УТВЕРЖДЕНО

BY.РТНК.00001-03.01 34 01-15-ЛУ

Программно-аппаратный комплекс
«Шлюз безопасности Bel VPN Gate 3.0.1»

РУКОВОДСТВО ОПЕРАТОРА

Руководство администратора

Bel VPN Gate 3.0.1 Приложение

BY.РТНК.00001-03.01 34 01-15

Листов 56

Подп. и дата	
Инв. № дубл.	
Взам. инд. №	
Подп. и дата	
Инд. № подл.	

2012

Bel VPN Gate 3.0.1 Приложение

1. ОТЛИЧИТЕЛЬНЫЕ ОСОБЕННОСТИ РАБОТЫ ПРОДУКТА ПОД УПРАВЛЕНИЕМ ОС LINUX 3
2. НАЧАЛЬНАЯ КОНФИГУРАЦИЯ ДЛЯ УДАЛЕННОГО УПРАВЛЕНИЯ ШЛЮЗОМ 4
3. ОПИСАНИЕ ФОРМАТА INI-ФАЙЛОВ 14
3.1. Формат допустимых строк 14
4. КОНВЕРТОР 16
4.1. Основная логика работы 16
4.2. Ограничения на конвертор 17
4.3. Логика запуска конвертора 24
4.4. Алгоритм работы конвертора 27
4.5. Внутренние настройки консоли и конвертора 30
4.6. Управление конвертором с помощью INI-файла 31
4.7. Сообщения в логе при конвертировании 34
4.8. Описание обработки интерфейсов 37
4.9. Формирование имен структур LSP при конвертировании 45
5. СОЗДАНИЕ ЛОКАЛЬНОГО СЕРТИФИКАТА ПРИ ИСПОЛЬЗОВАНИИ СКЗИ "AVCRYPT VER. 5.1" (РБ.ЮСКИ.09000-02) 48
5.1. Утилита cryptocont.exe 50

1. Отличительные особенности работы продукта под управлением ОС Linux

Отличительными особенностями продукта Bel VPN Gate под управлением операционной системы Linux являются:

- для аппаратной платформы в качестве терминала можно использовать компьютер, подключенный к последовательному порту.
  Для подключения терминального компьютера к аппаратной платформе используется нуль-модемный кабель (5 проводов). На компьютере можно использовать терминальную программу, например, Windows HyperTerminal.

В терминальной программе HyperTerminal проведите настройки:

File-> Properties-> Settings-> Emulation-> VT100.

Во вкладке Connect To нажать на кнопку Configure и провести следующие настройки COM-порта:

Bits per second: 115200

Data bits: 8

Parity: None

Stop bits: 1

Flow control: None

Вход в операционную систему:

Имя пользователя – root

Пароль- отсутствует.

2. Начальная конфигурация для удаленного управления шлюзом

Если планируется удаленно настраивать локальную политику безопасности шлюза при помощи консоли по протоколу SSH1, то после инсталляции Bel VPN Gate рекомендуется загрузить начальную конфигурацию, которая позволит в дальнейшем создать *защищенный канал* для настройки шлюза. При использовании протокола SSH2 загрузка начальной конфигурации не нужна.

Загрузка начальной конфигурации на шлюз безопасности должна осуществляться с локального терминала с помощью cisco-like консоли.

Для создания защищенного канала также необходимо на компьютер, с которого будет осуществляться удаленная настройка шлюза, установить Bel VPN Client с согласованной начальной конфигурацией для создания IPSEC SA между этим компьютером и шлюзом.

Ниже приведен пример настройки начальной конфигурации на шлюзе и удаленном компьютере.

Предположим, что на шлюзе безопасности один сетевой интерфейс FastEthernet0/1 с IP-адресом 192.168.13.1 (Рисунок 1) подключен к локальной сети и на нем установлен продукт Bel VPN Gate. К этой же локальной сети подключен компьютер с IP-адресом 192.168.13.2 для удаленной настройки шлюза и на нем установлен административный пакет Bel VPN Client AdminTool для создания инсталляционного пакета Bel VPN Client. Аутентификация сторон осуществляется при помощи предопределенного ключа со значением “adm123456”.

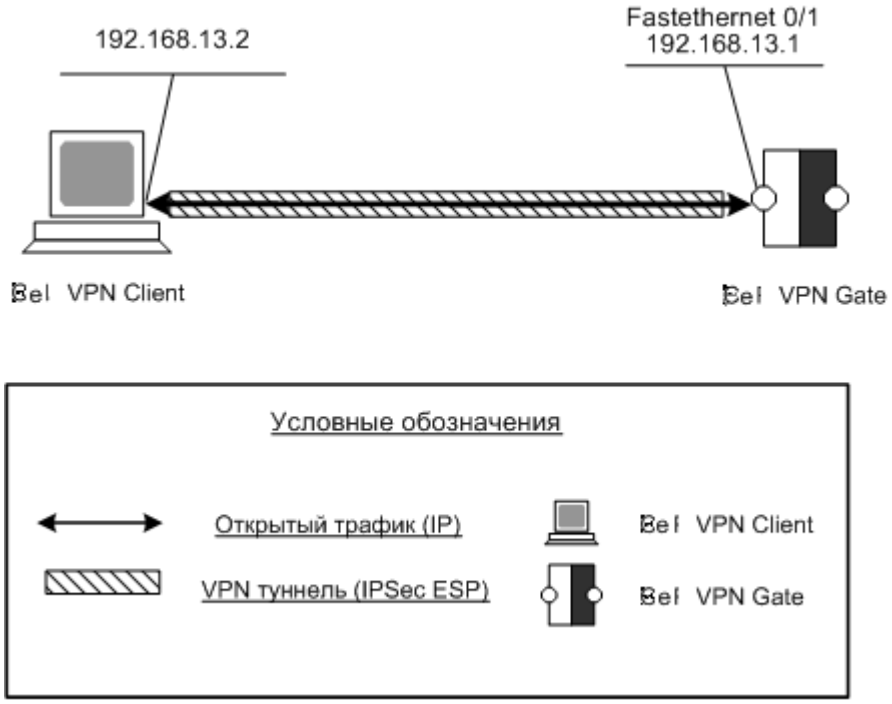

Рисунок 1

Начальная конфигурация на шлюзе

Для создания начальной конфигурации на шлюзе безопасности Bel VPN Gate запускаем cisco-like console командой `cs_console` из каталога `/opt/VPNagent/bin` (см. документ «Cisco-like команды») и вводим следующие команды:

```
configure terminal
crypto isakmp policy 1
    hash md5
    encryption des
    authentication pre-share
    group 2
exit
crypto isakmp key adm123456 address 192.168.3.2
crypto ipsec transform-set adm esp-des esp-md5-hmac
exit
ip access-list extended adm
    permit  tcp  host 192.168.3.1 eq 22 host 192.168.3.2
    permit  tcp  host 192.168.3.1 eq 80 host 192.168.3.2
exit
crypto map adm 1 ipsec-isakmp
    match address adm
    set transform-set adm
    set pfs group2
    set peer 192.168.3.2
exit
interface FastEthernet0/1
    crypto map adm
exit
end
```

При выходе из конфигурационного режима политика безопасности будет загружена на шлюз безопасности.

Начальная конфигурация на клиенте

На компьютере с установленным административным пакетом Bel VPN Client AdminTool (см. документацию «Bel VPN Client. Руководство администратора») запускаем графический интерфейс (`Start – Programs – Bel VPN Client AdminTool – Package Maker`) и создаем согласованную со шлюзом политику для создания защищенного соединения между ними.

Во вкладке Auth заполняем поля для настройки аутентификации на предопределенном ключе “adm123456” и выбираем идентификатор.

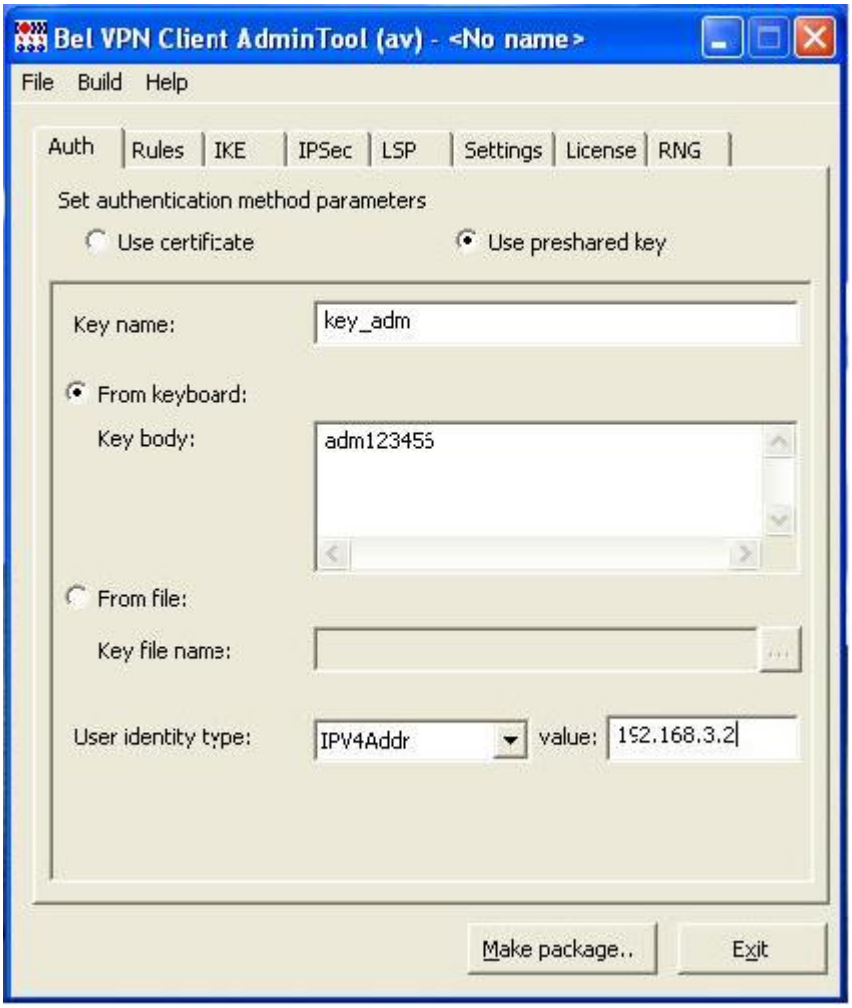

Рисунок 2

Во вкладке Rules создаем правило для создания защищенного соединения. Для этого нажимаем кнопку Add и устанавливаем параметры для правила (более подробно о работе с правилами см. «Bel VPN Client. Руководство администратора»).

В результате форма Add Rule должна иметь следующий вид (Рисунок 3):

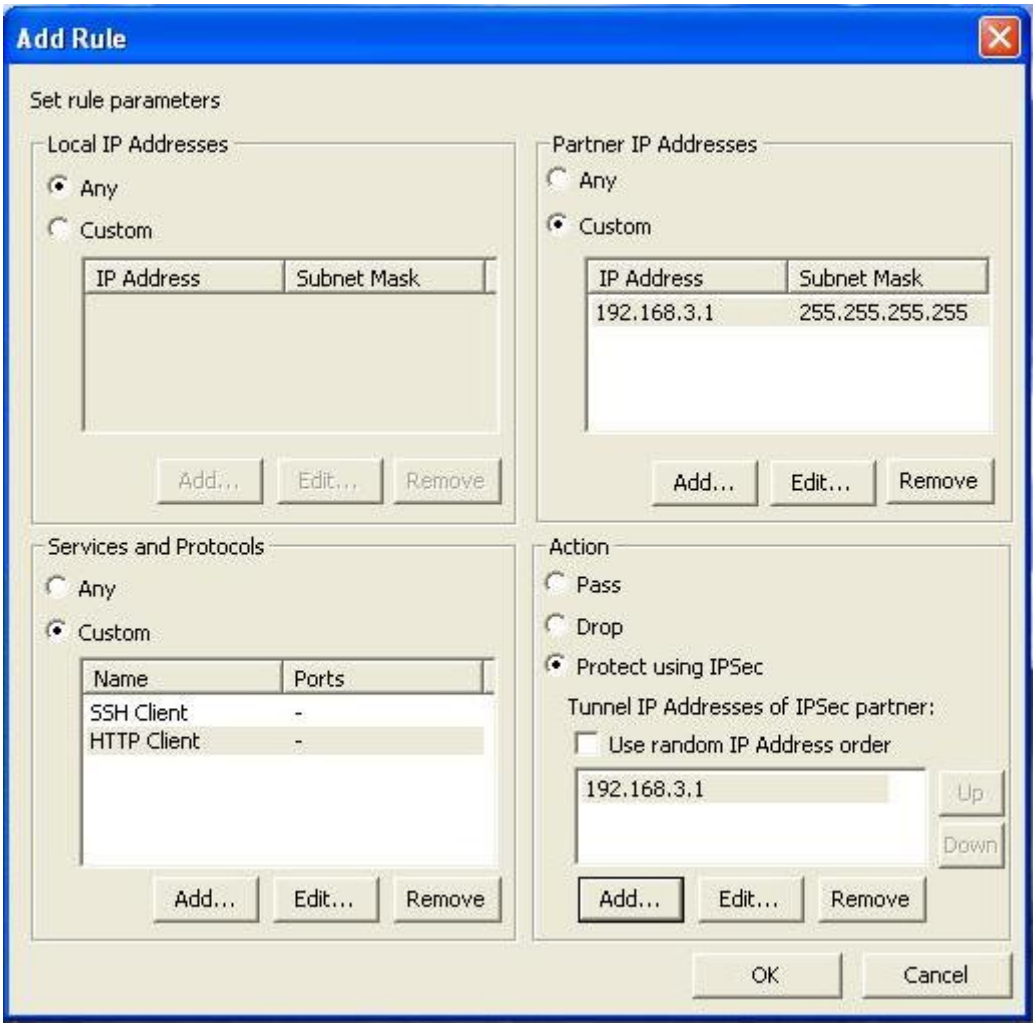

Рисунок 3

Нажимаем кнопку OK для сохранения правила.

Во вкладке Rules для выделенного нового правила нажимаем кнопку Up для повышения приоритета. Вкладка Rules будет иметь следующий вид:

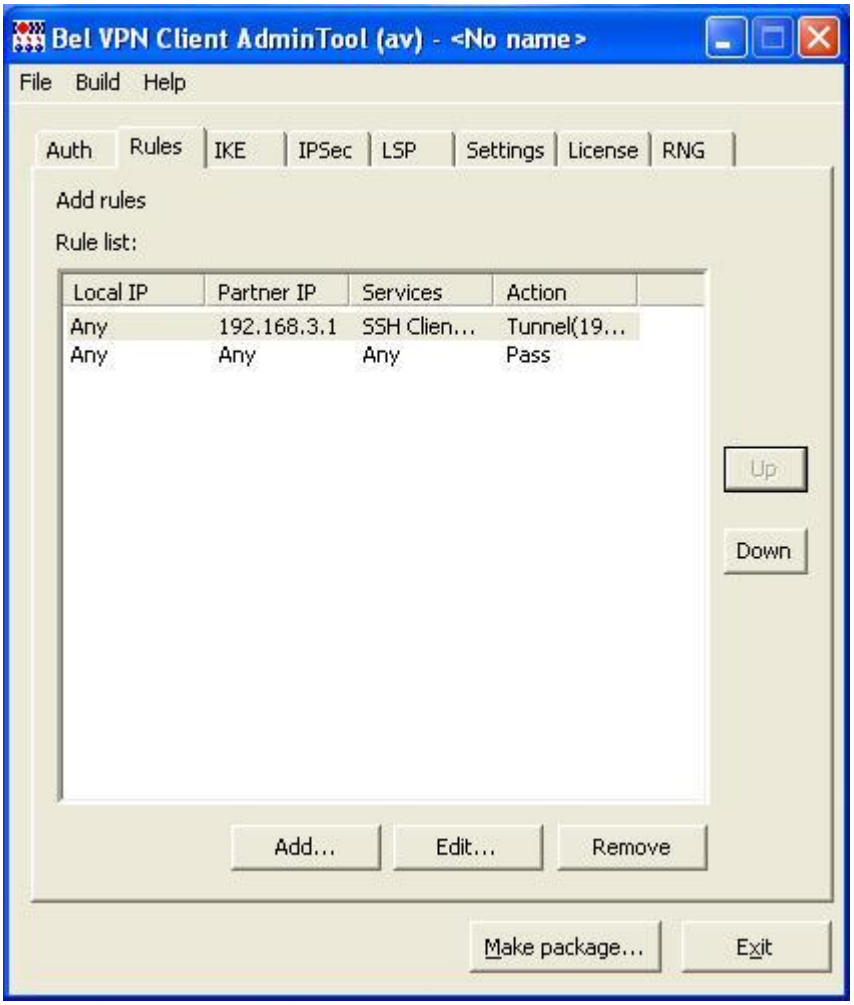

Рисунок 4

Во вкладке IPsec указываем значение группы MODP_1024 для параметра Group:

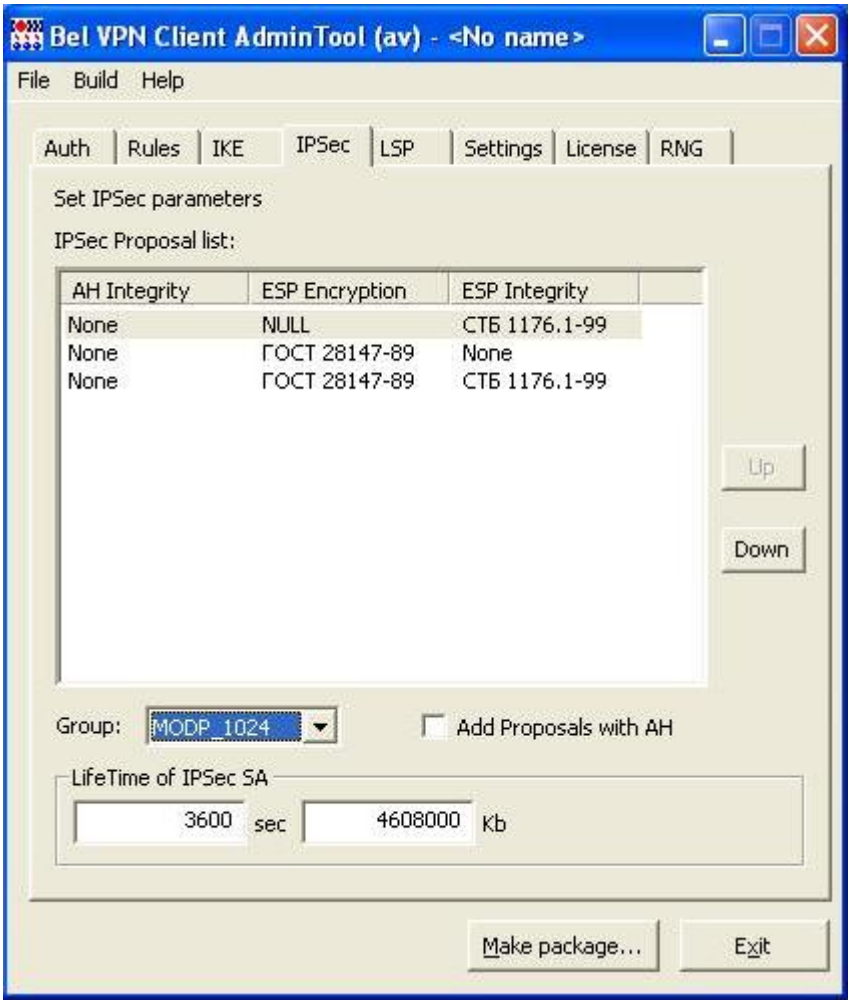

Рисунок 5

Переходим во вкладку LSP и нажимаем кнопку Refresh LSP. Созданная в предыдущих вкладках политика безопасности имеет следующий вид:

```
GlobalParameters (
  Title = "This LSP was automatically generated by Bel
VPN Client AdminTool (cp) at 2008.05.04 22:03:00"
  Version = "2.1"
  CRLHandlingMode = BEST_EFFORT
)
LDAPSettings (
  ResponseTimeout = 200
  HoldConnectTimeout = 60
  DropConnectTimeout = 5
)
IdentityEntry auth_identity_01(
  IPv4Address *= 192.168.3.2
)
AuthMethodPreshared auth_method_01(
  SharedIKESecret = "key_adm"
  LocalID = auth_identity_01
```



```
)
IKEParameters (
  DefaultPort = 500
  SendRetries = 5
  RetryTimeBase = 1
  RetryTimeMax = 30
  SACreationTimeMax = 60
  InitiatorSessionsMax = 30
  ResponderSessionsMax = 20
  BlacklogSessionsMax = 16
  BlacklogSessionsMin = 0
  BlacklogSilentSessions = 4
  BlacklogRelaxTime = 120
)
IKETransform ike_trf_01(
  LifetimeSeconds = 28800
  CipherAlg  *= "G2814789CPRO1-K256-CBC-65530"
  HashAlg *= "STB1176199-65530"
  GroupID *= MODP_1536
)
IKETransform ike_trf_02(
  LifetimeSeconds = 28800
  CipherAlg  *= "G2814789CPRO1-K256-CBC-65530"
  HashAlg *= "STB1176199-65530"
  GroupID *= MODP_1024
)
IKETransform ike_trf_03(
  LifetimeSeconds = 28800
  CipherAlg  *= "G2814789CPRO1-K256-CBC-65530"
  HashAlg *= "STB1176199-65530"
  GroupID *= MODP_768
)
ESPTransform esp_trf_01(
  IntegrityAlg *= "STB1176199-H96-HMAC-65530"
  CipherAlg *= "NULL"
  LifetimeSeconds = 3600
  LifetimeKilobytes = 4608000
)
ESPProposal esp_proposal_01(
  Transform *=esp_trf_01
)
ESPTransform esp_trf_02(
  CipherAlg *= "G2814789CPRO1-K256-CBC-250"
  LifetimeSeconds = 3600
  LifetimeKilobytes = 4608000
)
ESPProposal esp_proposal_02(
  Transform *=esp_trf_02
)
ESPTransform esp_trf_03(
  IntegrityAlg *= "STB1176199-H96-HMAC-65530"
```



```
  CipherAlg *= "G2814789CPRO1-K256-CBC-250"
  LifetimeSeconds = 3600
  LifetimeKilobytes = 4608000
)
ESPProposal esp_proposal_03(
  Transform *=esp_trf_03
)
IKERule ike_rule(
  DoNotUseDPD = FALSE
  DPDIdleDuration = 60
  DPDResponseDuration = 5
  DPDRetries = 3
  MainModeAuthMethod *= auth_method_01
  Transform *= ike_trf_01,ike_trf_02,ike_trf_03
  IKECFGRequestAddress = TRUE
  DoAutopass = TRUE
)
IPsecAction ipsec_action_01(
  TunnelingParameters *=
        TunnelEntry(
              PeerIPAddress = 192.168.3.1
        )
  ContainedProposals *=
(esp_proposal_01),(esp_proposal_02),(esp_proposal_03)
  GroupID *= MODP_1024,MODP_1536,MODP_768
  IKERule = ike_rule
)
FilterEntry local_entry_00_00(
  ProtocolID *= 6
)
FilterEntry remote_entry_00_00(
  IPAddress *= 192.168.3.1
  ProtocolID *= 6
  Port *= 80
)
FilteringRule filter_rule_00_00(
  LocalIPFilter *= local_entry_00_00
  PeerIPFilter *= remote_entry_00_00
  Action *= (ipsec_action_01)
  RefuseTCPPeerInit = FALSE
)
FilterEntry local_entry_00_01(
  ProtocolID *= 6
)
FilterEntry remote_entry_00_01(
  IPAddress *= 192.168.3.1
  ProtocolID *= 6
  Port *= 22
)
FilteringRule filter_rule_00_01(
  LocalIPFilter *= local_entry_00_01
```

```
  PeerIPFilter *= remote_entry_00_01
  Action *= (ipsec_action_01)
  RefuseTCPPeerInit = FALSE
)
FilterEntry local_entry_00_02(
  ProtocolID *= 17
)
FilterEntry remote_entry_00_02(
  IPAddress *= 192.168.3.1
  ProtocolID *= 17
  Port *= 22
)
FilteringRule filter_rule_00_02(
  LocalIPFilter *= local_entry_00_02
  PeerIPFilter *= remote_entry_00_02
  Action *= (ipsec_action_01)
  RefuseTCPPeerInit = FALSE
)
FilterEntry local_entry_01_00(
)
FilterEntry remote_entry_01_00(
)
FilteringRule filter_rule_01_00(
  LocalIPFilter *= local_entry_01_00
  PeerIPFilter *= remote_entry_01_00
  Action *= (PASS)
)
```

Во вкладке Settings укажите настройки протоколирования событий, а во вкладке License введите регистрационные данные на продукт Bel VPN Client с бланка Лицензии (см. «Bel VPN Client. Руководство администратора»).

После заполнения всех вкладок нажмите копку `Make package`, выберите тип инсталляции и сохраните инсталляционный файл на диске:

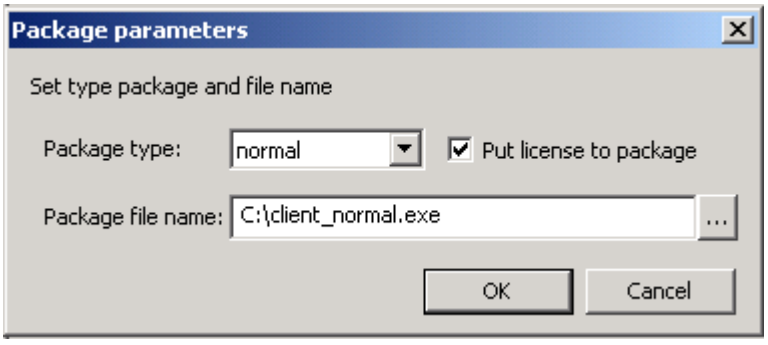

Рисунок 6

Установите на этом же компьютере Bel VPN Client, запустив созданный инсталляционный файл.

Таким образом, создана согласованная политика на шлюзе безопасности и клиенте, которая позволяет создавать *защищенный канал* для удаленной настройки шлюза при помощи консоли по протоколу SSH.

Дальнейшие действия по настройке шлюза описаны в документе «Cisco-like команды».

При изменении политики безопасности шлюза необходимо сохранять правило, созданное для его настройки.

3. Описание формата ini-файлов

3.1. Формат допустимых строк

1. Все `ini`-файлы, используемые в продукте Bel VPN Gate, могут содержать следующие типы строк:
   - *Строка имени секции* – строка, обозначающая начало новой секции переменных
   - *Строка описания переменной* – строка, содержащая имя и значение переменной
   - *Строка комментариев* – строка, содержащая комментарии или пустая строка.

Строка имени секции имеет следующий формат:

```
[SECTION_NAME]
```

Пробелы и символы табуляции, стоящие до открывающей и после закрывающей квадратных скобок, игнорируются. Именем секции считается строка, помещенная между квадратными скобками (любой символ является значимым). Допускается пустое имя секции.

```
[]
```

Строка описания переменной имеет следующий формат:

```
var_name = var_value
```

Пробелы и знаки табуляции, стоящие до и после `var_name и var_value` игнорируются.

Строка комментариев имеет следующий формат:

```
! comment string
```

Пробелы и знаки табуляции, стоящие до символа `!` игнорируются.

Пустая строка (не содержащая ничего, кроме пробелов и знаков табуляции) приравнивается к строке комментария.

Любая строка, не удовлетворяющая ни одному типу описанных строк, является ошибочной, и будет приводить к ошибке чтения `ini-файла`.

2. Повторяющиеся имена секций.

В файле допустимо многократное использование *строки имени секции* с одним и тем же именем секции. В этом случае каждая последующая секция считается продолжением предыдущих (*строки описания переменных* объединяются)

3. Повторяющиеся имена переменных.

В файле допустимо многократное использование *строки описания переменной* с одним и тем же именем переменной. В том случае, если эти строки принадлежат к одной секции - действительным считается значение, описанное последней строчкой (*строки описания переменных* накладываются).

4. Переменные, не принадлежащие ни одной из секций.

В файле допускается указание *строк описания переменных,* не принадлежащих ни одной секции (в начале файла идут строки переменных без задания имени секции). Этот случай эквивалентен заданию секции с пустым именем. Следующие ситуации эквивалентны:

```
File01.ini
 Var01=value01
 Var02=value02
 Vat03=valeue03

File02.ini
 []
 Var01=value01
 Var02=value02
 Vat03=valeue03
```

5. Перезапись `ini`-файла.

В процессе работы некоторые `ini`-файлы могут модернизироваться продуктом. При этом все комментарии и пустые строки будут сохранены.

- В случае повторяющихся имен секций подобные секции будут объединены в одну (первая дополняется переменными последующих). В этом случае комментарии, принадлежащие секциям, объединяются.
- В случае повторяющихся имен переменных (принадлежащих одной секции) будет оставлено только последнее ее описание. В этом случае комментарии, принадлежащие переменной, объединяются.

6. Принадлежность строк комментариев.

Любая *строка комментариев,* расположенная перед *строкой имени секции* или *строкой описания переменной,* считается принадлежащей этой строке.

4. Конвертор

В данной главе описана внутренняя логика работы конвертора VPN агента из Cisco-like конфигурации в Native-конфигурацию, как управлять конвертированием с помощью INI-файла, формирование имен структур при конвертировании, а также указан список сообщений, предупреждений и ошибок, которые могут быть посланы при протоколировании событий.

4.1. Основная логика работы

1. Конвертор выполнен в виде динамической библиотеки `s_converter.dll (Win32) / libs_converter.so (Solaris)`. Также используются некоторые вспомогательные файлы и агентские библиотеки.
2. Конвертор работает в рамках программы `cs_console`.
3. При выходе из конфигурационного режима, если в конфигурацию были внесены какие-то изменения, `cs_console` вызывает конвертор и передает ему внутреннее представление Cisco-конфигурации.
4. Во время работы конвертора используются настройки конвертора, описанные в разделе "Внутренние настройки консоли и конвертора". Некоторые из настроек могут редактироваться пользователем. В результатае работы конвертора формируется LSP в Native-формате.
5. Логика формирования имен структур в Native-конфигурации представлена в разделе "Формирование имен структур LSP при конвертировании".
6. Далее происходит попытка загрузки LSP в Native-формате в агента.
   - Если по каким-то причинам произошла ошибка при загрузке, Native-конфигурация пишется в файл `erroneous_lsp.txt`, расположенный в:

     `Windows` – в каталоге агента

     `Solaris и Linux` – в каталоге `/var/cspvpn`.
7. В конце работы выдается результат (успех/неуспех) обратно в `cs_console`.

4.2. Ограничения на конвертор

1. Поддерживается набор команд, определенный в документе «Cisco-like команды».
2. В Cisco используется примерно следующая логика работы с `access list`:

   `<interface_acl> -> <crypto_map_acl> -> <interface_acl>`,
   где `<interface_acl>` – access list в интерфейсе,
   а `<crypto_map_acl>` – access list в crypto map.

   В конверторе используется логика разворачивания access list-ов в сквозную модель правил. При этом возможны некоторые несоответствия и несовместимости (подробнее см."Описание обработки интерфейсов").

   - В правилах в access list в маске подсети допускается указание только непрерывной линейки из установленных битов в конце (т.е. 00…01…1, например 0.0.0.255 0.0.0.63 и т.п.). Не допускается разрывов в полях установленных и сброшенных битов (например, маски вида 0.255.0.255). В случае появления запрещенной маски, конвертация завершается с ошибкой [3.9].
3. Ряд ограничений на `ca trustpoint`:
   - enrollment игнорируется (только ручное задание сертификатов).
   - Читаются только CA-сертификаты, локальные сертификаты игнорируются.
     - Небольшое пояснение: в Cisco по команде `crypro ca certificate chain` показываются CA сертификаты и локальные сертификаты. Через эту команду все сертификаты можно посмотреть, удалить и ввести CA сертификаты. Однако, локальные сертификаты нельзя ввести таким образом (они будут неработоспособны без секретного ключа). В `cs_console` данная команда используется только для работы с CA сертификатами.
   - Под обозначением RSA-сертификатов (другие в Cisco не используются) могут использоваться RSA, ГОСТ и DSA-сертификаты.
   - В Cisco в пределах одного `trustpoint` могут вписываться только сертификаты из одной цепочки. В `cs_converter` допускаются любые CA сертификаты.
   - Задается строгое соответствие: RSA CA сертификат подписывает только RSA-сертификаты, ГОСТ CA сертификат подписывает только ГОСТ сертификаты, DSA CA сертификат подписывает только DSA-сертификаты.
   - Следует учитывать, что в конфигурации не задается точных критериев выбора локального сертификата (в терминах Native LSP задается USER_SPECIFIC_DATA). В связи с этим возможны ситуации, при которых не установится соединение, если присутствуют больше одного локального сертификата, подписанного разными CA.
     - Пример подобной ситуации: у партнера не прописана посылка Certificate Request, и партнер ожидает от агента конкретный сертификат (который действительно присутствует), но агент по своим критериям выбирает другой сертификат, который не подходит партнеру.
     - Как правило, таких проблем не возникает, если соблюдаются следующие условия:
       - У обоих партнеров прописана отсылка Certificate Request. По умолчанию конвертер именно так и делает. Cisco в большинстве случаев поступает также.
       - Не используется `Aggressive Mode` при работе с сертификатами (экзотический случай).

- У партнера должны быть явно указаны CA-сертификаты, которыми может быть подписан локальный сертификат агента. В Native LSP агента – атрибут `AcceptCredentialFrom` (cs_converter вписывает все CA-сертификаты, лежащие в базе). В Cisco – должен быть прописан подходящий `trustpoint`.

4. Ограничение на LDAP url: допускается только задание IP-адреса и, возможно, порта. Если задано DNS-name – данный url игнорируется.
5. Допускается только одно ISAKMP правило для одного IPSec-правила.
6. Если для данного crypto-map удалось подобрать несколько ISAKMP policy с разными Transform и методами аутентификации, то формируется одна IKERule, в которой пишутся ВСЕ трансформы и методы аутентификации, что приводит к несколько иной логике (т.е. теряется связь между трансформами и методами аутентификации).

Пример

#Фрагмент исходной конфигурации:

```
crypto isakmp policy 1
encr des
hash md5
authentication rsa-sig

crypto isakmp policy 2
encr 3des
hash sha
authentication pre-share
group 2

crypto isakmp policy 3
encr aes 128
hash md5
authentication rsa-sig
```

#Фрагмент Native-LSP (в ситуации, когда подходят все три policy):

```
AuthMethodRSASign auth_ca(
...
)
AuthMethodPreshared IKE_auth_key_192_168_11_110(
...
)
IKERule IKE_router_mc_fastethernet0_0_crypto_1(
    Transform* = IKETransform(
        CipherAlg   *= "DES-CBC"
        HashAlg     *= "MD5"
        GroupID     *= MODP_768
        LifetimeSeconds = 86400
    ),
    IKETransform(
        CipherAlg   *= "DES3-K168-CBC"
        HashAlg     *= "SHA1"
        GroupID     *= MODP_1024
        LifetimeSeconds = 86400
```

```
    ),
    IKETransform(
        CipherAlg   *= "AES-K128-CBC-7"
        HashAlg     *= "MD5"
        GroupID     *= MODP_768
        LifetimeSeconds = 86400
    )
    MainModeAuthMethod  *= auth_ca,
IKE_auth_key_192_168_11_110
    AggrModeAuthMethod  *= auth_ca,
IKE_auth_key_192_168_11_110
    DoAutopass          = TRUE
)
```

7. В Bel VPN Gate в фильтрах, в которых прописан локальный адрес ANY, в Native-LSP прописывается диапазон 0.0.0.0..255.255.255.255.
8. Если в crypto map прописаны несколько peer, каждый из которых аутентифицируется по preshared key, то используется следующий подход:
   - прописывается туннель и аутентификация для первого по счету peer
   - для остальных peer-ов проверяются preshared keys:
     - если preshared key совпадает с ключом для первого peer, то этот peer прописывается в качестве туннеля;
     - если preshared key не совпадает с ключом для первого peer, то для данного peer формируется отдельный `AuthMethodPreshared, IKERule и IPsecAction`. При этом в `IKERule` прописывается параметр:

       ```
       IKEPeerIPFilter* = FilterEntry(
                          IPAddress *= <IP-адрес peer> )
       ```

       В этом случае в `FilteringRule` для данной `crypto map` перечисляется список сформированных `IPsecAction`.

Пример подобного случая

```
#Фрагмент исходной конфигурации:
crypto isakmp key 1234 address 1.1.1.1
crypto isakmp key 5678 address 2.2.2.2
...
crypto map cmap 1 ipsec-isakmp
set peer 1.1.1.1
set peer 2.2.2.2
...
#Фрагмент Native-LSP:
AuthMethodPreshared IKE_auth_cs_key_1_1_1_1 (
    RemoteID = IdentityEntry(
        IPv4Address *= 1.1.1.1
    )
    SharedIKESecret = "cs_key_1_1_1_1"
)
IKERule IKE_cmap_1 (
   IKEPeerIPFilter* = FilterEntry(IPAddress *= 1.1.1.1)
...
```



```
    AggrModeAuthMethod  *= IKE_auth_cs_key_1_1_1_1
    MainModeAuthMethod  *= IKE_auth_cs_key_1_1_1_1
...
)
IPsecAction cmap_1 (
    TunnelingParameters *= TunnelEntry(
        PeerIPAddress = 1.1.1.1
...
    )
    IKERule = IKE_cmap_1
)
AuthMethodPreshared IKE_auth_cs_key_2_2_2_2 (
    RemoteID = IdentityEntry(
        IPv4Address *= 2.2.2.2
    )
    SharedIKESecret = "cs_key_2_2_2_2"
)
IKERule IKE_cmap_1_1 (
    IKEPeerIPFilter* = FilterEntry( IPAddress *= 2.2.2.2
)
...
    AggrModeAuthMethod  *= IKE_auth_cs_key_2_2_2_2
    MainModeAuthMethod  *= IKE_auth_cs_key_2_2_2_2
...
)
IPsecAction cmap_1_1 (
    TunnelingParameters *= TunnelEntry(
        PeerIPAddress = 2.2.2.2
...
    )
...
    IKERule = IKE_cmap_1_1
)
FilteringRule Filter_...(
...
    Action *= ( cmap_1 ), ( cmap_1_1 )
)
```

Следует учитывать, что подобная конфигурация приведет к тому, что работа со вторым peer будет возможна только в качестве ответчика. В качестве инициатора работа возможна только с первым peer.

Рекомендуется по возможности избегать таких ситуаций. Для этого, в случае указания в crypto map нескольких peers, следует либо использовать аутентификацию по сертификатам либо, в случае использования аутентификации на preshared keys, использовать одинаковый ключ для всех peers, перечисленных в одной crypto map.

В подобной ситуации выдается сообщение [2.10].

Если присутствует подобная конфигурация с несовпадающими preshared ключами и, кроме того, существует аутентификация на сертификатах; тогда к вышеперечисленным наборам `AuthMethodPreshared`, `IKERule` и `IPsecAction` добавится еще один, описывающий аутентификацию на сертификатах. При этом в `IPsecAction` будут прописаны все peers.


```
#Пример фрагмента LSP (отличия от предыдущего примера):
IKERule IKE_cmap_1_2 (
...
    AggrModeAuthMethod  *= auth_ca
    MainModeAuthMethod  *= auth_ca
...
)

IPsecAction cmap_1_2 (
    TunnelingParameters *= TunnelEntry(
        PeerIPAddress = 1.1.1.1
...
    ),
    TunnelEntry(
        PeerIPAddress = 2.2.2.2
...
    )
...
    IKERule = IKE_cmap_1_2
)
FilteringRule Filter_... (
...
    Action *= ( cmap_1 ), ( cmap_1_1 ), ( cmap_1_2 )
)
```

9. Существует специфический подход в случае, если в `crypto map set` присутствует несколько `crypto maps`, а в их `crypto-map-acls` существуют пересечения по адресам, причем в части правил присутствует `permit`, а в других правилах – `deny`. Подробнее логика конвертирования для данной ситуации описана в п.5 раздела "Описание обработки интерфейсов".
10. Возможен только симметричный маршрут, поэтому в интерфейсах используется только `access-group in`, без `access-group out`. Если `access-group out` задано, оно игнорируется с выдачей предупреждения.
11. Существуют особенности в настройке маршрутизации:
    - Если добавить из консоли `routing`, который уже присутствует в системной таблице маршрутизации, то он будет добавлен в текущую конфигурацию с диагностикой в файле лога.
    - При отгрузке сконвертированной конфигурации (по любой причине), из системной таблицы маршрутизации будут удалены все записи, добавленные из консоли, которые также могли существовавать и до запуска консоли (например, добавленные с помощью команды `route add`).
12. Существуют дополнительные команды, которые отсутствуют у Cisco:
    - команда `set pool` – задает IKE-CFG pool, привязанный к конкретной `crypto map`. Работает в конфигурационном режиме `crypto map` и `crypto dynamic-map`.
      - особый случай – команда `set pool <none>` – убирает для конкретной `crypto map` или `crypto dynamic-map` настройки IKE-CFG, которые, возможно, выставлены с помощью команды `crypto map client configuration address` или `crypto dynamic-map client configuration address` (они работают для `crypto map set`).

- команда `crypto dynamic-map client configuration address`. Работает аналогично команде `crypto map client configuration address`, но для `dynamic map set`.

13. Команды для задания ограничений по трафику и времени имеют больший диапазон, чем в Cisco:
    - `security-association lifetime kilobytes`: в Cisco `2560-536870912`, у нас – `1-4294967295`.
    - `security-association lifetime seconds`: в Cisco `120-86400`, у нас – `1-4294967295`.
    - `IKE lifetime (seconds)`: в Cisco 60-86400, у нас – `1-4294967295`.

Примечание: Cisco-like консоли как и в Cisco отсутствует возможность убрать ограничения по трафику и времени (unlimited).

14. Существуют особенности при настройке шлюза для работы с мобильным клиентом. Можно использовать один из двух подходов:
    - с точки зрения логики настройки Bel VPN Gate, в acl, привязанном к crypto dynamic map, в качестве remote-адресов для мобильных клиентов необходимо указывать any. Таким образом указывается, что допускается любой физический адрес мобильного клиента.
    - с точки зрения логики настройки Cisco, в acl, привязанном к crypto dynamic map, в качестве remote-адресов для мобильных клиентов указывается пул, из которого роутер раздает адреса мобильным клиентам. Таким образом указывается, что область действия этой crypto dynamic map распространяется только для мобильных клиентов из пула:

Пример

#Фрагмент конфигурации:

```
ip local pool p1 10.0.0.0 10.0.0.255
!
crypto ipsec transform-set ts1 esp-des esp-md5-hmac
 mode tunnel
!
ip access-list extended acl
 permit  ip  0.0.0.0 255.255.255.255 10.0.0.0 0.0.0.255
!
crypto dynamic-map   dmap 1
 match address acl
 set transform-set ts1
 set pool p1
!
crypto map cmap 1 ipsec-isakmp dynamic dmap
!
!
!
interface FastEthernet0/0
 ip address 10.0.15.100 255.255.0.0
 crypto map cmap
```

#Фрагмент сконвертированной LSP:

```
AddressPool p1
(
    IPAddresses *= 10.0.0.0..10.0.0.255
)
```



```
IPsecAction dmap_1
(
    TunnelingParameters *= TunnelEntry(
        DFHandling=COPY
    )
    ContainedProposals *= ( ESP_ts1 )
    IKERule = IKE_dmap_1
)

FilteringRule Filter_nil_acl_dmap_1
(
    LocalIPFilter *= FilterEntry(
                IPAddress *= 0.0.0.0..255.255.255.255 )
    PeerIPFilter  *= FilterEntry(
                IPAddress *= 10.0.0.0/24 )
    NetworkInterfaces *= "pcn0"
    Action *= ( dmap_1 )
)
```

- в сконвертированной LSP видно, что в структуре IPsecAction dmap_1 в атрибуте TunnelEntry отсутствует поле PeerIPAddress, то в качестве IKE-партнера для шлюза может выступать мобильный клиент с любым физическим адресом
- в тоже время, если мобильный клиент является пассивным IKE-CFG клиентом (IKECFGRequestAddress=FALSE – не инициирует посылку запроса на получение адреса из пула), то защищенное соединение построено не будет. Присылаемый мобильным клиентом его физический адрес в качестве identity QM не подходит под соответствующее FilteringRule и QM шлюзом отвергается
- для успешного создания соединения при такой конфигурации шлюза мобильный клиент должен выступать в качестве активного IKE-CFG клиента (IKECFGRequestAddress=TRUE). В этом случае в качестве identity QM используется выданный клиенту адрес из пула и соответствующее FilteringRule на шлюзе срабатывает

15. Если задается dynamic map без указания set peer (обычная ситуация), то формируются цепочки правил для всех возможных вариантов аутентификации. Например, если заданы несколько preshared keys для разных хостов, то будут сформированы правила для всех этих preshared keys и соответствующих им хостов.
    - если задается dynamic map с указанием set peer (экзотический, но допустимый вариант), то данная dynamic map конвертируется аналогично static map.
16. В TunnelingParameters прописывается значение DFHandling:
    - используется значение crypto ipsec df-bit для интерфейса, если оно присутствует в конфигурации.
    - в противном случае используется глобальное значение crypto ipsec df-bit.

4.3. Логика запуска конвертора

1. В базу локальных настроек добавляется признак источника загруженной LSP: из утилиты `lsp_mgr` или из `cs_converter` (в дальнейшем возможно расширение списка).
2. При входе в конфигурационный режим проверяется источник текущей LSP. Возможны следующие варианты:
   - LSP загружена из `cs_converter` (согласованные политики).
   - LSP загружена из другого источника или вообще не загружена (рассогласованные политики).
3. Кроме того, при старте проверяются следующие изменения:
   - добавление или удаление сертификата в базе локальных настроек
   - удаление preshared ключа, заданного в Cisco-like конфигурации
   - изменение состава сетевых интерфейсов в агенте (добавлены или удалены с помощью `if_mgr`)
   - изменение адреса сетевого интерфейса.

Во всех этих случаях данные изменения могут отразиться на LSP, формируемой с помощью конвертора. Поэтому, если зафиксировано одно или несколько упомянутых изменений, то данная ситуация также трактуется как рассогласование политик. При этом проверка загруженной LSP уже не делается.

Следует отметить, что проверка на данные ситуации делается только при старте консоли. При повторном входе в конфигурационный режим в рамках одной сессии считается, что данные изменения уже корректно отработаны. При этом проверка LSP делается при каждом входе в конфигурационный режим.

4. В файл `cs_conv.ini` добавляется новый параметр – режим синхронизации политик, который отвечает за логику работы консоли в случае второго варианта (рассогласование политик). См. описание настройки `policy_sync`.
   - Данный параметр влияет на конвертирование текущей конфигурации при входе в режим `configure terminal`.
   - Данный параметр влияет на включение и выключение инкрементальной конфигурации (для случая рассогласованной политики): если этот режим включен, то для некоторых команд (сейчас – `routing и SNMP traps`) формируется и загружается инкрементальная конфигурация немедленно после прописывания этой команды. Следует учитывать, что при выключенной инкрементальной настройке возможны побочные эффекты, связанные с командами редактирования маршрутизации (как минимум): возможно добавление неправильной команды с корректным синтаксисом (например, может быть добавлен маршрут через шлюз, к которому не определен маршрут). В этом случае команда добавится в Cisco-like конфигурацию, а затем при конвертировании этой конфигурации возникнет ошибка на стадии загрузки сконвертированной Native-LSP.
   - Подробнее логика работы данного параметра описана в таблице.

5. Логика запуска конвертора для разных вариантов:

Режим синхронизации политик	**Включен (значение по умолчанию)**	**Выключен**
Стартовая LSP загружена из `cs_converter`.	При входе в режиме `configure terminal` не происходит конвертирования. Инкрементальная настройка включена. Запуск конвертора осуществляется при выходе из режима `configure terminal` только в том случае, если были произведены какие-либо изменения в Cisco-like конфигурации.	
Стартовая LSP загружена из другого источника или не загружена вообще. Выдается сообщение в лог.[1.7].	При входе в конфигурационный режим делается попытка сконвертировать текущую Cisco-like конфигурацию. Далее логика различается в зависимости от того, удалось сконвертировать конфигурацию или нет: Если удалось сконвертировать: инкрементальная настройка включена предыдущая агентская конфигурация (если она есть) сохраняется в файл `non_cscons.lsp` (расположение файла см. в Примечании). Об этом выдается сообщение в лог [1.6]. Если по каким-либо причинам не удалось сохранить файл, то сообщается в лог [3.11]. запуск конвертора осуществляется при выходе из режима `configure terminal` только в том случае, если были произведены какие-либо изменения в Cisco-like конфигурации. Если не удалось сконвертировать, то ,выдается сообщение в лог [1.9]. Инкрементальное конфигу-рирование ВЫКЛЮЧЕНО, поскольку нет LSP, к которой можно было бы корректно приложить инкрементальную LSP (политики рассогласованы). При выходе из режима `configure terminal` вызывается конвертор только в том случае, если были произведены какие-либо измене-ния в Cisco-like конфигурации. Если изменений сделано не было, то данную конфигурацию не удастся сконвертировать, поэтому конвертор вызывать не нужно (сообщается в лог [2.8]).	При входе в режим `configure terminal` не происходит конвертирования. Инкрементальная настройка выключена. Выдается сообщение в лог [1.8]. При выходе из режима `configure terminal` вызывается конвертор вне зависимости от того, были внесены изменения в Cisco-like конфигурацию или нет. Если конвертирование прошло успешно, то предыдущая конфигурация агента (если она есть) сохраняется в файл `non_cscons.lsp lsp` (расположение файла см. в Примечании). Об этом выдается сообщение в лог [1.6]. Если по каким-либо причинам не удалось сохранить файл, то сообщается в лог [3.11].

Примечание:

расположение файла `non_cscons.lsp` зависит от ОС:

`Windows` – в каталоге агента

`Solaris и Linux` – в каталоге `/var/cspvpn`.

6. Если при выходе из конфигурационного режима происходит конвертирование конфигурации, и это конвертирование завершается с ошибкой; то на консоль выдается сообщение об ошибке: "`LSP conversion failed. You can use the "show load-message" command to obtain the additional information.`" ("Конвертирование LSP завершилось с ошибкой. Вы можете использовать команду "`show load-message`" для получения дополнительной информации").

4.4. Алгоритм работы конвертора

1. Подготовительный этап:
   - Инициализация лога.
   - Инициализация локальных настроек.
2. Написание служебной информации в комментариях:
   - Фраза `"This is automatically generated LSP"`.
   - Дата и время конвертации.
3. Создание заголовка конфигурации:
   - Служебная информация (версия и т.п.).
   - Настройки LDAP и CRL-processing.
   - Глобальные настройки лога.
4. Обработка интерфейсов. Подробнее см. "Описание обработки интерфейсов". При этом формируются правила фильтрации, в том числе и IPsec (APPLY).
5. Для APPLY правила происходит поиск подходящего ISAKMP правила:
   - Сначала делается попытка найти подходящее правило на `Preshared key`.
   - Также берется первое по счету правило `rsa-sig`.
6. Если подобрано правило на `Preshared key` – прописывается аутентификационная информация с соответствующим именем ключа.
   - Для `main mode LocalID` прописывается в зависимости от команды `crypto isakmp identity`:
     - Если `crypto isakmp identity address` (вариант по умолчанию), а также в случае `crypto isakmp identity dn` (вариант, неприменимый для `preshared keys`), `LocalID` в конфигурацию не пишется (обозначает – использовать локальный IP-адрес).
     - Если `crypto isakmp identity hostname`, пишется:
       `LocalID = IdentityEntry( KeyID *= "<local_id>" )`
       где `<local_id>` – представление в виде Hex-string полного DNS-адреса, составленного из локального `hostname`, заданного с помощью команды `hostname <…>` и доменного имени, заданного с помощью команды `ip domain name <…>`. Если доменное имя отсутствует, или в `hostname` присутствует хотя бы одна точка, `<local_id>` составляется только из `hostname`.
   - `RemoteID` прописывается по следующим правилам:
     - Если ключ привязан к IP-адресу с помощью команды `crypto isakmp key <…> address <…>`, то формируется правило с аутентификацией по данному ключу:

       ```
       AuthMethodPreshared <...> (
           [ LocalID  = IdentityEntry( KeyID *= "<local_id>" ) ] – если необходимо
           RemoteID = IdentityEntry( IPv4Address *= <peer_ip> )
           SharedIKESecret = <...>
       )
       ```

     - Если ключ привязан к `hostname` с помощью команды `crypto isakmp key <…> hostname <…>`, а `hostname` привязан к IP-адресу с помощью команды `ip host <hostname> <ip-addr>`, то формируется правило с аутентификацией по `hostname и IP-address`:

```
AuthMethodPreshared <...> (
    [ LocalID  = IdentityEntry( KeyID *= "<local_id>" ) ] –
    если необходимо
    RemoteID = IdentityEntry(
    IPv4Address *= <peer_ip> KeyID *= "<peer_id>" )
    SharedIKESecret = <...>
)
```

где `<peer_id>` – представление в виде Hex-string hostname-а из команды `crypto isakmp key <…> hostname <…>`.

- Если ключ привязан к `hostname` с помощью команды `crypto isakmp key <…> hostname <…>` и используется динамический `crypto map`, формируется правило с аутентификацией по `hostname`:

```
AuthMethodPreshared <...> (
    [ LocalID  = IdentityEntry( KeyID *= "<local_id>" ) ] –
    если необходимо
    RemoteID = IdentityEntry( KeyID *= "<peer_id>" )
    SharedIKESecret = <...>
)
```

- Если удается подобрать дополнительные IP-адреса и hostname, для которых задаются те же самые ключи, тогда эти IP-адреса и KeyID (из hostname-ов) добавляются в RemoteID.

7. Если подобрано правило `RSA sig` – прописывается правило аутентификации в виде:

```
{ AuthMethodRSASign | AuthMethodGostSign } auth_ca(
    LocalID        =  IdentityEntry( ... )
    [ RemoteID       =  IdentityEntry( ... ) ] |
    [ DoNotMapRemoteIDToCert = TRUE ]
    AcceptCredentialFrom      *= CertDescription(
       X509IssuerDN  *= "..."
       SerialNumber  =  "..."
       X509SubjectDN *= "..."
    )
    SendRequestMode =  { AUTO | NEVER | ALWAYS }
    SendCertMode    =  { AUTO | NEVER | ALWAYS }
)
ModeAuthMethod *= auth_ca
```

- `LocalID для main mode` формируется по следующим правилам:
  - Если `crypto isakmp identity hostname`, пишется:
    `LocalID =  IdentityEntry( FQDN* = USER_SPECIFIC_DATA )`
  - Если `crypto isakmp identity dn`:
    `LocalID =IdentityEntry( DistinguishedName* = USER_SPECIFIC_DATA )`
  - Если `crypto isakmp identity address`:
    `LocalID = IdentityEntry(IPv4Address* = USER_SPECIFIC_DATA)`
- `RemoteID` формируется по следующим правилам:
  - Если к crypto map привязана identity, в которой прописан один или несколько dn, пишется:

```
RemoteID = IdentityEntry (
  DistinguishedName* = CertDescription(Subject* ="<dn1>"),
                       CertDescription( Subject* ="<dn2>"),
...
  FQDN* = "<fqdn1>", "<fqdn2>" ...
)
```


- Если к crypto map не привязана `identity`, то пишется `DoNotMapRemoteIDToCert = TRUE`
  - данная логика предназначена для того, чтобы в качестве remote identity работал IP-адрес и в том случае, когда он отсутствует в сертификате (типичная ситуация). Это бывает полезно при использовании разных типов аутентификации в пределах одной конфигурации - на сертификатах и на preshared keys.

8. В зависимости от наличия команды `crypto isakmp keepalive` и ее параметров в `IKERule` прописываются настройки `DPD`:
   - Если команда `crypto isakmp keepalive` не задана (по умолчанию), прописывается `DoNotUseDPD = TRUE`.
   - Если команда задана, пишутся параметры:

     ```
     DPDIdleDuration = <secs>
     DPDResponseDuration = <retries>
     DPDRetries = <dpd_retries>
     ```

     где `<secs>` – первый аргумент команды;
     `<retries>` – второй аргумент команды;
     `<dpd_retries>` – параметр `dpd_retries` из файла `cs_conv.ini`. См. описание настройки `dpd_retries`.

     Внимание!!! Поведение по умолчанию отличается от настроек DPD в версии 2.0. Там всегда (вне зависимости от `crypto isakmp keepalive`) выставлялись настройки DPD по умолчанию, характерные для Native-LSP: `DPDIdleDuration = 60; DPDResponseDuration = 5; DPDRetries = 3`. Сейчас по умолчанию DPD выключен.

9. Для сочетания `crypto map` и интерфейса запоминается сформированное поле `Action` в структуре `FilteringRule`. Если данное сочетание снова встречается при обработке другого фильтра – сразу прописывается запомненная строка `Action`.
10. В конце конвертирования делается попытка загрузить сформированную конфигурацию.
    - Если конфигурацию не удалось загрузить, она сохраняется в файле `erroneous_lsp.txt` (с выдачей сообщения в лог). Файл расположен в:

      `Windows` – в каталоге агента.

      `Solaris и Linux` – в каталоге `/var/cspvpn`.

4.5. Внутренние настройки консоли и конвертора

1. Внутренние настройки конвертора хранятся в файле `cs_cons_reg.ini`, расположенном в каталоге агента.
2. Данный файл используется для хранения внутренних настроек консоли и конвертора. Он автоматически модифицируется при запуске консоли. Редактирование этого файла вручную не рекомендуется.
3. Если файл отсутствует на момент старта консоли, он автоматически создается.
4. Формат файла:
   - Обычный текстовый файл в кодировке ASCII. Используется кодирование окончания строк принятое для операционной системы (Windows/UNIX).
   - Пустые строки и строки, начинающиеся с `!` (восклицательный знак) игнорируются.
   - Файл состоит из секций. Каждая секция начинается с названия секции, заключенного в квадратные скобки.
5. В настоящее время присутствует одна секция: описание перекодировки интерфейсов из формата Cisco в Native формат агента:
   - Название секции: `[interface_list]`
   - Формат строк: <Native_interface_name> = <Cisco_interface_name>. Например: `I0 = 0/0`
   - Данная секция редактируется автоматически при старте консоли:
     - Если в Cisco-like конфигурации и в INI-файле не описан native interface (например, первый старт консоли или данный native interface был добавлен с помощью if_mgr между двумя стартами консоли), то этому интерфейсу присваивается свободное <Cisco_interface_name>. Этот интерфейс добавляется в Cisco-like конфигурацию и в INI-файл.
     - Если здесь описан native interface, который отсутствует в агенте (например, был удален с помощью if_mgr между двумя запусками консоли), то этот интерфейс удаляется как из INI-файла, так и из Cisco-like конфигурации.
     - Если в текущей Cisco-like конфигурации присутствует интерфейс, не описанный в INI-файле (нештатная ситуация), этот интерфейс удаляется из Cisco-like конфигурации.
     - Если в INI-файле присутствует интерфейс, которого нет в текущей Cisco-like конфигурации (нештатная ситуация), этот интерфейс удаляется из INI-файла.
6. В случае, если произошли какие-либо изменения, описанные в предыдущем пункте, то обновленный файл сохраняется. Если сохранение по тем или иным причинам не удалось, то в лог выдается сообщение об ошибке [3.12].

4.6. Управление конвертором с помощью INI-файла

1. Настройки конвертора хранятся в INI-файле `cs_conv.ini`, расположенном в каталоге агента.
2. INI-файл хранит служебную информацию, необходимую для конвертора, включающую в себя все пользовательские настройки.
3. Формат файла:
   - обычный текстовый файл в кодировке ASCII. Используется кодирование окончания строк, принятое для операционной системы (Windows/UNIX)
   - пустые строки и строки, начинающиеся с ! (восклицательный знак) игнорируются
   - файл состоит из нескольких секций. Каждая секция начинается с названия секции, заключенного в квадратные скобки. Например: `[interface_list]`.
4. Описание секций файла:
   - Описание отдельных глобальных настроек:
     - Название секции: `[global_settings]`
     - Формат строк:
       - `product_type = {SERVER | GATE}` – тип агента. Пользователю не следует менять данный параметр.
       - `ike_autopass = { on|off }` – включение/выключение прописывания ike autopass в конфигурации. По умолчанию – on. Пользователю редактировать данный параметр не рекомендуется.
       - `dpd_retries = {1 – 10}` – количество попыток проведения DPD-обмена. По умолчанию – 5. Пользователь может настраивать данный параметр для получения оптимального количества DPD-retries.
       - `tunnel_local_ip = {on | off}` – включение/выключение прописывания локального IP-адреса в TunnelEntry. По умолчанию – `off`. Пользователю редактировать данный параметр не рекомендуется: только при возникновении ситуаций, когда прописывание локального адреса необходимо (пока не выявлено).
       - `policy_sync = = { on | off }` – включение/выключение режима синхронизации политик. (См. п. 4 раздела "Логика запуска конвертора"). По умолчанию – `off`. Пользователь может по своему усмотрению выключить данный параметр, если для него удобнее соответствующее поведение консоли.
   - Описание перекодировки алгоритмов:
     - Название секции: `[algorithm_list]`
     - Формат строк: `<Generic_algorithm_name> = <Native_algorithm_name>`. В качестве `<Generic _algorithm_name>` могут использоваться следующие алгоритмы (регистр важен): `ike-hash-md5, ah-integrity-md5, esp-integrity-md5, esp-cipher-des, ike-cipher-des, ike-hash-sha1, ah-integrity-sha1, esp-integrity-sha1, esp-cipher-3des, esp-cipher-aes, esp-cipher-aes-192, esp-cipher-aes-256, esp-cipher-null, ike-cipher-3des, ike-cipher-aes, ike-cipher-aes-192, ike-cipher-aes-256.` `Например: ike-cipher-3des=DES3-K168-CBC`

- Редактирование пользователем данной секции, как правило, не требуется, но возможно при необходимости перенести перекодировку алгоритмов ГОСТ на другие алгоритмы, или для полного отказа от перекодировки ГОСТ на агенте.

- Описание логики работы с `Cert request` и посылки сертификатов в процессе IKE:
  - Название секции: `[auth_cert]`
  - Формат строк: `<param_name>=<param_val>`. Список названий `<param_name>`:
    - `send_request`. По умолчанию – ALWAYS.
    - `send_cert`. По умолчанию – ALWAYS.
  - Редактирование пользователем данной секции как правило не требуется, но возможно при необходимости изменить логику работы с Cert request и посылки сертификатов в процессе IKE.

5. По умолчанию в `cs_conv.ini` используется следующая подстановка: `MD5 и DES` заменяются на алгоритмы СТБ 1176.1-99 и ГОСТ 28147, соответственно.
6. Варианты файлов, поставляемых в составе продукта:

Пример INI-файла **cs_conv.ini** для Gate:

```
[global_settings]
ike_autopass    = on
dpd_retries     = 5
product_type    = GATE
tunnel_local_ip = off
policy_sync     = on

[algorithm_list]

! Original algorithms:

! ike-hash-md5        = MD5
! ah-integrity-md5    = MD5-H96-HMAC
! esp-integrity-md5   = MD5-H96-HMAC
! esp-cipher-des      = DES-CBC
! ike-cipher-des      = DES-CBC

! Replaced algorithms:

ike-hash-md5        = STB1176199-65530
ah-integrity-md5    = STB1176199-H96-HMAC-250
esp-integrity-md5   = STB1176199-H96-HMAC-65530
esp-cipher-des      = G2814789CPRO1-K256-CBC-250
ike-cipher-des      = G2814789CPRO1-K256-CBC-65530

ike-hash-sha1       = SHA1
ah-integrity-sha1   = SHA1-H96-HMAC
esp-integrity-sha1  = SHA1-H96-HMAC
esp-cipher-3des     = DES3-K168-CBC
ike-cipher-3des     = DES3-K168-CBC
```



```
esp-cipher-aes      = AES-K128-CBC-12
esp-cipher-aes-192  = AES-K192-CBC-12
esp-cipher-aes-256  = AES-K256-CBC-12
esp-cipher-null     = NULL
ike-cipher-aes      = AES-K128-CBC-7
ike-cipher-aes-192  = AES-K192-CBC-7
ike-cipher-aes-256  = AES-K256-CBC-7

[auth_cert]
send_request            = ALWAYS
send_cert               = ALWAYS
```


4.7. Сообщения в логе при конвертировании

При работе конвертора могут посылаться сообщения в Syslog.

Формат строк сообщений:

```
<Date_Time> <Level:> <Message>, где Level – INFO, Warning или
ERROR.
```

Пример сообщения:

```
Wed Oct 29 18:19:50 2003 INFO: LSP conversion complete.
Warnings: 2
```

Список сообщений, предупреждений и ошибок, выдаваемых в логе, представлен в таблице.

1. **Информационные сообщения**

	Сообщение	Комментарий
1.1	LSP conversion started	Начат процесс конвертирования
1.2	LSP conversion complete	Процесс конвертирования завершен успешно. Предупреждения не выдавались.
1.3	LSP conversion complete. Warnings: {1}	Процесс конвертирования завершен успешно. Выдано {1} предупреждений.
1.4	Host mode is enabled.	Включен Host-режим
1.5	File "{1}" opened for writing	Файл {1} открыт для записи конфигурации
1.6	Previous user-defined LSP saved in file "{1}"	Предыдущая пользовательская LSP сохранена в файле “{1}”
1.7	Non-synchronized policy detected. Policy type: <type> где <type> один из: DDP Drop All User-defined (Source: <source>), где <source> – Agent или Command-line utility.	Обнаружена несинхронизированная политика. Тип политики: <type>
1.8	Incremental policy loading disabled by policy_sync setting (file cs_conv.ini)	Инкрементальная политика отключена из-за настройки policy_sync (файл cs_conv.ini)
1.9	Incremental policy loading disabled due to policy synchronization fail	Инкрементальная политика отключена из-за того, что не удалось провести синхронизацию политик

2. **Предупреждения**

	Сообщение	Комментарий
2.1	LDAP url "{1}" ignored. IP address and port allowed only.	Введенный LDAP url {1} проигнорирован, поскольку допускаются только IP-адрес и порт.

	Сообщение	Комментарий
2.2	OUT access group in the interface "{1}" ignored. Only IN access group is used.	Проигнорирован access-group out в интерфейсе {1}, поскольку допускается только access-group in.
2.3	Only one interface is used while host mode is on. Other interfaces ignored.	При включенном Host-режиме допускается только один интерфейс. Остальные интерфейсы игнорируются.
2.4	Only one CA certificate imported. Other certs ignored.	Импортирован только первый по списку CA-сертификат. End-User сертификаты и оставшиеся CA-сертификаты проигнорированы.
2.5	Crypto map "{1}" contains several peers. Peer(s) "{2}" ignored due to authentication information mismatch.	В crypto map {1} прописаны несколько peer-ов. Peer(s) {2} проигнорированы из-за того, что для них не совпадает аутентификационная информация.
2.6	Crypto map(s) "{1}" contain transform sets with different encapsulation modes. Tunnel mode is used.	Crypto maps {1} содержат transform sets, в которых заданы разные encapsulation режимы. Используется туннельный режим.
2.7	Crypto map(s) "{1}" contain transform sets with different encapsulation modes. Transport mode is used.	Crypto maps {1} содержат transform sets, в которых заданы разные encapsulation режимы. Используется транспортный режим.
2.8	Incorrect config detected. Policy conversion ignored	Обнаружена некорректная политика. Конвертирование политики не делается.
2.9	Crypto map set(s) "{1}" contain static crypto map(s) with priorities lower than dynamic.	Crypto map set(s) {1} содержат статические crypto map(s) с приоритетом ниже, чем у динамических
2.10	Crypto map "{1}" contains several peers with different preshared keys. This is not recommended.	Crypto map {1} содержит несколько peers с разными preshared keys. Это не рекомендуемая ситуация. Подробнее см. п.8 для несовпадающих Preshared keys.

3. **Ошибки**

	Сообщение	Комментарий
3.1	Cannot read settings form INI file. Conversion failed.	Невозможно прочитать настройки из INI-файла.
3.2	LSP file "{1}" open for write failed.	Не удается открыть файл {1} на запись агентской LSP
3.3	No interfaces were found in the INI file. Configure interfaces or set host mode to proceed.	Не заданы интерфейсы в INI-файле при выключенном Host-режиме. Необходимо настроить интерфейсы или включить Host-режим.
3.4	No interfaces were found in the configuration.	В импортируемой конфигурации не заданы интерфейсы. Конвертирование не имеет смысла.
3.5	Interface "{1}" not found in the INI file. Conversion aborted.	Интерфейс {1} не задан в INI-файле. Конвертирование остановлено.
3.6	Certificate parse failed	Не удалось разобрать введенный сертификат.

	Сообщение	Комментарий
3.7	Could not convert crypto map “{1}”. Reason: <Reason> где <Reason>: There is no isakmp policy. There is no CA or appropriate preshared key. Also isakmp policy can have wrong type (rsa-sig or pre-share). There is no peer. There are no transform sets. Crypto map is incomplete. Unknown.	Невозможно сконвертировать crypto map “{1}”. Причина: <Причина> где <Причина> одна из: Отсутствует isakmp policy. Отсутствует CA или подходящий Preshared Key, либо isakmp policy неправильного типа (rsa-sig или pre-share). Отсутствует peer. Отсутствуют transform sets Crypto map неполная (не хватает crypto ACL, transform set или peer). Неизвестная причина.
3.8	LSP load failed	Не удалось загрузить сформированную LSP
3.9	Unsupported network wildcard "{1}"	Не поддерживается данный формат маски подсети
3.10	LSP conversion failed	Произошла некоторая невыясненная ошибка
3.11	Could not save previous user-defined LSP in file "{1}"	Не удалось сохранить предыдущую пользовательскую LSP в файл {1}
3.12	Could not save internal settings in file "{1}"	Не удалось сохранить внутренние настройки в файл {1}

4.8. Описание обработки интерфейсов

1. Из интерфейса читается `access list`, прописанный в команде `ip access-group <access_list> in` (режим настройки интерфейса).
   - Далее для простоты такой `access list` будет указываться как `acl-in`.
   - Если такая команда не вводилась, то считается, что прописан `access list` с неявным правилом `Pass All` (на интерфейсе).
   - Поскольку в данном `access list` прописывается условно входящий трафик, поле `Source` транслируется в поле `PeerIPFilter`, а поле `Destination` – `LocalIPFilter`.
   - В поле `NetworkInterfaces` структуры `FilteringRule` прописывается внутреннее ("агентское") имя текущего интерфейса (см. Описание перекодировки алгоритмов).
   - Если в данной команде стоит ссылка на `access list`, то в конце данного `access list` предполагается неявное правило `Drop All`.

   Пример

   ```
   Interface FastEthernet0/0
   ip address 1.1.1.1 255.255.0.0
   ip access-group acl-in in
   ...
   Exit
   ```

2. Из интерфейса последовательно читаются `crypto maps` из `crypto map set`, прописанного в команде `crypto map <crypto_map>` (конфигурационный режим интерфейса).
   - Из описания `crypto map` читается `access list`, прописанный в команде `match address <access_list>` (конфигурационный режим `crypto map`).
   - Далее для простоты такой `access list` будет указываться как `crypto-map-acl`.
   - Если такой `access-list` не прописан, то считается, что прописан `access list` с неявным правилом `Pass All`.
   - Поскольку в `crypto-map-acl` прописывается условно выходящий трафик, трансляция адресов производится зеркально по отношению к `acl-in`.

   Пример `(для интерфейса и crypto map прописываются фактически одинаковые access lists):`

   ```
   ip access-list ex acl-in
   permit udp 1.1.1.1 0.0.0.0 2.2.2.2 0.0.0.0
   exit

   ip access-list ex crypto-map-acl
   permit udp 2.2.2.2 0.0.0.0 1.1.1.1 0.0.0.0
   exit

   crypto map cr-map 1 ipsec-isakmp
   ...
   match address crypto-map-acl
   exit
   ```



```
Interface FastEthernet0/0
...
ip access-group acl-in in
crypto map cr-map
...
exit
```

3. Если никакой `crypto map` не привязан к интерфейсу:
   - Происходит однозначное перекодирование из `acl-in` в Native фильтры: `deny -> DROP, permit -> PASS`.

Пример

```
# Cisco-like конфигурация:
!...
ip access-list ex acl-1
deny udp 1.1.1.1 0.0.0.0 1.1.1.2 0.0.0.0
permit 1 1.1.1.1 0.0.0.0 1.1.1.2 0.0.0.0
exit
!
Interface FastEthernet0/0
ip address 1.1.1.2 255.255.0.0
ip access-group acl-1 in
exit
!...
```

```
# Native-LSP конфигурация:
...

FilteringRule acl_1
(
    LocalIPFilter *= FilterEntry( IPAddress *= 1.1.1.2
ProtocolID *= 17 )
    PeerIPFilter  *= FilterEntry( IPAddress *= 1.1.1.1
ProtocolID *= 17 )
    NetworkInterfaces *= "if0"
    Action *= ( DROP )
)

FilteringRule acl_1_1
(
    LocalIPFilter *= FilterEntry( IPAddress *= 1.1.1.2
ProtocolID *= 1 )
    PeerIPFilter  *= FilterEntry( IPAddress *= 1.1.1.1
ProtocolID *= 1 )
    NetworkInterfaces *= "if0"
    Action *= ( PASS )
)

FilteringRule acl_1_2
(
```

```
    LocalIPFilter *= FilterEntry( IPAddress *=
0.0.0.0..255.255.255.255 )
    PeerIPFilter  *= FilterEntry( IPAddress *=
0.0.0.0..255.255.255.255 )
    Action *= ( DROP )
    NetworkInterfaces *= "if0"
)
# ...
```

4. Если `crypto map` присутствует, то происходит чтение правил в `acl-in`:
   - Правило `deny` напрямую перекодируется в `DROP`.
   - В случае правила `permit` делается проход по `crypto-map-acl`:
     - Берется правило из `crypto-map-acl`. Сравнивается адресная информация из правила `acl-in` с адресной информацией в правиле `crypto-map-acl` с учетом смены `source` и `destination` (например: `1.1.1.0 0.0.0.255 range 10 20 2.2.2.0 0.0.0.255 -> 2.2.2.0 0.0.0.255 1.1.1.0 0.0.0.255 range 10 20`). Делается попытка определить пересечение этих подмножеств адресов (частными случаями пересечения являются также полное совпадение и включение одного из подмножеств в другое). В случае удачного сравнения формируется адрес `work_address`, содержащий в себе пересечение подмножеств адресов. Далее, для определенности, предполагается, что в `work_address` используется порядок `source/destination`, как в `crypto-map-acl`.

Примеры

acl-in address	crypto-map-acl address	work_address
tcp host 1.1.1.1 any	ip any any	tcp any host 1.1.1.1
udp 1.1.1.0 0.0.0.255 eq 10 2.2.2.0 0.0.0.255 range 10 50	udp host 1.1.1.1 eq 10 host 2.2.2.2 range 20 30	udp host 1.1.1.1 eq 10 host 2.2.2.2 range 20 30
udp host 1.1.1.1 host 2.2.2.2	udp host 1.1.1.1 host 2.2.2.2	udp host 1.1.1.1 host 2.2.2.2

     - Если не удалось определить пересечение адресов (иначе говоря, нулевое пересечение); тогда правило из `crypto-map-acl` игнорируется.
     - Если удалось определить пересечение адресов и сформировать `work_address`, прописывается правило с адресной информацией из `work_address`:
       - Если в правиле `crypto-map-acl` прописан `deny -> правило PASS`.
       - Если в правиле `crypto-map-acl` прописан `permit -> правило APPLY (IPSec)`. При этом пишутся параметры из данного `crypto map`.
   - В конце прохода по `crypto map` прописывается правило `PASS` с адресной информацией из правила из `acl-in`.
5. В случае, если в `crypto map set` присутствует ссылка на `dynamic template set` (задается командами `crypto dynamic map`), в котором есть несколько `dynamic crypto maps`, в `crypto-map-acls` которых существуют пересечения по адресам, в `FilteringRule` происходит объединение правил.

Пример

#Фрагмент Cisco-like конфигурации:

```
ip access-list extended a1
 permit  icmp  192.168.101.0 0.0.0.255 10.0.12.240
0.0.0.15
!
crypto dynamic-map dmap 1
 match address a1

...

crypto dynamic-map dmap 2
 match address a1

...
crypto map cmap 1 ipsec-isakmp dynamic dmap

interface FastEthernet0/0
 crypto map cmap
```

#Фрагмент Native-LSP:

```
FilteringRule Filter_nil_acl_dmap_1
(
    LocalIPFilter *= FilterEntry(
        IPAddress *= 192.168.101.0/24 ProtocolID *= 1 )
    PeerIPFilter  *= FilterEntry(
        IPAddress *= 10.0.12.240/28 ProtocolID *= 1 )
    NetworkInterfaces *= "I0"
    Action *= ( dmap_1 ), ( dmap_2 )
)
```

- Объединение правил для статических `crypto maps` не производится (ни между разными статическими `crypto maps`, ни между статическими и динамическими `crypto maps`).
- В случае, если статическая `crypto map` имеет приоритет ниже, чем динамическая, то могут возникать логические неувязки. Настоятельно рекомендуется давать статическим `crypto maps` приоритет выше, чем динамическим. Следует отметить, что в документации Cisco также присутствует эта рекомендация.
  - Если данная рекомендация не выполнена – выдается предупреждение [2.9].
- Не производится объединение правил для динамических `crypto maps`, которые входят в разные `dynamic template sets`, которые в свою очередь входят в один `crypto map set`.

Пример

#Фрагмент Cisco-like конфигурации:

```
ip access-list extended a1
 permit  icmp  192.168.101.0 0.0.0.255 10.0.12.240
0.0.0.15
!
crypto dynamic-map dmap1 1
match address a1
...
crypto dynamic-map dmap1 2
match address a1
...
crypto dynamic-map dmap2 1
match address a1
...
crypto map cmap 1 ipsec-isakmp dynamic dmap1

crypto map cmap 2 ipsec-isakmp dynamic dmap2

interface FastEthernet0/0
 crypto map cmap
```

#Фрагмент Native-LSP
(dmap2 не попала в конфигурацию, поскольку объединение правил для нее не выполнялось, а сформированный фильтр не был прописан, поскольку полностью совпал с фильтром, который был прописан ранее; подробнее см. ниже):

```
FilteringRule Filter_nil_acl_dmap1_1
(
    LocalIPFilter *= FilterEntry(
        IPAddress *= 192.168.101.0/24 ProtocolID *= 1 )
    PeerIPFilter  *= FilterEntry(
        IPAddress *= 10.0.12.240/28 ProtocolID *= 1 )
    NetworkInterfaces *= "I0"
    Action *= ( dmap1_1 ), ( dmap1_2 )
)
```

- В случае, если в `dynamic template set` существует пересечение по адресам правил, в которых для одних `dynamic templates` прописаны правила `permit`, а для других – `deny`, в `FilteringRule` прописывается правило вида `(PASS), (Action1), …, (ActionN).`

Пример

#Фрагмент Cisco-like конфигурации:

```
ip access-list extended a1
 permit  icmp  192.168.101.0 0.0.0.255 10.0.12.240
0.0.0.15
!
ip access-list extended a2
```



```
 deny  icmp  192.168.101.0 0.0.0.255 10.0.12.240 0.0.0.15
!
crypto dynamic-map dmap 1
 match address a1
...
crypto dynamic-map dmap 2
 match address a2
...
crypto dynamic-map dmap 3
 match address a1
...
crypto map cmap 1 ipsec-isakmp dynamic dmap
...
interface FastEthernet0/0
 crypto map cmap
```

#Фрагмент Native-LSP:

```
FilteringRule Filter_nil_acl_dmap_1
(
    LocalIPFilter *= FilterEntry(
       IPAddress *= 192.168.101.0/24 ProtocolID *= 1 )
    PeerIPFilter  *= FilterEntry(
       IPAddress *= 10.0.12.240/28 ProtocolID *= 1 )
    NetworkInterfaces *= "I0"
    Action *= ( PASS ), ( dmap_1 ), ( dmap_3 )
)
```

- Логика формирования данных фильтров может существенно отличаться от логики Cisco.
- В данном примере продемонстрирован особый прием: специально для прописывания `PASS`-правила сделан `crypto dynamic-map dmap 2` (на самом деле приоритет этого `dynamic map` в данном конкретном случае не важен), в котором нет ничего, кроме связи с `ACL`, состоящим из `deny`-правила (правил): отсутствуют `transform sets` и т.п. Следует отметить, что данный способ может использоваться только с агентом, и неприменим на реальных устройствах Cisco.
- Данная логика действует только на явно прописанные `deny`-правила. Для неявных правил `deny ip any any`, которые предполагаются в конце каждого `access list`, никаких объединений правил не делается.

Например, `если из предыдущего примера убрать dmap 2:`

```
ip access-list extended a1
 permit  icmp  192.168.101.0 0.0.0.255 10.0.12.240
0.0.0.15
!
crypto dynamic-map dmap 1
 match address a1
...
crypto dynamic-map dmap 3
 match address a1
...
```



```
crypto map cmap 1 ipsec-isakmp dynamic dmap
...
interface FastEthernet0/0
 crypto map cmap
```

#Фрагмент Native-LSP будет уже без PASS-правила:

```
FilteringRule Filter_nil_acl_dmap_1
(
    LocalIPFilter *= FilterEntry(
         IPAddress *= 192.168.101.0/24 ProtocolID *= 1 )
    PeerIPFilter  *= FilterEntry(
         IPAddress *= 10.0.12.240/28 ProtocolID *= 1 )
    NetworkInterfaces *= "I0"
    Action *= ( dmap_1 ), ( dmap_3 )
)
```

6. При формировании `FilteringRule` дополнительно соблюдается следующее правило: для сервера в фильтрах, в которых прописан локальный адрес `ANY`, в Native-LSP прописывается адрес `LOCAL_IP_ADDRESSES`. Для Bel VPN Gate – пишется диапазон `0.0.0.0..255.255.255.255`.
7. Происходит проверка: нужно ли прописывать данный фильтр. Если этот фильтр совпадает или полностью включается в один из предыдущих фильтров, прописанных для данного интерфейса, тогда этот фильтр не прописывается в LSP.

Пример

Cisco-like конфигурация:

```
!...
ip access-list ex crypto-acl-1
deny udp 1.1.1.1 0.0.0.0 eq 500 2.2.2.2 0.0.0.0 eq 500
permit 1 1.1.1.1 0.0.0.0 2.2.2.2 0.0.0.0
exit
!
crypto map crypto-map1 1 ipsec-isakmp
set peer 2.2.2.2
set transform-set transform-1
match address crypto-acl-1
exit
!
Interface FastEthernet0/0
ip address 1.1.1.1 255.255.0.0
crypto map crypto-map1
exit
!...
```

Native LSP

```
 ...
FilteringRule Filter_nil_acl_crypto_map1_1
(
    LocalIPFilter *= FilterEntry( IPAddress *= 1.1.1.1
ProtocolID *= 17 Port *= 500 )
    PeerIPFilter  *= FilterEntry( IPAddress *= 2.2.2.2
ProtocolID *= 17 Port *= 500 )
```

```
    NetworkInterfaces *= "if0"
    Action *= ( PASS )
)

# .IKE & IPsec parameters

IPsecAction crypto_map1_1
(
    TunnelingParameters *= TunnelEntry(
        LocalIPAddress = 1.1.1.1
        PeerIPAddress = 2.2.2.2
        DFHandling=...
    )
    ContainedProposals *= ( ... )
    IKERule = ...
)

FilteringRule Filter_nil_acl_crypto_map1_1_1
(
    LocalIPFilter *= FilterEntry( IPAddress *= 1.1.1.1
ProtocolID *= 1 )
    PeerIPFilter  *= FilterEntry( IPAddress *= 2.2.2.2
ProtocolID *= 1 )
    NetworkInterfaces *= "if0"
    Action *= ( crypto_map1_1 )
)

FilteringRule Filter_nil_acl
(
    LocalIPFilter *= FilterEntry( IPAddress *=
0.0.0.0..255.255.255.255 )
    PeerIPFilter  *= FilterEntry( IPAddress *=
0.0.0.0..255.255.255.255 )
    NetworkInterfaces *= "if0"
    Action *= ( PASS )
)
# ...
```


4.9. Формирование имен структур LSP при конвертировании

1. При конвертировании Cisco-like конфигурации в LSP конфигурацию имена структур LSP формируются из имен и индексов объектов Cisco-like конфигурации. При этом следует учитывать ряд ограничений:
   - В объектах Cisco-like конфигурации разных типов могут использоваться одинаковые имена. В LSP имя объекта должно быть уникальным.
   - Могут использоваться цифровые индексы. В LSP требуется задавать идентификаторы, начинающиеся с буквы.
   - Как правило, синтаксис Cisco-like имен более свободный (например допускаются символы, которые нельзя использовать в идентификаторах LSP).
   - В некоторых случаях требуется формировать имя структуры LSP из группы объектов Cisco-like конфигурации.
   - Один объект Cisco-like конфигурации (или группа объектов) может порождать несколько LSP объектов (каждый из которых должен обладать уникальным именем).
2. Общие сведения по формированию имен:
   - Сначала готовится прототип имени объекта. Для этого прототипа нет каких-то специальных требований: например это может быть имя объекта Cisco-like конфигурации, константная строка, сочетание префикса и имен нескольких объектов и т.п.
   - Далее производится нормализация имени:
     - Все символы, кроме букв латинского алфавита и цифр преобразуются к символу подчеркивания.
     - Если имя начинается с цифры, перед ним ставится буква n.
   - Далее производится поиск полученного имени среди уже сформированных (для обеспечения уникальности):
     - Если имя не найдено, считаем его окончательно сформированным.
     - Если имя найдено, добавляем к нему последовательно суффиксы _1, _2 и т.д. до тех пор, пока не будет найдено имя, которое еще не использовалось.
   - Полученное имя записывается в конфигурацию и запоминается для того, чтобы оно не было использовано для другого объекта.
3. Далее описываются конкретные правила формирования прототипов имен объектов:

Имя структуры	Вариант использования	Правило формирования	Примеры
FilteringRule	Фильтр (без IPsec)	Если на интерфейсе отсутствует фильтрующий ACL, то используется слово "Filter_nil_acl".	Filter_nil_acl
		Если на интерфейсе присутствует фильтрующий ACL, заданный командой access-list, то производится конкатенация (соединение) префикса "Filter" и числового access-list-number из команды access-list.	Filter_101
		Если на интерфейсе присутствует фильтрующий ACL, заданный командой ip access-list – имя ACL, заданное в команде ip access-list.	Filter_acl5
	IPsec, заданный с помощью статической crypto map	Формируется конкатенацией имени FilteringRule без IPSec (см. предыдущий пункт), знака подчеркивания, имени crypto map, знака подчеркивания, индекса crypto map. Примечание: в случае, если в правиле задано несколько правил, имя FilteringRule формируется по первому правилу.	Filter_acl10_cmap_1 Filter_nil_acl_cmap_12
	IPsec, заданный с помощью динамической crypto map	Формируется аналогично, как для статической crypto map. Отличие в том, что вместо имени crypto map используется имя dynamic crypto map template (задается в команде crypto dynamic-map).	Filter_acl5_dmap_2
IKETransform		Конкатенация префикса "IKETransform_" и индекса ISAKMP policy (команда crypto isakmp policy).	IKETransform_10
AHProposal		Конкатенация "AH_" и имени transform-set (команда crypto ipsec transform-set)	AH_trset1
ESPProposal		Конкатенация "ESP_" и имени transform-set (команда crypto ipsec transform-set)	ESP_trset1

Имя структуры	Вариант использования	Правило формирования	Примеры
IKERule	Статическая crypto map	Конкатенация "IKE_", имени crypto map, знака подчеркивания, индекса crypto map.	IKE_cmap_1
	Динамическая crypto map	Формируется аналогично, как для статической crypto map. Отличие в том, что вместо имени crypto map используется имя dynamic crypto map template (задается в команде crypto dynamic-map).	IKE_dmap_2
AuthMethodRSASign AuthMethodDSSSign AuthMethodGOSTSign		auth_ca	auth_ca
AuthMethodPreshared		Конкатенация "IKE_auth_" и имени ключа (как он кладется в базу). Имя ключа формируется как cs_key_<ip_addr> или cs_key_<hostname> (точки заменяются на знак подчеркивания).	IKE_auth_cs_key_192_168_1_2 IKE_auth_cs_key_host1_company_com
CertDescription		ca	ca
IPsecAction	Статическая crypto map	Конкатенация имени crypto map, знака подчеркивания, индекса crypto map.	cmap_1
	Динамическая crypto map	Формируется аналогично, как для статической crypto map. Отличие в том, что вместо имени crypto map используется имя dynamic crypto map template (задается в команде crypto dynamic-map).	dmap_1
AddressPool		Имя pool (команда ip local pool)	pool1

5. Создание локального сертификата при использовании "AvCrypt ver. 5.1" (BY.ЮСКИ.09000-02)

Создание локального сертификата для шлюза безопасности можно осуществить с использованием утилиты `cryptocont`, созданной компанией “АВЕСТ”, и которая входит в состав дистрибутива Продукта Bel VPN Gate. Утилита используются для создания ключевой пары, запроса на локальный сертификат, создания контейнера и др. После инсталляции Продукта утилита размещена в каталоге `/opt/Avest/bin`.

Создание ключевой пары и формирование запроса на локальный сертификат шлюза безопасности выполняются администратором безопасности на программно-аппаратном комплексе Bel VPN Gate. Опишем все эти действия подробно.

Шаг1: Создайте контейнер, содержащий личный ключ, используя утилиту `cryptocont`, созданную компанией «АВЕСТ».

Выполните команду:

```
cryptocont n –n=<Container> [-p=<Password>]
[-y=<SysRandomSource>] [-r=<RandomFile>]
[-key_alg=<KeyAlgOid>] [–u]
```

где

`Container` – имя контейнера

`Password` – пароль к контейнеру, может отсутствовать, в этом случае пользователю предлагается дважды ввести пароль с клавиатуры

`SysRandomSource` – системный источник энтропии используемой при генерации случайности. Для Windows параметр указывает код типа используемого криптопровайдера (для криптопровайдера компании «АВЕСТ» указывается код «420» или «421», для криптопровайдеров других производителей – код «1»), для linux и solaris параметр игнорируется.

`RandomFile` – имя файла, используемого при генерации случайности. Файл должен быть не пуст, содержимое файла свыше 64 килобайт игнорируется.

`KeyAlgOid` – OID алгоритма ключа ЭЦП. Возможные значения (без кавычек):
`"1.3.6.1.4.1.12656.1.38"` или `"bds"` - ЭЦП согласно СТБ 1176.2-99
`"1.3.6.1.4.1.12656.1.35"` или `"bdspro"` - ЭЦП согласно СТБ 1176.2-99 с предварительным хэшированием
Значение по умолчанию – `"bds"`

`-u` – неинтерактивный режим генерации случайности.

Шаг2: Создайте запрос на сертификат, содержащий открытый ключ ЭЦП СТБ 1176.2-99 и экспортируйте запрос в файл используя утилиту `cryptocont`. Открытый ключ вычисляется на основе личного ключа, хранящегося в указанном контейнере.

Выполните команду:

```
cryptocont r {-f=<RequestFileName> -s=<SubjectName> -
c=<Country> [-k=<KeyUsage>] –n=<ContainerName> [-
p=<Password>]} {-i=IniFile}
```

где

`ContainerName` – имя контейнера, содержащего личный ключ ЭЦП СТБ 1176.2-99.

`Password` – пароль к контейнеру, может отсутствовать, в этом случае пароль вводится с клавиатуры.

`RequestFileName` – имя создаваемого файла запроса.

`SubjectName` - имя абонента.

`Country` – идентификатор страны абонента, например BY.

`KeyUsage` – область применения ключа согласно X.509, этот параметр рекомендуется не использовать – будет применено значение по умолчанию – “100000000” (ЭЦП).

Если указан параметр `-i`, данные берутся из .ini-файла `IniFile`, раздел request, поля filename, subject, keyusage, country, а также раздел container, поля name, pin.

пример `ini` файла:

```
[container]
name=cont1
pin=12345678

[request]
filename=req.req
subject=test subject
country=BY
keyusage=100010000
```

Шаг3: Отправьте созданный запрос доступным вам способом на сервер доверенного Удостоверяющего Центра, где по данному запросу будет создан локальный сертификат. В качестве УЦ может использоваться программа «Центр цифровых сертификатов Авест».

Шаг4: Получите из Удостоверяющего Центра локальный сертификат, цепочку сертификатов издателя и списки отозванных сертификатов в виде файлов и доставьте их на шлюз безопасности.

Шаг5: Зарегистрируйте в базе Продукта Bel VPN Gate сертификаты издателей и локальный сертификат, а также списки отозванных сертификатов, используя утилиту cert_mgr из состава Продукта. Такая регистрация описана в документе “Специализированные команды”.

5.1. Утилита cryptocont.exe

Формат вызова:

```
cryptocont <команда> <параметры>
```

Возможные команды:

Проверка контейнера

Производится проверка существования контейнера, и его пароль, если он указан.

```
cryptocont x –n=<Container> [-p=<Password>]
```

`Container` - имя контейнера

`Password` - пароль к контейнеру, может отсутствовать, в этом случае проверка пароля не производится.

Возможные коды ошибок:

AVCN_CONTAINER_NOT_FOUND – контейнер с указанным именем не существует.

AVCN_INVALID_PASSWORD - указан неверный пароль

AVCN_DATA_ERROR – нарушена структура данных контейнера

Удаление контейнера

```
cryptocont e –n=<Container> [-p=<Password>]
```

`Container` - имя контейнера

`Password` - пароль к контейнеру, может отсутствовать, в этом случае удаление происходит без проверки пароля.

Возможные коды ошибок:

AVCN_CONTAINER_NOT_FOUND – контейнер с указанным именем не существует.

AVCN_INVALID_PASSWORD - указан неверный пароль

AVCN_DATA_ERROR – нарушена структура данных контейнера, невозможно проверить пароль, удаление не произведено.

Создание контейнера

```
cryptocont n –n=<Container> [-p=<Password>] [-y=<SysRandomSource>]
[-r=<RandomFile>] [-key_alg=<KeyAlgOid>] [-u]
```

`Container` - имя контейнера

`Password` - пароль к контейнеру, может отсутствовать, в этом случае пользователю предлагается дважды ввести пароль с клавиатуры.

`SysRandomSource` – системный источник энтропии используемой при генерации случайности. Для Windows параметр указывает код типа используемого криптопровайдера, для linux и solaris параметр игнорируется.

`RandomFile` – имя файла, используемого при генерации случайности. Файл должен быть не пуст, содержимое файла свыше 64 килобайт игнорируется.

KeyAlgOid – OID алгоритма ключа ЭЦП. Возможные значения (без кавычек):

"1.3.6.1.4.1.12656.1.38" или "bds" - ЭЦП согласно СТБ 1176.2-99

"1.3.6.1.4.1.12656.1.35" или "bdspro" - ЭЦП согласно СТБ 1176.2-99 с предварительным хэшированием

Значение по умолчанию – "bds"

-u – неинтерактивный режим генерации случайности.

Создаваемый контейнер содержит личный ключ ЭЦП СТБ 1176.99-2 и параметры ДСЧП на основе функции хэширования СТБ 117.99-1. Для генерации случайности используются:

- системные источники энтропии.
- содержимое файла, указанного параметром –r (опционально).
- если не указан параметр –u, пользователю предлагается произвольным образом нажимать на клавиши до тех пор, пока не будет собрано достаточное количество случайных данных.

Возможные коды ошибок:

AVCN_ALREADY_EXIST – контейнер с указанным именем уже существует.

AVCN_SHORT_PASSWORD – длина пароля меньше 8 символов.

AVCN_BAD_CMDLINE – неверные параметры командной строки.

AVCN_FILE_NOT_FOUND – указанный файл не найден.

AVCN_RAND_FILE_EMPTY – файл случайности имеет нулевой размер.

Копирование контейнера

```
cryptocont c –n=<Container1> [-p=<Password1>] –d=<Container2> [-
q=<Password2>]
```

Container1 - имя контейнера-источника

Password1 - пароль к контейнеру-источнику, может отсутствовать, в этом случае пароль вводится с клавиатуры.

Container2 - имя контейнера-приёмника

Password2 - пароль к создаваемому контейнеру-приёмнику, может отсутствовать, в этом случае пользователю предлагается дважды ввести пароль с клавиатуры.

Возможные коды ошибок:

AVCN_CONTAINER_NOT_FOUND – контейнер с указанным именем (источник) не существует.

AVCN_INVALID_PASSWORD - указан неверный пароль.

AVCN_DATA_ERROR – нарушена структура данных контейнера-источника.

AVCN_READ_ERROR – ошибка чтения данных источника

AVCN_WRITE_ERROR – ошибка записи при создании контейнера-приёмника.

AVCN_SHORT_PASSWORD – длина пароля меньше 8 символов.

AVCN_ALREADY_EXIST – контейнер с указанным именем (приёмник) уже существует.

Создание запроса PKCS#10

Создаётся запрос на сертификат, содержащий открытый ключ ЭЦП СТБ 1176.99-2. Открытый ключ вычисляется на основе личного ключа, хранящегося в указаном контейнере.

```
cryptocont r {-f=<RequestFileName> -n=<Container> [-p=<Password>] [-
cn=<CommonName>] [-c=<Country>] [-o=<Organization>] [-
g=<StateOrProvince>] [-a=<StretAddress>] [-t=<OrganizationalUnit>]
[-e=<EmailAddress>] [-k=<KeyUsage>] } | {-i=<IniFile>}
```

`ContainerName` - имя контейнера, содержащего личный ключ ЭЦП СТБ 1176.99-2.

`Password` - пароль к контейнеру, может отсутствовать, в этом случае пароль вводится с клавиатуры.

`RequestFileName` - имя создаваемого файла запроса.

`CommonName` - имя абонента.

`Country` – идентификатор страны абонента, например BY.

`Organization` – название организации

`OrganizationalUnit` – название подразделения

`StateOrProvince` – название области

`StretAddress` – городской адрес

`EmailAddress` – адрес электронной почты

`KeyUsage` - область применения ключа согласно X.509, комбинация битов:

```
100000000 (digitalSignature)
010000000 (nonRepudiation)
001000000 (keyEncipherment)
000100000 (dataEncipherment)
000010000 (keyAgreement)
000001000 (keyCertSign)
000000100 (CRLSign)
000000010 (encipherOnly)
000000001 (decipherOnly)
```

Параметр `KeyUsage` может отсутствовать, значение по умолчанию – "`100000000`" (ЭЦП).

Если указан параметр `-i`, данные берутся из `.ini`-файла `IniFile`, раздел `request`, поля `filename, commonname, keyusage, country, organization, stateorprovince, address, organizationalUnit , email` а также раздел `container`, поля `name, pin`.

пример ini файла:

```
[container]
name=Container1
pin=12345678

[request]
country=BY
organization=Avest
stateorprovince=Minskaya
address=PravdaStreet5
organizationalUnit=Avest1
commonname=ivanov
```

```
email=ivanov@avest.by
keyusage=100010000
```

Возможные коды ошибок:

AVCN_CONTAINER_NOT_FOUND – контейнер с указанным именем не существует.

AVCN_INVALID_PASSWORD - указан неверный пароль.

AVCN_READ_ERROR – ошибка чтения данных контейнера.

AVCN_WRITE_ERROR – ошибка записи файла запроса.

AVCN_DATA_ERROR – нарушена структура данных контейнера.

AVCN_BAD_KEYUSAGE – указано неверное значение KeyUsage.

AVCN_FILE_NOT_FOUND – ini–файл не найден.

Получение списка существующих контейнеров

```
cryptocont l
```

формат вывода:

```
имя_контейнера1
имя_контейнера2
...
```

Контейнер, содержащий ДПСЧП по умолчанию, в списке не отображается.

Возможные коды ошибок:

AVCN_IO_ERROR – ошибка ввода-вывода при получении списка.

Получение списка подключённых токенов (носителей)

```
cryptocont t
```

формат вывода:

```
имя_носителя1
имя_носителя2
...
```

в текущей версии функция не реализована.

Инициализация или проверка датчика псевдослучайной последовательности по умолчанию.

```
cryptocont i [-n=<Container>] [-p=<Password>] [-y=<SysRandomSource>]
[-r=<RandomFile>] [-k]
```

`Container` – имя контейнера, используемого для генерации параметров датчика случайных чисел по умолчанию.

`Password` – пароль контейнера.

SysRandomSource – системный источник энтропии используемой при генерации случайности. Для Windows параметр указывает код типа используемого криптопровайдера, для linux/solaris параметр игнорируется.

RandomFile – имя файла, используемого при генерации случайности. Файл должен быть не пуст, содержимое файла свыше 64 килобайт игнорируется.

-k – производится проверка существования контейнера ДПСЧП по умолчанию.

Если не указан параметр "-k", команда создаёт контейнер с именем prdparams и паролем prdparams, содержащий ДПСЧП, используемый по умолчанию. Если указано имя и пароль контейнера, его содержимое используется для генерации параметров ДПСЧП, иначе пользователю предлагается произвольным образом нажимать на клавиши до тех пор, пока не будет собрано достаточное количество случайных данных.

При инициализации создаваемого контейнера для генерации случайности также используются:

- системные источники энтропии.
- содержимое файла, указанного параметром -r (опционально).

Возможные коды ошибок:

AVCN_CONTAINER_NOT_FOUND – контейнер с указанным именем не существует.

AVCN_INVALID_PASSWORD - указан неверный пароль.

AVCN_DATA_ERROR – нарушена структура данных контейнера

AVCN_READ_ERROR – ошибка чтения данных контейнера

AVCN_WRITE_ERROR – ошибка записи файла

AVCN_SHORT_PASSWORD – длина пароля меньше 8 символов.

Экспорт содержимого контейнера

```
cryptocont ex –n=<Container> [-p=<Password1>] -f=<FileName> [-
q=<Password2>]
```

Container - имя экспортируемого контейнера

Password1 - пароль к контейнеру, может отсутствовать, в этом случае пароль вводится с клавиатуры.

FileName - имя файла экспорта

Password2 - пароль к экспортируемому файлу

Возможные коды ошибок:

AVCN_CONTAINER_NOT_FOUND – контейнер с указанным именем не существует.

AVCN_INVALID_PASSWORD - указан неверный пароль.

AVCN_DATA_ERROR – нарушена структура данных контейнера

AVCN_READ_ERROR – ошибка чтения данных контейнера

AVCN_WRITE_ERROR – ошибка записи файла

AVCN_SHORT_PASSWORD – длина пароля меньше 8 символов.

AVCN_ALREADY_EXIST – файл с указанным именем уже существует.

Импорт содержимого контейнера

```
cryptocont im -f=<FileName> [-p=<Password1>] -n=<Container> [-
q=<Password2>]  [-y=<SysRandomSource>] [-r=<RandomFile>] [-u]
```

FileName - имя файла импорта

Password1 - пароль к файлу, указанный при экспорте

Container - имя создаваемого контейнера

Password2 - пароль к контейнеру, может отсутствовать, в этом случае пароль дважды вводится с клавиатуры.

SysRandomSource – системный источник энтропии используемой при генерации случайности. Для Windows параметр указывает код типа используемого криптопровайдера, для linux/solaris параметр игнорируется.

RandomFile – имя файла, используемого при генерации случайности. Файл должен быть не пуст, содержимое файла свыше 64 килобайт игнорируется.

-u – неинтерактивный режим генерации случайности.

Команда создаёт новый контейнер и импортирует в него ключи, сохранённые в файле FileName при экспорте. При создании контейнера производится инициализация ДПСЧП. Создаваемый контейнер содержит личный ключ ЭЦП СТБ 1176.99-2 и параметры ДПСЧП на основе функции хэширования СТБ 117.99-1. Для генерации случайности используются:

- системные источники энтропии.
- содержимое файла, указанного параметром -r (опционально).
- если не указан параметр -u, пользователю предлагается произвольным образом нажимать на клавиши до тех пор, пока не будет собрано достаточное количество случайных данных.

Возможные коды ошибок:

AVCN_FILE_NOT_FOUND – файл либо файл случайности не найден.

AVCN_INVALID_PASSWORD - указан неверный пароль к файлу.

AVCN_DATA_ERROR – нарушена структура данных файла импорта

AVCN_READ_ERROR – ошибка чтения данных файла импорта

AVCN_WRITE_ERROR – ошибка записи данных при создании контейнера

AVCN_SHORT_PASSWORD – длина пароля меньше 8 символов.

AVCN_ALREADY_EXIST – контейнер с указанным именем уже существует.

AVCN_RAND_FILE_EMPTY – файл случайности имеет нулевой размер.

Генерация псевдослучайной последовательности

```
cryptocont  -f=<FileName> -l=<RandomSize> [-n=<Container>] [-
p=<Password>]
```

FileName – имя файла для сохранения сгенерированной последовательности.

RandomSize – длина генерируемой последовательности в байтах.

Container – имя контейнера, используемого для генерации. Если параметр не задан, используется контейнер ДПСЧП по умолчанию.

Password – пароль к контейнеру.

Имена контейнеров

Ключевой контейнер может располагаться либо на локальном жестком диске, либо на отчуждаемом носителе (токене). Если для хранения используется токен, полное имя контейнера имеет вид:

```
имя_носителя:имя_контейнера
```

Имена носителя и контейнера не должны включать символ "/".

Для носителя AvPass имя носителя будет avpass.

Если контейнер расположен на локальном диске, имя носителя не указывается, полное имя контейнера имеет вид:

```
имя_контейнера
```

Имя контейнера не должно содержать пробелы и символ "/".

Коды возврата

AVCN_BAD_CMDLINE	1	неверные параметры командной строки
AVCN_CONTAINER_NOT_FOUND	2	контейнер с указанным именем не существует.
AVCN_INVALID_PASSWORD	3	указан неверный пароль
AVCN_ALREADY_EXIST	4	контейнер с указанным именем уже существует
AVCN_BAD_VERSION	5	неверная версия структуры данных контейнера
AVCN_READ_ERROR	6	ошибка чтения
AVCN_WRITE_ERROR	7	ошибка записи
AVCN_IO_ERROR	8	общая ошибка ввода-вывода
AVCN_SHORT_PASSWORD	9	длина пароля меньше 8 символов
AVCN_DATA_ERROR	10	нарушена структура данных контейнера
AVCN_ACCESS_DENIED	11	ошибка доступа при операциях ввода-вывода
AVCN_BAD_KEYUSAGE	12	указано неверное значение KeyUsage
AVCN_FILE_NOT_FOUND	13	файл не найден

