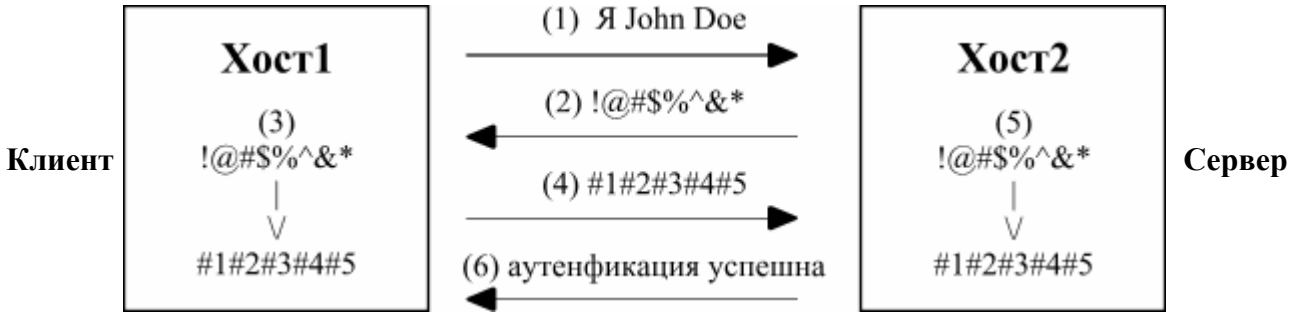
Недостатки системы защиты Win2K и способы их преодоления

Введение. Как и любой другой современной многопользовательской операционной системе Win2k обладает некоторыми средствами защиты, которые позволяют ограничивать пользователей в их правах. И так же как и в любом программном продукте в ней присутствуют ошибки программистов, которые делают возможным обхождение средства защиты этой операционной системы.На втором месте по причинам взлома операционных систем находится недостаточная квалификация администратора, который на закрывает все возможные дыры, проставленные по умолчанию или присутствующие в другом виде. В данном эссе предполагается рассмотреть неоторые такие возможные дыры, как они используются для взлома, и меры административной защиты, которые необходимо наложить для их устранения.

Прежде всего, мне хотелось бы рассказать, как устроена процедура входа в домен или локальный вход в Windows NT и где хранятся пароли. Вход в систему с регистрацией на контролере домена реализован по алгоритму *CHAP* (*Challenge Handshake Autenfication Protocol*). Схема передачи пароля:



Рассмотрим этапы подробнее:

1. Клиент передает серверу запрос об аутенфикации пользователя (John Doe).
2. Сервер генерирует случайную последовательность данных (challenge) и передает клиенту.
3. Клиент, получив данные, с поощью хеш-функции генерирует хеш (от английского "hash" - мешанина) где входными данными являются пароль и полученные данные.
4. Передача полученного хеша серверу.
5. Сервер генерирует на своей стороне хеш, используя те же входные данные (пароль и случайные данные).
6. Сверив два хеша сообщается результат аутенфикации.

Хеш функция необратима, т.е. нельзя получить пароль, имея только хеш (не перебирая все варианты). Как видно - при данной схеме избегается передача пароля в незашифрованном виде. Даже если злоумышленник перехватит хеш, то при правильно выбранном пароле узнать пароль будет невозможно (перебор всех комбинаций займет длительный период). Кроме того, использование каждый раз при генерации хеша в качестве входных данных случайные данные, защищает от повторного использования перехваченного злоумышленником хеша, который может быть получен при авторизации подлинного пользователя. В Windows NT пароли, а вернее хеши паролей, для локального и удаленного входа в систему хранятся в файле *%systemroot%\system32\SAM*. Однако просмотреть этот файл, даже имея права администратора, не удастся - система блокирует обращения к этому файлу. В файле *SAM* (Security Account Manager) хранятся хеши паролей для каждого пользователя в структуре, называемой, *V-блок*. Он имеет размер 32 байта и содержит в себе хеш пароля для локального входа (NT hash - 16 байт), а также хеш,

используемый при аутенфикации при попытке использовать общие ресурсы других хостов (LanMan hash - 16 байт).
Алгоритм формирования NT hash:

1. введенный пароль перекодируются в юникод.
2. на основе полученной строки генерируется хеш (MD4).
3. полученный хеш шифруется алгоритмом DES. В качестве ключа используется RID (младшая часть SID - ID пользователя). Этот шаг используется для того, чтобы два пользователя с одинаковыми паролями имели разные хеши.

Алгоритм формирования LanMan hash:

1. введенный пароль переводится в верхний регистр.
2. затем константная строка шифруется алгоритмом DES, используя в качестве ключа 7 первых байт пароля (пароль может быть максимум 14 символов, если он короче, то добавляется нулями). Другая постоянная строка шифруется байтами 7-14 пароля.
3. затем с полученной строкой производится манипуляция как и в шаге 3 для NT hash.

Для повышения безопасности в системах с сервис паком (*Service Pack*) выше третьего присутствует утилита *syskey*. Однако переход на ее использование необратим, поэтому перед ее использованием рекомендую сделать резервную копию системных фалов. Утилита *syskey* повышает надежность хранения паролей путем хранения в файле *SAM* не хеша паролей, а хеша хеша паролей. Т.е. перед записью хеша пароля в *SAM* генерируется еще один хеш и только потом записывается. Необходимые данные при генерации хеша могут храниться как на жестком диске, так и на дискете. Т.е. до появления стандартного диалога с логином и паролем необходимо будет ввести дополнительный пароль. Во втором случае, локальный вход в систему может быть осуществлен только при наличии в дисководе дискеты с этой информацией.

Теперь рассмотрим различные методы обхода этой защиты. Возможны три различные ситуации:

- Физический доступ к компьютеру;
- Доступ в локальной сети;
- Доступ в глобальной сети

Итак при наличии физического доступа

Самое простое – не узнать, а изменить пароль администратора. Так например Исследователь Radim "EliCZ" Picha (Bugs@EliCZ.cjb.net) обнаружил серьезную уязвимость в безопасности Windows NT и Windows 2000. Им была написана программа (exploit), демонстрирующая очевидную слабость локальной подсистемы безопасности NT/2000 и полностью компрометирующая всю систему безопасности этих операционных систем. Программа, названая DebPloit (от английских слов Debug и Exploit), использует "дыру" в подсистеме отладки (debugging subsystem) и позволяет ЛЮБОМУ пользователю с ЛЮБЫМИ привилегиями (даже пользователям входящим в группы Guests и Restricted Users), выполнять программный код с правами администратора и/или локальной системы. Другими словами, любой человек имеющий доступ к локальному компьютеру может стать администратором и делать на этом компьютере все, что угодно. Принцип работы DebPloit: программа "просит" отладочную подсистему (smss.exe) вернуть описатель (handle) процесса, запущенного с правами администратора или локальной системы (в системе всегда находится большое кол-во процессов, работающих с правами локальной системы):

1. Становимся dbgss-клиентом (функция DbgUiConnectToDbg).
2. Подключаемся к LCP-порту DbgSsApiPort (ф-ция ZwConnectPort). Любой пользователь имеет доступ к этому порту!

3. Посылаем запрос на отладку процесса к dbgss, точно так же как это делает CreateProcess (ф-ция ZwRequestPort).
4. Ожидаем ответа (CREATE_PROCESS_DEBUG_EVENT) от dbgss (ф-ция WaitForDebugEvent). Ответ будет содержать описатель (handle) процесса.
5. Переключам свой текущий контекст безопасности на контекст безопасности описателя, полученного на шаге 4.
6. Исполняем код (например запускаем внешнюю программу) с правами выбранного для отладки процесса.

Данная уязвимость превращает ранее считаемые достаточно защищенными системы Windows NT/2000 в такую же беззащитную как Windows 9x. Хочется отметить что подобная уязвимость уже не присутствует в Windows XP. Будем надеяться что впакет исправлений Service Pack 3 для Windows 2000 будет включена заплатка для данной уязвимости. Скачать DebPloit можно http://www.anticracking.sk/EliCZ/bugs/DebPloit.zip (исходные тексты прилагаются). В данном архиве имеется программа для тестирования системы на наличие данной уязвимости (IsAvailable), а также и сами эксплоиты для Windows NT/2000 (ERunAsX, ERunAs2X). Все что нужно это залогиниться в систему под любым пользователем, а затем запустить, к примеру на Windows 2000, ERunAs2X.exe CMD.EXE. При этом процесс CMD.EXE запустится с системными привилегиями. С данной консоли можно вызвать MMC (Microsof Managment Console) и после этого можно управлять системой как заблагорассудится.
Что бы закрыть эту "дыру" в безопасности NT/2000, автор DebPloit написал специальный драйвер DebPloitFix, который устанавливает новые права для LPC-порта DbgSsApiPort. После запуска DebPloitFix, доступ к DbgSsApiPort будет иметь только локальная система. Загрузить DebPloitFix и исходный код к нему можно http://www.smartline.ru/software/DebPloitFix.zip. Интересная получается ситуация: официальной заплатки от Microsoft не выпускалось - ее пришлось написать автору уязвимости... Спасение утопающих - дело рук самих утопающих?

Существуют варианты взлома, скоторый позволяют использовать файл SAM который собственно и хранит зашифрованные пароли пользователей. Операционная система запрещает доступ к этому файлу любому пользователю, даже обладающему административными правами (но это возможно, только если файл SAM находиться на разделе NTFS, если же используется раздел FAT вытянуть и взломать этот файл не составит труда). Поэтому для того, чтобы скопировать этот файл необходимо наличие других операционных систем, таких как DOS или Linux. Загрузившись под какой нибудь из этих операционных систем необходимо скопировать файл SAM на какой-нибудь жесткий носитель и тогда станет возможным с ним работать. При его просмотре мы увидим примерно следующую картину:

_>ABL:0:A1BDB9ED706F3C47C9C7FAD571FDC1D5:BECB0BF86A97B65C118B22E25C984623::_AB@>5==0O CG5B=0O 70?8AL 4;O 4>ABC?0 3>AB59 :
:><?LNB5@C/4><5=C:
Stupid:500:1B9A5B6GG9F99DA65D3B68BFDA66BC84:C87E40758D4FFF8D3B3C0390ADA7E136::_AB@>5==0O CG5B=0O 70?8AL 04<8=8AB@0B>@0 :><?LNB5@0/4><5=0:
ZaDNiCa:1000:C819160E87A9CGADHA5F8C243A93ACB3:5D67D210E1D913F72BCD8EDD CB5172DB:Indeed ZaDNiCa::

Что видим? Присутствуют 3 пользователя. Первый - это Гость (программа английская и пользователей с русскими логинами выдает в таком виде).
Нас интересует пользователь, имеющий во втором поле значение 500, что соответствует встроенной учетной записи администратора. Зачастую его можно определить по логину (*Administrator* - для английской версии ОС, для русской версии это будет набор непонятных символов). Рекомендую определять встроенную учетную запись исключительно по ID =500. потому как встроенную учетную запись администратора

можно переименовать во что-нибудь непривлекательное, а записи гостя дать имя *Administrator*
Дальше меняем V-блок в имеющейся у нас копии SAM:
chntpw.exe -u Stupid SAM
Username: Stupid, RID = 500 (0x1f4)
[file offset: 03b1c]
RID : 0500 [01f4]
Username: Stupid
fullname:
comment : |AB@>5==0O CG5B=0O 70?8AL 04<8=8AB@0B>@0 :><?LNB5@0/4><5=0
homedir :

Crypted NT pw: d6 7e 8f 6e ae 54 62 74 c4 c3 05 c5 82 3f 89 64
Crypted LM pw: 32 53 a7 8e a7 5b 56 fd d4 30 dd f8 e0 40 8d 4c
MD4 hash : c8 7e 40 75 8d 4f ff 8d 3b 3c 03 90 ad a7 e1 36
LANMAN hash : 1b 9a 5b 6f 19 f9 9d a6 5d 3b 68 bf da 66 bc 84
Нам выводится имеющийся V-блок и предлагается ввести новый пароль. После его ввода выдается новый V-блок, который и записывается в нашу копию SAM-файла. Осталось немного - поместить файл SAM на системный диск, откуда он и был взят.
Какие есть недостатки у этого метода - если для повышения безопасности используется утилита *syskey* - то сначала придется отключить ее, а потом только менять пароль
Как отключить *syskey*? Вообще считается (так и сообщается переходом на ее использование, а также по официальным заявлениям Microsoft), что отменить использование утилиты syskey нельзя. На самом деле отключить ее можно (правда не совсем корректно), изменив следующие ключи реестра:
HKLM\SAM\Domains\Account\F
необходимо обнулить содержимое этого ключа. Эта структура хранит *SID* компьютера, а также другую системную информацию. Здесь же хранится копия статуса ключа *SecureBoot*.
HKLM\System\CurrentControlSet\Control\Lsa\SecureBoot
необходимо также присвоить значение 0. Этот ключ хранит режим *syskey*:
1 - ключ в реестре
2 - ключ вводится пользователем
3 - ключ на дискете
Обнуление этих двух ключей отключает syskey для системы Windows NT 4.0. Для Windows 2000 нужно поправить еще один ключ:
HKLM\security\Policy\PolSecretEncryptionKey
здесь хранится еще одна копия режима работы syskey. Его также необходимо установить в 0
В чем заключается некорректность отключения *syskey* таким методом? Дело в том что, после отключения *syskey* **НЕВОЗМОЖНО** будет войти в систему ни под каким пользователем. Это происходит потому, что хеши в файле *SAM* неверные. Поэтому необходимо будет изменить их прямой записью хеша пароля в файл *SAM.*
Главное запомнить - отключение *syskey* таким образом нужно применять только в крайнем случае! Наиболее корректным методом является откат системы с помощью дискеты с резервными файлами (естественно, что перед переходом на использование syskey необходимо будет сделать backup системных файлов с помощью утилиты *rdisk*).
Также имея на руках файл SAM необходимо найти какой-нибудь переборщик паролей (*L0phtCrack*, *LC3*, *LCP*), затем выбрать нужный набор символов и ждать пока будет найден пароль. Против такого перебора целесообразно применять не простые пароли, которые состоят из простых слов на русском или английском языке, а использовать

специальные цифры и символы, такие как !@#$%^&* и т.д. Также практически ни подобный взломщик не способен подобрать пароль более чем из 14 символов.
Защита от подобного взлома состаит в том, чтобы выполнять следующие простые рекомендации:

1. Никогда на ставить Win2k на файловую систену FAT или FAT32
2. Запретить в BIOS компьютера загрузку с любых других носителей, кроме как жесткого диска (таких как дискета или CD-rom).
3. Не ставить параллельно с Win2k такие операционные системы как Win9x, DOS, Linux-подобные системы (последние можно настроить на не предоставление доступа разделам Win2k).
4. Использовать сервиспаки предаставляемые Microsoft в частности SP3 начиная с которого используется для повышения безопаснсти утилита SYSKEY.
5. Не использовать простые пароли, на позвлять посторонним людем следить за их набором, а также желательно использовать пароли, длина которых превышает 14 символов.

Также после того, как администратор (или любой другой пользователь) залогинивается в систему, его пароль остается в реестре в зашифрованном виде. Поэтому если администратор забывает залочить машину перед уходом, злоумышленник может воспользоваться утилитой pwdump или подобной Она сдампирует все хеши в файл, к которым потом можно будет подобрать пароли. Она хороша тем, что имеет маленький размер и не требует инсталяции. Также возможно воспользоваться утилитой *RDISK*. Она создает копию резервных файлов в *%systemroo%\repair* (и в том числе и файла *SAM*). После того сохранения там файлов остается лишь списать оттуда файл *SAM._* на дискету. Операция занимает от 1 до 5 минут. Поэтому администратор НИКОГДА не долже оставлять машину незалоченной во время своего ухода, или даже небольшой отлучки.

Другие варианты – не подбирать или менять пароль, а изменить сам файл, которые осуществляет его проверку. Данная проверка осуществляется в библиотеке *MSV1_0.DLL*. Фрагмент кода выглядит так (для Windows NT 4.0 SP5):
call RtlCompareMemory cmp EAX, 10h je
искать необходимо команду *cmp eax, 10h* (рекомендую hex-редактор *hiew*), либо ее шестнадцатеричный вид *83 F8 10*. Такой фрагмент встречается в файле 5 раз по данным смещениям в файле *MSV1_0.DLL*:

Смещение	
относительно от начала	вирутальное
1F6C	(.75B81F6C)
1F95	(.75B81F95)
226A	(.75B8226A)
22AE	(.75B822AE)
22F3	(.75B822F3)

Проверка по смещению *1F9E (.75B81F9E)* в файле - это проверка правильности пароля для локального входа. Необходимо поправить (убрать проверку и поставить простой дальний переход):

1F9E:	nop	(90)
1F9F:	jmp .75B8233A	(E996030000)

Теперь, если заменить существующую библиотеку пропатченной, то локально входить можно под любым пользователем, используя при этом какой вздумается пароль (или пустой пароль).
Этот метод можно применить и для Windows 2000. Однако в ней искомый фрагмент будет встречаться около 10 раз. Изменив все вождения, получим пропатченную версию *MSV1_0.DLL* для Windows 2000. Этот метод наиболее универсален - он подходит для Windows NT/2000 и главное не играет роли используется ли утилита *syskey*.
Хочется отметить, что это наиболее эффективный способ. Его преимущества - применим, даже если используется утилита *syskey*, старый пароль администратора остается прежним (а, значит, он не заподозрит ничего). Единственное что будет изменено - это имя последнего пользователя, осуществившего успешный вход в систему, особенно если была осуществлена попытка получения прав локального администратора на машине, у который обычный вход осуществляется в домен. Но и это легко поправить. Достаточно присвоить следующим ключам реестра необходимые значения:
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon
DefaultDomainName
DefaultUserName
Для этого уже необязательно делать это под DOS. Дело в том, что удалить (или переписать) оригинальный файл *MSV1_0.DLL* невозможно. Но зато можно его переименовать в *MSV1_0.BAK*, а пропатченную библиотеку *MSV1_0.DLL* записать в *%systemroot%\system32*. Естественно, что сделать это можно имя права на запись данный каталог (после того как права администратора были получены это уже не проблема).

Существует еще один оригинальный метод получения прав администратора - *He4Admin*. Суть этого метода состоит в том, чтобы найти сервис (service) запускающий от имени system и расположенный в каталоге, куда обычный пользователь имеет полный доступ. В дефолтовые сервисы не подходят, потому как они расположены в системных каталогах. Это может быть файрвол или какие-нибудь другие дополнительно установленные сервисы. Для поиска подходящих сервисов в дистрибутив входит *He4FindWin32Services.exe*. Необходимо попытаться переименовать найденные ею файлы, а *He4Win32Srv.exe* переименовать в имя исходного файла. Если не удается можно, попытаться его остановить, и после этого переименовать. Если все удалось - необходимо чтобы этот сервис запустился. Для этого необходимо перегрузить систему (после рестарта вместо исходного сервиса запустится *He4Win32Srv.exe*). После этого достаточно запустить:
NewName_He4Win32Service -start:%systemroot%\system32\CMD.EXE
Запущенная консоль унаследует права системы, следовательно, можно из нее запустить *%SystemRoot%\system32\musrmgr.exe* (диспетчер пользователей) и добавить себя в группу администраторов либо сменить пароль администратора и т.д. Недостаток данного метода в том, что далеко не на всех системах можно будет найти нужные сервисы.
Еще один подобный метод реализуется следующим методом - файл *logon.scr* (хранитель экрана, который запускается, если не осуществить вход в систему в течение 15 минут после загрузки ОС) заменяется на файл cmd.exe. В результате запускается командная строка с правами SYSTEM. Далее можно для удобства запустить explorer (очень интересная получается ситуация - залогинился пользователь *SYSTEM*!) и затем добавить нового пользователя с правами администратор, либо сменить пароль уже имеющемуся администратору.
Чтобы уменьшить время, через которое запустится хранитель экрана нужно отредактировать следующий ключ в реестре:
HKEY_USERS\.DEFAULT\Control Panel\Desktop\ScreenSaveTimeOut
значение указано в секундах.

Или же можно не подменять файлы, а изменить всего лишь ключ реестра:
HKEY_USERS\.DEFAULT\Control Panel\Desktop\SCRNSAVE.EXE
"повернуть" на cmd.exe.
Подмену сервисом можно осуществить еще и следующим методом. Необходимо в реестре прописать следующие записи:
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Spooler
ImagePath="C:\temp\srvany.exe"
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Spooler\Parameters
Application="c:\winnt\system32\net.exe"
AppParameters="user Administrator password"
где под *Administrator* понимается встроенная учетная запись администратора.
После перезагрузки можно будет войти в систему, используя пароль password для встроенной учетной записи *Administrator*. Теперь остается только восстановить исходные значения реестра.
Смысл описанного представить нетрудно - утилита *srvany.exe* используется для запуска *net.exe*, которая меняет пароль пользователю. Почему используется именно *srvany.exe* и где ее взять? Взять ее можно из *Windows NT Resource Kit*, а используется она, потому что мы запускаем обыкновенную программу как сервис (вместо *spools.exe* - службы буферизации печати). Теперь встает вопрос - как изменить ключи реестра? Ответ прост. И он не один. Можно воспользоваться все той же linux-загрузочной дискетой, можно подключить диск с исходной системой к любому другому компьютеру с аналогичной операционной системой Windows NT/2000, а можно даже установить еще одну ОС на диск и с нее редактировать реестр (лишнюю версию потом можно будет удалить). Подключить реестр исходной системы, можно запустив *regedt32.exe* и выбрав *load hive*

А как можно получить пароль к системе удаленно? Все тем же перебором - только теперь придется перебирать пароли к общим ресурсам. Дело в том, что в Windows NT по умолчанию всегда присутствуют скрытые общие ресурсы, к ним и надо пробовать подбирать пароли. Теперь остается получить имя строенной учетной записи администратора. Это можно сделать с помощью разных утилит, к примеру, *Red Button*, *Red Shadow*, *Retina* и т.д. Кроме того, эти программы покажут все скрытые ресурсы. Недостаток в том, что осуществить это все можно только с Windows NT/2000. Если "натравить" *Retin*'у то она выдаст много полезной информации, а именно список всех пользователей, которые входили в систему, инфо'у то она выдаст много полезной информации, а именно список всех пользователей, которые входили в систему, информацию о политике безопасности, а также какие пользователи имеют права администратора и какая запись является встроенной. Дело в том, что вс! троенную учетную запись зачастую переименовывают и ставят на нее длинные пароли (оставляя ее на всякий случай для входа в систему). А помимо нее заводят еще нескольких пользователей с правами администратора для повседневной работы. И очень часто пароли на эти аккаунты легче подобрать. Для перебора можно воспользоваться такими утилитами как RedShadow, *NAT* (*Network Auditing Tool*), *Brurus-AE* - выбор их велик и найти их с помощью поисковиков совсем не сложно. Перебирать пароли можно как по словарю, так и простым перебором. Однако последний будет более эффективен в локальной сети, где скорость намного больше чем по интернету.
Еще одним способом захвата прав администратора в локальной сети является использование особенности реализации *NPFS* в Windows NT. Не буду вдаваться в технические подробности - кому они интересны, советую прочитать следующую статью, в ней автор (Вадим Проскурин) детально все изложил *"Проблемы защиты сетевых соединений в Windows NT"*. Вкратце в чем суть: пользователь на своем хосте запускает программу, которая ожидает подключения с удаленной системы администратора (удаленная правка или просмотр реестра, добавление принтера и т.д.). После подключения

администратора, программа, используя его права, создает пользователя и добавляет его в группу локальных администраторов. Самая большая проблема в том, что подключившийся администратор никак не сможет заметить данных манипуляций (правда он может заметить в последствии их результат). Ссылка на программу, реализую! щую данную атаку и немного подправленную, приведена в конце статьи. Автор статьи был очень удивлен - если эту программу запустить и указать "слушать" обращения к службе *spoolss.exe*, то атака будет успешной даже если администратор просто просмотрит папку "Принтеры" на удаленном хосте! Обязательно прочитайте комментарий к программе.
Если система, к которой необходимо получить пароли находится в локальной сети, можно попробовать сниффить траффик, с целью перехвата хешей паролей при авторизации в домен (*NT hash* или *LanMan hash*). Это можно осуществить как с помощью просто программ-снифферов (в данном случае придется "руками" извлекать хеш и пакетов и сохранять его для дальнейшего взлома программами подбора пароля к хешу), так и программой *L0phtCrack* (*LC3*) - что намного удобнее. В ней имеется функции прослушивания сети на предмет передачи *LanMan* и *NT* хешей. Однако проинсталлировать и запустить ее в такой режим можно лишь с правами администратора (на Windows NT). В системах Windows 9x никаких ограничений нет, поэтому ее удобнее использовать именно на них. Почему может не получиться поймать ни одного хеша? Может быть, локальная сеть коммутирована свитчами.
В локальной сети где используются ОС Windows NT без домена, пользователи имеют общие папки (для обмена файлами - что-то вроде Download и Upload) встроенная учетная запись "гость" часто не блокируется - для того чтобы можно было получить доступ к ресурсу *IPC$* и просмотреть список общих ресурсов. Однако есть еще один побочный эффект - разрешен доступ к системному реестру (ветке *HKCU* - характерно для Windows NT) с полным доступом. Это дает многое. В частности можно в общую папку, используемую для обмена файлами, положить программу, а в ключе:
HKCU\Software\Microsoft\Windows\CurrentVersion\Run
создать параметр с путем к этой программе. При следующем входе в систему под эти пользователем программа запуститься и системой можно будет управлять удаленно.
Еще один метод внедрения средства удаленного администрирования (особенно это касается локальных сетей) стал возможным после обнаружения Гунинским ошибки *explorer* при обработке расширений файлов. Эта ошибка позволяет при просмотре файлов *explorer*'ом сделать так, чтобы запускаемый файл отображался с расширением *txt*. Теперь немного фантазии и появляется следующий пакетный файл с именем *readme.txt.{3050F4D8-98B5-11CF-BB82-00AA00BDCE0B}*:
Еще один метод внедрения средства удаленного администрирования (особенно это касается локальных сетей) стал возможным после обнаружения Гунинским ошибки *explorer* при обработке расширений файлов. Эта ошибка позволяет при просмотре файлов *explorer*'ом сделать так, чтобы запускаемый файл отображался с расширением *txt*. Теперь немного фантазии и появляется следующий пакетный файл с именем *readme.txt.{3050F4D8-98B5-11CF-BB82-00AA00BDCE0B}*:
Данный файл можно разместить на общем ресурсе, а также туда же положить файл с именем *"1"* и тот, кто, купившись на уведенное им невинное расширение *txt*, попытается открыть его в explorer'е создаст общий ресурс (*TEMP$*) с полным доступом, который будет указывать на С-диск. А также ему откроется содержимое файла *"1"*. Он будет думать, что открыл файл readme.txt. Единственное что может насторожить - странно дополнительное окно. Что делать дальше, уже ясно: достаточно положить в автозагрузку программу удаленного администрирования и после перезагрузки хостом можно будет управлять.