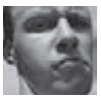
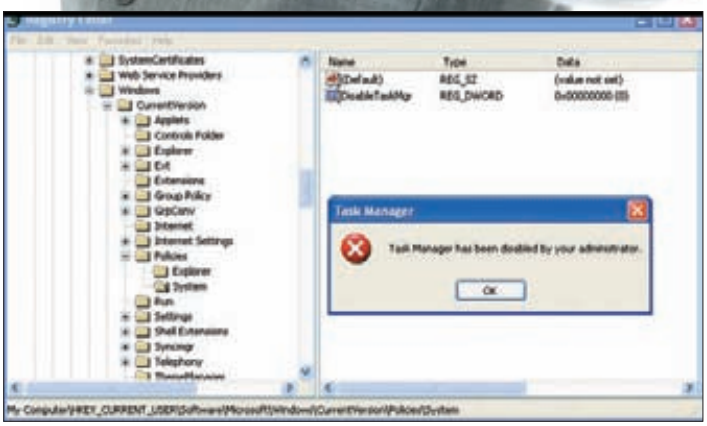
Алексей »GreenDog» Тюрин, DSecRG.ru, Digital Security (agrrrdog@gmail.com)



Недостаточно прав? Достаточно

8 ПРИЕМОВ ДЛЯ ОБХОДА ГРУППОВЫХ ПОЛИТИК В ДОМЕНЕ



Не знаю, чем руководствовались люди из Microsoft, когда проектировали и создавали систему групповых политик в Windows, но получилось у них не очень. Система получилась гибкой и функциональной, но с немалым количеством лазеек, позволяющих обойти ограничения и добраться до тех мест ОС, доступ к которым запрещен.

По правде говоря, групповые политики были исследованы вдоль и поперек еще пять лет назад. Однако используемые решения мало чем изменились. Многие баги по-прежнему работают. Кроме того, появляются новые приемы для обхода групповых политик. Поэтому мы решили собрать для тебя некоторый набор рецептов, чтобы ты, во-первых, мог взять на вооружение, а во-вторых, перестал верить в то, что групповые политики — это панацея. При всем удобстве их использования они не могут обеспечить должный уровень защиты. Но обо всем по порядку.

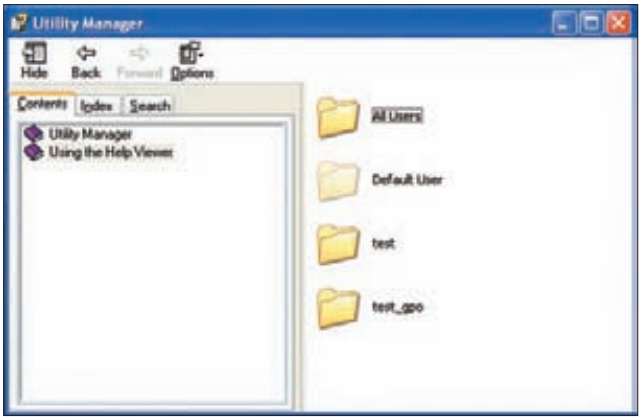
ЧТО ЭТО ТАКОЕ?

Если верить скучным определениям, то групповые политики (или Group Policy) — это эффективный и централизованный механизм управления многочисленными параметрами операционных систем и приложений. Групповые политики позволяют админам определять правила, в соответствии с которыми настраиваются параметры рабочей среды как для пользователей, так и для компьютеров. Проще говоря, это довольно мощный инструмент для ограничения в действиях обычных пользователей. Существует масса различных политик и прав, с помощью которых можно запретить вызов диспетчера задач или редактора реестра, запретить доступ к меню «Пуск», а также довольно гибко ограничить запуск программного обеспечения (это реализуется с помощью так называемых Software Restriction Policies). Является ли этот механизм эффективным? Лишь отчасти. Доступ к шорткатам, запуск левого ПО и системных приложений, изменение настроек — все это достаточно легко запрещается с помощью групповых политик, и с этой точки зрения можно сказать спасибо разработчикам ОС. Но, увы, как это обычно бывает, эти политики зачастую можно обойти. Тут стоит сделать оговорку. Все политики можно разбить на две категории — для компов и для пользователей. Групповые политики доступны как в домене, так и на локальном компе. Если речь идет о локальной машине, то их можно посмотреть через специальную оснастку gpedit.msc (secpol.msc). В данной статье основной акцент сделан именно на доменные групповые политики. Итак, приступим.

ТРЮК 1. ОБХОДИМ ЗАГРУЗКУ ПОЛИТИК

Давай разберемся, как вообще каждая машина в локальной сети получает групповые политики из домена? Процесс создания групповых политик можно разбить на следующие условные этапы:

1. Админ создает объект групповой политики.
2. Привязывает его к каким-то элементам домена.
3. При входе в домен комп отправляет запрос на получение политик и получает их в ответ от домена.
4. При входе пользователя выполняется аналогичный запрос, но уже по пользовательским политикам.



Win+U + Help + Jump to url = Полноценный Explorer

Итак, что мы здесь видим: политики подгружаются на стадии входа в систему. Здесь есть небольшая фича. По умолчанию обновление политик выполняется каждые 5 минут. Но если политики не были получены во время входа в систему, то обновляться они не будут! Вырисовывается элементарный способ, как эту особенность можно использовать:

1. Вынимаем патч-корд из компа.
2. Включаем комп и логинимся под своей учеткой.
3. Подключаем патч-корд обратно.

Даже при отсутствии доступа в сеть мы сможем войти в домен, так как винда ранее закешировала наш логин и пароль (это произошло во время предыдущего входа в систему). Но уже без применения групповых политик. При этом мы спокойно сможем использовать ресурсы сети, так как патч-корд к этому моменту будет на месте, а со всякими авторизациями на удаленных ресурсах справится сама винда. Стоит добавить, что в безопасном режиме винды групповые политики вообще не действуют. Комментарии, я думаю, излишни.

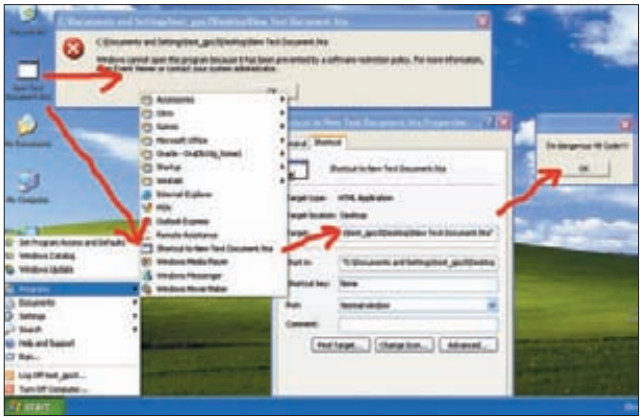
ТРЮК 2. КАК ПРОИСХОДИТ ПРОВЕРКА ПОЛИТИК?

Важно понимать, на каком этапе происходит сопоставление действия, которое хочет выполнить пользователь, с теми ограничениями групповых политик, которые на него накладываются. Сперва давай разберемся, где расположены политики.

Изначально, конечно, на контроллере домена, откуда уже передаются на машины в локальной сети. После получения групповых политик на клиентской машине они сохраняются в реестре винды в следующих местах (приведены основные ветки):

```
Политики для компа:
• HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\
CurrentVersion\Policies\
• HKEY_LOCAL_MACHINE\Software\Policies\
Политики для пользователей:
• HKEY_CURENT_USER\Software\Microsoft\Windows\CurrentVersion\
Policies\
• HKEY_CURENT_USER\Software\Policies\
```

Когда запускается какой-то процесс, то в нем (то есть в userspace'е) производится проверка данных веток реестра (через подгруженную библиотеку advapi.dll) на те или иные ограничения, которые потом кешируются/сохраняются в памяти процесса. Они проверяются, когда пользователь выполняет какое-то действие, например запуск ПО. В чем подвох? В том, что контроль производится из самого процесса. То есть если процесс «не захочет» проверять политики, то ничто его не заставит их соблюдать. Никакого общего мониторинга не производится! Отсюда вывод: если мы каким-то образом сможем запустить произвольный процесс, то политики нам уже не страшны. Сделать как правило — не проблема. Даже если нет возможности закачать программу на хост, можно выполнить ее удаленно (например, через шару).



Обход через некорректную обработку ярлыков

ТРЮК 3. ОБХОДИМ SRP

Увы, дальше на нашем пути возникает другой механизм ограничений — SRP (Software Restriction Policies). Это группа политик, с помощью которых админ может ограничить список ПО, которое может запускать пользователь, через черный и белый списки. Blacklist и Whitelist определяются с помощью правил, которые можно задавать несколькими способами: по зонам и по сертификатам (первые два варианта практически не используются), а также по пути до файла и по его хешу. О том, что в системе действуют политики SRP, указывает соответствующий пункт в реестре — HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\Safer\CodeIdentifiers\TransparentEnabled со значением большим 0, который, как уже было сказано выше, проверяется при запуске процесса. Наша задача, соответственно, отрубить эту проверку внутри запускаемого процесса. Марк Руссинович (goo.gl/KNauh) еще в далеком 2005 году опубликовал пост в блоге об обходе SRP и представил тулзу GPdisable. Она производит DLL-инъекцию в заданный процесс, подгружая специальную DLL'ку. Когда процесс попытается получить значение ключа реестра HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\Safer\CodeIdentifiers\TransparentEnabled, то есть будет проверять присутствие политик SRP, данная библиотека перехватит запрос и возвратит STATUS_OBJECT_NAME_NOT_FOUND. Таким образом, процесс думает, что все ОК и SRP политики в системе не действуют.

После покупки компании Sysinternals Майкрософтом GPdisable перестал был официально доступным (но его по-прежнему легко найти в Сети (bit.ly/nnzjN9). Есть еще более продвинутые решения. Утилита GPCul8or (bit.ly/nJAYrj) от Eric'a Rachner'a выполняет аналогичные функции, но доступна в исходниках. Что это нам дает? Мы можем добавить в GPCul8or любые другие значения реестра винды (DisableTaskMgr, ProxySettingsPerUser к примеру) и таким образом обойти все возможные ограничения политик. Какие именно значения, спросишь ты. Тебе в помощь RegMon от Марка Руссиновича, хотя, по сути — это все значения из ветки Policies.

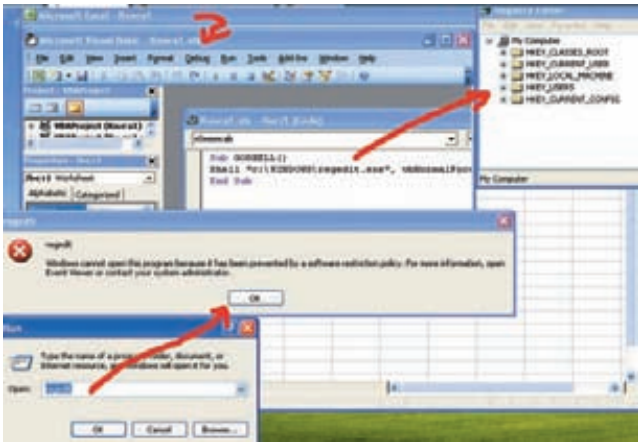
Другой оригинальный способ в своем блоге опубликовал Дидье Стивенс (goo.gl/LE1M0). Используя свою тулзу bpmtk (Basic Process Manipulation Tool Kit), он предложил прямо в памяти процесса изменять значение необходимого для групповой политики ветки реестра.

ТРЮК 4. BINARY PLANTING

Утилита GPdisable состоит из двух файлов:

- gpdisable.exe — инъектирует DLL в процесс;
- gpdisable.dll — специальная DLL для обхода SRP.

Как я уже сказал, если мы можем запустить приложение, то можем легко обойти SRP и другие политики (через GPdisable, bpmtk,

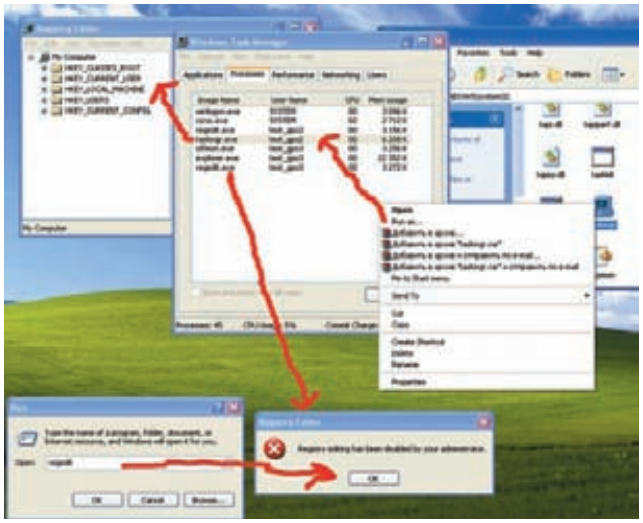


Макросы не палятся политиками SRP

GPCul8or — неважно). Однако в реальной системе может оказаться не так уж просто запустить эти приложения. Но мы можем подгрузить DLL (в том числе gpdisable.dll). Тут есть важный нюанс. Групповые политики при запуске ПО могут проверять и DLL'ки, но при этом достаточно сильно падает производительность системы, потому по умолчанию эта опция отключена. И мы это можем использовать! Очень кстати приходится недавнее исследование от компании Across Security (bit.ly/ov7EAz), которое рассказывает о новых (достаточно извращенных, но работающих) методах подгрузки кода в процессы. Прием называется Binary planting (и как его классический пример — dll hijacking), при его изучении у меня возникла мысль: «а почему не использовать его для обхода групповых политик?». Если система разрешает запуск приложений только из белого списка (пускай даже только Word), то этого уже достаточно, чтобы подгрузить нашу полезную DLL для обхода SRP. Итак, попробуем скрестить dll hijacking от парней из Across и GPdisable:

ГРУППОВЫЕ ПОЛИТИКИ В ТОНКИХ КЛИЕНТАХ

Хочется еще рассказать про такие системы, как Citrix XenApp. Что это такое? XenApp, если говорить простым языком, это система «доставки» приложений (хотя это только часть функционала). По сути, это что-то типа терминального сервера винды, но когда пользователю доступно только конкретное приложение. В жизни это выглядит так. Пользователь коннектится клиентом к Citrix-серверу — ему выводится список доступного ПО. Далее юзер запускает какое-то приложение и начинает в нем работать. Основная фича в том, что фактически процесс приложения выполняется на Citrix-сервере. По сути, данный подход хорош (особенно с тонкими клиентами), но у него есть пучок косяков с точки зрения безопасности. Так как процесс — на сервере, то, значит, пользователю доступны все ресурсы сервера (с учетом пользовательских прав, конечно). Это не очень хорошо, так как предполагается, что у пользователя должен быть доступ только к запущенной программе, а не к ОС. Что еще хуже, добраться-то до самой ОС — не проблема. Даже если у самого ПО нет возможности по взаимодействию с ОС (нет меню «Открыть», «Сохранить как»), то стандартные возможности винды все еще работают: нажимаем в Citrix-приложении <Ctrl+Shift+Esc> — нам открывается диспетчер задач Citrix-сервера, или правый клик по раскладке клавиатуры, а оттуда в файл справки с возможностью листинга директорий. Лично я столкнулся с групповыми политиками именно в этом контексте — при взломе Citrix.



Обход политик через Runas

1. Переименовываем gpdisable.dll в ehTrace.dll.
2. Создаем папку с именем куку.{2E095DD0-AF56-47E4-A099-EAC038DECC24} (название любое, текст после точки исчезнет).
3. Кидаем ehTrace.dll в только что созданную папку.
4. Заходим в папку и создаем там любой документ в Word, Excel или, к примеру, PDF'ку.
5. Теперь открываем только что созданный файл.
6. Соответствующая программа должна запуститься. И запустить вместе с подгруженной DLL'кой!
8. Все, политики нам не страшны.

ТРЮК 5. ИСПОЛЬЗУЕМ ИСКЛЮЧЕНИЯ

Часто можно обойтись и без подобных ухищрений, если знать тонкости политик, в результате которых их действия распространяются:

- на программы, запущенные от имени учетной записи SYSTEM;
- драйверы и другие приложения уровня ядра;
- макросы внутри документов Microsoft Office;
- программы, написанные для общей многоязыковой библиотеки времени выполнения (Common Language Runtime).

Итак, процессы от SYSTEM не контролируются. Первый финт ушами: если есть доступ к какому-то ПО, запущенному под такой учеткой, — атакуем. Например, нажимаем Win+U — запускаются «специальные возможности» (лупа и экранная клавиатура). Utilman.exe (процесс «специальных возможностей») при этом запускается от SYSTEM. Далее идем там в «Справку». Она тоже должна открыться с нужными привилегиями, так как запущена в контексте процесса с правами SYSTEM. Если винда не самая новая (до Vista), то кликаем правой кнопкой на синей верхней панельке «Jump to url», там печатаем «C:\» и получаем настоящий встроенный explorer. Если более новая, то можно по правому клику в тексте хелпа просмотреть исходный код (View Source) через блокнот, откуда далее добраться до файлов. Или другой вариант — «добавить» новый принтер, получив опять же доступ к листингу файлов.

Другая интересная категория — макросы внутри документов Microsoft Office. Это страшное дело. Попробуем для начала реализовать запуск ПО. Хотя если запуск заблочен обычными политиками (не SRP), как, например, блокировкой диспетчера задач, то этот обход не сработает. Но нам-то главное — запустить специальный exe'шник. Поэтому в любом документе смело создаем следующий макрос и пробуем запустить его:

```
Sub GOSHELL()
Shell "C:\windows\system32\regedit.exe", vbNormalFocus
End Sub
```

В результате, как ты можешь догадаться, мы получаем запущенный exe. Хардконный метод предложил опять же Дидье Стивенс (goo.gl/kSPK3). Используя в макросе MS Excel функции VirtualAlloc, WriteProcessMemory и CreateThread, он сумел подгрузить шеллкод из макроса в память процесса. Данный шеллкод подгружает DLL'ку в память процесса, а DLL'ка — не что иное, как cmd.exe. Кстати, ее исходники взяты из проекта ReactOS. Как я уже сказал, SRP может препятствовать запуску DLL'ек (хотя и не делает этого по умолчанию), но если подгрузку библиотек осуществлять, используя функцию LoadLibraryEx с LOAD_IGNORE_CODE_AUTHZ_LEVEL вместо LoadLibrary, то проверка на принадлежность подгружаемой dll к white-листу не происходит!

ТРЮК 6. ИСПОЛЬЗУЕМ ПЕРЕМЕННЫЕ СРЕДЫ

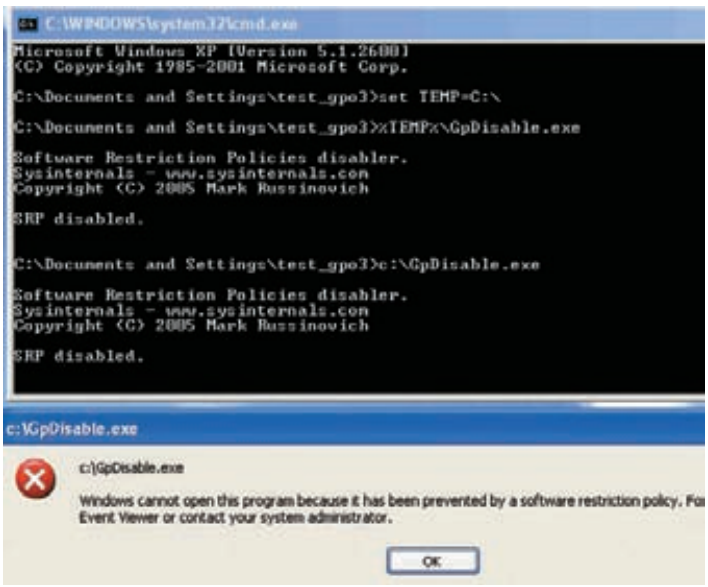
Когда начинаешь мучить групповые политики, то приходит осознание, что для создания защищенной системы потребуется попотеть. Дело трудное и с большим количеством тонкостей. Например, разработчики предлагают админам использовать удобный хинт — указывать переменные среды в качестве путей для ограничений SRP. Да вот здесь проблема. У пользователя, если их жестко не прищучить, есть возможность их переопределять. Указал, например, админ, что из папки %TEMP% можно запускать exe'шники, а юзер взял да и переопределил следующей командой:

```
Set TEMP C:\
```

И вот так просто получил возможность запускать файлы из корня C:. Кроме того, не стоит забывать про стандартные директории, из которых разрешен запуск exe-файлов:

```
• %HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\
CurrentVersion\SystemRoot%
• %HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\
CurrentVersion\SystemRoot%*.exe
• %HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\
CurrentVersion\SystemRoot%System32\*.exe
• %HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\
CurrentVersion\ProgramFilesDir%
```

Они разрешают запуск ПО только из папки Windows и Program Files для пользователей. У обычного пользователя нет возможности записи в них, но и здесь могут быть проблемы. Так как на самом деле права на запись у пользователя есть — по дефолту в папку C:\windows\system32\spool\Printers и C:\windows\temp. Если у пользователя будет возможность писать в какой-то каталог с софтом, то, считай, соответствующие политики SRP уже не сработают. Кстати, для того чтобы на практике поверить, какие у пользователя есть права, поможет тулза — AccessChk от все того же Руссиновича (goo.gl/jQ9tt).



Обход через изменение переменной окружения

ТРЮК 7. ИСПОЛЬЗУЕМ ДРУГОГО ПОЛЬЗОВАТЕЛЯ

Есть способ не подпустить подгрузки политик, но для этого трика нам понадобятся логин и пароль другого пользователя. Суть в том, что нам надо войти в систему «еще раз», но не под собой. Тут два варианта:

1. <Shift> + правый клик на запускаемом файле, далее в контекстном меню выбираем «Run as...».
2. Через консоль набираем команду: runas /noprofile <название exe-файла>.

Другой пользователь, под которым ты запускаешь программку, как и ты, может быть обычным пользователем с ограниченными правами. Но политики на запущенную программку уже не будут действовать! См. рисунок.

На нем пользователь test_gpo3 не может запустить regedit из-за политик. Но, запустив под test_gpo2 любой exe'шник (диспетчер задач например), он уже ничем не ограничен и поэтому может запустить regedit. Кроме того, если у нас есть возможность удаленного входа в систему (по RDP, например), то мы можем провести аналогичный финт, но только с одной учеткой (демонстрацию можешь посмотреть в этом видео — bit.ly/pXsBj6).

ТРЮК 8. ВСПОМИНАЕМ ПРО HTA

Последний хинт касается неофициальных исключений, на которые не действуют групповые политики. Вадимс Поданс написал в блоге отличную серию постов, посвященных SRP-политикам. В частности, он обнаружил отличный путь для их обхода и запуска произвольного кода (goo.gl/BmBsm) с использованием приложения HTA (HTML Application). Итак, последовательность действий:

1. Создаем файлик с примерно таким текстом:

```
<HTML>
<script language="vbscript">
  msgbox "I'm dangerous VB Code!!!"
</script>
</HTML>
```

2. Сохраняем его с расширением .hta (например, execute_this.hta).
3. Создаем ярлык для него.
4. Открываем ссылку — и hta запускается.

Надо ли говорить, что вместо вызова безобидного MessageBox'а VB-код может сделать в системе что угодно? Политики SRP должны проверять весь код, который может исполняться, в том числе и всевозможные скрипты. Однако из-за тонкостей работы групповых политик данный обход работает. К аналогичным «глюковатым» расширениям помимо HTA Вадимс относит REG, MSC, HTA, CHM. Точно так же ситуация наблюдается и с com=файлами (в том числе всякими олдскульными DOC'овскими программами, которые все еще разбросаны в папке винды). Они не учитывают правила групповых политик, так как работают в виртуальной машине DOS.

НАШ ИТОГ

Как ты можешь заметить, всевозможных лазеек полно. Их разнообразие и ухищрения не могут не удивлять. То, что я скажу дальше, может тебя удивить. Даже после перечисления всех этих способов для обхода ограничений я все же настоятельно рекомендую не пренебрегать их использованием. К тому же теперь ты можешь учитывать возможность применения этих лазеек и с намного большей вероятностью настроить систему, которая будет достаточно защищена.