопасности своей организации от подобных угроз.

Этот анализ особенно полезен специалистам по безопасности, стремящимся разобраться в тонкостях уязвимостей VPN и их последствиях для безопасности предприятия. Он также служит ресурсом для ИТ-администраторов, ответственных за поддержание безопасных конфигураций VPN, и для заинтересованных сторон отрасли, заинтересованных в более широком влиянии таких уязвимостей на цифровую безопасность и соответствие требованиям.

I. Введение

Northwave Cybersecurity выявила несколько уязвимостей в Ivanti Secure Access VPN (Pulse Secure VPN). Было обнаружено, что эти уязвимости, в частности CVE-2023–38043, CVE-2023–35080 и CVE-2023–38543, затрагивают программное обеспечение VPN, используемое более чем 40 000 организациями по всему миру. Основная обсуждаемая уязвимость позволяет повысить привилегии из-за драйвера ядра, установленного ПО VPN, который создаёт устройство, доступное для чтения и записи любому пользователю. Это потенциально может привести к повреждению ядра или повышению привилегий.

II. Уязвимости

Уязвимости CVE-2023-38043, CVE-2023-35080, CVE-2023-38543 обнаружены во всех версиях клиента Ivanti Secure Access Client ниже 22.6R1.1 и может позволить злоумышленнику, прошедшему локальную аутентификацию, использовать уязвимую конфигурацию, что потенциально может привести к отказу в обслуживании (DoS) или раскрытие информации. Успешная эксплуатация уязвимости может позволить злоумышленнику получить повышенные привилегии в уязвимой системе. Серьёзность этой уязвимости оценивается как высокая: базовый балл CVSS 3.x составляет 7,8 от NIST и 8,8 от HackerOne, что указывает на значительное влияние на конфиденциальность, целостность и доступность.

A. Схема атаки

- **Первоначальный доступ:** злоумышленник должен сначала получить возможность выполнять код в целевой системе, что достигается различными способами, такими как фишинг, использование другой уязвимости или получение легитимного доступа к учётной записи пользователя в системе.
- **Эксплуатация:** как только злоумышленник получит возможность выполнить код в целевой системе, он использует уязвимую конфигурацию клиента безопасного доступа Ivanti путём отправки специально созданного запроса компоненту клиента Ivanti Secure Access Client.
- **Отказ в обслуживании.** Успешная эксплуатация уязвимости может привести к DoS-состоянию, при котором затронутая машина перестанет отвечать на запросы или выйдет из строя.
- **Компрометация системы.** В некоторых сценариях уязвимость может быть использована для получения повышенных привилегий или выполнения произвольного кода, что приводит к полной компрометации системы.

B. Затронутые отрасли

CVE затрагивают различные отрасли, которые используют VPN клиент для безопасного удалённого доступа к своим сетям.

- **Здравоохранение.** Больницы и поставщики медицинских услуг используют VPN-клиенты, для безопасного удалённого доступа к записям пациентов и внутренним системам, что делает их потенциальными целями.
- **Финансовые услуги.** Банки, страховые компании и другие финансовые учреждения полагаются на безопасный VPN-доступ для удалённых сотрудников и защиту конфиденциальных финансовых данных.
- **Государственный сектор:** Государственные учреждения используют VPN-клиенты для обеспечения безопасной связи и удалённого

доступа к конфиденциальным правительственным ресурсам.

- **Образование.** Университеты и учебные заведения используют VPN-клиенты для безопасного доступа к академическим ресурсам, а также для обеспечения удалённого обучения и администрирования.
- **Технологии и ИТ-услуги.** Компании технологического сектора, включая поставщиков ИТ-услуг, используют VPN-клиенты для безопасного удалённого доступа к сетевым ресурсам и клиентским средам.
- **Производство и критическая инфраструктура.** Производственные компании и поставщики критической инфраструктуры используют VPN-клиенты для безопасного подключения к системам промышленного управления и сетям операционных технологий.
- **Розничная торговля и потребительские товары.** Розничные торговцы используют VPN-клиенты для безопасного удалённого доступа к управлению запасами, системам торговых точек и другим критически важным бизнес-приложениям.

1) Здравоохранение

В текущей отрасли последствия использования такой уязвимости могут включать:

- **Нарушение предоставления медицинских услуг.** Атака типа «отказ в обслуживании» может нарушить доступ к критически важным системам здравоохранения и данным пациентов, что повлияет на уход за пациентами и потенциально приведёт к задержкам в лечении или диагностике.
- **Компрометация конфиденциальных данных.** Повышенные привилегии могут позволить злоумышленникам получать доступ, изменять или удалять конфиденциальные данные пациентов, нарушая конфиденциальность пациентов и потенциально приводя к краже личных данных или мошенничеству.
- **Нарушения нормативных требований и нормативных требований.** На медицинские организации распространяются строгие нормативные требования по защите данных пациентов, поэтому последствия уязвимости могут привести к штрафам со стороны регулирующих органов и юридическим последствиям.
- **Ущерб репутации.** Инцидент безопасности может нанести ущерб репутации затронутой организации здравоохранения, что приведёт к потере доверия среди пациентов и партнёров.
- **Финансовые затраты:** Реагирование на нарушение безопасности и восстановление после него может быть дорогостоящим, включая расходы, связанные с расследованием, исправлением ситуации, судебными издержками и потенциальными выплатами или штрафами.

2) Индустрия финансовых услуг

В текущей отрасли последствия использования такой уязвимости могут включать:

- **Нарушение финансовых операций.** Атака типа «отказ в обслуживании» может нарушить доступ к критически важным финансовым системам, повлиять на транзакции, торговлю и другие срочные операции, что потенциально может привести к финансовым потерям.
- **Кража конфиденциальных финансовых данных.** Повышенные привилегии могут позволить злоумышленникам получить доступ, изменить или украсть конфиденциальные финансовые данные, включая учётные записи клиентов, истории транзакций и собственные торговые алгоритмы, что приведёт к финансовому мошенничеству и конкурентному ущербу.
- **Нарушения нормативных требований и требований:** к финансовым учреждениям предъявляются строгие нормативные требования в отношении защиты данных и кибербезопасности. Нарушение безопасности, вызванное этой уязвимостью, может привести к штрафам со стороны регулирующих органов, санкциям и усилению контроля.
- **Репутационный ущерб:** инциденты безопасности могут серьёзно подорвать репутацию финансовых учреждений, подрывая доверие клиентов и потенциально приводя к потере бизнеса, поскольку клиенты перемещают свои активы в кажущиеся более безопасными учреждения.
- **Финансовые затраты.** Затраты, связанные с реагированием на нарушение безопасности и восстановлением после него, могут быть значительными, включая криминалистические расследования, исправление системы, судебные издержки и потенциальную компенсацию пострадавшим клиентам.

3) Государственный сектор

В текущей отрасли последствия использования такой уязвимости могут включать:

- **Нарушение предоставления основных услуг:** госучреждения предоставляют населению основные услуги, включая службы экстренной помощи, социальные услуги и управление инфраструктурой. DoS-атака может нарушить работу этих критически важных служб, что повлияет на общественную безопасность и благосостояние.
- **Раскрытие конфиденциальной информации.** Правительственные учреждения обрабатывают конфиденциальную информацию, включая личные данные граждан, секретную информацию национальной безопасности и данные критической

инфраструктуры. Полная компрометация системы может привести к раскрытию такой информации с серьёзными последствиями для национальной безопасности и конфиденциальности личности.

- **Потеря общественного доверия.** Любое нарушение или сбой в работе государственных служб из-за инцидента в области кибербезопасности может привести к значительной потере общественного доверия к государственным учреждениям. Восстановление этого доверия может оказаться долгим и трудным процессом.
- **Нормативно-правовые последствия:** Государственные учреждения подчиняются строгим нормативным и правовым нормам в отношении защиты данных и кибербезопасности. Нарушение, вызванное этой уязвимостью, может привести к судебным разбирательствам, расследованиям и наложению штрафов.
- **Финансовые последствия:** Реагирование на инциденты кибербезопасности и восстановление после них может оказаться дорогостоящим. Сюда входят затраты, связанные с криминалистическими расследованиями, восстановлением системы, потенциальными юридическими обязательствами и мерами по предотвращению будущих инцидентов.

4) Образовательная индустрия

В текущей отрасли последствия использования такой уязвимости могут включать:

- **Нарушение образовательных услуг.** Атака может нарушить доступ к системам управления обучением, виртуальным классам и другим онлайн-образовательным ресурсам.
- **Раскрытие конфиденциальных данных.** Если уязвимость приведёт к компрометации системы, конфиденциальные данные, такие как записи студентов, данные исследований и личная информация преподавателей и студентов, могут стать публично доступными.
- **Вопросы регулирования и соответствия:** образовательные учреждения часто подчиняются правилам, касающимся защиты данных учащихся. Нарушение безопасности может привести к несоблюдению этих правил, что приведёт к юридическим и финансовым последствиям.
- **Репутационный ущерб:** Инцидент безопасности может нанести ущерб репутации учебного заведения, что потенциально может повлиять на набор студентов и партнёрские отношения с другими организациями.
- **Финансовые затраты.** Затраты, связанные с реагированием на нарушение безопасности, включая расследования, исправление системы и потенциальную юридическую ответственность, могут быть значительными для образовательных учреждений.

5) Отрасль технологий и ИТ-услуг

В текущей отрасли последствия использования такой уязвимости могут включать:

- **Нарушение ИТ- и технологических услуг.** Атака может нарушить доступ к критически важной ИТ-инфраструктуре и услугам, затрагивая как поставщиков услуг, так и их клиентов. Это может привести к простоям, снижению производительности и нарушению соглашений об уровне обслуживания (SLA).
- **Компрометация конфиденциальных данных.** Уязвимость потенциально может привести к полной компрометации системы, обеспечивая несанкционированный доступ к конфиденциальным данным, таким как интеллектуальная собственность, исходный код, данные клиентов и внутренние коммуникации. Это может иметь серьёзные последствия для конфиденциальности и целостности данных.
- **Регуляторные риски и риски, связанные с соблюдением требований.** Многие компании, занимающиеся технологиями и ИТ-услугами, подчиняются нормативным требованиям, касающимся защиты данных и кибербезопасности. Атака может привести к несоблюдению требований, что приведёт к штрафам, судебным искам и усилению контроля со стороны регулирующих органов.
- **Репутационный ущерб.** Репутация компаний, предоставляющих технологические и ИТ-услуги, во многом зависит от их способности защитить свои собственные данные и данные своих клиентов. Инцидент безопасности может подорвать доверие, что потенциально может привести к потере клиентов и трудностям в приобретении нового бизнеса.
- **Финансовые затраты.** Финансовые последствия реагирования на нарушения безопасности и восстановления после них могут быть существенными. Затраты могут включать криминалистические расследования, восстановление системы, судебные издержки и компенсации пострадавшим сторонам.

6) Отрасль производства и критической инфраструктуры

В текущей отрасли последствия использования такой уязвимости могут включать:

- **Нарушение операционной деятельности.** DoS-атака может нарушить доступ к критически важным системам и сетям, затрагивая производственные линии, управление цепочками поставок и среду операционных технологий.
- **Компрометация конфиденциальных данных.** Повышенные привилегии могут позволить злоумышленникам получить доступ, изменить или

украсть конфиденциальные данные, включая запатентованные производственные процессы, данные систем управления инфраструктурой и информацию о сотрудниках.

- **Риски безопасности.** В критически важных секторах инфраструктуры, таких как энергетика, водоснабжение и транспорт, нарушение системы может создать прямые риски для безопасности населения и окружающей среды.
- **Нарушения нормативных требований и требований.** Многие производственные организации и организации критической инфраструктуры подчиняются нормативным требованиям в отношении кибербезопасности. Нарушение безопасности может привести к несоблюдению требований, что приведёт к штрафам и судебным искам.
- **Репутационный ущерб.** Инцидент безопасности в этих отраслях может привести к потере доверия со стороны клиентов, партнёров и регулирующих органов, что потенциально повлияет на будущие возможности бизнеса.
- **Финансовые затраты.** Финансовые последствия нарушения безопасности могут быть значительными, включая затраты на реагирование на инциденты, восстановление системы и потенциальную юридическую ответственность.

7) Розничная торговля и производство потребительских товаров

В текущей отрасли последствия использования такой уязвимости могут включать:

- **Нарушение операций розничной торговли.** Атака типа «отказ в обслуживании» может нарушить доступ к критически важным системам розничной торговли, что повлияет на продажи, управление запасами и обслуживание клиентов, потере дохода.
- **Компрометация конфиденциальных данных.** Если уязвимость приводит к компрометации системы, конфиденциальные данные и платёжная информация клиентов, конфиденциальные бизнес-данные и информация о сотрудниках, могут стать доступными.
- **Вопросы регулирования и соответствия.** Розничные торговцы часто подчиняются правилам, касающимся защиты данных потребителей. Нарушение безопасности может привести к несоблюдению этих правил, что приведёт к юридическим и финансовым последствиям.
- **Репутационный ущерб.** Инцидент безопасности может нанести ущерб репутации ритейлера, что потенциально повлияет на лояльность клиентов и ценность бренда.
- **Финансовые затраты.** Затраты, связанные с реагированием на нарушение безопасности, включая расследования, исправление системы и потенциальную юридическую ответственность, могут быть значительными для организаций розничной торговли.

III. Детали

IOCTL 0x80002018 связан с уязвимой функцией в callback’е IRP_MJ_DEVICE_CONTROL драйвера ядра. Эта функция предназначена для обработки кодов управления вводом-выводом (IOCTL), которые отправляются из приложений пользовательского режима драйверу. Код, обрабатывающий этот IOCTL, содержит уязвимость повышения привилегий из-за следующей последовательности операций:

- Загружается указатель на входные данные, переданные из пользовательского режима (системного буфера).
- Первое значение внутри этого ввода принимается как указатель на структуру, специфичную для драйвера.
- Внутри этой структуры загружается указатель по смещению +28h.
- Указатель на смещение +50h внутри памяти, на которое указывает предыдущий указатель, передаётся API ядра IoCsqRemoveIrp.
- Кроме того, второй аргумент, предоставляемый вызову IoCsqRemoveIrp, который находится в регистре RDX, также находится под контролем пользователя.

Функция IoCsqRemoveIrp — это API ядра, который удаляет IRP (пакет запроса ввода-вывода) из очереди с помощью указателей функций (callback), содержащихся в первом аргументе, переданном API. Уязвимость возникает потому, что пользователь контролирует этот первый аргумент, что означает, что он может манипулировать указателями функций, используемыми IoCsqRemoveIrp, для выполнения произвольного кода с привилегиями ядра.

Сама функция IoCsqRemoveIrp относительно проста и использует процедуры диспетчеризации очереди для удаления, указанного IRP из очереди. Однако критическая проблема безопасности здесь заключается в том, что пользователь может управлять регистрами RCX и RDX, которые используются в качестве аргументов функции. Внутри функции есть несколько мест, где указатель загружается из первого аргумента (RCX) и затем передаётся в _guard_dispatch_icall. Эта внутренняя функция предназначена для вызова любого указателя функции в регистре RAX, но у нее есть существенное ограничение: указатель в RAX должен находиться в начале допустимой функции, которая является частью образа ядра. Это означает, что функции шеллкода или не-изображения ядра не могут быть вызваны напрямую.

Таким образом, уязвимость в коде обработки IOCTL позволяет злоумышленнику контролировать указатели

функций, используемые IoCsqRemoveIrp, что потенциально может привести к выполнению произвольного кода с привилегиями ядра. Это серьёзный недостаток безопасности, который можно использовать для повышения привилегий, позволяя злоумышленнику с локальным доступом к системе получить полный контроль над ней.

Ограничения, описанные в сценарии с уязвимой обработкой IOCTL в драйвере ядра, иллюстрируют сложность и проблемы разработки надёжного эксплойта для уязвимости ядра. Давайте разберём эти ограничения и их последствия для разработки эксплойтов:

A. Ограничение 1: гарантированный синий экран

Автоматическое освобождение предоставленного пользователем указателя через ExFreePoolWithTag в конце процедуры обработки IOCTL представляет собой серьёзную проблему. Для этой операции требуется действительный указатель ядра. Даже если злоумышленнику удастся предоставить действительный указатель, его освобождение может привести к нестабильности или повреждению ядра, что, вероятно, приведёт к сбою системы (синий экран). Это ограничение значительно усложняет разработку стабильного эксплойта, поскольку требует, чтобы эксплойт либо избегал запуска этого освобождения, либо гарантировал, что освобождение не приведёт к неблагоприятному воздействию на стабильность системы.

B. Ограничение 2: Сильно ограниченный контроль аргументов

Ограниченный контроль над аргументами, передаваемыми функциям, вызываемым IoCsqRemoveIrp через _guard_dispatch_icall, создаёт ещё одну проблему. Эксплойт контролирует регистр RCX (указывающий на область памяти с указателями функций) и, в одном случае, регистр RDX (указывающий на контролируемую область памяти). Однако для других вызовов RDX указывает на область стека, находящуюся вне контроля злоумышленника, а регистр R8, который потенциально может содержать дополнительные данные, не используется в контексте этих вызовов функций. Это ограничение серьёзно влияет на возможности эксплойта манипулировать потоком выполнения вызываемых функций, что затрудняет выполнение произвольного кода без сбоя системы.

C. Ограничение 3: защищенные вызовы

Использование _guard_dispatch_icall в качестве защитной меры со стороны Microsoft ещё больше усложняет разработку эксплойтов. Этот механизм гарантирует, что могут быть вызваны только указатели на легитимные функции в образе ntoskrnl.exe, эффективно предотвращая выполнение произвольного шеллкода или функций вне образа ядра. Найти в ядре последовательность из трех функций, которую можно вызвать с ограниченным доступным управлением аргументами, не вызывая при этом сбоя, является серьёзной проблемой. Это ограничение требует глубокого понимания внутреннего устройства ядра и доступных функций, чтобы определить жизнеспособную цепочку, которая может привести к успешной эксплуатации.

D. Обход синего экрана

Чтобы решить проблему обхода синего экрана после использования уязвимости, предполагается использование последнего вызова функции перед сбоем системы. Идея состоит в том, чтобы предотвратить продолжение выполнения после последнего вызова функции, не вызывая при этом сбоя системы. Предлагаемое решение включает использование функций синхронизации и блокировки, в частности, нацеленных на функцию синхронизации ядра, которая может блокировать весь поток на неопределённый срок, предотвращая тем самым его доступ к вызову ExFreePoolWithTag, который приводит к появлению синего экрана.

Для этой цели выбрана функция KxWaitForSpinLockAndAcquire. Эта функция принимает указатель в регистре RCX и проверяет, не равно ли значение в начале памяти, на которую он указывает, нулю. Если это так, функция входит в цикл, многократно проверяя значение, пока оно не станет нулевым. Установив для первых 8 байт памяти, на которые указывает RCX, ненулевое значение, поток можно заблокировать в бесконечном цикле, эффективно предотвращая появление синего экрана без сбоя системы.

Однако блокировка потока ядра в бесконечном цикле может существенно повлиять на производительность системы, вызывая замедление работы компьютера после многократного выполнения эксплойта. Чтобы решить проблему, эксплойт может установить минимально возможный приоритет потока через API SetThreadPriority() с параметром THREAD_PRIORITY_LOWEST. Это гарантирует, что заблокированный поток получит наименьшее количество процессорного времени, сводя к минимуму его влияние на производительность системы.

Таким образом, стратегия обхода синего экрана включает в себя:

- Использование функции KxWaitForSpinLockAndAcquire для блокировки потока в бесконечном цикле, не позволяя ему достичь вызова ExFreePoolWithTag.
- Установка минимально возможного приоритета заблокированного потока, чтобы минимизировать его влияние на производительность системы.

E. Уязвимый код

Чтобы добраться до уязвимого кода и правильно настроить входной буфер IOCTL для вызова IoCsqRemoveIrp, в предоставленном фрагменте кода выполняются следующие шаги:

- HANDLE устройства получается путём вызова CreateFile с DEVICE_NAME.
- Входной буфер выделяется и инициализируется нулем с помощью calloc.
- Первые 8 байт входного буфера указывают на начальный_буфер.

- Затем в Initial_buffer устанавливаются указатели со смещениями 0x28 и 0x30, указывающие на buff_28h и buff_30h соответственно.
- Функция DeviceIoControl вызывается с кодом VULN_IOCTL и подготовленным входным буфером.

Фрагмент кода предназначен для выполнения проверок, выполняемых драйвером входного буфера перед вызовом IoCsqRemoveIrp. В частности, это гарантирует, что:

- Первое значение во входном буфере — это ненулевой указатель на другой буфер (initial_buffer).
- Initial_buffer содержит указатели, отличные от NULL, по смещениям +0x28 и +0x30.
- Эти указатели используются для передачи указателя на смещение +0x50 в буфере, на которое указывает buff_28h в качестве первого аргумента IoCsqRemoveIrp.
- Указатель, загруженный со смещения +0x28 (buff_28h), передаётся в качестве второго аргумента функции.

Настраивая таким образом входной буфер и вызывая DeviceIoControl, код достигает уязвимой области кода драйвера, где вызывается IoCsqRemoveIrp, что подтверждается попаданием точки останова в отладчике.

Функция IoCsqRemoveIrp — это API ядра, который удаляет IRP (пакет запроса ввода-вывода) из очереди с помощью указателей функций (callback), содержащихся в первом аргументе, переданном API. Уязвимость в коде обработки IOCTL позволяет злоумышленнику контролировать указатели функций, используемые IoCsqRemoveIrp, что потенциально может привести к выполнению произвольного кода с привилегиями ядра.

F. Управление IoCsqRemoveIrp

Чтобы управлять функцией IoCsqRemoveIrp и подготовить входные данные для выполнения всех внутренних проверок, выполняются следующие шаги:

- Входной буфер настроен на доступ к вызову IoCsqRemoveIrp, гарантируя, что первые 8 байтов входного буфера интерпретируются как указатель на другой буфер и что этот указатель не равен NULL.
- Буфер, на который указывают первые 8 байтов входного буфера, затем устанавливается с помощью указателей со смещениями +0x28 и +0x30, указывающих на buff_28h и buff_30h соответственно.
- Буфер buff_28h подготовлен с указателями функций для трех вызовов функций, которые выполнит IoCsqRemoveIrp. Эти указатели размещаются по соответствующим смещениям внутри buff_28h:
    - Первый указатель вызова функции размещается по смещению +0x20.
    - Второй указатель вызова функции размещается по смещению +0x10.
    - Третий указатель вызова функции размещается по смещению +0x28.
- Выделяется отдельный буфер iocsq_rsi_plus_8h, а ненулевое значение помещается по смещению +0x68, чтобы обеспечить проверку внутри IoCsqRemoveIrp.
- Буфер buff_30h настроен так, чтобы указывать на iocsq_rsi_plus_8h по смещению +0x08, а ненулевое значение также помещается по смещению +0x68 в пределах buff_30h.
- Чтобы предотвратить появление синего экрана после использования уязвимости, для третьего вызова функции установлено значение KxWaitForSpinLockAndAcquire, которое заблокирует поток на неопределённый срок и не позволит ему достичь вызова ExFreePoolWithTag, который мог бы вызвать синий экран.
- Первые два вызова функции настроены на HalMakeBeep, безвредную функцию ядра, которая не дает сбоя и не принимает аргументов.
- Буферу buff_28h по смещению +0x50 присваивается ненулевое значение, чтобы предоставить заблокированный объект спин-блокировки KxWaitForSpinLockAndAcquire.

Настраивая таким образом входной буфер и вызывая DeviceIoControl с кодом VULN_IOCTL, эксплойт может достичь уязвимой области кода драйвера, где вызывается IoCsqRemoveIrp, и контролировать указатели функций, используемые IoCsqRemoveIrp, что потенциально может привести к выполнению произвольного кода с привилегии ядра

G. Повышение привилегий

Чтобы повысить привилегии и получить полный контроль над системой, злоумышленник может использовать соответствующие уязвимости. Одним из распространённых методов является манипулирование токенами доступа, которые представляют собой объекты, описывающие контекст безопасности процесса или потока, включая личность и привилегии учётной записи пользователя, связанной с процессом. Получив токен с более высокими привилегиями, злоумышленник может создать новый процесс с повышенными правами или заменить токен существующего процесса. Условие записи «что-где» — это уязвимость, которая позволяет злоумышленнику записать произвольное значение в произвольное место в памяти. Это можно использовать для перезаписи критических структур данных или указателей функций, что приводит к выполнению произвольного кода.

В контексте уязвимостей Ivanti Secure Access VPN, CVE-2023–38043, CVE-2023–35080 и CVE-2023–38543, процесс эксплуатации включает остановку VPN-клиента во избежание повреждения памяти, а затем использование уязвимостей для повышения привилегий. Уязвимости

позволяют повысить привилегии из-за драйвера ядра, установленного программным обеспечением VPN, который создаёт устройство, доступное для чтения и записи любому пользователю, что потенциально может привести к повреждению ядра или повышению привилегий.

Процесс эксплуатации может включать поиск указателя ядра для объекта токена с использованием класса SystemExtendedHandleInformation в API NtQuerySystemInformation, а затем использование примитива записи для перезаписи полей TOKEN->_SEP_TOKEN_PRIVILEGES->Enabled и TOKEN->_SEP_TOKEN_PRIVILEGES->Present для предоставления системного уровень привилегий для процесса. За этим может последовать создание оболочки с повышенными привилегиями.

H. Включение уязвимого драйвера

Чтобы включить уязвимый драйвер в Ivanti Secure Access VPN, который обычно отключён по умолчанию, злоумышленник может воспроизвести поведение, которое автоматически запускает драйвер, когда пользователь подключается к VPN-серверу с включённым аварийным переключением TDI. Это можно сделать, установив мошеннический VPN-сервер Ivanti Secure Access и настроив его на использование аварийного переключения TDI:

- **Загрузить VM образ:** необходимо скачать образ виртуальной машины сервера Ivanti Secure Access VPN с официального сайта.
- **Установить сервер:** установить загруженный образ виртуальной машины на виртуальный частный сервер (VPS) или локально и указать на него доменное имя, например vpn.rogue-server.com.
- **Завершить настройку виртуальной машины:** после загрузки VM-образа и завершить настройку и получить доступ к порталу администрирования.
- **Настроить действительный сертификат.** Необходимо получить действительный сертификат для домена «мошеннического» сервера (например, vpn.rogue-server.com), используя службу Let's Encrypt и загрузить файлы fullchain.pem и privkey.pem на портал администрирования в разделе «Система» -> «Конфигурация» -> «Сертификаты» -> «Сертификат устройства» с удалением предварительно настроенных самоподписанных сертификатов.
- **Ограничить VPN и настроить TDI-Failover:** на портале администрирования в разделе «Пользователи» -> «Роли пользователей» -> «Пользователи» снять флажки со всех функций доступа, кроме подпункта «Диспетчер безопасных приложений и версия для Windows/Mac». Затем включите «Включить аварийное переключение на TDI для подключения Pulse SAM» на вкладке SAM -> «Параметры».
- **Создать пользователя VPN:** «Аутентификация» -> «Аутентификация». Серверы -> Локальная система -> вкладка Пользователи и создать нового пользователя со статическим именем пользователя и паролем. Этот пользователь будет использоваться для подключения к мошенническому VPN.
- **Подключение к мошенническому серверу.** Необходимо что жертва выполнила подключение к мошенническому серверу, указав URL-адрес, имя пользователя и пароль созданного пользователя, а также область, в которой находится этот пользователь (по умолчанию — «Пользователи»). Для подключения:

  "%programfiles(x86)%\Common Files\Pulse Secure\Integration\pulselauncher.exe" -url YOUR_DOMAIN -u YOUR_USER -p YOUR_PASS -r Users

  Например:

  "%programfiles(x86)%\Common Files\Pulse Secure\Integration\pulselauncher.exe" -url vpn.rogue-server.com -u steve -p Welcome01! -r Users
- Остановить VPN-клиент. Прежде чем запускать эксплойт повышения привилегий, необходимо остановить VPN-клиент, чтобы предотвратить повреждение памяти, с помощью команды:

  "%programfiles(x86)%\Common Files\Pulse Secure\Integration\pulselauncher.exe" -stop

Выполнив эти шаги, злоумышленник может включить уязвимый драйвер и потенциально использовать уязвимости CVE-2023–38043, CVE-2023–35080 и CVE-2023–38543 в Ivanti Secure Access VPN для повышения привилегий.

IV. PoC «main.c»

Код предназначен для использования уязвимости в VPN-клиенте, позволяющей повысить привилегии, отказ в обслуживании или раскрытие информации.

A. Как работает код

- **Настройка приоритета потока.** Код начинается с попытки установить приоритет текущего потока в фоновый режим, чтобы минимизировать его влияние на производительность системы.
- **Распределение и настройка памяти:** он выделяет память для различных буферов (input_buffer, Initial_buffer, buff_30h, iocsq_rsi_plus_8h) и настраивает их для создания вредоносной полезной нагрузки. Сюда входит настройка указателя (buff_28h) для хранения значения байта, предназначенного для записи в уязвимый компонент в пространстве памяти драйвера.
- **Получение базового адреса ядра:** код извлекает базовый адрес ядра (ntoskrnl_base) для вычисления адресов конкретных функций или смещений внутри ядра, которыми эксплойт намеревается манипулировать.

- **Настройка указателей функций:** он устанавливает указатели функций в подготовленных буферах, чтобы они указывали на вредоносные или контролируемые сегменты кода или вызывали уязвимость в драйвере клиента Ivanti Secure Access Client.
- **Запуск уязвимости.** Эксплойт запускает уязвимость, выполняя вызов DeviceIoControl с подготовленным input_buffer, который содержит вредоносную полезную нагрузку, предназначенную для использования уязвимости.
- **Повышение привилегий:** в случае успеха эксплойт изменяет привилегии токена текущего процесса или выполняет другие несанкционированные действия, что приводит к повышению привилегий, DoS или раскрытию информации.

B. Входные данные:

- **Путь к целевому устройству:** путь к уязвимому устройству или драйверу, на который нацелен эксплойт.
- **Значение байта (что):** конкретное значение байта, которое эксплойт намеревается записать в целевую ячейку памяти.
- **Целевой адрес памяти (где):** адрес памяти внутри уязвимого компонента или драйвера, куда эксплойт намеревается записать значение байта.

C. Выходные данные/результат

- **Сообщения о состоянии эксплойта:** код выдает сообщения о состоянии, указывающие на успех или неудачу различных шагов, таких как установка приоритета потока, создание потоков и выполнение эксплойта.
- **Привилегированный доступ:** если эксплойт успешен, он получает повышенные привилегии для текущего процесса, позволяя ему выполнять действия, которые ранее были ограничены.
- **Потенциальная модификация системы:** в зависимости от цели эксплойта он может изменить настройки системы, отключить меры безопасности или выполнить другие несанкционированные действия в результате повышения привилегий.

V. PoC «kernel.c»

Код является частью эксплойта, нацеленноого на уязвимость в системном драйвере, написан на C и включает в себя несколько функций, которые взаимодействуют с операционной системой Windows на низком уровне для управления дескрипторами устройства и памятью.

A. Как работает код

- **BuildDevicePath:** создаёт строку пути к устройству для уязвимого драйвера.
- **OpenDevice:** открывает дескриптор устройства с помощью функции CreateFileW, которая позволяет выполнять чтение и запись на устройство.
- **CloseDevice:** закрывает дескриптор устройства и освобождает связанную память.
- **GetFunctionOffset:** извлекает смещение функции в файле ntoskrnl.exe, который является ядром Windows NT.
- **GetKernelBase:** определяет базовый адрес ядра путём запроса системной информации.
- **GetObjectPointedByHandle:** извлекает объект ядра, на который указывает данный дескриптор, который можно использовать для манипулирования или чтения информации из этого объекта.

B. Входные данные

- DevicePath: строка, представляющая путь к уязвимому устройству или драйверу.
- DEVICE_NAME_W: имя устройства, которое используется для построения пути к устройству.
- hDevice: указатель на дескриптор, который будет использоваться для взаимодействия с устройством.
- fnName: имя функции, смещение которой извлекается.

h: Дескриптор, чей указанный объект извлекается.

C. Выходные данные/результат

- **DevicePath:** полная строка пути к устройству, которая создаётся и используется для открытия дескриптора устройства.
- **hDevice:** ручка, получаемая при открытии устройства, которую можно использовать для дальнейшего взаимодействия с устройством.
- **FnOffset:** смещение указанной функции в исполняемом образе ядра.
- **KernelBase:** базовый адрес ядра, полученный из системной информации.
- **Объект:** объект ядра, на который указывает указанный дескриптор, которым можно манипулировать или читать.

Код предназначен для выполнения низкоуровневых операций, которые обычно являются частью цепочки эксплойтов. Эти операции включают в себя открытие дескриптора уязвимого драйвера, определение местоположения определённых функций или данных в ядре и потенциальное использование этой информации для манипулирования системой таким образом, чтобы использовать уязвимость.