Светлана А. Голуб[1], Игорь Ю. Коркин[2]
Национальный исследовательский ядерный университет «МИФИ»,
Каширское ш., 31, Москва, 115409, Россия
[1]e-mail: glb.svtln@gmail.com, https://orcid.org/0000-0002-2395-0661
[2]e-mail: igor.korkin@gmail.com, https://orcid.org/0000-0001-7640-2792

АНАЛИЗ БЕЗОПАСНОСТИ ПОДСИСТЕМ ЛОКАЛЬНОЙ АУТЕНТИФИКАЦИИ ОС СЕМЕЙСТВ WINDOWS И LINUX

DOI: http://dx.doi.org/10.26583/bit.2022.1.06

Аннотация. Работа посвящена одному из ключевых вопросов безопасности современных операционных систем Windows и Linux – анализу защищённости парольной информации пользователей. Для операционной системы Windows проводится анализ процессов локальной аутентификации и аутентификации с использованием контроллера домена. Для демонстрации атак на подсистему аутентификации рассмотрено программное средство Mimikatz (Франция), позволяющее извлекать парольную информацию из памяти процесса LSASS. Представлен анализ штатных средств ОС Windows для защиты памяти процессов: Security Reference Monitor, Protected Process Light и Virtualization-Based Security. Нарушитель с помощью драйвера ядра может получить доступ к парольной информации пользователя в обход штатных средств защиты. Для Linux-подобных систем представлен результат аналогичного анализа безопасности подсистем локальной аутентификации. Показано, что в модуле GNOME памяти процесса gnome-keyring-daemon можно обнаружить пароли пользователей в открытом виде, которые нарушитель может извлечь, используя привилегии прикладной программы пользовательского уровня. Данная проблема остаётся актуальной для многих современных ОС Linux на базе дистрибутива компании RedHat, таких как CentOS, Ubuntu, GNU/Linux Rolling. Для устранения описанной проблемы исследователями были разработаны программные средства для поиска и удаления паролей из памяти: MimiPenguin (США) и Mimipy (США). Сравнительный анализ этих средств показал их недостатки: средства не могут осуществлять поиск и удаление паролей, состоящих из символов Юникод (Unicode) кодировки, а также имеют медленную скорость работы. Предлагаемое в работе программное средство защиты MimiDove расширяет возможности имеющихся средств и позволяет находить и удалять из памяти пароли, содержащие символы из наборов ASCII и Unicode, затрачивая значительно меньше времени.

Ключевые слова: извлечение данных пользователя, пароли в памяти, ASCII и Unicode символы, безопасность операционных систем, gnome-keyring-daemon, LSASS, Mimikatz.
Для цитирования: ГОЛУБ, Светлана А.; КОРКИН, Игорь Ю. АНАЛИЗ БЕЗОПАСНОСТИ ПОДСИСТЕМ ЛОКАЛЬНОЙ АУТЕНТИФИКАЦИИ ОС СЕМЕЙСТВ WINDOWS И LINUX. Безопасность информационных технологий, [S.l.], т. 29, № 1, с. 57–69, 2022. ISSN 2074-7136. URL: https://bit.mephi.ru/index.php/bit/article/view/1402. DOI: http://dx.doi.org/10.26583/bit.2022.1.06.

Svetlana A. Golub[1], Igor Y. Korkin[2]
National Research Nuclear University MEPhI (Moscow Engineering Physics Institute),
Kashirskoe shosse, 31, Moscow, 115409, Russia
[1]e-mail: glb.svtln@gmail.com, https://orcid.org/0000-0002-2395-0661
[2]e-mail: igor.korkin@gmail.com, https://orcid.org/0000-0001-7640-2792

An Analysis of Local Security Authority Subsystem Services for Windows and Linux

DOI: http://dx.doi.org/10.26583/bit.2022.1.06

Abstract. The paper is devoted to the security analysis of authority subsystem services for Windows and Linux operating systems. The paper provides security analysis for both local and network-based authentication in Windows. The Mimikatz (France) will be presented to demonstrate attacks on the authentication subsystem. Mimikatz is a software tool that can extract users' credentials and password information from the memory of the LSASS process. To prevent such attacks on process memory

Windows OS includes several security mechanisms: Security Reference Monitor, Protected Process Light, and Virtualization-Based Security. However, attackers can bypass these mechanisms to get illegal access to the process memory and steal users' credentials. A similar analysis of the local authority subsystem for Linux OSes shows that gnome-keyring-daemon stores the users' passwords in plain text. As a result, attackers can easily extract this sensitive information using memory forensics techniques via user-mode applications. Several modern Linux Distributions based on Red Hat Enterprise Linux (RHEL) still have this security issue: CentOS, Ubuntu, GNU/ Linux Rolling. Experts have developed software tools to locate and remove passwords from the memory to tackle this security challenge: MimiPenguin (USA) and Mimipy (USA). Comparison analysis of these tools reveals their drawbacks: these security tools cannot locate passwords with Unicode characters, and these tools have low speed. The proposed security solution called MimiDove is designed to solve both these issues. MimiDove expands features of MimiPenguin and Mimipy by locating and deleting passwords with ASCII and Unicode characters. MimiDove is faster than MimiPenguin and Mimipy.

Keywords: extraction credentials, passwords in memory, ASCII and Unicode passwords, operating system security, gnome-keyring-daemon, LSASS, Mimikatz.

For citation: GOLUB, Svetlana A.; KORKIN, Igor Y. An Analysis of Local Security Authority Subsystem Services for Windows and Linux. IT Security (Russia), [S.l.], v. 29, n. 1, p. 57–69, 2022. ISSN 2074-7136. URL: https://bit.mephi.ru/index.php/bit/article/view/1402. DOI: http://dx.doi.org/10.26583/bit.2022.1.06.

Введение

Операционные системы семейств Windows и Linux являются доминирующими на рынках персональных и серверных компьютерных систем. Согласно опубликованной статистике компании Майкрософт, операционная система Windows 10 обеспечивает работу более 1 миллиарда персональных устройство в 200 странах мира. Linux занимает менее 2% объёма рынка персональных компьютеров, однако более 96% всех веб-серверов работают под управлением Linux.

Обеспечение безопасности парольной информации пользователей является первостепенной задачей в любой информационной инфраструктуре. В настоящее время существует множество атак, позволяющих получить несанкционированный доступ к парольным данным пользователей. Проведение успешных атак на подсистемы локальной аутентификации позволяет нарушителям получить несанкционированный доступ к учётным данным пользователя. Программные средства для эксплуатации уязвимостей в подсистеме аутентификации использовались такими известными хакерскими группировками, как APT28, APT39, Carbanak, Axiom и многими другими [1]. Компании, которые подверглись их атакам, понесли большие материальные и репутационные потери.

В работе проводится анализ безопасности подсистем аутентификации в ОС Windows и Linux. Windows занимает большую часть рынка операционных систем для персональных компьютеров и атаки на эту ОС наиболее распространены. В то же время Linux-подобные системы сейчас являются основой корпоративной инфраструктуры. Серверы крупных компаний работают под управлением ОС Linux. Для исследования была выбрана операционная система Community Enterprise Operating System (CentOS), основанная на ОС Red Hat Enterprise Linux, и используемая при развертывании критически важных приложений на мировых биржах, в финансовых учреждениях и в ведущих телекоммуникационных компаниях.

Исследователями было обнаружено, что пароли пользователей хранятся в открытом виде длительное время в памяти сервисов ОС Linux, предназначенных для безопасного хранения информации. Эта уязвимость была подтверждена и зарегистрирована как CVE-2018-20781 [2]. Нарушители могут извлечь содержимое паролей пользователей путём чтения памяти соответствующего процесса. Некоторые

процессы в дистрибутивах ОС Linux последних версий до сих пор остаются уязвимыми к атакам, позволяющим извлекать учётные данные из памяти.

В работе проводится анализ атак на подсистемы локальной аутентификации в ОС Windows и ОС Linux и предлагаются способы им противодействия. Описывается разработанное программное средство для удаления парольной информации, содержащей как ASCII, так и Unicode символы из памяти ОС Linux, имеющее конкурентные преимущества.

1. Анализ безопасности подсистемы локальной аутентификации ОС Windows: современные атаки и защита от них

Все широко используемые современные операционные системы являются многопользовательскими. Для того, чтобы получить доступ к учётной записи, пользователю необходимо пройти процесс аутентификации, который отвечает за подтверждение подлинности пользователя для дальнейшей авторизации в системе. Данный процесс входит в подсистему локальной аутентификации операционной системы.

В ОС Windows учётные данные локальных пользователей хранятся в базе данных Security Account Manager (SAM) в виде хеш-кода NTLM [3]. Для его вычисления при локальной аутентификации используется хеш-функция MD4. Хеш-коды вычисляются единожды за сессию без добавления «соли» и впоследствии не меняются. Иерархическая база данных SAM расположена в ключе реестра HKEY_LOCAL_MACHINE\SAM\SAM и по умолчанию недоступна ни обычным пользователям, ни администраторам. В ветке реестра каждого пользователя можно найти информацию об учётной записи, домашней директории, попытках входа, хеш-код LM (если хеширование LM не отключено) и хеш-код NTLM.

За реализацию локальной политики безопасности отвечает сервис проверки подлинности локальной системы безопасности (Local Security Authority Subsystem Service, LSASS) [4, с. 155]. После того как пользователь вводит свои учетные данные, вычисляется хеш-код NTLM. Для его проверки служба WinLogon загружает графическую динамическую библиотеку GINA для интерактивной идентификации и аутентификации, которая передаёт их в функцию LsaLogonUser [5]. Сервис проверки подлинности LSASS использует пакет аутентификации MSV1_0 для обработки введённых данных. Пакет MSV1_0 обращается к базе учётных данных SAM, чтобы проверить подлинность пользователя, а затем возвращает результат попытки входа в систему сервису проверки подлинности LSASS. Процесс LSASS.exe сохраняет в памяти хеш-коды NTLM паролей пользователей с активными сеансами для реализации возможности единого входа (Single Sign-On, SSO), который позволяет пользователю получать доступ к различным службам, не проходя повторную аутентификацию. Процесс локальной аутентификации представлен на рис. 1.

В доменных сетях может использоваться другой механизм аутентификации. Доменная сеть позволяет централизованно управлять компьютерами, подключёнными к одной сети. В домене Windows вся информация об учетных записях пользователей, компьютерах, подключенных устройствах и политиках безопасности записана в центральной базе данных, расположенной на одном или нескольких главных компьютерах, которые называются контроллерами домена. Эта информация об учетных записях пользователей может храниться в локальной базе учётных данных SAM, либо удалённо, с использованием службы Active Directory и контроллера домена [6].

Аутентификация в домене Windows осуществляется с помощью контроллеров домена.

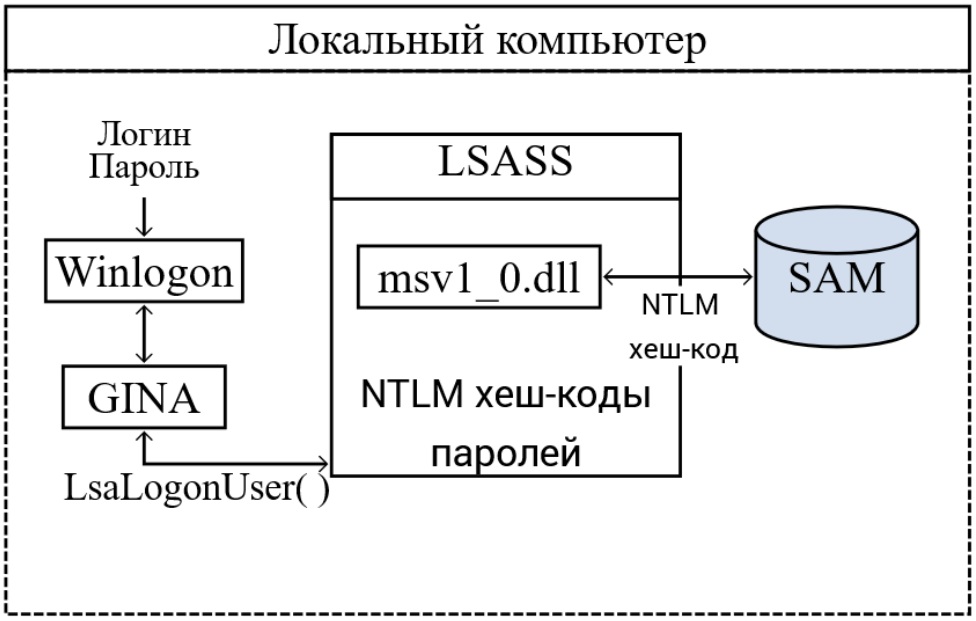

Рис. 1. Процесс локальной аутентификации
Fig. 1. The scheme of the local authentication

Аутентификация в домене также возможна с помощью протокола NTLM. Для этого используется пакет MSV, он делится на две части: одна часть MSV1_0 осуществляет преобразование открытого пароля пользователя в хеш-код пароля LM и/или хеш-код пароля NTLM, затем хеш-код передаётся в службу NetLogon. На следующем этапе служба NetLogon вызывает вторую часть пакета MSV, расположенную на контроллере домена, и которая, в свою очередь, осуществляет проверку хеш-кода пароля с помощью базы учётных записей контроллера домена SAM. Результат проверки через первую часть пакета MSV на локальном компьютере передаётся сервису проверки подлинности LSASS, который принимает решение, давать ли пользователю доступ к системе. Процесс аутентификации пользователя в домене представлен на рис. 2.

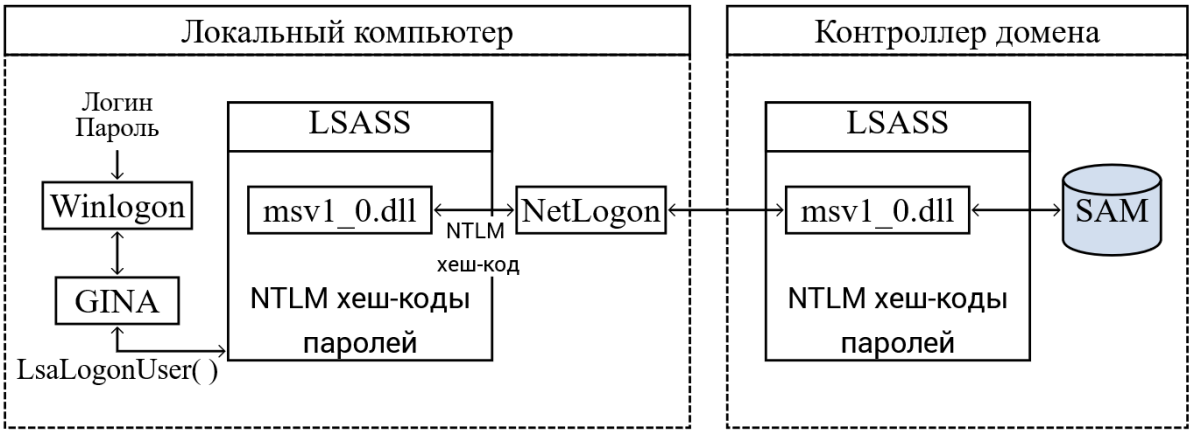

Рис. 2. Процесс аутентификации в домене
Fig. 2. The scheme of domain-based authentication

1.1. Анализ программного средства Mimikatz для извлечения парольной информации из памяти процессов Windows

Французский исследователь Бенджамин Делпи разработал утилиту Mimikatz для ОС Windows, которая с помощью драйвера позволяет извлекать следующую парольную информацию: пароли пользователя в открытом виде, хеш-коды NTLM, билеты Kerberos, сертификаты SSL и ключи шифрования [7]. Сама по себе программа не является вредоносной, но может быть использована нарушителями для осуществления следующих атак: «pass-the-ticket» [6] и «pass-the-hash» [8].

1.1.1. Атака «pass-the-hash»

В случае, если в операционной системе для локальной аутентификации используется протокол LM или NTLM, пароли пользователей передаются по каналу в виде хеш-кодов. Для восстановления пароля из хеш-кода нарушителю требуется проведения перебора значений хеш-функции, что требует значительных временных и вычислительных ресурсов. Соответственно на этапе ответа в схеме аутентификации «запрос-ответ» предоставляются хеш-коды паролей без использования «соли». Таким образом, нарушитель может использовать для несанкционированного доступа только значение хеш-кода пароля без необходимости получения пароля пользователя.

Программа Mimikatz может извлечь хеш-коды NTLM с помощью дампа памяти процесса LSASS.exe. Для этого Mimikatz получает привилегии отладки, благодаря которым может получить доступ к процессам, запускаемым от системных учётных записей. По умолчанию в локальной политике безопасности привилегия отладки выдаётся группе BUILTIN\Administrators. Это значит, что при выставленном идентификаторе безопасности (SID) этой группы можно получить данную привилегию. Таким образом, Mimikatz получает доступ к процессу LSASS.exe, следовательно, и к хеш-кодам NTLM. Имея хеш-код NTLM, можно напрямую пройти аутентификацию в системе, что показано на рис. 3.

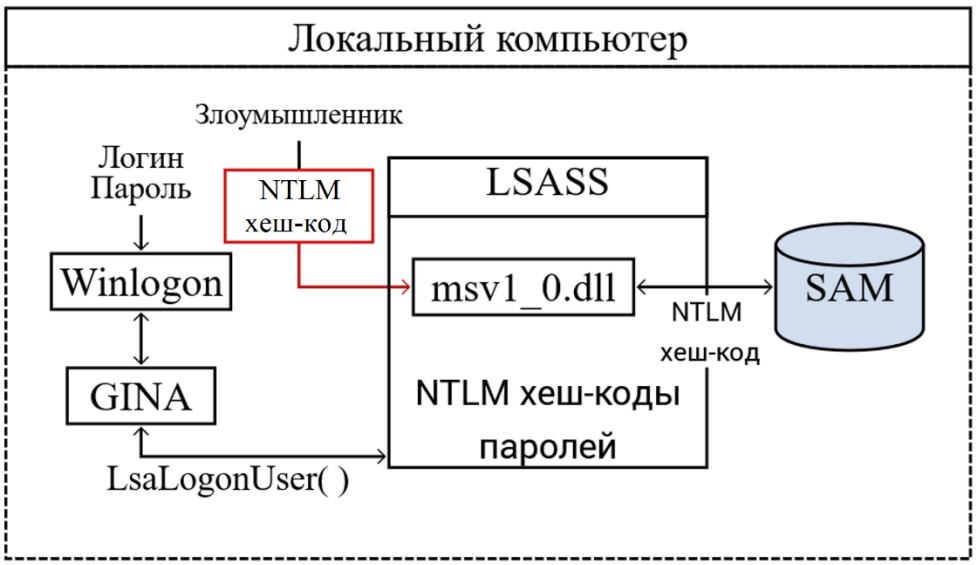

Рис. 3. Атака «pass-the-hash»
Fig. 3. "Pass-The-Hash" Attack

1.1.2. Атака «pass-the-ticket»

В модели клиент-сервер взаимная аутентификация осуществляется с помощью протокола Kerberos, который предполагает наличие третьей независимой стороны.

Посредником между клиентом и сервером выступает доверенный центр аутентификации – центр распределения ключей (Key Distribution Center, KDC), который хранит информацию об учётных записях всех клиентов сети, в частности, долговременные ключи, которые создаются на основе пароля пользователя. Центр распределения ключей Kerberos интегрирован с другими службами безопасности Windows Server, работающими на контроллере домена. KDC использует службу Active Directory в качестве базы данных учетных записей пользователей.

Введённый пользователем пароль преобразуется в хеш-код NTLM и, с использованием библиотеки Kerberos, передаётся в контроллер домена для проверки подлинности пользователя. Процесс аутентификации по протоколу Kerberos представлен на рис. 4.

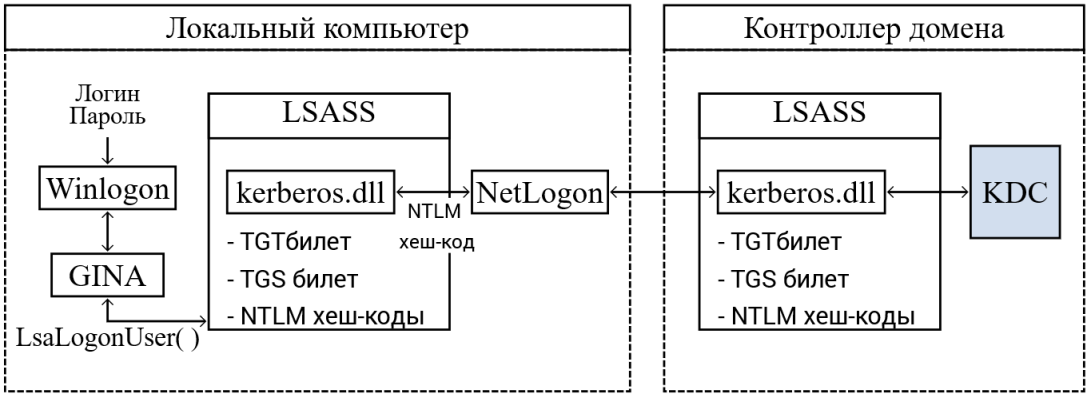

Рис. 4. Схема процесса аутентификации с использованием протокола Kerberos
Fig. 4. The Kerberos authentication protocol scheme

Для получения доступа к какой-либо службе клиенту необходимо осуществить первичную аутентификацию. На первом этапе клиент должен предоставить хеш-код NTLM для пароля, своё имя и зашифрованную временную метку. Если они подлинные, KDC выдаёт пользователю зашифрованный билет – Ticket Granting Ticket (TGT) с ограниченным временем жизни и сеансовый ключ. TGT билет включает в себя имя пользователя, запрашивающего билет, сгенерированный сессионный ключ пользователя, время жизни билета и сертификат атрибута привилегий (Privilege Attribute Certificate, PAC), который определяет права пользователя в системе. Ключом для шифрования является хеш-код NTLM для пароля, а алгоритм шифрования, может быть, одним из следующих: RC4, AES128, AES256. В дальнейшем, предъявляя TGT билет, идентификатор сервиса и зашифрованные сеансовым ключом временную метку и имя, клиент может получить от KDC зашифрованный Ticket Granting Service (TGS) для доступа к конкретному сервису. TGS состоит из сеансового ключа сервиса, имени пользователя, запрашивающего доступ, времени жизни билета и список привилегий PAC пользователя относительно запрашиваемого сервиса. Ключом шифрования является хеш-код NTLM владельца сервиса. Такой же билет доверенный центр выдаёт и сервису, посредством чего достигается взаимная аутентификация. Оба билета имеют ограниченный срок жизни. Все билеты сохраняются в памяти процесса LSASS для осуществления дальнейшего доступа к сервисам. Упрощённый процесс получения билета TGS представлен на рис. 5.

Благодаря программе Mimikatz можно несколькими способами получить доступ к сервису, не зная паролей пользователя. Такая атака называется «pass-the-ticket». Для получения TGT пользователь отправляет свой ключ, который является хеш-кодом пароля. Нарушитель может получить хеш-код NTLM и, используя его, пройти дальнейшую

аутентификацию. Нарушитель может подменить все TGT для администратора домена. После чего KDC предоставляет нарушителю TGS, и он получает доступ к сервису, как показано на рис. 6.

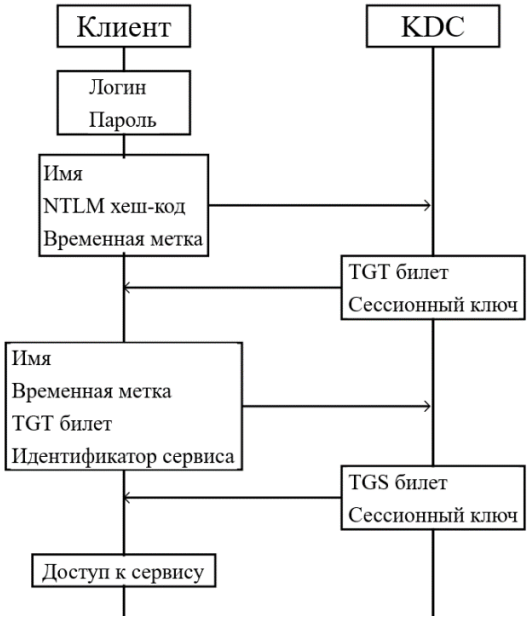

Рис. 5. Процесс получения билета TGS
Fig. 5. The scheme of gathering TGS tickets

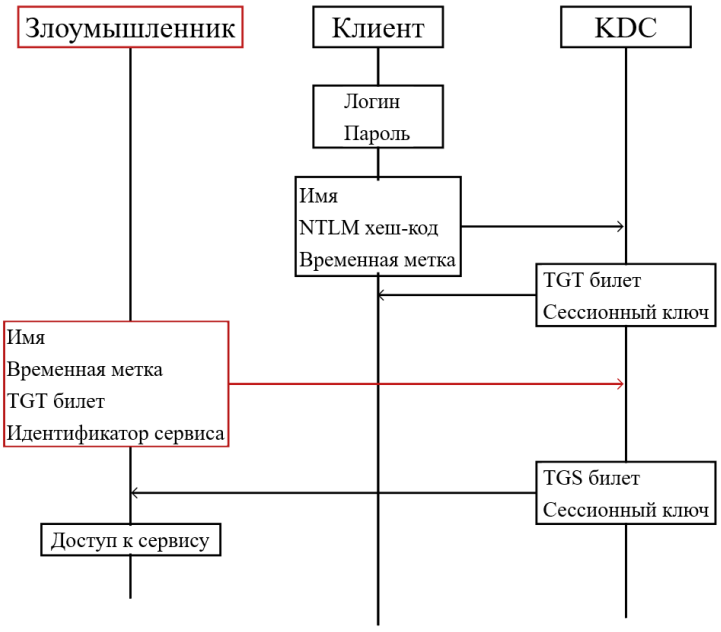

Рис. 6. Атака «pass-the-ticket» через TGT
Fig. 6. The scheme of «pass-the-ticket» attack via TGT

По такому же принципу нарушитель может получить и TGS. Этот вариант аутентификации представлен на рис. 7.

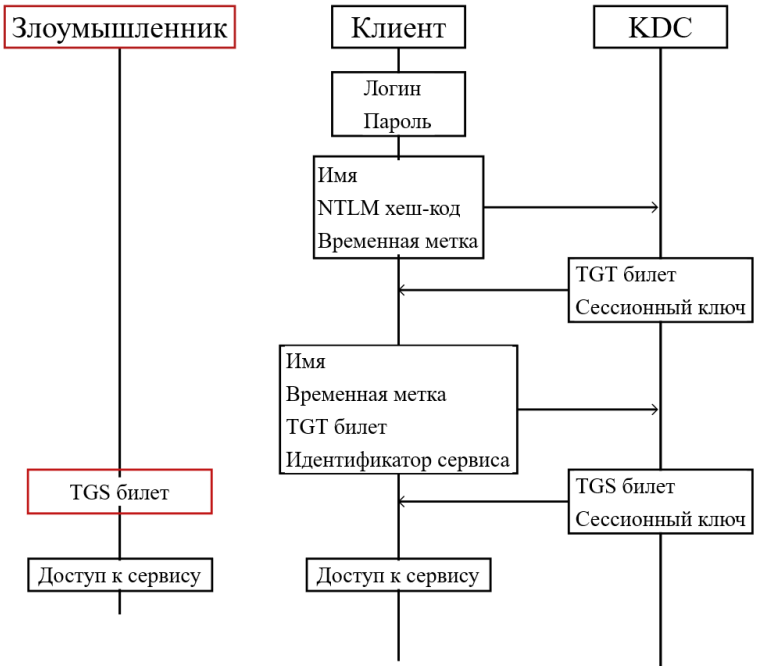

Рис. 7. Атака «pass-the-ticket» через TGS
Fig. 7. The scheme of «pass-the-ticket» attack via TGS

1.2. Анализ встроенных механизмов защиты памяти процессов на примере LSASS

Для защиты парольной информации пользователя, хранящейся в памяти процесса LSASS, от несанкционированного доступа, разработчики Windows реализовали ряд средств защиты: SRM, PPL, VBS. Проведем анализ этих средств и рассмотрим примеры атак, направленных на противодействие этим средствам защиты.

1.2.1. Монитор безопасности (Security Reference Monitor, SRM)

Штатный механизм безопасности ОС Windows, называемый монитором безопасности (Security Reference Monitor, SRM), обеспечивает контроль доступа к памяти программ на основе данных токена и дескриптора безопасности. Данный механизм позволяет предотвратить доступ к памяти программ, работающих с более высокими привилегиями, со стороны менее привилегированных программ. Однако, если вредоносная программа имеет привилегии отладки SeDebugPrivilege, то монитор безопасности всегда предоставляет такой программе полный доступ к памяти других программ без каких-либо дополнительных проверок безопасности, что может быть использовано нарушителем.

1.2.2. Защищённые процессы (Protected Process Light, PPL)

Для предотвращения такого несанкционированного доступа и для защиты мультимедийного контента от копирования (Digital Rights Management), памяти процесса LSASS.EXE и антивирусных средств защиты (Early Launch AntiMalware, ELAM) в ОС Windows был добавлен новый механизм для защиты памяти процессов, называемый «Защищённые процессы» (Protected Process Light, PPL). Процессы LSASS.EXE и Windows Defender запускаются как «PPL-защищённые». Процессы, имеющие специальную цифровую подпись, загружаются как «PPL-защищённые», процессы без подписи загружаются как обычные или «PPL-незащищённые». Механизм PPL блокирует любой доступ к памяти «PPL-защищённых» процессов со стороны «PPL-незащищённых», даже если они были запущены с привилегиями отладки. Информация о том, является ли

работающий процесс защищённым, хранится в поле Protection типа PS_PROTECTION, которое было добавлено в структуру EPROCESS. При каждом запросе на получение доступа к процессу с помощью вызова функций CreateProcess() или OpenProcess() ядро Windows производит чтение содержимого поля Protection.

Механизм PPL, в свою очередь, также уязвим и может быть отключён. Нарушитель с помощью драйвера может изменить значение поля Protection: либо понизить привилегии для «PPL-защищённого» процесса, либо наоборот повысить привилегии для вредоносного «PPL-незащищённого» процесса. В результате такой манипуляции доступ к памяти «PPL-защищённого» процесса будет предоставлен без срабатывания штатных средств самозащиты ядра, таких как Kernel Patch Protection (KPP, PatchGuard).

При включённом механизме PPL процесс LSASS запускается как «PPL-защищённый». С помощью драйвера Mimikatz может переписать нулями поле Protection для процесса LSASS, в результате LSASS процесс станет «PPL-незащищённым» и доступ к нему можно будет получить по описанной выше схеме.

1.2.3. Защита памяти с использованием технологии аппаратной виртуализации (Virtualization Based Security, VBS)

Для компенсации недостатков механизмов PPL и SRM компания Microsoft выпустила новый механизм для защиты памяти процесса LSASS и пользовательских учётных данных. Благодаря гипервизору на базе технологии аппаратной виртуализации Hyper-V стало возможным использовать новый режим работы Virtual Secure Mode (VSM), в результате которого часть ядра и защищаемые прикладные программы стали выполняться изолированно от основного ядра и других программ. Таким образом, стало возможным обеспечить защиту памяти с использованием технологии аппаратной виртуализации (Virtualization Based Security, VBS). В настоящее время режим VSM поддерживает два уровня доверия Virtual Trust Levels (VTLs) [9]. На нулевом уровне VTL0 (normal mode) происходит выполнение большинства прикладных программ и основной части ядра ОС Windows; в то время как первый уровень VTL1 (secure mode) предназначен для работы программ и драйверов с повышенными требованиями по безопасности. Для запуска на уровне VTL1 программа должна быть подписана специальным цифровым сертификатом, что исключает запуск вредоносных программ в режиме VTL1. Режим VSM гарантирует отсутствие доступа драйверов и программ из VTL0 в VTL1.

Программы в VTL1 выполняются в изолированном пользовательском режиме (Isolated User Mode, UIM) и называются доверенными приложениями или трастлетами (trustlet, trusted application). Драйвера в VTL1 работают в защищённом режиме ядра (secure kernel).

При включённом режиме VSM программа LSASS.EXE (LSAISO.EXE) работает как трастлет, то есть выполняется как процесс изолированного режима. Учётные данные пользователей хранятся в памяти процесса LSAISO.exe, который является изолированным и взаимодействует с LSASS с помощью механизма удалённого вызова процедур. VSM гарантирует отсутствие доступа к памяти процесса LSAISO.exe со стороны прикладных программ и драйверов, запущенных в VTL0.

Описанный механизм защиты памяти с использованием технологии аппаратной виртуализации доступен в Windows ОС начиная с версии 10 только для редакций Enterprise, в то время как другие версии операционной системы всё ещё остаются уязвимыми для описанных атак. Для защиты памяти процесса LSASS, а также для предотвращения несанкционированного повышения PPL для вредоносных процессов возможно использовать альтернативное решение на базе гипервизора MemoryRanger [10].

2. Анализ безопасности подсистемы локальной аутентификации ОС Linux: современные атаки и защита от них

Операционные системы на базе ядра Linux также подвержены аналогичной атаке на подсистему локальной аутентификации: вредоносные процессы, запущенные с необходимыми привилегиями, могут получить несанкционированный доступ к памяти других работающих процессов для чтения и перезаписи обрабатываемых данных.

В табл. 1 представлен список процессов и данные пользователя, которые могут быть извлечены.

Таблица 1. Список процессов и соответствующая парольно-адресная информация пользователя

Linux-процесс	Парольно-адресная информация пользователя в памяти процесса
gnome-keyring-daemon	• ключи и сертификаты; • имена и пароли текущих пользователей.
apache2	• пароли аутентифицированных пользователей для доступа к веб-ресурсам.
vsftpd	• информация об активных FTP подключениях клиента;
sshd	• информация об активных SSH подключениях.

Исследователь Seong-Joong Kim [11] обнаружил, что в GNOME Keyring версии 3.18.3 пароли всех активных пользователей находятся в открытом виде в памяти процесса gnome-keyring-daemon. Причиной этому является отсутствие вызова функций для перезаписи содержимого памяти из-за особенностей работы компиляторов и оптимизаторов [11]. Эксперты по безопасности разработали ряд программных средств для извлечения и удаления парольно-адресной информации пользователя из памяти процессов. Далее будет осуществлён анализ двух таких средств: MimiPenguin и MimiPy.

2.1. Анализ работы средства MimiPenguin по извлечению парольной информации

Исследователь Hunter J. Gregal из США разработал программное средство MimiPenguin [12] для поиска паролей активных пользователей в памяти процесса gnome-keyring-daemon. Программа MimiPenguin извлекает из памяти пароли активных учётных записей в открытом виде. Для проверки извлечённых паролей MimiPenguin осуществляет их хеширование и сравнение со значениями хеш-кодов паролей из файла /etc/shadow.

MimiPenguin имеет следующие недостатки:

- поддержка поиска паролей только из печатных ASCII символов, отсутствие возможностей поиска паролей из Unicode символов;
- отсутствие функций перезаписи найденной парольной информации;
- низкая скорость работы.

2.2. Анализ работы средства Mimipy по извлечению парольной информации

Исследователь Nicolas Verdier из США на основе средства MimiPenguin разработал программное средство Mimipy [13]. Он добавил возможность перезаписи найденных в памяти паролей.

Программа Mimipy имеет следующие недостатки:

- поддержка поиска паролей только из печатных ASCII символов, отсутствие возможностей поиска паролей из Unicode символов;
- низкая скорость работы.

3. Авторское программное средство MimiDove исключает недостатки конкурентов по защите парольной информации

Для противодействия угрозам утечки парольной информации необходимо иметь возможность перезаписи найденных в памяти паролей вне зависимости от алфавита их символов. На основе программы MimiPenguin было разработано программное средство MimiDove [14].

3.1. Расширение словаря паролей

Linux-подобные операционные системы широко используются в различных странах, и в общем случае пароли пользователя могут содержать символы национальных языков, которые могут быть представлены с использованием кодировок ASCII и Unicode. В отличие от существующих программных средств MimiPenguin и MimiPy, где используется поиск только по печатным символам ASCII, в данной работе поиск осуществляется с учётом всех возможных символов, за исключением нулевого символа. Нулевой байт является индикатором конца предыдущей строки и начала новой со следующего байта.

3.2. Ускорение алгоритма поиска

Для ускорения поиска проведён анализ памяти процессов gnome-keyring-daemon разных пользователей. Это позволило определить расположение паролей. Они находятся в одной из последних областей памяти среди тех, что обозначены в файлах /proc/PID/maps.

Файл включает в себя следующую информацию о каждом блоке памяти процесса:

- адрес начала и конца блока в адресном пространстве процесса;
- права доступа к блоку памяти (read, write, execute);
- смещение, индексный дескриптор inode файла и имя файла, если область памяти была отображена из другого файла (процесса) с помощью утилиты mmap;
- основной и второстепенный номера устройств, если область памяти была отображена из специального файла устройства.

В процессе gnome-keyring-daemon учётные данные во время обработки находятся в стек-памяти. Причиной нахождения паролей в памяти процесса является неправильная работа со стеком во время обработки чувствительных данных в процессе, отвечающем за аутентификацию gnome-keyring-daemon. Программное средство MimiDove с помощью файла /proc/<PID>/maps вычисляет относительные адреса областей памяти процессов, на которые был отображен стек родительских процессов. Данный приём позволил значительно сократить время поиска паролей.

3.3. Удаление паролей из памяти процессов

Парольно–адресная информация пользователей не должна храниться в памяти процессов в открытом виде. Общим решением этой задачи является исправление кода уязвимых программных продуктов: корректная работа с указателями, использование безопасных функций. Но зачастую этот процесс бывает длительным, и не всегда существует возможность оперативно внести изменения в исходный текст программы и установить обновленное программное обеспечение. В качестве частного решения (adhoc) можно рассмотреть возможность автоматического удаления остаточной информации из памяти процессов.

В программном средстве MimiPenguin была реализована возможность удаления паролей из памяти процессов: при этом найденный пароль можно перезаписывать как нулевыми символами, так и ненулевыми символами, для сокрытия факта очистки паролей.

3.4. Сравнение существующих средств по поиску и перезаписи паролей

Разработанное программное средство MimiDove имеет следующие конкурентные преимущества:

- возможность поиска и перезаписи паролей, состоящих из ASCII и Unicode символов;
- высокая скорость работы.

Сравнительные характеристики программ приведены в табл. 2.

Таблица 2. Сравнительные характеристики программ

Название программы	Среднее время работы, сек.	Возможность поиска паролей из символов		Возможность перезаписи паролей из символов	
		ASCII	Unicode	ASCII	Unicode
MimiPenguin	90	+	–	–	–
Mimipy	50	+	–	+	–
MimiDove	2	**+**	**+**	**+**	**+**

Заключение

В настоящей работе проведён анализ механизмов работы подсистем аутентификации в современных операционных системах Windows и Linux на примере техник извлечения из памяти процессов парольно-адресной информации. Определены места хранения парольных данных пользователя, рассмотрены возможные атаки с целью получения доступа к парольным данным пользователей и возможные способы защиты от этих атак. Проведён анализ средств по извлечению парольной- информации пользователя: Mimikatz (Франция), MimiPenguin (США) и Mimipy (США). Рассмотрено влияние расширения паролей на работу программ для извлечения учётных данных.

Реализованное программное средство MimiDove позволяет находить и перезаписывать пароли из памяти процессов вне зависимости от используемого алфавита, что исключает несанкционированный доступ к парольной информации.

СПИСОК ЛИТЕРАТУРЫ:

1. OS Credential Dumping, MITRE ATT&CK, 2018. URL: https://attack.mitre.org/techniques/T1003/ (дата обращения: 20.01.2022).
2. CVE-2018-20781. MITRE, 2018. URL: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-20781 (дата обращения: 20.01.2022).
3. Du J., Li J. Analysis the Structure of SAM and Cracking Password Base on Windows Operating System. In: International Journal of Future Computer and Communication (IJFCC). 2016, vol. 5, no. 2, p. 112–115. DOI: https://doi.org/10.18178/ijfcc.2016.5.2.455.
4. Ligh M., Case A., Levy J., Walters. A. The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory. Indianapolis, Indiana. John Wiley & Sons. 2014. – 912 p.
5. Bassil Y. Windows and Linux Operating Systems from a Security Perspective. Journal of Global Research in Computer Science. 2012, vol. 3, no. 2. URL: https://arxiv.org/ftp/arxiv/papers/1204/1204.0197.pdf (дата обращения: 20.01.2022).
6. Kotlaba L. Active Directory Kerberoasting Attack: Monitoring and Detection Techniques. In Proceedings of the 6th International Conference on Information Systems Security and Privacy (ICISSP 2020). 2020, p. 432–439. DOI: https://doi.org/10.5220/0008955004320439.
7. Delpy B. Mimikatz. URL: https://github.com/gentilkiwi/mimikatz (дата обращения: 20.01.2022).
8. Dimov D., Tzonev Y. Pass-the-Hash: One of the Most Prevalent Yet Underrated Attacks for Credentials Theft and Reuse. In Proceedings of the 18th International Conference on Computer Systems and Technologies (CompSysTech'17). 2017, p. 149–154. DOI: https://doi.org/10.1145/3134302.3134338.
9. Isolated User Mode (IUM) Processes. URL: https://docs.microsoft.com/en-us/windows/win32/procthread/isolated-user-mode--ium--processes (дата обращения: 20.01.2022).

10. Korkin I. Protected Process Light is not Protected: MemoryRanger Fills the Gap Again. Systematic Approaches to Digital Forensic Engineering (SADFE) International Workshop in conjunction with the 42nd IEEE Symposium on Security and Privacy. 2021, p. 298–308. DOI: https://doi.org/10.1109/SPW53761.2021.00050.
11. Besson F., Dang A., Jensen T. Securing Compilation Against Memory Probing. In Proceedings of the 13th Workshop on Programming Languages and Analysis for Security (PLAS'18). 2018, p. 29–40. DOI: https://doi.org/10.1145/3264820.3264822.
12. Gregal H. Mimipenguin. URL: https://github.com/huntergregal/mimipenguin (дата обращения: 20.01.2022).
13. Verdier N. Mimipy. URL: https://github.com/n1nj4sec/mimipy (дата обращения: 20.01.2022).
14. Golub S. MimiDove. URL: https://github.com/SvetlanaGolub/MimiDove (дата обращения: 20.01.2022).

REFERENCES:

[1] OS Credential Dumping, MITRE ATT&CK, 2018. URL: https://attack.mitre.org/techniques/T1003/ (accessed: 20.01.2022).
[2] CVE-2018-20781. MITRE, 2018. URL: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-20781 (accessed: 20.01.2022).
[3] Du J., Li J. Analysis the Structure of SAM and Cracking Password Base on Windows Operating System. In: International Journal of Future Computer and Communication (IJFCC). 2016, vol. 5, no. 2, p. 112–115. DOI: https://doi.org/10.18178/ijfcc.2016.5.2.455.
[4] Ligh M., Case A., Levy J., Walters. A. The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory. Indianapolis, Indiana. John Wiley & Sons. 2014. – 912 p.
[5] Bassil Y. Windows and Linux Operating Systems from a Security Perspective. Journal of Global Research in Computer Science. 2012, vol. 3, no. 2. URL: https://arxiv.org/ftp/arxiv/papers/1204/1204.0197.pdf (accessed: 20.01.2022).
[6] Kotlaba L. Active Directory Kerberoasting Attack: Monitoring and Detection Techniques. In Proceedings of the 6th International Conference on Information Systems Security and Privacy (ICISSP 2020). 2020, p. 432–439. DOI: https://doi.org/10.5220/0008955004320439.
[7] Delpy B. Mimikatz. URL: https://github.com/gentilkiwi/mimikatz (accessed: 20.01.2022).
[8] Dimov D., Tzonev Y. Pass-the-Hash: One of the Most Prevalent Yet Underrated Attacks for Credentials Theft and Reuse. In Proceedings of the 18th International Conference on Computer Systems and Technologies (CompSysTech'17). 2017, p. 149–154. DOI: https://doi.org/10.1145/3134302.3134338.
[9] Isolated User Mode (IUM) Processes. URL: https://docs.microsoft.com/en-us/windows/win32/procthread/isolated-user-mode--ium--processes (accessed: 20.01.2022).
[10] Korkin I. Protected Process Light is not Protected: MemoryRanger Fills the Gap Again. Systematic Approaches to Digital Forensic Engineering (SADFE) International Workshop in conjunction with the 42nd IEEE Symposium on Security and Privacy. 2021, p. 298–308. DOI: https://doi.org/10.1109/SPW53761.2021.00050.
[11] Besson F., Dang A., Jensen T. Securing Compilation Against Memory Probing. In Proceedings of the 13th Workshop on Programming Languages and Analysis for Security (PLAS'18). 2018, p. 29–40. DOI: https://doi.org/10.1145/3264820.3264822.
[12] Gregal H. Mimipenguin. URL: https://github.com/huntergregal/mimipenguin (accessed: 20.01.2022).
[13] Verdier N. Mimipy. URL: https://github.com/n1nj4sec/mimipy (accessed: 20.01.2022).
[14] Golub S. MimiDove. URL: https://github.com/SvetlanaGolub/MimiDove (accessed: 20.01.2022).

Поступила в редакцию – 20 декабря 2021 г. Окончательный вариант – 13 февраля, 2022.
Received – December 20, 2021. The final version – February 13, 2022.