

Документация по Hardlock

Документация по Hardlock познакомит вас со всеми компонентами продуктовой линейки Hardlock и с возможностями защиты вашего программного обеспечения (ПО).

Документация включает в себя следующие разделы.

- **Техническое руководство по Hardlock** описывает принципы защиты ПО с помощью всех компонентов продуктовой линейки. Оно поможет вам выбрать правильную защиту для выполнения ваших задач.
- **Руководство по HL-Bistro** (визуальной среде программирования для Hardlock) описывает этот программный пакет, который позволяет вам проектировать систему защиты в привычной среде Windows.
- **Руководство по HL-Server** описывает применение Hardlock в сетевых приложениях.
- **Руководство по HL-Crypt** описывает применение автоматической защиты вашего ПО.
- **Руководство по HL-API** содержит обзор интерфейса прикладных базовых систем (спецификацию API) для ручной защиты ваших программ.
- **Раздел Понятия** поясняет основные термины и понятия, используемые в документации по Hardlock.

Все разделы дополняются интерактивной помощью (Help) к различным программам. Самые последние изменения и дополнения вы найдете в файлах Readme к программам. В руководствах и файлах интерактивной помощи (Help) применяются следующие типы шрифтовых выделений:

Обозначение	Функция	Пример
жирный	Понятия верхнего уровня Продукты	DatePCTV aus Hardlock Hardlock Twin
ЗАГЛАВНЫЕ БУКВЫ	файлы и пути	HLDRV.EXE
Courier	Синтаксис	hlpatchup -m 29809

Содержание

1 Принцип защиты Hardlock: аппаратная защита программного обеспечения	1
1.1 Как защищаются ваши программы ключами Hardlock?	1
1.2 Кодирование	3
1.3 Шифрование	3
1.4 Применение зашифрованных программ	4
1.5 Дифференцированное разграничение доступа (HL-LiMaS)	4
1.6 Изменение лицензирования (HL-LiMaS)	5
1.7 Сопровождение данных (HL-LiMaS)	5
1.8 Сценарии защиты	6
1.8.1 Простая быстрая защита	6
1.8.2 Дифференцированная защита с помощью Hardlock LiMaS	6
1.9 Тестирование ключа	7
1.9.1 Тестовый пакет	7
1.9.2 Пробное шифрование программ	8
1.9.3 Тестирование функций банка данных HL-DB (HL-LiMaS)	8
1.9.4 Тестирование функции лицензирования HL-RUS (LiMaS)	8
2 Продукты Hardlock. Настоящая защита на все случаи жизни	9
2.1 Ключи Hardlock для конечных пользователей	9
2.1.1 Ключи Hardlock E-Y-E, Hardlock Twin, Hardlock SE для параллельных и/или последовательных портов	10
2.1.2 Плата расширения Hardlock Internal	13
2.1.3 В сети: ключ Hardlock Server Internal и External	16
2.1.4 Ключ в виде карты PC-Card (PCMCIA): Hardlock PC-Card	19
2.1.5 Для порта USB: ключ Hardlock USB	20
2.2 Аппаратное обеспечение для кодирования ключей Hardlock	22
2.2.1 Сертификация CE	22
2.2.2 Криптокарта	22
2.2.3 Адаптер для криптокарты	23
2.2.4 Мастер Hardlock	23

2.2.5	USB-Master-Hardlock	24
2.3	Программное обеспечение для кодирования	24
2.3.1	CP.EXE, CP32.EXE, CPLuna.EXE	24
2.3.2	Cappuccino	24
2.4	Защита программ в ручном режиме при помощи HL-API	25
2.4.1	HL-API — ручной режим	25
2.4.2	Latteccino: тестирование API и генерация кода.	26
2.5	Программное обеспечение для защиты в автоматическом режиме с использованием HL-Crypt и Espresso	26
2.5.1	HL-Crypt – автоматическая защита	26
2.5.2	Espresso	28
2.6	Отслеживание данных в HL-DB	28
2.6.1	Программа для обновления данных (Daten-Tracking): Gazzetta	28
2.6.2	Присоединение банков данных	29
3	Основы криптографии	30
3.1	Поточные шифры против блочных шифров	30
3.1.1	Поточный шифр	30
3.1.2	Блочный шифр	31
3.2	Шифрование в системе Hardlock	33
3.3	Аппаратная генерация ключа	34
3.4	Критерий безопасности	34
4	Договор о разработке и лицензировании	37
4.1	Лицензия	37
4.2	Использование программного обеспечения, продажа ключей Hardlock®	37
4.3	Гарантия	38
4.4	Возмещение ущерба	38
4.5	Окончание договора	39

1 Принцип защиты Hardlock: аппаратная защита программного обеспечения

1.1 Как защищаются ваши программы ключами Hardlock?

Программы шифруются алгоритмом, который ни вы, ни мы точно не знаем. Шифр хранится в аппаратном модуле, именуемом **Hardlock-Modul**. Необходимые модули Hardlock создаются с помощью индивидуально запрограммированных аппаратных средств.

Для защиты ваших программ ключом Hardlock необходимо выполнить следующие шаги.

1. Модули Hardlock кодируются Криптокартой (входит в Мастер-комплект Hardlock).

Защитные модули Hardlock после этого приобретают совершенно индивидуальные особенности, которые можно воссоздать только с помощью тех же самых аппаратных средств.

2. Для защиты программ необходимо использовать тот самый шифр, который используется в ключе. При этом защита может быть настроена индивидуально для каждого комплекта программы.

Защищенные программы будут нормально функционировать только при подключении того ключа Hardlock, который может расшифровать зашифрованные части программ.

Ключи **Hardlock** обеспечивают дифференцированные способы защиты, которые отвечают вашим требованиям и удобны вашим заказчикам. Так, при выборе защитных модулей вы можете подстроиться под предпочтения ваших заказчиков к аппаратным средствам, не затрачивая при этом лишних сил.

Для защиты ваших программ вы получите от нас следующие продукты:

- различные ключи **Hardlock** для конечных потребителей (см. главу 2.1);

- **Криптокарту** для кодирования ключей Hardlock (см. главу 2.2.2.) и **Мастер Hardlocks** для применения новых сигнатур (см. главу 2.2.4);
- **USB-Master-Hardlock** для кодирования **Hardlock USB** (см. гл. 2.2.5);
- программу для управления кодированием **HL-Bistro Cappuccino** для Windows (см. главу 2.3.2), а для пакетного режима — **CP, CP32** или **CPLuna** (см. главу 2.3.1);
- программу для автоматической защиты **HL-Crypt** (см. главу 2.5.1) и **HL-Bistro Espresso** (см. главу 2.5.2 и описание *HL-Bistro*);
- библиотеку **HL-API** для ручного режима защиты (см. главу 2.4 и описание *HL-API*);
- средство тестирования функций API и создания исходных текстов **HL-Bistro Latteccino** (см. главу 2.4.2 и описание *HL-Bistro*);
- базу данных **HL-DB (Hardlock LiMaS)** с **HL-Bistro Gazzetta** для обработки данных по заказчикам и ключам (см. главу 2.6. и описание *HL-Bistro*);
- **HL-RUS (Hardlock LiMaS)** для создания дифференцированных конфигураций защиты, таких, как сублицензирование с дальнейшим отключением – в программах **HL-Bistro Cappuccino** (см. главу 2.3.2 и описание *HL-Bistro*) и **HL-API** (см. главу 2.4 и описание *HL-API*);
- **HL-Server** – для защиты программного обеспечения в сети с использованием всего одного ключа HL-Server (см. описание *HL-Server*).

Подробную информацию о конфигурации, шифровании, применении зашифрованных программ и индивидуальных возможностях вы найдете в следующих главах.

Дополнительно мы предоставляем вам для защиты ваших страничек в Интернете программу **HL-WEB**. С помощью этой программы вы так защитите ваши странички, что доступ будет возможен только с определенным ключом. Например, вы можете так защитить странички техподдержки, что доступ к ним будут иметь лишь те заказчики, которые имеют ключ. Дополнительную информацию по этой программе можно получить в файлах Help.

1.2 Кодирование

При кодировании вы присоединяете сигнатуру из **Криптокарты** или **Мастер-комплекта** и базисный адрес модуля к одной из ваших подпрограмм и адресу модуля пользователя. Каждая Криптокарта и Мастер Hardlock имеют одну уникальную сигнатуру, которую нельзя изменить.

Сигнатура и субкод вместе определяют криптографическое поведение ключа. Базисный адрес модуля и адрес модуля пользователя образуют вместе адрес модуля ключа Hardlock. По адресу модуля зашифрованная программа найдет нужный ключ для расшифровки.

Примечание. Тестовые модули имеют адрес 29809 и тестовое кодирование. Эти ключи не могут быть изменены без Криптокарты.

При кодировании ключей Hardlock с памятью (опция Memory) вы также можете описать память. У этих ключей вы можете провести дифференцированное сублицензирование, чтобы в дальнейшем создать файлы модификации, с которыми далее вы могли бы подключить другие программные компоненты или увеличить время работы (с помощью функции **HL-RUS**). Если вы захотите использовать эту функцию, часть памяти будет резервирована. Информация о лицензиях вписывается цифровым способом с помощью ключа вендора и, тем самым, защищается от манипуляций.

1.3 Шифрование

После того, как в результате кодирования ключа было определено его поведение при шифровании, вы можете так зашифровать ваши программы, что они будут работать только с этим ключом и именно с этим особенным поведением при шифровании.

При этом у вас есть выбор между автоматической и ручной установкой защиты, которые вы можете комбинировать для повышения безопасности.

При автоматической защите дополнительно шифруется готовая программа. Для этого служат программа **HL-Crypt** (см. главу 2.5.1 и описание *HL-Crypt*), а для Windows – программа **Espresso** из пакета **HL-Bistro** (см. главу 2.5.2 и описание *HL-Bistro*). При

этом у вас есть возможность определять защитные механизмы и степень защиты (например, фоновые запросы и ограничение по времени).

Для ручной установки защиты вы можете использовать библиотеку функций **HL-API** (см. главу 2.4.1 и описание *HL-API*). **HL-API** предоставляет вам разные функции, которые при программировании можно непосредственно встроить в программу.

Hardlock Bistro совместно с **Latteccino** предоставляют дополнительные возможности для тестирования влияния отдельных функций на программу (см. главу 2.4.2 и описание *HL-Bistro*).

1.4 Применение зашифрованных программ

После шифрования вы можете отправить программу вместе с ключом своим заказчикам. Вы и ваши заказчики можете выбрать разные типы интерфейсов для ключей Hardlock (см. главу 2.1). Ключи Hardlock для разных интерфейсов могут, несмотря на разный внешний вид, иметь одно и то же поведение при шифровании. Так, при выборе ключа можно учесть особенности аппаратного обеспечения ваших заказчиков, чтобы не затрачивать дополнительных усилий на шифрование.

Данные по ключам для разных заказчиков можно сохранить в базе данных **HL-DB** (см. главу 2.6) и работать с ними в программах **Gazzetta**, **Espresso** и **Cappuccino** (см. описание *HL-Bistro*).

1.5 Дифференцированное разграничение доступа (HL-LiMaS)

Лицензирование и защиту ваших программ можно дифференцировать с помощью функции **HL-RUS** программы **Hardlock LiMaS** настолько, насколько вам это потребуется. В частности, предоставляются следующие возможности:

- защищать и освобождать отдельные программные компоненты;
- установить глобальный счетчик;
- установить срок пользования программы;

- установить число лицензий (при использовании в сети с помощью HL-Server).

Отдельные условия закладываются в программу (с помощью **Espresso** или **HL-API**) и при кодировании ключей устанавливаются отдельно для каждого заказчика (с помощью программы **Cappuccino**). Эта дифференцированная информация может быть позднее изменена, когда ключ модифицирован.

1.6 Изменение лицензирования (HL-LiMaS)

Дифференцированная информация по лицензиям, которая закладывается в каждый отдельный ключ, может быть позднее изменена вашим заказчиком на месте. Условием для такой модификации является применение функции **HL-RUS** при кодировании ключа.

Модификация производится на основе уже записанных в Hardlock данных. Таким образом, и речи нет о новом кодировании.

Данные в Hardlock и файлы модификации защищаются сгенерированным вами ключом вендора. С помощью этого ключа информация по модификации подписывается вами. Она может быть создана только вами. Кроме того, файл модификации может быть применен только к тому ключу с уникальным серийным номером, для которого он был создан.

1.7 Сопровождение данных (HL-LiMaS)

Данные по отдельным ключам записываются в базу данных **HL-DB**. С помощью функции **HL-DB** программы **Hardlock LiMaS** вы можете не только сохранять данные о ключах, но и отсортировать их по своим заказчикам. Вы можете либо использовать базу данных, которую мы вам предоставляем, либо привязаться к собственной базе данных. При этом данные по заказчикам будут однозначно связаны с данными по ключу.

В различных программах **Hardlock Bistro** вы можете видеть и использовать данные по Hardlock и заказчикам, например, для генерации файлов модификации.

Связь между базой данных и программами **Hardlock Bistro** представлена в таблице.

База данных	Средство доступа на чтение	Средство доступа на запись
База данных Hardlock	Cappuccino Gazzetta Espresso	Cappuccino Gazzetta
База данных по заказчикам	Gazzetta Cappuccino	Gazzetta (если применяется поставляемая база данных по заказчикам)

Дальнейшую информацию вы найдете в описании *HL-Bistro*.

1.8 Сценарии защиты

1.8.1 Простая быстрая защита

Вы разработали программу, которую вы хотели бы защитить. Чтобы защитить готовую программу, поступайте следующим образом:

1. ключ Hardlock кодируется с помощью программы **Cappuccino**;
2. ваша программа защищается с помощью **Espresso**. При этом указывается, какой адрес имеет ключ Hardlock. Далее программа шифруется с помощью присоединенного ключа;
3. последующие ключи Hardlock кодируются так же, как и первый ключ;
4. вы продаете вашу программу с ключом Hardlock, который расшифровывает ее во время работы.

Программа может быть расшифрована только подходящим ключом. Пиратские копии не работают, так как отсутствует аппаратное обеспечение, которое может расшифровать программу.

1.8.2 Дифференцированная защита с помощью Hardlock LiMaS

1. С помощью **Cappuccino** кодируется ключ Hardlock с памятью. При этом указывается, что будет использована функция **HL-RUS**. Кроме того, изготавливается ключ

вендора с помощью **Vendor-Key-Manager**, который запускается программой **Cappuccino**, для защиты лицензиями.

2. Программа защищается с указанием ключа вендора с помощью **Espresso**. При этом указывается, при каких условиях функционируют программные файлы. Имеется выбор в комбинации условий. Как вариант, можно установить условия прямо в исходный текст программы с помощью функций **HL-API**.
3. Далее кодируются ключи Hardlock для ваших заказчиков. При этом допускается включать программные файлы по отдельности. Можно обратиться к условиям, которые были определены ранее, установить глобальное время окончания действия или устанавливать счетчик. Вся информация сохраняется в базе данных **HL-DB**.
4. В **Gazzetta** можно использовать данные по ключу Hardlock в базе данных **HL-DB**, например, чтобы целенаправленно обратиться к заказчикам, чьи лицензии истекли.
5. Если заказчик решает приобрести программу в полном объеме, необходимо внести соответствующие данные в **Cappuccino** из базы данных, чтобы создать файл модификации для ключа заказчика. Этот файл можно передать заказчику (например, по электронной почте).

1.9 Тестирование ключа

1.9.1 Тестовый пакет

С помощью пакета тестов вы можете проверить, как функционирует защитная система Hardlock. Вы получите необходимые для этого программы, документацию и один ключ для тестирования с тестовым кодированием и адресом модуля 29809. Используйте **Espresso Assistant** из **Hardlock Bistro**, чтобы произвести быстрое шифрование тест-ключом для тестирования защиты.

Обратите внимание, что шифрование этим ключом не представляет настоящей защиты, так как все тест-модули имеют одно и то же поведение при шифровании. Действенную защиту вы получите, только если вы сами закодируете ключ Hardlock.

С помощью тестового пакета вы можете познакомиться со всеми программами из **Hardlock Bistro**.

1.9.2 Пробное шифрование программ

Используйте **Espresso Assistant** из **Hardlock Bistro**, чтобы провести быстрое шифрование с помощью тестового модуля и проверить защиту.

Обратите внимание, что шифрование этим ключом не представляет настоящей защиты. Действительную защиту вы получите, только закодировав самостоятельно ключ Hardlock.

1.9.3 Тестирование функций банка данных HL-DB (HL-LiMaS)

Функция базы данных **HL-DB** ограничена 10 заказчиками в банке данных. Это значит, что в **Cappuccino** и **Gazzetta** можно увидеть часть базы данных по ключу Hardlock и заказчикам и, таким образом, опробовать управление базой этими обеими программами.

1.9.4 Тестирование функции лицензирования HL-RUS (LiMaS)

Функция **HL-RUS** в **Cappuccino** может быть применена только для сублицензирования и изготовления файлов модификации и только для ключа с адресом 29809. Созданные файлы модификации и информацию по лицензиям вы можете защитить только демонстрационным ключом вендора. Он доступен всем пользователям **HL-Bistro** и не защищен паролем. Демонстрационный ключ вендора не является настоящей защитой, вы не можете создавать сами ключи вендора, так как у вас нет доступа к **Vendor-Key-Manager**.

2 Продукты Hardlock. Настоящая защита на все случаи жизни

2.1 Ключи Hardlock для конечных пользователей

Защитные ключи Hardlock имеют несколько вариантов, которые совместимы со всей защитной системой. Так, вы можете подобрать ключ, который подходил бы аппаратному обеспечению ваших заказчиков, не занимаясь дополнительно программированием. У вас есть выбор между локальными ключами для параллельных портов (**E-Y-E**), для параллельных или последовательных портов (**Twin**), для интерфейса USB (**USB**), как PC-Card или как плата расширения ISA (**Internal**). Ключи для различных интерфейсов есть с памятью (**Memo**) или без памяти (**Standard**). Для эксплуатации в сети у вас есть **Hardlock Server Internal** (как плата расширения) или **Hardlock Server External** (для параллельного порта). Для всех ключей должны быть установлены драйверы (см. ниже).

Ключи Hardlock и порты

Без памяти	С памятью	Порт
Hardlock E-Y-E Standard	Hardlock E-Y-E Memo	параллельный
Hardlock Twin Standard	Hardlock Twin Memo	параллельный последовательный
—	Hardlock USB	USB
Hardlock Internal Standard	Hardlock Internal Memo	ISA-плата расширения
Hardlock PC-Card Standard	Hardlock PC-Card Memo	PCMCIA- плата расширения
—	Hardlock Server External	параллельный
—	Hardlock Server Internal	ISA - плата расширения

Далее мы предлагаем вам параллельную плату расширения **AladdinCARD** для стандартных слотов расширения, на которой могут быть установлены от одного до нескольких ключей для одного компьютера. Отдельные ключи нельзя удалить, не вскрывая компьютера.

Драйвер

Для установки ключей должны быть установлены соответствующие драйверы. Для этого запустите соответствующий файл .EXE, подходящий для вашей операционной системы. На установочном CD-ROM имеются следующие файлы:

Операционная система и драйверы

Операционная система	С графическим интерфейсом	Для командной строки
Windows 3.x	HLDRV16.EXE	INSTVXD.EXE
Windows 95/98, NT	HLDRV32.EXE	HLDINST.EXE

Эти программы управляют системой запуска драйверов для ключей Hardlock. При этом для всех ключей устанавливается драйвер HARDLOCK.VXD (в Windows 9x) или HARDLOCK.SYS (в Windows NT). Для ключа Hardlock USB дополнительно устанавливается AKSUSB.SYS или AKSUSB95.SYS.

СЕ обозначение

Все наши продукты соответствуют требованиям стандарта EG 89/336/EWG «Электромагнитная совместимость» и 72/23/EWG «Низковольтные приборы» с перечнем изменений 93/68/EWG. Наши продукты имеют обозначение СЕ. Если ваш ключ из-за технических обстоятельств не имеет обозначения СЕ, мы вам предоставим соответствующие наклейки.

2.1.1 Ключи Hardlock E-Y-E, Hardlock Twin, Hardlock SE для параллельных и/или последовательных портов

Для параллельных и/или последовательных портов у вас есть ключи **Hardlock E-Y-E** и **Hardlock Twin** для локального использования и **Hardlock Server External** для использования в сети (по **Hardlock Server** см. главу 2.1.3). **Hardlock E-Y-E** и **Hardlock Twin** имеют чип ASIC и не имеют память (**Standard**) или имеют память (**Memo**).

Hardlock E-Y-E

Hardlock E-Y-E может быть использован только на параллельном порте. При этом может быть присоединено множество ключей, а ключей без памяти – теоретически сколько угодно.

Hardlock Twin

Ключ **Hardlock Twin** может быть использован как на параллельных, так и на последовательных портах. Таким образом, у ваших заказчиков есть огромный выбор при применении поставляемых вами ключей. При этом может быть присоединено множество защитных ключей, а если это ключи без памяти на параллельных портах, то теоретически сколько угодно. На последовательных портах защитные ключи могут быть установлены по два на каждом порту. При применении ключа Hardlock Twin на последовательном порте следует учитывать небольшую скорость загрузки.

Память

Ключи **Hardlock E-Y-E Memory** и **Hardlock Twin Memory** имеют дополнительно к чипу ASIC энергонезависимую память (EEPROM) объемом 128 байт. Память состоит из области ROM и из области RAM в 32 байта.

Область ROM (Read-Only-Memory) допускает запись только при помощи Криптокарты. После программирования ключа она может только читаться из прикладной программы. Изменение возможно только при перекодировке. Область RAM (Random-Access-Memory) может перезаписываться независимо от кодирования.

Вы можете использовать память, чтобы защитить различные программные компоненты одним ключом, установить счетчики для тестируемых версий и специфические для программы данные, фиксированные данные в памяти ROM и изменяемые данные конфигурации в RAM. Вы можете развивать эти функции индивидуально или использовать функцию **HL-RUS**. Если вы используете эту функцию, то часть обеих областей памяти будет для нее зарезервирована, чтобы провести там дифференцированное лицензирование, установить лимит глобальных счетчиков и т. д.

Внутреннюю память вы можете обработать при кодировании ключа Hardlock в программе **Cappuccino**.

Технические характеристики Hardlock Twin, Hardlock E-Y-E

Параметр	Данные (с памятью и без памяти)
температура хранения	-25° до +70° C
рабочая температура	0° до 55° C
влажность воздуха	20 - 80% относительная влажность
размеры вкл. разъем	44.5 x 54.7 x 16 мм
тип разъема	DB 25
сигнальные шины	DATA 0 ... DATA 7, BUSY, INIT, STROBE, GROUND
выходные шины	BUSY (серия: TxD, RTS, GND, DTR, Ri)
потребляемый ток	< 100 mA (< 2mA последовательно)
мин. рабочее напряжение	са. 2.8 V
батарея	отсутствует
максимально число посл. подключений	теоретически любое
ASIC технология	CMOS 1 мс ячейками EEPROM
сложность	свыше 1300 элементов
емкость кода	2 ⁴⁸
емкость адресов модулей	2 ¹⁵
число циклов программирования	10 000 для ASIC, 1 млн. для память
емкость памяти	отсутствует или 128 байт, (из них 96 байт только для чтения)
программирование памяти	никакого или 96 байт только с Криптокартой и 32 байта для прикладных программ
сохранность данных в ASIC	10 лет
сохранность данных в памяти	40 лет

Ключ Hardlock SE

Этот ключ предназначен для последовательного порта и может быть использован на рабочих станциях. Он выполнен по другой технологии (не Hardlock ASIC) и, следовательно, не совместим с другими продуктами этого ряда. Этот продукт не поддерживает каскадирования. Для получения информации по этому продукту вы можете связаться с нашим отделом продаж.

Технические характеристики Hardlock SE

Параметр	Характеристики
температура хранения	-25° до +70° C
рабочая температура	0° до 55° C
влажность воздуха	20 - 80% относительной влажности
габаритные размеры (вкл. разъем)	70 x 54.7 x 21 mm
тип разъема	DB 25
использованные сигнальные шины	TxD, RxD GND
потребление эл. энергии	< 10 mA
батарея	нет
емкость кода	2 ⁴⁸

2.1.2 Плата расширения Hardlock Internal

Альтернативой внешнему ключу Hardlock являются платы расширения **Hardlock Internal** – для локального использования и **Hardlock Server Internal** – для использования в сети (**о Hardlock Server Internal** см. главу 2.1.3). Таким образом, не нужен ни параллельный, ни последовательный порт, защита внешне не заметна и не может быть удалена без вскрытия компьютера. Поэтому **Hardlock Internal** подходит для комплексных решений и компьютеров со многими пользователями.

Hardlock Internal, как и внешний модуль, может быть как без памяти, так и с памятью (**Memory** или **Standard**).

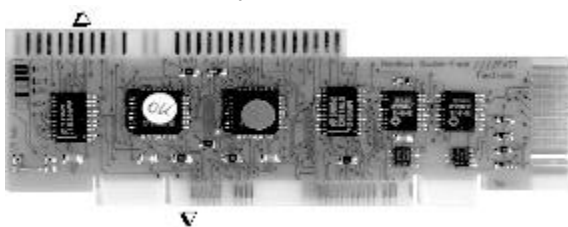
Технические характеристики Hardlock Internal

Параметр	Характеристики (с и без памяти)
габаритные размеры	138 x 47 мм
подключение программатора	26-контактный концевой разъем
архитектура	Micro Channel и ISA
Micro Channel ID	6769 (6769.ADF)
ISA I/O- адрес	\$278, \$378, \$3BC через Jumper
технология ASIC	CMOS 1 м с EEPROM
объем памяти	никакого или 128 байт, из них 96 байт только для чтения
программирование памяти	никакого или 96 байт только для Криптокарты и 32 байт программно

Установка платы расширения Hardlock Internal

Эта плата может быть встроена как в АТ – совместимые компьютеры, так и в компьютеры с архитектурой Micro Channel (MCA). В этом разделе описываются установка для обоих вариантов.

Разъем для подключения к
АТ - компьютерам



Разъем для подключения к
MCA компьютерам

Разъем для програм-
мирования
при помощи
Криптокарты

1. **MCA.** Перед установкой скопируйте файл @6769.ADF с дискеты на справочную дискету MCA компьютера.
2. Выключите ваш компьютер и снимите крышку.

3. Аккуратно распакуйте плату ключа, не касайтесь ее электронных элементов, чтобы не повредить их статическим электричеством.
4. Установите адресные перемычки на плате подходящим для вашего компьютера образом.

АТ: при установке ключа для этого типа компьютера необходимо выбрать свободный начальный адрес (адрес, не занятый параллельным портом) при помощи двух адресных перемычек. Поскольку внутренний ключ имеет две адресные перемычки, вы можете установить один из трех адресов ввода/вывода. Правильные адреса указаны на плате в шестнадцатеричном представлении; возможные конфигурации переключателей показаны с использованием символов.

МСА: если вы устанавливаете ключ – карту в компьютер **МСА**, убедитесь, что обе адресные перемычки установлены должным образом.

Примечание. Последние модели внутренних ключей имеют вместо перемычек два мини-переключателя. Они функционируют аналогично перемычкам.

5. Установите ключ–карту в соответствующий слот компьютера. Убедитесь, что карта установлена правильной стороной.

МСА: для моделей PS/2 70 и старше вы можете установить ключ только в 16-битный слот.

6. **АТ:** закройте корпус и включите компьютер.
7. **МСА:** при установке внутреннего ключа в МСА компьютер дополнительно нужны следующие шаги.

МСА до 90 модели: установите копию справочной дискеты в дисковод А: и загрузите систему.

МСА от 90 модели и старше: достаньте все дискеты из дисководов.

- а) Нажмите комбинацию клавиш **CTRL+ALT+DEL**. Следите за курсором. Как только он достигнет правого верхнего угла экрана, нажмите комбинацию **CTRL+ALT+INSERT** и снова отпустите все три клавиши. На эти операции у вас восемь секунд времени.

На экране появится основное меню. Выберите Copy Diskette for System Expansion и следуйте указаниям на экране.

б) Выберите подменю Configuration.

с) Запустите **Automatic Configuration**. Внесите указанный адрес внутреннего ключа, например:

Slot x: Hardlock Internal
IO Adresse Low Byte = ADR_xxBC
High Byte = ADR_03xx
(В итоге – шестнадцатиричный адрес 3BC)

Разрешение автоматической конфигурации может вызвать проблемы, когда частично меняются параметры установленной карты. Например, возможен конфликт с менеджерами памяти, с сетевыми картами, драйверы которых читают адрес RAM из файла конфигурации, а не с самой карты, с некоторыми графическими картами высокого разрешения.

8. **MCA**: сохраните новую конфигурацию.

9. **MCA до 90 модели**: достаньте дискету и перезагрузите компьютер.

MCA 90 модели и старше: перезагрузите компьютер.

На этом установка внутреннего ключа закончена.

Если у вас возникнут проблемы с установкой внутреннего ключа, обратитесь к руководству по установке дополнительных плат расширения вашего компьютера.

2.1.3 В сети: ключ Hardlock Server Internal и External

Для использования ключей в сети применяются внешние защитные ключи **Hardlock Server External** и платы расширения **Hardlock Server Internal**. Эти ключи всегда содержат память.

При использовании системы защиты Hardlock-Server функции защиты распространяются на все рабочие станции сети без необходимости установки на каждой из них отдельного ключа Hardlock. Дополнительную информацию по лицензиям хранят в специальном файле.

Для пользования системой Hardlock-Server кроме ключей нужно программное обеспечение, которое защищено вторым чипом ASIC на защитном ключе. Программное обеспечение для Hardlock-Server описано в руководстве *HL-Server*.

Технические данные: Hardlock-Server

Параметр	Extern	Intern
температура хранения	-25° до +70° C	-25° до +70° C
рабочая температура	0° до 55° C	0° до 55° C
влажность воздуха	20 - 80% относительная влажность	20 - 80% относительная влажность
габаритные размеры (вкл. разъем)	44.5 x 54.7 x 16 мм	138 x 47 мм
тип разъема	DB 25	AT-Bus/MCA
использованные сигнальные шины	DATA 0 ... DATA 7, BUSY, INIT, STROBE, GROUND	
выходные шины	BUSY	
потребляемый ток	< 200 mA	< 35mA
мин. рабочее напряжение	2.8 V	4,5 V
батарея	нет	нет
макс. число послед. подключений	5	-
технология ASIC	CMOS 1 мс E2 ячейками	CMOS 1 мс E2 ячейками
сложность	приблизительно 1300 элементов	приблизительно 1300 элементов
емкость кода	2 ⁴⁸	2 ⁴⁸
адресное пространство	215	215
число программных циклов	10 000 для ASIC, 1 млн. для памяти	10 000 для ASIC, 1 млн. для памяти
объем памяти	128 байт, из них 96 байт только для чтения	128 байт, из них 96 байт только для чтения
программирование	96 байт только Криптокартой, 32 байт - программно	96 байт только Криптокартой, 32 байт - программно
разъем для программатора		концевой разъем 26-контактный
архитектура		Micro Channel и ISA
Micro Channel ID		6769 (6769.ADF)

Параметр	Extern	Intern
ISA I/O адрес		\$278, \$378, \$3BC через адресные перемычки
программное обеспечение	резидентная программа DOS на любой станции DOS- (свыше 70 KB может быть загружена в область верхней памяти). Напрямую использует уровень сетевых протоколов (IPX, NetBios). Для сервера Novell NetWare или программ для Win32	резидентная программа DOS на любой станции DOS- (свыше 70 KB может быть загружена в область верхней памяти). Напрямую использует уровень сетевых протоколов (IPX, NetBios). Для сервера Novell NetWare или программ для Win32
сохранение данных ASIC	10 лет	10 лет
сохранение данных память	40 лет	40 лет

Установка платы расширения

При установке **Hardlock Server Internal** действуйте так же, как и при установке **Hardlock Internal** (см. главу 2.1.2).

2.1.4 Ключ в виде карты PC-Card (PCMCIA): Hardlock PC-Card

Hardlock PC-Card – это удобное средство для переносных компьютеров, которое может, как и другие ключи, быть с памятью и без (**Memory** или **Standard**).

Технические характеристики PC-Card (PCMCIA)

Параметр	Характеристика
тип карты	PCMCIA PC Card Type II
рабочая температура	0° до 55° C
влажность воздуха	20 - 80% относительная влажность
потребляемый ток	< 100 mA (типично 50 mA)
рабочее напряжение	5 вольт
батарея	нет
ASIC технология	CMOS 1 м с EEPROM
сложность	приблизительно 1300 элементов
емкость кода	2 ⁴⁸
адресное пространство	2 ¹⁵
число циклов программирования	10.000 для ASIC, 1 млн. для памяти
сохраность данных в ASIC	10 лет
сохраность данных в памяти	40 лет

2.1.5 Для порта USB: ключ Hardlock USB

Ключ **Hardlock USB** объединяет в себе преимущества Hardlock и интерфейса USB. Условием этого является то, что операционная система поддерживает интерфейс USB. Пока это возможно в Windows 95 OSR2 с установленной полддержкой USB, а так же в Windows 98 и в Windows 2000.

Для использования ключа Hardlock USB устанавливается дополнительный драйвер к уже имеющемуся HARDLOCK.VXD: AKSUSB.SYS для Windows 2000 и Windows 98 или AKSUSB95.SYS для Windows 95 (см. главу 2.1).

Этот ключ совместим со всей палитрой ключей и не требует дополнительных усилий при программировании.

Технические параметры. Ключ Hardlock USB

Параметр	Характеристика
тип карты	PCMCIA PC Card Type II
рабочая температура	0° до 55° C
влажность воздуха	20 - 80% относительная влажность
потребляемый ток	< 100 mA (типично 50 mA)
рабочее напряжение	5 вольт
батарея	нет
ASIC технология	CMOS 1 м с EEPROM
сложность	приблизительно 1300 элементов
емкость кода	2 ⁴⁸
адресное пространство	2 ¹⁵
число циклов программирования	10.000 для ASIC, 1 млн. для памяти
сохраность данных в ASIC	10 лет
сохраность данных в памяти	40 лет

Hardlock USB оснащен светодиодным индикатором (LED), который индицирует его статус.

Индикация LED	Статус
LED не светится	модуль не распознается. Возможные причины: нет драйвера; нет поддержки; USB- модуль не работает
LED мигает с частотой около 2Гц	ключ распознан, но в нем есть ошибки. Возможные причины: дефект памяти, ключ не был закодирован как Hardlock модуль; ключ не работает
LED светится	ключ распознан и установлен

2.2 Аппаратное обеспечение для кодирования ключей Hardlock

2.2.1 Сертификация CE

Наше аппаратное обеспечение отвечает всем требованиям стандарта 89/336/EWG «Электромагнитная совместимость» и 72/23/EWG «Требования к низковольтному изделию» с дополнениями 93/68/EWG.

Поэтому наши продукты несут имеют символ **CE**. Будет ли соответствовать ваше аппаратное обеспечение сертификату **CE**, мы оставляем на ваше усмотрение и ответственность.

2.2.2 Криптокарта

Все защитные ключи Hardlock, кроме ключа Hardlock USB, на-
iöyîôp èëè ãäâç ääàïðäð (ñì. главу 2.2.3) кодируются при помощи **Криптокарты**. (О кодировании Hardlock USB см. гла-
ву 2.2.5).

Каждая Криптокарта имеет сигнатуру, которая при кодировании переносится в ключ Hardlock. Каждая сигнатура поставляется компанией Aladdin Knowledge System только один раз.

Технические характеристики Криптокарты

Параметр	Характеристики
габаритные размеры	108 x 110 мм
тип разъема	IBM PC совместимый 8 Bit Slot
напряжение программирования	18 Volt \pm 3%
IBM PC I/O адрес	278 Hex или 3BC Hex
длительность программирования	< 1 s
защита от копирования	через микросхему ключа и сигнатуру памяти
кабель адаптера	1.5 метра, 25-pin
рабочая программа	CP, CP32, CPLuna, Cappuccino

Сигнатуру одной Криптокарты вы можете комбинировать с 43.680 субкодами и с 30 адресами модулей (Кодирование –

см. главу 1.2). Если этого вам не хватает, нет необходимости получать еще CPC. Вместо этого вы можете применить еще один Master-Hardlock, который имеет другую сигнатуру и другое кодирование.

Установка

Криптокарта устанавливается в PC/AT и совместимые с ними компьютеры, требует один свободный слот ISA на 8 бит.

Криптокарта использует один из двух адресов портов \$278 или \$3BC. Убедитесь до установки Криптокарты, что выбранный вами адрес не занят другой картой.

Чтобы установить криптокарту, поступайте следующим образом:

1. выключите компьютер и откройте корпус;
2. установите на криптокарте с помощью перемычек адрес;
3. установите криптокарту в свободный слот ISA 8 бит;
4. снова закройте компьютер.

Теперь вы можете кодировать модули с помощью криптокарты в программе **Cappuccino**. При этом адрес криптокарты вы можете установить сами или автоматически.

2.2.3 Адаптер для криптокарты

С помощью криптокарты кодируются все ключи, кроме ключа Hardlock USB. Внешние модули устанавливаются напрямую на криптокарте, в то время как другие модули (платы расширения и карты PC-Card) соединяются через адаптер. Необходимый адаптер вы приобретаете вместе с ключом Hardlock.

2.2.4 Мастер Hardlock

Мастер Hardlock вам потребуется, если вы захотите использовать новую сигнатуру для кодирования. Мастер Hardlock используется вместе с криптокартой. Он не должен быть присоединен при кодировании к криптокарте, так как он может быть случайно перекодирован. Устанавливайте **Мастер Hardlock** на параллельный порт.

Технические характеристики **Мастер Hardlock** соответствуют техническим характеристикам **Hardlock E-Y-E Memory** (см. главу 2.1.1).

2.2.5 USB-Master-Hardlock

USB-Master-Hardlock вам потребуется, когда вы будете кодировать ключ **Hardlock USB**. **USB-Master-Hardlock** не должен при кодировании быть присоединен к криптокарте, так как он может быть случайно закодирован заново. Вместо этого установите его на параллельный порт.

Технические данные **USB-Master-Hardlock** соответствуют техническим данным **Hardlock E-Y-E Memory** (см. главу 2.1.1).

2.3 Программное обеспечение для кодирования

Аппаратные средства кодирования ключей Hardlock- могут управляться различными программами. Для интерфейса Windows имеется программа **Cappuccino** из пакета **Hardlock Bistro**. Только в ней вы сможете использовать в полном объеме функции Hardlock. Для некоторых функций можно применять консольные приложения, такие как CP.EXE, CPLuna.EXE и CP32.EXE, но тогда нельзя использовать функцию **HL-RUS** для сублицензирования и модификации.

2.3.1 CP.EXE, CP32.EXE, CPLuna.EXE

CP32.EXE и другие консольные приложения, это программы, с помощью которых вы можете управлять кодированием ключей Hardlock. Параметры, используемые в этих программах, вы найдете в файлах Readme.

Если вы хотите использовать все возможности ключей Hardlock, советуем вам использовать не консольные приложения, а программу **Cappuccino** из **Hardlock Bistro** (см. ниже и раздел *HL-Bistro*).

2.3.2 Cappuccino

Программа **Cappuccino** обеспечивает доступ ко всем функциям кодирования ключей и обладает привычным интерфейсом Windows. При этом вы можете проводить дифференцированное лицензирование с помощью **HL-RUS** и использовать возможности модификации. В программе **Cappuccino** вы можете использовать преимущества базы данных и, с помощью **HL-DB**,

получить доступ к базе данных заказчиков. Подробнее вы получите информацию о **Cappuccino** в разделе *HL-Bistro*.

2.4 Защита программ в ручном режиме при помощи HL-API

2.4.1 HL-API — ручной режим

С помощью библиотеки функций HL-API (Application Programming Interface) вы можете установить защиту напрямую и индивидуально для вашей программы. Если вы хотите защитить вашу программу не только автоматически, но и вручную, вы должны в фазе разработки концепции запланировать эффективную защиту. Если вы не хотите затрачивать лишних усилий, вы можете все же защитить вашу программу автоматическим путем с помощью **HL-Crypt** или **Espresso**.

Соединение ключа с подлежащей защите программой является самым слабым звеном в цепи защиты. При ручной установке защиты качество защиты зависит от знаний программиста и его квалификации. Помните, что много небольших защитных мер лучше, чем одна или несколько более качественных мер.

Aladdin Knowledge Systems предоставляет вам вспомогательные средства для ручной установки защиты с помощью интерфейса **HL-API**.

- Все необходимые подпрограммы для управления ключами из ваших приложений и переноса данных между программой и ключом.
- Программу шифрования для выполнения фиксированного алгоритма на основе блочного шифра и информации ключа.
- Удобный программный интерфейс на основе вызовов на языке высокого уровня. Высокоуровневый интерфейс упрощает управление механизмами защиты и позволяет применять инструменты и примеры программ на всех популярных языках программирования.

Интерфейс **HL-API** поддерживает все популярные операционные системы и компиляторы.

Более подробную информацию о **HL-API** можно получить в разделе *HL-API*.

2.4.2 Latteccino: тестирование API и генерация кода.

Средством тестирования для **HL-API** является программа для Windows **Latteccino** из пакета **Hardlock Bistro**. С ее помощью можно протестировать действие всех функций HL-API. При этом вы можете сгенерировать C-код.

В **Latteccino** у вас есть доступ к подробной документации по функциям **HL-API**, которые вы также найдете в разделе *HL-API*. Более подробную информацию по **Latteccino** вы найдете в разделе *HL-Bistro*.

2.5 Программное обеспечение для защиты в автоматическом режиме с использованием HL-Crypt и Espresso

Простой и надежный метод защитить ваши программы – это автоматическая защита с помощью **HL-Crypt** и **Espresso**. Автоматическую защиту вы сможете провести уже после создания программного обеспечения. Вы можете использовать DOS-программу **HL-Crypt** или работать в Windows-программе **Espresso**, которая обращается к **HL-Crypt**.

2.5.1 HL-Crypt – автоматическая защита

Система **HL-Crypt** – это простой и надежный метод, для того чтобы интегрировать ключ в вашу программу, не внося изменений в исходный код программы. Автоматическая защита экономит ваше время и позволяет увеличить степень защиты, наряду с другими методами. Программа **Espresso** из пакета **Hardlock Bistro** предоставляет вам удобный Windows-интерфейс для **HL-Crypt**.

HL-Crypt предоставляет следующие возможности:

- **HL-Crypt** выполняет обращения к ключу во время выполнения программы, а не только при запуске программы. Частоту выполнения обращений вы можете установить сами;
- все типы файлов, входящие в состав приложения, могут быть зашифрованы. Это означает, что HL-Crypt также

защищает и файлы данных;

- **HL-Crypt** обеспечивает уникальность вашей системы путем выбора различных модулей защиты. Каждый модуль обеспечивает свой метод защиты;
- не возникает никаких проблем с оверлейными программами для DOS. И внешние, и внутренние оверлеи могут быть защищены независимо от используемого языка программирования;
- шифрование в реальном масштабе времени позволяет защитить программы, написанные на языках-интерпретаторах;
- настраиваемые механизмы защиты обеспечивают максимально возможную степень защиты против «крэков» (например, при использовании отладчиков);
- простая защита программ, использующих расширения DOS.

Система **HL-Crypt** может быть использована для защиты следующих типов программ:

HL-Crypt	Защита для	Операционная система
HL-CRYPT.EXE (HL-EVALU.EXE)	COM, EXE	DOS
HLWCRIPT.EXE (HLWEVALU.EXE)	EXE, DLL	Windows 3.x, 16-битные программы
HLCWIN32.EXE (HLEWIN32.EXE)	EXE, DLL	Win32 (Windows 95-, Win32s-, NT программы)

При этом система **HL-Crypt** сама является программой для DOS и работает под MS- или PC-DOS начиная с версии 3.1 и выше (см. главу 2.5.2 и раздел *HL-Bistro*).

Даже если у вас еще нет полной версии (программы **HL-Crypt**, защищенной ключом в красном корпусе), вы можете протестировать все возможности программы **HL-Crypt**. Для этого можно использовать демо-программы, которые указаны в скобках в предыдущей таблице (например, программу HL-EVALU.EXE для DOS). Демо-версии программ работают с демонстрационным ключом, имеющим адрес 28909. Единственное ограничение состоит в том, что при помощи демо-версий программ приложение можно защитить только демонстрационным ключом, который является свободно доступным.

Если не оговорено отдельно, все приведенные примеры ссылаются на демо-версию ключа Hardlock с адресом 29809. Такой ключ поставляется в демонстрационном пакете. Кроме того, все «незакодированные» ключи программируются с этим адресом при их производстве.

2.5.2 Espresso

С помощью **Espresso** вы можете обращаться ко всем функциям **HL-Crypt** и при этом работать в привычной оболочке Windows (подробнее в разделе *HL-Bistro*).

2.6 Отслеживание данных в HL-DB

Функция **HL-DB** позволяет вам в разных программах **Bistro** обращаться в базу данных, где находится информация по закодированным ключам. Перечень возможностей вы найдете в *главе 1.7*.

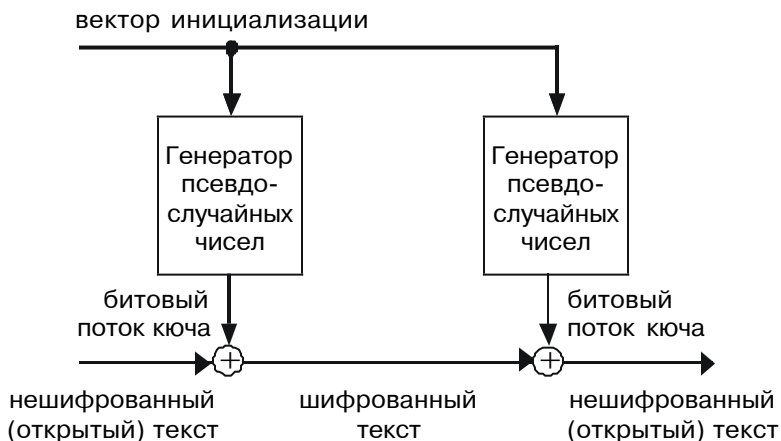
Данные по ключам могут быть связаны с данными по заказчикам, которые берутся либо из базы данных системы Aladdin Knowledge Systems, или из вашей собственной присоединенной базы.

2.6.1 Программа для обновления данных (Daten-Tracking): Gazzetta

С помощью **Gazzetta** из **HL-Bistro** вы можете работать с информацией из базы данных и при этом еще и обращаться к данным из вашей базы данных о заказчиках. Вы можете сортировать данные, фильтровать и экспортировать для дальнейшей обработки. Так могут быть, например, найдены все заказчики с определенным видом защитного ключа Hardlock. Подробнее о программе **Gazzetta** – в разделе *HL-Bistro*.

2.6.2 Присоединение банков данных

Для заказчиков вы можете использовать таблицу в банке данных Hardlock или применить свой собственный банк заказчиков. Условием присоединения такого банка является однозначное обозначение данных номерным ключом. Как присоединять банк данных, читайте в описании *HL-Bistro*.



3 Основы криптографии

3.1 Поточные шифры против блочных шифров

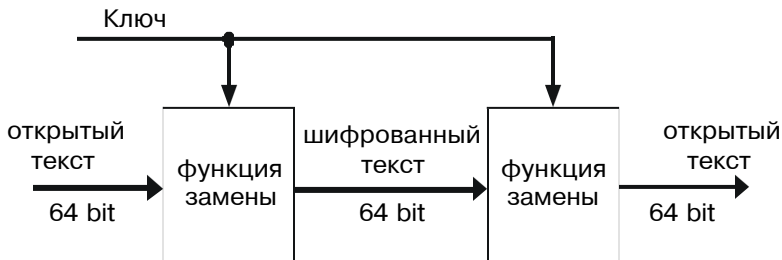
Блочное и поточное шифрование имеют в своей основе различие, основанное на способе подготовки открытых данных к шифрованию. Данные могут быть сгруппированы в потоки фиксированной длины (блоки) или представляться последовательным потоком. Оба метода имеют специфические преимущества и недостатки и подходят для различных приложений.

3.1.1 Поточный шифр

Поток битов открытого текста подвергается операции XOR (сложение по модулю 2) с потоком битов ключа. Для восстановления исходного текста из зашифрованных данных процедура просто повторяется с тем же самым ключом.

На практике (передача ключа получателю, требования по памяти для ключа) длина ключа может не совпадать с длиной потока данных. Следовательно, реально использовать параметрический генератор псевдослучайных чисел для создания потока бит ключа.

Параметры генератора псевдослучайных чисел формируют ключ, и они же используются для создания потока битов, необходимого для расшифровывания. Для получения гарантии безопасности этого метода поток псевдослучайных чисел не должен повторяться (т.е. должен быть большой продолжительности), не должен быть предсказуем (не должно быть зависимости между последовательными величинами).



При использовании коротких блоков данных существует другая проблема. Поскольку поток битов ключа всегда начинается одинаково, злоумышленник может вычислить последовательность символов ключа из пары открытый/закрытый известный текст. По этой причине вводится понятие вектора инициализации, который является задающей величиной для генератора псевдослучайных чисел. Не путайте его с реальным ключом.

Вектор инициализации не должен быть тайным, но важно его регулярно менять. Он должен храниться вместе с зашифрованными данными, так как он необходим для расшифровывания.

3.1.2 Блочный шифр

При блочном шифровании блок открытого текста фиксированной длины заменяется на блок зашифрованного текста аналогичной длины при помощи обратимого процесса замены. Ключ для шифрования блока генерируется из параметров функции замены.

Отсюда следующие требования к функции замены:

- блок должен быть достаточно большой длины, чтобы замена не могла быть описана при помощи таблицы. В основном достаточно 64 бит, поскольку нереально попытаться создать таблицу с количеством значений для каждого ключа 2^{64} ;

- пространство ключей должно быть достаточно большим, чтобы проверка всех возможных ключей для пары открытый/зашифрованный текст занимала много времени и ресурсов;
- преобразование должно выполняться таким образом, чтобы число ключей, которое необходимо проверить для пары открытый/зашифрованный текст, нельзя было бы уменьшить математическими или статистическими методами. Не должно быть математического метода усечения процедуры поиска ключей. Ниже приведен список некоторых рекомендаций для достижения этих требований:

Полнота

Каждый выходной бит, т.е. бит зашифрованного текста, должен зависеть от каждого входящего, т.е. бита открытого текста. Должен соблюдаться критерий строгого потока. Вероятность того, что каждый выходящий бит будет изменен при изменении входного бита, должна быть 50%.

Нелинейность

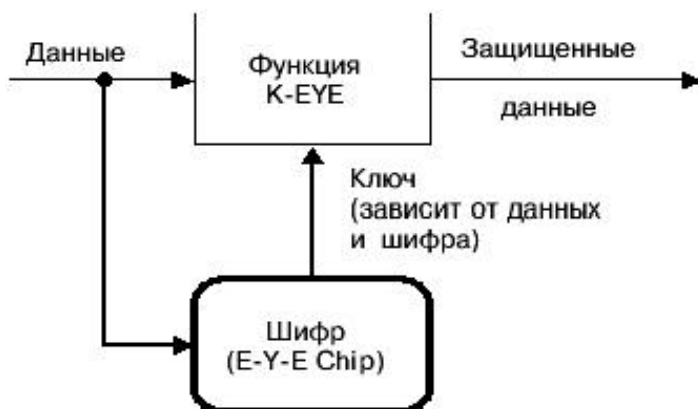
Не должно быть линейной зависимости между открытым и зашифрованным текстом.

Блочные шифры сложнее в применении, чем поточные. Наиболее популярные методы шифрования DES (Data Encryption Standard) и Public Cryptosystem RSA (по именам изобретателей Rivest, Shamir, Adleman) основаны на принципах блочного шифрования. Блочные шифры имеют только один недостаток перед поточными:

при шифровании записей данных, имеющих длину, не кратную целому числу длин блоков, с использованием блочного шифра необходимо использовать специальные меры для предотвращения увеличения размера данных.

С другой стороны, поточный шифр использует вектор инициализации, что также ведет к увеличению размера данных.

3.2 Шифрование в системе Hardlock



При создании системы Hardlock нашей целью было найти метод шифрования, отвечающий следующим требованиям:

- простота применения;
- наличие аппаратно генерируемых ключей;
- автоматическое изменение запроса.

Последнее означает, что для различных данных выполняются различные запросы к аппаратным средствам. Увеличение количества запросов увеличивает также необходимое на вскрытие защиты время. Гораздо сложнее анализировать меняющиеся, чем остающиеся постоянными ответы. Для большей ясности мы приведем более детальное описание двух типов шифрования.

Для того, чтобы добиться «автоматического изменения запроса» шифра ключа, мы использовали метод Feistel, специально предназначенный для этого приложения. Этот блочный шифр основан на использовании обратимого переплетения данных (напоминающего косу) с наборами субключей, которые используются в обратном порядке при расшифровывании.

Грамотно установленная зависимость между субключами и данными позволяет генерировать частичные ключи аппаратными

средствами и восстанавливать их из зашифрованных данных в процессе расшифровывания.

3.3 Аппаратная генерация ключа

Блочное шифрование выполняется подпрограммой HL_CODE, при этом система Hardlock предоставляет нужные ключи.

Как и во многих системах шифрования, так и в HL_CODE создается набор субключей как расширения ключа. Задача программы расширения – формирование запросов к ключу. При этом аргументы запросов формируются из данных и из данных предыдущих запросов.

Таким образом, генерируемые субключи используют в качестве шифра Feistel-Chiffre (сравните, например Luzipher, DES) для двунаправленной замены 64-битного блока данных на 64-битный зашифрованный блок. Поскольку процесс генерации ключей сильно нелинейный, нет необходимости использовать дополнительную нелинейную функцию. Это не тот случай с DES, когда требуются SBOXы и последовательное преобразование. 32-битные субключи зависят от каждого бита связанных с ними аргументов. В результате, шифр заканчивается только после нескольких итераций (т.о. каждый выходной бит зависит от каждого входного бита).

Когда шифр применяется повторно к зашифрованному тексту, аппаратные средства генерируют субключи в обратном порядке. В отличие от DES, нет разницы между шифрованием и расшифровыванием. Это упрощает использование программы HL_CODE.

Реальный ключ шифра представляет собой параметры алгоритма Hardlock. Пространство ключей равно пространству Hardlock - 2^{48} .

3.4 Критерий безопасности

Для того, чтобы оценить количество времени и усилий, необходимых злоумышленнику для расшифровки зашифрованных при помощи Hardlock файлов, мы должны сделать некоторые предположения относительно его возможностей. В следующих разделах представлены результаты оценки для трех сценариев.

Сценарий А

Условия. Хакер имеет ясное представление о том, является ли подобранный ключ правильным. Это вполне реалистичное допущение, если открытые данные являются текстом.

Хакер не знает алгоритма «Аладдина», который использует Hardlock для генерации ключа. Этот алгоритм держится в тайне.

Хакер обладает мощным процессором (600 MHz Pentium III), который выполняет шифрование блока со сравнением каждые 20 микросекунд.

Усилия. Если хакер не знает алгоритма Hardlock, он должен искать ключ из пространства ключей для каждого нового блока данных, так как субключ для каждого блока данных различный.

Программа HL_CODE получает 48-битный сеансовый ключ из аппаратных средств при шифровании каждого блока, таким образом, возможности должны быть рассчитаны для каждого 64-битного блока. Благодаря симметричности шифра HL_CODE и вероятности поиска правильного субключа после 50% попыток, это число уменьшается на фактор 4 до 2^{46} . Время, затраченное для расшифровывания блока данных длиной в 1Кб (128x64Бит), будет составлять:

$$128 \times 2^{46} \times 20 \mu S = 1,8 \times 10^{11} s = 5,7 \times 10^3 \text{ лет}$$

Сценарий В

Условия (дополнительно к А). Хакер обладает специальным аппаратным средством, выполняющим алгоритм HL_CODE и проверяющим результат шифрования за одну микросекунду.

Предположим, что такое оборудование стоит 200 DM и что хакер может задействовать 10000 таких машин одновременно.

Усилия.

$$128 \times 2^{46} \times 1 \mu S / 10.000 = 9 \times 10^5 s = 10,5 \text{ дней.}$$

Даже при таких нереальных условиях злоумышленнику потребуется 10.5 дней на каждый 1Кб блок.

Атака с любыми знаниями алгоритма Hardlock будет настолько дорога и потребует столько времени, что хакер вместо этого решит проанализировать чип Hardlock, чтобы найти алгоритм генерации ключа.

Сценарий С

Условия (дополнительно к А). Хакер обладает алгоритмом генерации ключа, полученным путем шпионажа или анализа схемы ключа Hardlock. Компьютер хакера выполняет алгоритм генерации ключа за 20 микросекунд.

Усилия. Хакеру необходимо найти область параметров ключа Hardlock, т.е. усилия больше не зависят от количества данных, которое надо расшифровать. Алгоритм Hardlock несимметричный, следовательно, требуемые усилия будут следующими:

$$2^{47} \times (20 + 20) \mu s = 5,6 \times 10^{10} s = 1785 \text{ лет}$$

Это время может быть уменьшено до нескольких дней или часов при использовании специализированного оборудования. Однако это будет дорогой и продолжительный анализ, а также производство оборудования и чипа. На это могут уйти миллионы DM.

Заключение

Размер пространства ключей и усилия, необходимые для перебора всех ключей, не являются единственным фактором, определяющим безопасность шифра. В большей степени есть опасность математического усечения, когда шифр разрабатывался не по принципам криптографии. Это значит, что есть такой метод, который позволяет сократить время и усилия, необходимые для расшифровывания зашифрованных данных, по сравнению со случайным поиском. При разработке HL_CODE мы постарались сделать такое математическое усечение невозможным.

Напротив, хакер, решая относительно простые линейные выражения, может легко сломать обычно используемые генераторы псевдослучайных чисел со сдвигом регистров, даже если при этом период длины и размер пространства этих шифров весьма большие.

4 Договор о разработке и лицензировании

Aladdin Knowledge Systems GmbH & Co. KG
Gabriele-Münter-Str.1
82110 Germering

Все продукты компании Aladdin Knowledge Systems GmbH & Co. KG и связанных с ней предприятий (в дальнейшем обозначаются как «Аладдин»), включая демонстрационный набор, аппаратное обеспечение, дискеты Hardlock®, ключи и руководства по эксплуатации (в дальнейшем обозначаются как «Продукт»), как и все возможные в дальнейшем заказы, выполняются на следующих нижеперечисленных условиях. Если вы не согласны с этими положениями, верните нам демонстрационный ключ и все книги в течение 7 дней. Вам будет возвращена заплаченная сумма, исключая транспортные расходы и расходы на обработку.

4.1 Лицензия

«Аладдин» владеет всеми правами на Продукт и передает вам право на его пользование в соответствие со следующими условиями распределять сублицензии.

За исключением случаев, приведенных в параграфе 2, вы не имеете права передавать продукт третьему лицу или делать его доступным кому-либо еще, изменять части продукта, расчленять, делать их несовместимыми, проводить реинжиниринг, перерабатывать, улучшать или искать исходный код, за исключением условий, перечисленных в параграфе § 69e UrhG .

4.2 Использование программного обеспечения, продажа ключей Hardlock®

Вы имеете право встроить нашу программу в свое программное обеспечение, чтобы защитить его от пиратского обращения, в соответствии с описанием в руководстве. Вы имеете право передавать сублицензии для встроенной программы дистрибьюторам и конечным заказчикам в соответствии с ус-

ловиями этого договора. Вы имеете право сделать копию программы, чтобы ее заархивировать.

Вы имеете право перепродавать полученные от нас ключи Hardlock® конечным заказчикам, в соответствии с нашими гарантиями (см. параграф 3(с)).

4.3 Гарантия

«Аладдин» гарантирует вам на 12 месяцев с момента поставки следующее:

- а) программное обеспечение соответствует описанию в руководстве, если оно установлено на нужном аппаратном обеспечении;
- в) средство, на котором сохранена программа, свободно от ошибок;
- с) ключ Hardlock® в основном свободен от ошибок.

В случае нарушения этой гарантии «Аладдин» обязан починить или заменить бесплатно Продукт или его часть. В случае, если «Аладдин» не может это выполнить, вы имеете право отступить от договора купли/продажи.

Сообщение о дефекте должно быть прислано в письменном виде в компанию «Аладдин» в течение гарантийного времени и не позднее семи дней после выявления дефекта. Дефектные продукты должны быть отправлены назад дистрибьютору, у которого они были взяты, если они не были получены прямо у компании «Аладдин». Оплату пересылки и транспортной страховки берет на себя покупатель.

КРОМЕ ВЫШЕ ПЕРЕЧИСЛЕННЫХ ГАРАНТИЙ, АЛАДДИН НЕ ДАЕТ НИ УСТНЫХ, НИ ПИСЬМЕННЫХ ОБЕЩАНИЙ ОТНОСИТЕЛЬНО ПРОДУКТА, ОСОБЕННО ОТНОСИТЕЛЬНО ЕГО ИСПОЛЬЗОВАНИЯ И ИСПОЛЬЗОВАНИЯ В ОПРЕДЕЛЕННЫХ ЦЕЛЯХ.

4.4 Возмещение ущерба

Если «Аладдин» по каким-либо причинам должен возместить ущерб, то размер возмещения ущерба ограничивается стоимостью продукта, который нанес ущерб, или по причине которого был нанесен ущерб. Ни в коем случае это положение не каса-

ется случая, когда «Аладдин» должен возмещать ущерб, нанесенный из-за грубой халатности.

Не возмещаются ни убытки, которые произошли из-за нарушения вами договора, ни сопровождающие убытки, ни убытки состояния, убытки на основе потери данных или из-за претензий третьего лица.

4.5 Окончание договора

Неисполнение вами условий этого договора приводит к окончанию лицензии и этого договора. Положения параграфов 3,4,5 остаются в силе и после окончания этого договора.

Документация по Hardlock

Документация по **Hardlock** познакомит вас со всеми компонентами продуктовой линейки **Hardlock** и с возможностями защиты вашего программного обеспечения (ПО).

Документация включает в себя следующие разделы.

- **Техническое руководство по Hardlock** описывает принципы защиты ПО с помощью всех компонентов продуктовой линейки. Оно поможет вам выбрать правильную защиту для выполнения ваших задач.
- **Руководство по HL-Bistro** (визуальной среде программирования для Hardlock) описывает этот программный пакет, который позволяет вам проектировать систему защиты в привычной среде Windows.
- **Руководство по HL-Server** описывает применение Hardlock в сетевых приложениях.
- **Руководство по HL-Crypt** описывает применение автоматической защиты вашего ПО.
- **Руководство по HL-API** содержит обзор интерфейса прикладных базовых систем (спецификацию API) для ручной защиты ваших программ.
- **Раздел Понятия** поясняет основные термины и понятия, используемые в документации по Hardlock.

Все разделы дополняются интерактивной помощью (Help) к различным программам. Самые последние изменения и дополнения вы найдете в файлах Readme к программам. В руководствах и файлах интерактивной помощи (Help) применяются следующие типы шрифтовых выделений:

Обозначение	Функция	Пример
жирный	Понятия верхнего уровня Продукты	DatePCTV Hardlock Hardlock Twin
ЗАГЛАВНЫЕ БУКВЫ	Файлы и пути	HLDRV.EXE
Courier	Синтаксис	hlpatchup -m 29809

Содержание

1	Hardlock Bistro 2.1	1
1.1	Линейка продуктов Hardlock Bistro	1
1.2	Действия при защите	2
1.3	Сублицензирование и модификация с HL-LiMaS	2
1.4	Установка	3
1.4.1	Основные требования	3
1.4.2	Установка	4
1.5	Тестовый пакет	4
1.6	Демонстрационный режим для HL-LiMaS	5
1.7	Установки Hardlock Bistro	6
1.7.1	Диалог настроек	6
1.7.2	Переменные в Hardlock Bistro	6
2	Espresso 2.1	7
2.1	Функции программы Espresso	7
2.2	Программа Espresso-Assistent	8
2.3	Защита	8
2.4	Установки в Espresso	9
2.4.1	Установки Hardlock Bistro	9
2.4.2	Проекты в Espresso	9
2.4.3	Общие защитные установки	10
2.4.4	Установка сообщений об ошибках	10
2.5	Защита программных файлов	11
2.5.1	Действия при защите	11
2.5.2	Защитные установки для программы Win32	12
2.5.3	Защитные установки для программ Win16	12
2.5.4	Защитные установки для программ DOS	12
2.6	Защита файлов данных	13
2.7	Перечень учитываемых параметров командной строки	13
3	Cappuccino 2.1	17
3.1	Функции программы Cappuccino	17
3.2	Аппаратное обеспечение	17
3.3	Установки	18
3.4	Кодирование ключей Hardlock	18
3.4.1	Процесс кодирования	18
3.4.2	Подготовка аппаратного обеспечения	19
3.4.3	Заказы на кодирование ключей	19
3.4.4	Проекты для кодирования	21
3.4.5	Список заказчиков (только с HL-DB)	21
3.4.6	Тестирование ключей	21

3.4.7	Создание дубликатов	22
3.5	Создание файлов модификации	22
3.5.1	Условия для модификации ключей Hardlock	23
3.5.2	Получение необходимых данных	23
3.5.3	Создание файлов модификации	24
3.5.4	Модификация у конечного заказчика	25
4	HL-Upgrade 2.1: аппаратное средство модификации для заказчика	26
4.1	Конфигурирование расширения HL-Upgrade	26
4.1.1	Создание конфигурации с помощью файла INI	26
4.1.2	Создание конфигурации с помощью PATCHUP	27
4.1.3	Запуск HL-Upgrade с параметрами командной строки	29
4.2	Считывание данных из ключа Hardlock	29
4.3	Проведение модификации с файлом .VTC или блоком VTC	30
4.4	Проведение модификации с файлом .EXE	30
5	HL-RUS: модификация и сублицензирование	31
5.1	Сублицензирование	32
5.2	Модификация	32
5.3	Менеджер ключа Vendor-Key	33
6	HL-DB	35
6.1	База данных HL-DB	35
6.2	Присоединение банков данных	35
6.2.1	Предпосылки	35
6.2.2	Формы/Запросы в HL-Bistro	36
6.2.3	Конфигурация INI-файла с помощью запроса к базе данных	38
6.2.4	Конфигурация файла .INI с помощью выбора оператора в явном виде	39
6.2.5	Присоединение с помощью Microsoft Data Link	39
7	Gazzetta 2.1	41
7.1	Функции Gazzetta	41
7.2	Распределение клиентов и данные по Hardlock	41
7.3	Обработка данных по клиентам	42
7.4	Установка фильтров	43
7.5	Передача данных	43

8	Latteccino 2.1	44
8.1	Функции программы Latteccino	44
8.2	Установки	44
8.3	Тестирование функций HL-API	44
8.4	Передача буферов данных	45
8.5	Обзор выполненных функций	46
8.6	Создание исходной программы	46
8.7	Подключение в помощь RefKey/VerKeys	46

1 Hardlock Bistro 2.1

1.1 Линейка продуктов Hardlock Bistro

Hardlock Bistro - это рабочая среда, которая поможет вам защитить ваше программное обеспечение и при этом работать в привычной оболочке Windows.

С помощью программы **Hardlock Bistro** вы сможете:

- установить общие настройки **Hardlock Bistro** для всех программ, см. главу 1.7;
- кодировать ключи Hardlock, создавать описания кодирования ключей и сохранять информацию в базе данных с помощью программы **Cappuccino**, см. главу 2, и **HL-DB**, см. главу 6;
- управлять данными по ключам и заказчикам с помощью **HL-DB**, см. главу 6, и **Gazzetta**, см. главу 7;
- проводить модификации лицензий и создавать файлы модификаций для заказчиков с помощью **Cappuccino** и **HL-RUS**, см. главы 3.5 и 5.2;
- генерировать ключи вендоров для защиты лицензий с помощью **Vendor-Key-Manager**, см. главу 5.3;
- конфигурировать обновление (**HL-Upgrade**) для конечных заказчиков, см. главу 4;
- фильтровать и экспортировать данные по ключам с помощью **HL-DB** и **Gazzetta**, см. главу 7.5;
- автоматически защищать программные и относящиеся к ним файлы данных и при этом индивидуально определять механизм защиты с помощью программы **Espresso**, см. главу 2;
- тестировать ручную установку защиты с помощью **HL-API** и создавать исходный код с помощью программы **Latteccino**, см. главу 8.2.

Из основного окна **HL-Bistro** вы можете запустить основные программы одним нажатием на кнопку мыши.

1.2 Действия при защите

1. Закодируйте один ключ Hardlock с помощью программы **Cappuccino** (можно с помощью CP.EXE или CP32.EXE, см. *Hardlock-Техническое описание*). Для этого вам потребуется криптокарта (CPC) или USB-мастер ключ (USB-CryptoKey), которые переносят сигнатуру в ключ Hardlock.

Ключ Hardlock имеет индивидуальное кодирование, с помощью которого вы можете защитить ваши программы.

2. Защитите вашу программу автоматически с помощью **Espresso** (или вручную с помощью **HL-API**).

Программа будет зашифрована информацией из ключа и заданным вами защитным механизмом.

При применении программы зашифрованные данные передаются в ключ и возвращаются из него расшифрованными. Если ключа нет, расшифровка не состоится.

3. Закодируйте следующие ключи таким же образом и выдайте эти ключи с их программами. Также у вас есть возможность проводить индивидуальное лицензирование с **HL-RUS/LiMaS** (см. главу 1.3).

Дальнейшую информацию по принципам защиты и техническим данным вы получите в главе *Техническое руководство*.

1.3 Сублицензирование и модификация с HL-LiMaS

С помощью **HL-RUS** вы можете провести дифференцированное сублицензирование ваших программ через слоты памяти ключа и провести модификацию вашего ключа с помощью файла модификации. Необходимо учесть, что речь идет о ключе с памятью, так как часть памяти используется **HL-RUS**.

Использование **HL-RUS** необходимо при кодировании ключей Hardlock с помощью программы **Cappuccino** и защите программ с помощью **Espresso** (в ручном режиме - с помощью **HL-Crypt** или **HL-API**).

При кодировании области памяти ROM и RAM резервируются в ключе для **HL-RUS**. Модифицировать данные в этих областях можно только в тех ключах, у которых возможность модификации заложена при кодировании.

При шифровании программы вы должны точно определить, к какому слоту обращается программный файл, каково показание счетчика запусков программы, использован ли период использования программы. Определение условий производится в **Espresso** (в **HL-Crypt** или при ручной установке функций в **HL-API**). Опции условий (описание подключения ключа, периода действия и счетчиков запусков программы) могут быть заложены заранее при кодировании (в **Cappuccino**) и записаны при защите с помощью **Espresso**.

Если вы хотите защитить не только программные файлы, но и соотнести с разными частями программы различные модули, вы можете интегрировать эти модули в программу только вручную с помощью **HL-API** (см. раздел *HL-API*).

Информация по лицензиям и файлы модификации имеют многократную защиту. Только тот может изменить информацию по лицензиям, кто сам кодировал ключ Hardlock. За этим проследит Vendor-Key, защищенный паролем. Созданный при кодировании ключа уникальный серийный номер обеспечивает следующее условие: только единственный файл модификации может быть использован при модификации единственного ключа, имеющего это серийный номер.

Дальнейшие указания по HL-RUS имеются в главе 5.

1.4 Установка

1.4.1 Основные требования

Требования к аппаратным средствам

- IBM-совместимый компьютер с 486 процессором (рекомендуется Pentium).
- Минимум 16 Мб оперативной памяти (рекомендуется 32Мб).
- 20 Мб свободного пространства на жестком диске.
- 40 Мб свободного пространства на жестком диске при дополнительной установке HL-API с документацией.
- Привод CD-ROM.
- Графическая карта с разрешением мин. 800х600 пикселей и 256 цветами.

Операционная система

Windows 95/98, Windows NT 4.0 или Windows 2000.

Необходимые компоненты Microsoft

Для применения **Hardlock Bistro** необходимы следующие компоненты:

Компонент	95	98	NT4	Адрес для загрузки
DCOM (Distributed Component Object Model)	x			http://www.microsoft.com/com/tech/dcom.asp
MDAC (Data Access Components)	x	x	x	http://www.microsoft.com/data/download
Common Control Update	x		x	http://www.microsoft.com/msdownload/ieplatform/ie/comctrlx86.asp

В Windows 2000 имеются все необходимые компоненты.

Все версии этих компонентов, соответствующие выходу Hardlock CD-ROM, вы найдете на нем в каталоге SETUP.

1.4.2 Установка

Установка **Hardlock Bistro** производится с помощью установочной программы при установке Hardlock-CD или при ручном запуске программы BISTRO2.EXE.

При запуске с CD-ROM вы можете сразу установить необходимые драйверы.

Следуйте указаниям установочных программ и внимательно их читайте, прежде чем продолжать установку.

Кроме того, установите драйверы для базы данных с помощью MDAC21.EXE. Если вы используете Windows 95, то прежде всего вы должны установить DCOM95.EXE.

1.5 Тестовый пакет

С помощью тестового пакета вы можете проверить, как работает система защиты Hardlock. Вы получите необходимые программы, документацию и ключ Hardlock с демонстрационной кодировкой и адресом модуля 29809.

Чтобы проверить защиту Hardlock, поступите следующим образом:

1. установите программы и драйверы с CD-ROM;
2. установите демонстрационный ключ на параллельный порт вашего компьютера;
3. запустите программу **Espresso Assistant**.

С помощью этой программы вы можете быстро, в четыре шага, закодировать любой файл программы с помощью демонстрационного ключа Demo-Hardlock.

После шифрования программа запускается только с помощью демонстрационного ключа;

4. отключите демонстрационный ключ от компьютера и попытайтесь запустить программу.

Примечание. Имейте в виду, что шифрование с помощью демонстрационного ключа не представляет настоящей защиты, так как все ключи имеют один и тот же способ защиты. Действенную защиту вы получите только тогда, когда вы сами закодируете ключ уникальным образом.

1.6 Демонстрационный режим для HL-LiMaS

Если вы не имеете **HL-LiMaS**, вы можете лишь проверить функции **HL-DB** и **HL-RUS**.

База данных **HL-DB** ограничена десятью первыми пользователями. Это значит, что вы можете увидеть лишь часть файла заказчиков и файла Hardlock в программах **Cappuccino** и **Gazzetta**, и проверить, насколько удобно управление базой данных с помощью этих программ.

Функция **HL-RUS** для сублицензирования и создания модифицированных файлов применяется только для демонстрационного ключа с адресом 29809. Вы не сможете применять эту функцию для ключей, закодированных вами. Все созданные модифицированные файлы вы можете обезопасить только с помощью ключа демо-ключа вендора (Demo-Vendor-Key), который доступен всем пользователям **HL-Bistro** и не защищен паролем. Ключ Demo-Vendor-Key не является настоящей защитой. Вы не можете создавать своих собственных ключей, так как не имеете доступа к менеджеру **Vendor-Key-Manager**.

1.7 Установки Hardlock Bistro

1.7.1 Диалог настроек

Каждая программа Bistro предлагает вам большое число настроек. К ним относятся:

- используемый язык;
- определение критериев поиска после того, как ПО защищено с помощью Hardlock Bistro;
- обозначения индивидуально конфигурируемых полей базы данных по заказчикам и ключам;
- стандартное описание кодируемого ключа;
- каталоги, где хранятся сохраненные файлы;
- установки для криптокарты;
- установки для передачи модифицированных файлов по электронной почте;
- установки для базы данных.

Информации по отдельным полям вы получите в on-line Help (контекстная помощь).

1.7.2 Переменные в Hardlock Bistro

При определении установок электронной почты и для предустановки описания ключа Hardlock вы можете использовать следующие переменные:

Переменная	Описание
%date	текущая дата и время
%template	имя примененной программы
%dongle	описание ключа
%ma	адрес ключа
%sid	уникальный серийный номер (HL-RUS)
%dores1	имя поля DongleReserved1
%dores2	имя поля DongleReserved2
%c1	ID заказчика
%c2	наименование фирмы
%c3	имя заказчика

2 Espresso 2.1

2.1 Функции программы Espresso

При автоматической защите ваших программ ключом Hardlock с помощью **Espresso** возможно зашифровать один или несколько программных файлов и файлов данных. **Espresso** – это Windows-оболочка для системы **HL-Crypt**. В разделе *HL-Crypt* вы получите информацию по основам автоматической защиты. В **Espresso** также имеются все необходимые компоненты для надежной защиты программ.

В частности, **Espresso** позволяет:

- создавать и сохранять проекты, чтобы позднее при создании новых версий программ без больших затрат времени использовать те же параметры;
- зашифровывать различные типы программных файлов внутри проекта;
- определять детальную защиту для каждого программного файла;
- зашифровывать файлы данных и определять их связь с программными файлами (указывать, какие файлы и типы данных зашифрованы);
- при защите программ Win32 детально определить условия их работы (HL-RUS/LiMaS);
- определять сообщения об ошибках, когда возникают проблемы доступа к ключу;
- детально следить за ходом выполнения с помощью Log-файлов.

Если вы еще никогда не использовали программу **Espresso** и **Hardlock**, вы можете проверить защиту с помощью **Espresso-Assistent** (см. главу 2.2).

2.2 Программа Espresso-Assistent

Программа Espresso-Assistent позволяет защитить файл в три шага. При этом программа копируется и сохраняется под тем же именем во временном каталоге. Используемый при этом проект сохраняется в установленном заранее каталоге. Позднее проект можно модифицировать. Каталоги по умолчанию можно изменить в окне **File/Settings/Directories**.

При помощи программы **Espresso Assistent** можно проверить, как действует защита с помощью ключа Hardlock.

1. Установите тестовый ключ с адресом 29809 или индивидуально закодированный ключ в нужный порт вашего компьютера.
2. Запустите **Espresso Assistent** и следуйте его указаниям. Программа шифруется и работает только при использовании ключа.
3. Отключите ключ от компьютера и попытайтесь запустить программу.
4. Снова установите ключ в нужный порт.

Теперь вы можете запустить программу. Если у вас есть проблемы с запуском, проверьте, установлены ли необходимые драйверы (см. *Техническое описание*).

2.3 Защита

При защите вашей программы необходимо учесть нижеприведенные сведения и указания и выполнить перечисленные операции:

- общие данные по применяемому ключу, данные по шифрованию данных и по опциям HL-RUS;
- добавление программных файлов, которые вы хотите защитить, в папку **Программы**;
- определение защитных опций для каждого программного файла;
- привязка программных файлов к маске файлов (для защиты файлов данных);
- добавить файлы данных (в папку **Данные**);

- присоединение ключа, которым вы хотите защитить файлы;
- защитить программные файлы и файлы данных, отдельно или вместе (через меню или контекстное меню или через интерфейс).

Если вы выполнили все шаги, заданные программные файлы защищены. Заданные файлы данных зашифрованы и будут расшифрованы при доступе из программных файлов.

2.4 Установки в Espresso

2.4.1 Установки Hardlock Bistro

Общие установки, которые характерны для всех программ Bistro, вы можете использовать во всех программах Bistro. К ним относятся каталоги Bistro, установка поиска ключа HL-LiMaS, который защищает программу Hardlock Bistro. Они находятся в диалоговом окне **Файл/Установки (File/Settings)**.

Дальнейшую информацию вы получите в главе 1.7.1 и в помощи on-line Help.

2.4.2 Проекты в Espresso

Файлы, которые вы хотите защитить, объединяются в программе **Espresso** как проекты. Проекты сохраняются в каталоге Directories с расширением .ES2.

Создание новых проектов:

- в **Espresso** через **File/New**;
- с помощью **Espresso-Assistent**.

Открытие проектов:

- из **Explorer** двойным щелчком;
- из **Espresso** через **Open/File (Открыть/Файл)**.

Примечание. Проекты, созданные в **Espresso 2.0**, можно непосредственно открыть в **Espresso 2.1**.

Сохранение проектов

С помощью **File/Save (Файл/Сохранить)** можно сохранить проект, чтобы, например, повторно использовать его при создании новых версий программы. Лог-файл при этом не сохраняется.

2.4.3 Общие защитные установки

Для каждого проекта защиты необходимы следующие данные:

- адрес ключа Hardlock;
- данные, где программа должна искать ключ (локально, в сети или и то, и другое);
- защитный ключ длиной в восемь знаков и алгоритм зашифровки файлов данных. Мы рекомендуем алгоритм 6 как самый быстрый и надежный;
- образец (опционально) и относящийся к нему ключ Vendor-Key, с помощью которого защищается информация по лицензиям (**HL-RUS/LiMaS**). По предъявлению образца вы позднее можете получить при настройке **HL-RUS** определенные модули по выбору.

2.4.4 Установка сообщений об ошибках

Вы можете установить, какие сообщения об ошибках будут передаваться, если возникнут проблемы доступа к ключу и расшифровке защищенной программы.

Для индикации возможных ошибок в диалоговом окне предусмотрено сообщение об ошибках на том языке, на котором вы используете **Espresso**. Вы можете изменить сообщение об ошибках по вашему желанию, например, перевести на другой язык.

При применении HL-RUS/LiMaS вы можете установить, когда должны появиться предупреждающие сообщения о приближающемся окончании срока действия и о заполнении счетчика запусков.

2.5 Защита программных файлов

2.5.1 Действия при защите

После ввода всех установок (см. главу 2.4.3) добавьте подлежащие защите программные файлы в проект. Тип программы обозначается соответствующим символом.

Для каждого отдельного файла вы можете определить различные защитные механизмы. Выделите программный файл и выберите установки в правой части окна. Ко всем типам программных файлов относятся:

- задание места сохранения оригинального файла;
- задание места сохранения защищенного файла. Установки вы можете ввести в **File/Settings/Directories (Файл/Установки/Директории)** в поле **TEMP**;
- опционально, дополнительные параметры командной строки. Но сначала проверьте, нет ли желаемых параметров в интерфейсе. Перечень параметров, которые надо учитывать, вы найдете в главе 2.7;
- задание частоты фоновых запросов;
- задание силы мер против отладки и дизассемблирования.

Дальнейшие возможности зависят от типа файла (см. ниже). Оптимальные защитные механизмы определяются интерактивно.

Вы можете установить для каждого программного файла, к какому закодированному файлу данных он обращается (см. главу 2.6).

После того, как вы добавили все программные файлы и файлы данных, вы можете их защитить по отдельности или все вместе.

Как правило, вы можете выполнить различные шаги с помощью как главного, так и контекстного меню или с помощью иконок на рабочей панели. Дальнейшую информацию вы получите в интерактивном Help.

2.5.2 Защитные установки для программы Win32

Для защиты программ Win32 предусмотрены следующие опции:

- выдача сообщения при старте программы. Сообщения вы можете поместить среди **Сообщений об ошибках** (WARN_WAITBOX);
- активизация поддержки для оверлея. При активизации этой опции программы, которые обращаются сами к себе, могут видеть зашифрованные данные незашифрованными;
- исключение отдельных секций программы из шифрования;
- установка условий HL-RUS для старта программы. Вы можете задать три разные комбинации условий. Условия внутри одной строки соединяются с помощью AND, а их комбинации – с помощью OR.

2.5.3 Защитные установки для программ Win16

При защите программ Win16 у вас есть вышеперечисленные возможности. Если необходимо, вы можете добавить и другие механизмы через поле **Опции**. Информацию по возможным параметрам вы получите в разделе *HL-Crypt*. Используемые параметры обозначаются буквой **W**.

2.5.4 Защитные установки для программ DOS

При защите программ DOS у вас есть следующие опции, с помощью которых вы сможете установить особенности вашей DOS-программы:

- добавить антиотладочные механизмы (Debug-Killer, Quizmaster-Module);
- установить защиту внутреннего оверлея;
- подавить поддержку механизма FCB;

- активировать звуковые сигналы при выполнении фоновых задач;
- подавить поддержку многозадачности;
- допустить многослойную защиту программы;
- сжатие основного кода программы;
- выбор загрузочного модуля;
- выбор опции PPS/Speichercheck (контрольной суммы).

Отдельные механизмы детально описаны в разделе *HL-Crypt*.

2.6 Защита файлов данных

Вы можете защитить любые файлы данных, к которым обращается ваша программа. Если вы хотите защитить файлы данных, необходимо выполнить следующие шаги:

- вы добавляете в проект файл, который хотите защитить. Если вы далее проводите защиту проекта, этот файл будет зашифрован. Для этого применяются ключи по защите данных и алгоритмы, которые вы задали в ключе **Hardlock**;
- для отдельных программных файлов необходимо указать, какие файлы данных, к которым обращается программа, должны быть зашифрованы. Только тогда программа расшифрует файлы.

При этом у вас есть возможность задать отдельные файлы данных и использовать маски (например, *.txt). Вы можете также через контекстное меню добавить в папку **Data** соответствующие маски файлов и добавить в программу.

Детальную информацию вы получите в on-line Help.

2.7 Перечень учитываемых параметров командной строки

Большинство параметров командной строки системы HL-Crypt выведены в программе **Espresso** в интерфейс. Если вы хотите ввести дополнительные опции сверх предусмотренных, проверьте сначала, используются ли они в диалоговом режиме по отдельности или в комбинации.

Перечень учитываемых в Espresso параметров HL-Crypt

Назначение	Параметр	Описание	Применение
Выбор модуля	-m:MOD	адрес ключа Hardlock	D,W,W32
Безопасность	-b:FREQ	фоновое задание	D,W,W32
	-c:CRYPT	число модулей кода / Anti-Debug and Reverse Engineering	D,W
	-sec:LEVEL	уровень безопасности при шифровании данных в DOS и при шифровании программы	D,W,W32
	-y	автоматическое распознавание отладки	D
	-lock	прервать программу Debug-Interrupts	D
	-qm	Quiz-Master Trick модуль	D
online кодирование	-d:FILE	маска файлов для шифрования	D,W,W32
	-de:FILE	исключение маски файлов для шифрования	D,W,W32
	-k:KEY	ключи для шифрования данных	D,W,W32
	-dlx	поддержка защиты данных для оверлеев	W32
	-qslren	быстрая функция переименования	D,W,W32
Сеть (HL-Server)	-acc:MODE	способ доступа	D,W,W32
	-time:MIN	управлять таймаутом HL-Server	D,W,W32
Сублицензирование с HL-RUS (LiMaS)	-rusvk:FILE	определение файла Vendor-Key	W32
	-rus:MODE	определение условий для HL-RUS	W32
	-rusexp warn:DAYS	передача предупреждения об опасности за определенное время до окончания работы программы	W32

Назначение	Параметр	Описание	Применение
Сублицензирование с HL-RUS (LiMaS)	-ruscntwarn-UNIT	передача предупреждения об опасности за определенное время до окончания работы счетчика	W32
Дополнительно	-exsecnr	исключить секции программы из шифрования	W32
	-waitbox	передать сообщение о начале старта программы	W32
	-def:FILE	файл конфигурации	D,W,W32
	-lfcf	FCB (File Control Block) отключить поддержку	D
	-lbeep	деактивировать звуковой сигнал	D
	-comp	Root-Code Kompression	D
	-bf	допустить многоразовое кодирование	D
	-lcf	выключить мультизадачность	D
	-n	активировать нормальный загрузчик	D
	-ldtiny	активировать Tinyloader	D
	-!ovl	отключение внутреннего шифрования оверлея	D,W
	-!no:ID	вспомогательные параметры для экстендера DOS	D
	-ex:BYTES	исключить части программы от шифрования	D
	-o	загрузчик оверлев Microsoft	D
	-t	загрузчик турбо-оверлея Turbo-VROOMM-Format	D

Назначение	Параметр	Описание	Применение
Дополнительно	-p	другие загрузчики оверлея (нестандартный формат, например, PLINK86, BLINKER, RTLINK)	D

В разделе *HL-Crypt* вы получите дальнейшую информацию по отдельным параметрам.

3 Cappuccino 2.1

3.1 Функции программы Cappuccino

С помощью программы **Cappuccino** вы можете кодировать ключи и – в комбинации с **HL-DB** - управлять соответствующими данными. Эта программа предоставляет вам следующие возможности:

- составить список заказов на кодирование (см. 3.4.3);
- составлять и руководить проектами по кодированию (см. 3.4.4);
- кодирование с помощью параметров из одного проекта;
- создание файлов модификации в различных формах с параметрами из банка данных **HL-DB** или другого происхождения (см. 3.5).

3.2 Аппаратное обеспечение

Для кодирования ключей вам потребуется **криптокарта** или **Master Hardlock**.

Программное обеспечение для кодирования	Аппаратное обеспечение для кодирования	Аппаратное обеспечение для конечного заказчика
Cappuccino (альтернативно CP.EXE или CP32.EXE)	Crypto- Programmer- Card	Hardlock E-Y-E
		Hardlock Twin
		Hardlock Server External
		Hardlock PCMCIA
		Hardlock Internal
		Hardlock Server Internal
	USB-CryptoKey	Hardlock USB

С криптокартой вы можете кодировать только ключи с одной и той же базисной сигнатурой. Это могут быть ключи с 30 различными адресами и с более чем 40000 субкодов. Если вы захотите закодировать другие ключи с помощью этой криптокарты, вам дополнительно потребуется **Master Hardlock**.

Для кодирования ключа **Hardlock USB** вам не нужна криптокарта, а потребуется только **USB-CryptoKey**.

Все ключи Hardlock, кроме **Hardlock Server**, имеют стандартное исполнение, т.е. имеют стандартный чип ASIC- и, для вариантов с памятью, память (EEPROM).

3.3 Установки

Общие установки, которые характерны для всех программ Bistro, вы можете использовать во всех программах Bistro. К ним относятся каталоги Bistro, установка поиска ключа HL-LiMaS, который защищает программу Hardlock Bistro. Они находятся в опции **Файл/Установки (File/Settings)**.

Дальнейшую информацию вы получите в главе 1.7.1 и в on-line Help.

3.4 Кодирование ключей Hardlock

3.4.1 Процесс кодирования

При кодировании вы объединяете заданную **криптокартой** или **Hardlock Master** сигнатуру и задаваемый ими базовый адрес с одним из субкодов и адресом ключа пользователя.

Адрес ключа пользователя и полученный из сигнатуры базовый адрес образуют вместе адрес модуля ключа Hardlock. Сигнатура и субкод определяют способ кодирования ключа Hardlock.

Примечание. Тестовые модули всегда имеют адрес 29809.

У ключей с памятью при кодировании может быть запрограммирована и память.

Для ключей с памятью, в которых используется функция **HL-RUS** для дифференцированного сублицензирования и создания файлов модификации, используйте дополнительно ключ Vendor-Key, который защищен паролем (см. главу 5.3).

3.4.2 Подготовка аппаратного обеспечения

Все ключи Hardlock, кроме Hardlock USB и Hardlock card

Подключите ключ Hardlock напрямую или через адаптер к **криптоплате** (об устройстве и установке **криптокарты** см. *Техническое описание*).

Hardlock USB

Если вы хотите кодировать **Hardlock USB**, установите его в порт USB. Установите **USB-CryptoKey** в параллельный порт.

Hardlock PCMCIA

Если вы хотите закодировать **Hardlock PC-Card (PCMCIA)**, то установите адаптер в компьютер на разъем криптоплаты, а ключ – в адаптер.

Hardlock Internal и Hardlock Server Internal

При кодировании платы расширения используйте адаптер. Установите адаптер в компьютер на разъем криптоплаты, а плату, которая должна быть закодирована, в адаптер.

3.4.3 Заказы на кодирование ключей

Если вы хотите кодировать ключи, составьте сначала список «заказов». Вы можете составлять его с нуля, обрабатывать и сохранять для последующего выполнения заказов.

Внутри самого списка должны быть заказы пользователей и просто заказы. Список заказчиков содержит список ключей: сколько их, какие они, для каких заказчиков, по каким проектам должны быть закодированы. Второй список содержит только заказы безотносительно заказчиков.

Примечание. Обратите внимание при выборе проекта, что он играет решающую роль в изменении особенностей ключа Hardlock и в возможности создания файлов модификации.

Список заказчиков

Чтобы составить список заказчиков, выберите в правом окне:

- одного или несколько заказчиков, для которых должны быть закодированы ключи;
- проект Hardlock, который определяет основные особенности ключа;
- число ключей, которое должно быть закодировано для этих заказчиков.

Выберите команду **Внести заказ** для внесения в список заказов.

Список заказов

Чтобы составить список заказов, выберите в правом окне проект, по которому вы хотите кодировать новые ключи. Укажите также, сколько ключей вы хотите кодировать.

Заказы на разные ключи вносятся в список под разными символами (в зависимости от вида заказа). Вы можете изменить особенности ключей. В правой половине окна представлены особенности ключей, отмеченных в списке. При этом определяется вариативность проекта.

Если вы выбрали проект для вашей программы, который имеет функцию лицензирования и модификации (**HL-RUS**), вам нужно указать пароль для ключа Vendor-Key (о производстве ключей Vendor-Key см. главу 5.3). Только тогда можно правильно закодировать ключ.

Через контекстное меню вы можете выполнить заказы. Но вы можете и защитить список заказов для дальнейшего их выполнения.

Успешное выполнение отразится в строке статуса. А данные кодирования будут сохранены в базе данных **HL-DB**.

Примечание. Даже если у вас нет базы данных, вы можете для тестирования создать список заказчиков. Но в этом случае у вас будет доступ только к ограниченному числу строк базы данных.

3.4.4 Проекты для кодирования

Список проектов содержит проекты для кодирования ключей. Он содержит несколько проектов-моделей, которые вам не следует изменять. На основе этих проектов вы можете создавать свои собственные. Выберите для этого **Tools/Template-Manager**. Детальную информацию по созданию проектов вы получите в опции Помощь, которая открывается нажатием клавиши **F1** или через меню.

Примечание. Обратите внимание, что вы не должны изменять или стирать проекты по кодированию ключей. Иначе, например, параметры нельзя будет использовать для отбора ключей с одинаковыми свойствами.

3.4.5 Список заказчиков (только с HL-DB)

Список заказчиков содержит данные из базы данных. Если вы используете базу данных, которая является частью **HL-DB**, вы можете обработать эти данные программой **Gazzetta**. Если у вас нет **HL-DB**, найдите в базе данных часть данных, на которых вы могли бы проверить эти функции.

При создании списков по кодированию ключей вы можете выбрать, для каких заказчиков вы хотите кодировать ключи.

Остальную информацию о заказчиках, такую, как адрес, номер телефона и так далее, вы получите через контекстное меню.

Сюда же относится информация, какие ключи уже есть у заказчика. Вы можете познакомиться с этой информацией, если откроете опцию **Hardlock-Parameter** из **HL-DB**. Но это важно только для создания файлов модификации.

3.4.6 Тестирование ключей

Вы можете протестировать ключи и проверить, насколько они функциональны и закодированы по вашим представлениям. Для этого служит функция тестирования. Выберите пункт **Tools/**

Hardlock test. Дополнительную информацию вы получите в меню Помощь (on-line Help).

3.4.7 Создание дубликатов

В исключительных случаях вы можете сделать дубликаты уже закодированных ключей. Проследите, чтобы эти дубликаты не были сохранены как таковые в банке данных. Делайте дубликат только тогда, когда предыдущий ключ больше не функционирует, так как иначе банк данных не будет отражать действительную ситуацию.

Чтобы создать дубликат, сделайте следующее:

1. отметьте в списке закодированных ключей тот ключ, чей дубликат вы хотите создать;
2. активируйте правой клавишей контекстное меню и выберите команду **Duplicate**.

Заказ поступит в список и будет отмечен как только что выполненный (помечен «молнией»);

3. отметьте заказ в списке и выберите **Hardlock code**.

Дубликат будет закодирован без соответствующей строки в банке данных.

3.5 Создание файлов модификации

В программе **Cappuccino** вы можете не только кодировать ключи, но и с помощью дополнительной функции **HL-RUS** создавать файлы модификации. С этими файлами ваши заказчики смогут сами модифицировать свои ключи.

Для проведения модификации вам необходима информация о ключе, для которого проводится модификация. У вас есть разные возможности получить эту информацию:

- из банка данных (с помощью **HL-DB/LiMaS**);
- от заказчика в файле CTV, который вы импортируете;
- от заказчика из текстового блока, который вы можете взять из промежуточного хранения;
- напрямую из ключа.

Вы создаете с помощью этой информации, учитывая желаемые изменения, файл модификации и передаете его своему заказчику. Вы можете создать файл .EXE или файл с расширением .VTC. В качестве альтернативы вы можете провести модификацию прямо в ключе.

Cappuccino дает вам возможность создать файл модификации как сообщение по электронной почте. У вас есть выбор между созданием вложения для e-mail в виде файлов .VTC или .EXE.

Заказчик может запустить файл с расширением .EXE или с помощью **HL-Upgrade** применить файл .VTC к своему ключу.

Для дополнительной безопасности при создании модификации необходим ключ Vendor-Key, который обеспечит возможность создания файла модификации только вами. Данными лицензий нельзя будет манипулировать. Этот ключ защищен паролем. Вы можете управлять ключом Vendor-Key и соответствующим паролем с помощью **Vendor-Key-Manager**. Можно использовать и пункт меню **Tools/Vendor-Key-Manager** программы **Cappuccino**.

3.5.1 Условия для модификации ключей Hardlock

С помощью **HL-RUS/LiMaS** вы можете предусмотреть для всех ключей с памятью возможность создавать модификации. При этом **HL-RUS** использует часть памяти ключа, чтобы идентифицировать ключ.

Поэтому сразу при первом кодировании ключа или при создании проекта нужно указать, хотите ли вы использовать в дальнейшем эту опцию. Иначе позднее создание модификации будет невозможно.

3.5.2 Получение необходимых данных

У вас есть различные возможности получить нужные лицензии для модификации.

Из банка данных (с помощью HL-DB/LiMaS)

Если у вас есть **HL-DB**, вы можете на базе данных заказчика и ключа создать файлы модификации из вашей базы данных. Для этого выберите из списка заказчиков одного. Активируйте опцию **Hardlock-Parameter** из **HL-DB**, выберите ключ, для ко-

торого вы создаете модификацию. Данные могут быть перенесены в список заказов.

От заказчика в файле .CTV, который вы импортируете

Файл CTV создается вашим заказчиком с использованием **HL-Upgrade**. При этом необходимые данные берутся из ключа, см. главу 4.

Вызовите команду **File/Import CTV-files** для импортирования файла .CTV. Данные будут сохранены как заказ в списке заказов.

Напрямую из ключа

Если вы получите данные не в виде файла, а как блок текста по электронной почте, вы можете его скопировать в промежуточный буфер и оттуда поместить в **Cappuccino**. Для этого выберите **File/Read CTV-Info from Clipboard**.

3.5.3 Создание файлов модификации

После того как вы внесли заказы в список заказов, вы можете для каждого файла в правом окне установить новые свойства ключа. Изменения возможны в следующих рамках:

- время окончания;
- лимит счетчика;
- свободные позиции (слоты).

Возможности модификации зависят от использованного проекта и от свойств закодированного ключа.

После того, как вы установили новые свойства, вы можете создать новые файлы модификации на рабочем столе, нажав **Generate** (создать). При этом у вас есть выбор разных форматов:

- как файл .VTC или .EXE. Файлы помещаются в заданной внизу опции **File/Settings** и могут быть переданы заказчику;
- напрямую — данные направляются прямо на присоединенный ключ;
- как e-mail — с вложением в виде файла .EXE блока VTC. Файл отправляется на заданный адрес e-mail, который

находится в банке данных. Установки и описание вы можете установить в опции **File/Settings**.

3.5.4 Модификация у конечного заказчика

Используя файл модификации .EXE, ваш заказчик проведет модификацию ключа Hardlock, либо использует программу HL-Upgrade, чтобы применить созданные вами файлы .VTC, см. главу 4.

4 HL-Upgrade 2.1: аппаратное средство модификации для заказчика

HL-Upgrade – это устройство, с помощью которого ваш заказчик для создания модификации может считывать необходимые файлы из своего ключа и с помощью которого он сможет провести модификацию своего ключа (о создании файла модификации см. главу 3.5.3).

HL-Upgrade конфигурируется автоматически, когда вы создаете файл EXE в **Cappuccino**. Если вы предоставляете заказчику расширение **HL-Upgrade**, чтобы считывать данные из Hardlock или чтобы применить файл .VTC к ключу, сначала сконфигурируйте расширение сами. Альтернативно заказчик может запустить расширение через командную строку.

4.1 Конфигурирование расширения HL-Upgrade

Не обязательно конфигурировать расширения **HL-Upgrade**, если вы берете данные для модификации из банка данных Hardlock и создаете файл модификации из файла .EXE.

Для конфигурации важно, когда ваш заказчик должен считать данные из Hardlock и когда должен применить модификацию в виде файла .VTC к ключу.

При конфигурации задайте, к какому ключу будут обращаться с помощью программы **HL-Upgrade**. Необходимые данные поместите либо в файле .INI, либо прямо в программный код с помощью программы **PATCHUP**.

4.1.1 Создание конфигурации с помощью файла INI

Создайте в текстовом редакторе файл .INI с таким же именем файла, как и у файла .EXE (как правило, HLUP.INI), где вы зададите опционально адрес ключа, а также зададите Reference Key и Verify Key. Задайте отступ с помощью [HLUPGRADE] и обозначьте параметры и аргументы в следующем формате:

- [Switch]=[Argument]

У вас есть следующие параметры на выбор:

Параметр	Данные
ModAdr	адрес ключа
RefKey	Reference Key (опционально), задано в шестнадцатеричном коде
VerKey	Verify Key (опционально), задано в шестнадцатеричном коде
Access	режим доступа к ключу Hardlock - <code>local_device</code> для локального ключа <code>net_device</code> для Hardlock Server <code>dont_care</code> для двух режимов доступа

Пример для файла .INI:

```
[HLUPGRADE]
modadr=29809
refkey=484152444c4f434b
verkey=184c97f0c07a0888
access=local_device
```

Сохраните файл .INI в том же каталоге, что и HLUP.EXE, и передайте его вместе с HLUP.EXE вашему заказчику.

4.1.2 Создание конфигурации с помощью PATCHUP

Откройте окно данных MS-DOS и запустите программу PATCHUP с перечисленными ниже параметрами. Задайте параметры и аргументы в следующей форме:

-[Switch] [Argument]

Задание большинства параметров опционально. Только адрес модуля должен быть задан.

Parameter	Данные
-m	адрес модуля
-r	Reference Key (опционально), данные задаются в шестнадцатеричном коде или auto для автоматического генерирования
-v	Verify Key (опционально), данные задаются в шестнадцатеричном коде или auto для автоматического генерирования
-i	имя файла ввода, опционально, если не задано - HLUP.EXE
-o	имя файла выхода (опционально), если не задано - как в файле ввода
-u	задайте здесь имя файла VTC, если вы хотите создать файл .EXE с именем файла VTC
-a или -асс	режим доступа к ключу Hardlock - local_device для локальных ключей net_device для Hardlock Server dont_care для двух режимов

Примечание. Для задания **Auto** должен быть присоединен соответствующий ключ Hardlock.

С этим параметром вы можете конфигурировать HLUP.EXE для определенного ключа Hardlock (с параметрами **-m**, **-r** и **-v**) или для определенного файла .VTC (с параметром **-u**).

У вас есть здесь возможность, например, определить имя программы модификации:

```
patchup
-m 29809
-o meinprogramm.exe
-i «c:\hardlock\hlup.bin»
-r 484152444c4f434b
-v 184c97f0c07a0888
```

Для модуля с адресом 29809 из HLUP.BIN создается программа MEINPROGRAMM.EXE. Вы можете передать ее вашему заказчику. Заказчик может с этой программой провести модификацию ключа с адресом 29809.

Примечание. Укажите данные, которые содержат пробелы, в кавычках, например: D:\My Files\HL-Upgrade.

Список параметров вы получите с помощью параметра **-?**.

4.1.3 Запуск HL-Upgrade с параметрами командной строки

Есть возможность запустить HL-Upgrade из командной строки (например, если нет файла .INI).

-[Switch] [Argument]

Задание большинства параметров опционально. Должен быть только указан адрес ключа.

Параметр	Данные
-m	адрес ключа
-r	Reference Key (опционально), задано в шестнадцатеричном коде
-v	Verify Key (опционально), задано в шестнадцатеричном коде
-a или -асс	режим доступа к ключу Hardlock - <code>local_device</code> для локального ключа <code>net_device</code> для Hardlock Server <code>dont_care</code> для двух режимов доступа

Когда ваш заказчик загрузит расширение **HL-Upgrade** с параметрами командной строки, файл .INI и конфигурация с помощью **PATCHUP** не будут учитываться.

4.2 Считывание данных из ключа Hardlock

Если вы не можете обратиться к необходимым данным для ключа в банк данных или напрямую в ключ или не хотите этого, вы можете получить данные от вашего заказчика. Заказчик считывает данные с расширением HL-Upgrade из своего ключа.

Передайте вашему заказчику программный файл, специально конфигурированный для ключа с подходящим файлом .INI.

1. Заказчик присоединяет свой ключ.
2. Заказчик запускает свою программу и выбирает опцию **Get Data**. При этом у него есть выбор, писать ли данные в промежуточный буфер, например, чтобы добавить их в e-mail, или создать файл RUS с расширением .CTV (Customer-To-Vendor). Имя соответствует серийному номеру ключа, файл сохраняется в активном каталоге.

4.3 Проведение модификации с файлом .VTC или блоком VTC

Если вы создаете файл модификации как файл .VTC или посылаете блок VTC по электронной почте, ваш заказчик может применить их с помощью программы **HL-Upgrade** к своему ключу.

1. Передайте вашему заказчику файл .VTC, чье имя соответствует серийному номеру ключа, или подходящий блок VTC (например, по электронной почте). Заказчик должен также располагать сконфигурированным программным файлом программы **HL-Upgrade**.
2. Заказчик присоединяет свой ключ и запускает программу расширения **HL-Upgrade**.
3. Заказчик запускает **RUS-Write Data**. При этом у него есть выбор, считывает ли он данные из файла .VTC или применяет блок VTC из промежуточного буфера.

Файлы модификации применяются к ключу, и активируется информация по лицензиям.

4.4 Проведение модификации с файлом .EXE

Если вы создаете файл модификации как файл .EXE, ваш заказчик запускает этот файл .EXE. Имя этого файла состоит из автоматически заданного серийного номера ключа Hardlock и расширения .EXE.

Заказчик присоединяет свой ключ и проводит **RUS-Write Data**.

Файл модификации применяется к ключу и активируется информация по лицензиям.

5 HL-RUS: модификация и сублицензирование

Функция **HL-RUS** позволяет провести дифференцированное сублицензирование ваших программ Win32 по модулям и тем самым создать возможность модификации вашего ключа с помощью файла модификации. Предпосылкой к этому является наличие ключа с памятью, так как часть памяти использует программа **HL-RUS**.

Применение функции **HL-RUS** закладывается при кодировании ключа программой **Cappuccino** и при защите программы с помощью **Espresso**, **HL-Crypt** и **HL-API**. При этом в ключе резервируются части ROM и RAM памяти для программы **HL-RUS**. Итак, вы можете создавать файлы модификации только для тех ключей, у которых использование функции заложено при кодировании.

При шифровании программ установите, к какой сублицензии относится программный файл, сколько единиц счетчиков использует программа при запуске и соответствует ли время окончания программы времени ее работы. Распределение происходит в **Espresso** (альтернативно в **HL-Crypt** или при ручном подключении с **HL-API**). Выбор сублицензий задается при кодировании проекта в **Cappuccino**.

Если вы хотите дифференцированно не только защитить программные файлы, а еще и назначить частям программы разные сублицензии, то возможно назначение сублицензий произвести только вручную с помощью **HL-API**.

Информация по лицензиям и файлы модификации имеют многократную защиту. Ключ Vendor-Key, имеющий пароль, обеспечивает возможность изменения информации по лицензиям только тому, кто кодировал ключ. Серийный номер ключа обеспечивает применение одного файла модификации только к тому ключу, который имеет этот неизменяемый номер.

5.1 Сублицензирование

По отдельности сублицензирование можно провести так, что разные сублицензии освобождаются независимо:

- при использовании Hardlock Server число лицензий на каждый модуль также может определяться отдельно;
- при использовании Hardlock Server можно установить время окончания действия для каждой позиции независимо;
- возможна установка лимита счетчиков запуска программы.

Данные по лицензиям записываются в зарезервированную для этого область и защищены от манипуляций ключом Vendor-Key. У ключей Hardlock-Server файл лицензии защищен файлом с расширением .ALF. Изменения возможны только с файлом модификации, который был создан при применении того же самого ключа Vendor-Key.

5.2 Модификация

Возможность модификации закладывается при первом кодировании ключей. При этом резервируется часть памяти для информации по лицензиям. Информация по лицензиям защищена от манипуляций ключом Vendor-Key.

Если вы заложили возможности модификации, вы сможете позднее создать для ваших заказчиков файлы модификации, с помощью которых новые позиции будут разрешены и будет установлена новая дата окончания работы. Файлы модификации вы создаете в программе **Cappuccino** (см. главу 3.5).

Вы можете передать файлы модификации с расширением .EXE, .VTC или как блок текста VTC вашим заказчикам. Уже на месте они могут провести эту модификацию с программой **HL-**

Upgrade на своем ключе. Конфигурация программы **HL-Upgrade** описывается в главе 4.1.

5.3 Менеджер ключа Vendor-Key

С помощью менеджера ключа **Vendor-Key** вы сможете создавать ключи Vendor-Key и управлять паролями. Один ключ Vendor-Key вам нужен, если вы используете функцию **HL-RUS**. Ключ Vendor-Key служит для того, чтобы снабдить ключ цифровой подписью (сигнатурой).

Цифровая подпись

- служит для того, чтобы файл модификации для ключа Hardlock смог создать только тот, кто закодировал ключ;
- защищает информацию по лицензиям от манипуляций.

Открытая часть ключа Vendor-Key (данные исполняющей системы) интегрируется в программу, которую нужно защитить. Вместе с данными исполняющей системы сигнатура, созданная в ключе, проверяется на ее правильность.

Примечание. Аккуратно храните свой ключ Vendor Key и берегите его от утери, так как без этого ключа вы не сможете больше изменить информацию по лицензиям.

Необходимые данные исполняющей системы для подлежащей защиты программы вы можете создать в менеджере ключа Vendor-Key-Manager. При этом у вас есть выбор в языках программирования и структуре данных. Вы можете создать данные исполнительной системы в виде файла. Файл сохранится там же, где и ключ Vendor-Key, в той же директории. Директо-

рию задайте в **Cappuccino** в диалоге **File/Settings/**. В качестве альтернативы возможно записать данные исполнительной системы в промежуточный буфер, чтобы напрямую с помощью текстового редактора ввести в вашу программу.

К менеджеру ключа Vendor-Key вы можете обратиться через основное меню в **Cappuccino** или из программы **Latteccino** через **Tools/Vendor-Key-Manager**.

6 HL-DB

6.1 База данных HL-DB

База данных **HL-DB** содержит данные по ключам Hardlock, включая информацию по проектам, список данных по заказчикам и данные об уникальных номерах ключей.

В программе **Gazzetta** вы можете увидеть как данные из базы данных по заказчикам, так и данные по ключам Hardlock. **Gazzetta**, кроме того, может вносить данные в базу данных по заказчикам, если вы используете вместе с базой данных ключей и базу данных заказчиков. Для создания своей собственной базы данных ниже будет приведена необходимая информация.

Другие программы Bistro имеют разный доступ в базу данных. **Espresso** читает только данные из базы данных Hardock, в то время как **Cappuccino** – из присоединенной базы данных заказчиков и базы данных Hardock и пишет в базу данных Hardock.

Доступ программы Bistro к базам данных:

База данных	Доступ на чтение	Доступ на запись
база данных Hardlock	Cappuccino Gazzetta Espresso	Cappuccino Gazzetta
база данных по заказчикам	Gazzetta Cappuccino	Gazzetta (если используется интегрированная БД заказчика)

6.2 Присоединение банков данных

6.2.1 Предпосылки

Прежде чем присоединить свою собственную базу данных, проверьте следующее:

- наличие для вашей базы данных или вашего сервера базы данных, драйвера или, соответственно, программное обеспечение клиента, которые могут быть запрошены через ODBC или OLEDB. Они должны быть установлены на вашем компьютере;

- имеете ли вы доступ к базе данных или таблице заказчиков и необходимые пароли для этого доступа;
- в таблице заказчиков должно иметься идентификационное поле с однозначно понимаемым номером с длиной в 4 байта (Long Integer), например, первичный ключ таблицы (primary key). Наличие индекса по этому полю повышает скорость доступа, но индекс не обязателен. Преимуществом является также, если значения идентификационного поля не будут повторяться, то есть если после погашения записи данных это значение больше не используется. В случае, если такое поле не существует, есть еще одна возможность: создать такое поле с помощью функции вашей базы данных;
- у вас имеется информация о строении таблицы заказчиков (имена полей и типы данных).

HL-Bistro устанавливает в качестве вычислительной машины базы данных Microsoft Access, который имеет механизм внутренних запросов через провайдера OLEDB для Jet машины 4.0. В качестве базы данных API применяются объекты ActiveX Data Objects (ADO).

Присоединение чужой базы данных по этому принципу к **HL-Bistro** не производится, а используется путем выбора подходящего OLEDB провайдера. Основные провайдеры позволяют подключиться с источниками данных из MS-Access, MS SQL-Server, Oracle и ODBC. Эти возможности предоставляются вам после установки Microsoft Data Access Components (MDAC), требуемой **HL-Bistro**.

Выясните у производителя вашей базы данных, какие из ее возможностей являются самыми быстрыми и беспроблемными.

6.2.2 Формы/Запросы в HL-Bistro

HL-Bistro требуются для общения со своей базой данных определенные запросы Query/View с установленными идентификаторами массива. Эта форма (View) будет занесена как SQL-Select-Statement в файл INI (HLDB.INI).

Список идентификаторов массива

Идентификатор массива	Тип данных	Описание
Nr	нумерационные (Long Integer)	однозначный номер идентификации
IdentNr	алфавитно-цифровой (текст)	-ID заказчика (номер заказчика)
Firma	алфавитно-цифровой (текст)	имя фирмы
Name	алфавитно-цифровой (текст)	имя партнера
Adresse	алфавитно-цифровой (текст)	адрес (например, улица, город и т.д.)
Tel	алфавитно-цифровой (текст)	номер телефона
Fax	алфавитно-цифровой (текст)	номер факса
eMail	алфавитно-цифровой (текст)	адрес электронной почты
Customer-Reserved1	алфавитно-цифровой (текст)	индивидуально сконфигурированное поле
Customer-Reserved2	алфавитно-цифровой (текст)	индивидуально сконфигурированное поле

Есть две возможности провести запрос:

- вы создаете форму в вашей базе данных, видимые поля этого запроса должны иметь вышеперечисленные наименования;
- запрос в явном виде будет внесен в файл .INI. Здесь также будут переставлены идентификаторы полей.

Последовательность этих идентификаторов должна соблюдаться. Все идентификаторы должны быть введены.

В файле .INI – HLDB.INI находятся строки загрузки, которые важны для присоединения базы данных в разделе Заказчик (Customer).

Ключ	Описание
Connection	ADO Connection String
View	Select-Statement для Query/View
Link	имя идентификатора массива, соответствует первому полю Query/View
SortFirma, SortName, SortIdent	оригинальные имена полей, по которым Query/View сортируется.

Ключ Connection сильно зависит от применяемой машины базы данных. В файле CONNECTIONEX.DOC на Hardlock-CD имеется описание наиболее распространенных конфигураций. Дополнительную информацию вы найдете в Интернете по адресу <http://www.microsoft.com/data>.

6.2.3 Конфигурация INI-файла с помощью запроса к базе данных

Если вы составили запрос под именем Bistro Customers с выше перечисленными идентификаторами массива в ваш банк данных, появятся следующие записи в HLDB.INI:

Ключ	Значение
View	"select * from Bistro Customer"
Link	"nr"
SortFirma	"Firma asc"
SortName	"Name asc" или, например, "Name asc, email desc"
SortIdent	"IdentNr asc" или "IdentNr desc"

Значение SQL asc или desc определяет, идет ли сортировка по возрастанию или по убыванию. Конечно, здесь могут быть внесены множественные сортировки по вашему желанию, если еще остались дополнительные поля в запросе. Здесь, думается, вместо IdentNr должна быть сортировка по дате создания записи данных заказчика с названием IdentNr desc.

6.2.4 Конфигурация файла .INI с помощью выбора оператора в явном виде

Если вы в явном виде зададите выбор оператора (например, из-за отсутствующих прав в базе данных), вам придется использовать оригинальные имена идентификаторов полей в ключах Link, SortName, SortFirma, SortIdent.

Допустим, ваша таблица заказчиков называется «Заказчики» и состоит из полей No., CustID, Company, FirstName, Surname, Street, ZIP, City, Tel, Fax, E-mail.

Ключ	Значение
View	" select No. as nr, CustID as IdentNr, Company, [Firstname]+" "+"[Surname] as name, [Street] + chr(13) + chr(10) + [ZIP] + " " + [City] as Adresse, telefon as tel, fax, email from Kunden"
Link	"No"
SortFirma	"Company asc"
SortName	"Surname asc, FirstName asc"
SortIdent	"CustID asc"

По этому выбору оператора можно определить, что и базы данных могут выглядеть полностью по-другому. На самом деле речь идет о способности упомянутых баз данных поддерживать такую функцию, которая обеспечивает наибольшую гибкость.

Так как программа **HL-Bistro** поддерживает многострочные поля «Фирма» и «Адрес», в этом примере адрес создается объединением многих полей и выражений.

6.2.5 Присоединение с помощью Microsoft Data Link

У вас есть возможность присоединить вашу базу данных через Microsoft Data Link (UDL-файл). Поступайте следующим образом:

- создайте ссылку, например, с помощью Explorer или правой клавиши мыши;

- щелкните дважды мышью на ссылку и введите необходимые данные для соединения и т.д. Используйте помощь для диалога.

Примечание. Указание пользователя и пароля в **Bistro** невозможно. Если доступ в базу данных зависит от этого, вы должны указать пользователя и пароль при создании ссылки Data Link и активировать опцию **Запись пароля**;

- занесите соединение в файл HLDB.INI:

`Connection = «File Name=\somepath\pubs.udl;»`

Присоединение через ссылку делает удобной передачу данных через диалог.

7 Gazzetta 2.1

7.1 Функции Gazzetta

В Gazzetta вы имеете доступ на запись и чтение к базе данных. Если вы используете базу данных заказчиков, которая является частью базы данных Hardlock, вы также имеете к ней доступ на запись. Если вы используете свою собственную базу данных заказчиков, то в программе **Gazzetta** вы сможете только просматривать ее.

В частности, программа **Gazzetta** позволяет:

- изменять список клиентов;
- добавлять, обрабатывать и сортировать данные по клиентам;
- устанавливать фильтры;
- экспортировать данные.

7.2 Распределение клиентов и данные по Hardlock

В программе **Gazzetta** вы можете просмотреть, каким клиентам какие ключи распределены. Вы можете изменить это распределение. Прежде всего это имеет смысл, если вы сначала кодируете ключи, не назначая их определенным клиентам, а только после при пересылке ключей распределяете их между клиентами.

При этом вы можете применить метод Drag&Drop:

1. выделите ключ, который вы хотите послать другому клиенту, левой кнопкой мыши. Держите кнопку нажатой и перетаскивайте ключ к клиенту, которому он предназначен;
2. отпустите кнопку.

Таким образом ключ распределен к нужному клиенту.

Вариант: вы можете вырезать и добавить строки регистрации с помощью контекстного или основного меню.


Примечание. Убедитесь, чтобы изменения в распределении клиентов прошли без неожиданных последствий, поскольку име-

ется опасность, что база данных Hardlock не будет показывать действительную ситуацию.

7.3 Обработка данных по клиентам

Если вы используете интегрированную таблицу заказчиков, вы можете обработать в программе Gazzetta данные по клиентам. Вы можете добавить новых клиентов, изменять и стирать данные по заказчикам.

Добавление заказчиков

Выберите **Data/New Customer** или символ  , чтобы добавить заказчика. В правой части окна вы можете ввести новые данные по заказчику.

Обработка данных по заказчикам

Нажмите в левой половине окна заказчиков, чьи данные вы хотите обработать.

Вариант: вы можете дважды щелкнуть мышью на заказчиков в правом списке, в правой части окна вы можете изменить данные по заказчикам.

Стереть данные по заказчикам

Выделите заказчика, чьи данные вы хотели бы стереть. Сотрите данные (погасите) через контекстное или основное меню с помощью опции **Edit/Delete**.


7.4 Установка фильтров

В программе **Gazzetta** вы можете применить заданные или свои фильтры на данные по Hardlock. Отфильтрованные данные можно затем экспортировать, для того, например, чтобы создать сообщения для заказчиков с выбранными ключами.

Установка заданных, заранее определенных фильтров

Выберите один из фильтров в подменю **Filter**, чтобы отфильтровать данные по заказчикам или хранящиеся данные по определенным критериям.


Установка собственных фильтров

Чтобы определить и установить собственные фильтры, выберите опцию **Filter/Manage** или символ . Только что определенные фильтры, как и определенные заранее, отображаются в меню **Filter**.

Выключение фильтров

Выберите опцию **Filter/Filter off** или символ , чтобы снова увидеть все данные.

7.5 Передача данных

Вы можете экспортировать отфильтрованные и неотфильтрованные данные, чтобы использовать их в дальнейшем в других программах. При этом вы можете задать их формат. Выберите опцию **Data/Export** или символ .

Для экспорта данных у вас есть выбор разных форматов. В зависимости от формата вы можете сохранить данные как в файле .CSV, так и экспортировать их в виде таблицы в существующую базу данных.

8 Latteccino 2.1

8.1 Функции программы Latteccino

Latteccino - это вспомогательное средство и средство тестирования для реализации ручной защиты при помощи ключа Hardlock. При этом вы защищаете свои программы введением вызовов функций в исходную программу.

Latteccino обеспечивает вам следующие возможности:

- вы можете проверить, как действует ручная защита с применением интерфейса прикладного уровня HL-API. При этом **Latteccino** ведет себя как защищенная программа, из которой вызываются функции HL-API (см. главу 8.3);
- у вас есть обзор выполненных функций, выданных сообщений о статусе и вызывающей программе;
- при выполнении функций или независимо от этого вы можете создать программу для вызова функций (см. 8.6);
- вы можете произвести пару ключей Reference-Key/Verify-Key, с помощью которых можно выяснить, что имеющийся ключ Hardlock именно тот, который нужен (см. главу 8.5).

Детальное описание функций графического интерфейса HL-API вы получите в описании *HL-API*.

8.2 Установки

Общие установки, которые относятся ко всем программам Bistro, вы можете создать во всех этих программах. К таким установкам относятся, например, каталоги Bistro и определение поиска ключа HL-LiMaS, который защищает Hardlock Bistro. Диалог ведется с помощью **File/Settings**. Дальнейшую информацию вы получите в главе 1.7.1. и в Help.

8.3 Тестирование функций HL-API

Прежде чем тестировать функции, вы должны подключить ключ Hardlock к вашему компьютеру. Различные функции будут запрашивать этот ключ. Перед выполнением дальнейших функций

вы должны выполнить функцию, с помощью которой программа подключается к ключу. Это функции HL_LOGIN или HLM_LOGIN.

В левой части окна программы вы можете найти перечень функций HL-API, разделенный на функциональные группы. Среди них вы можете выбрать функции, которые вы хотели бы выполнить. У некоторых функций должен быть буфер данных (см. главу 8.4).

В правой части окна вы найдете диалог. Теперь вы можете ввести параметры функции. Далее вы тестируете функции с помощью опции **Execute** под диалогом или с помощью символа на строке символов. Помощь вы получите, нажав **F1**.

Примечание. Помните, что выполнение некоторых функций может разрушить кодирование ключа, если он закодирован для HL-RUS.

Эти функции обозначены восклицательным знаком на желтом фоне.

В нижней части диалогового поля вы видите краткое пояснение к функциям **HL-API**. Чтобы получить подробное описание этих функций, нажмите <Shift> и F1 или иконку на строке символов.

После выполнения функции рядом с иконкой **Execute** видно заданное значение статуса в форме номера и в виде текста и ниже краткое объяснение текущего состояния.

8.4 Передача буферов данных

При выполнении некоторых функций ключа должен быть передан буфер данных. В первую очередь это функции HL_CODE и HL_WRITEBL, с помощью которых вы можете задать любые данные. **Latteccino** предоставляет вам панели ввода данных. С помощью этих панелей вы можете задать любые значения в шестнадцатеричном коде или в коде ASCII.

Для других функций, таких как HLM_LOGIN или HLM_WRITELICENSE, вы должны передать Vendor-Key или информацию по модификациям, которую вы можете загрузить из внешнего файла. Для функции HLM_LOGIN вы задаете исполняющую часть ключа Vendor-Keys как VKEY в формате RAW-. Такой файл вы можете создать с помощью менеджера **Vendor-Key** (из **HL-RUS/LiMaS**). Менеджер ключа **Vendor-Key** вы можете вызвать из меню **Tools** (по ключу Vendor-Key см. главу 5.3).

Для функции `HLM_WRITELICENSE` вы должны передать информацию по модификациям. Эта информация создается программой **Capriccino**.

8.5 Обзор выполненных функций

Вы можете посмотреть, какие функции с начала запуска программы были выполнены. Для этого выберите **Tools/API Call History**.

Вы увидите:

- таблицу вызовов функций и переданной информации по статусу;
- соответствующий исходный текст программы на C.

С помощью **Copy** вы можете записать исходную программу в промежуточный буфер и оттуда, например, ввести в вашу программу.

8.6 Создание исходной программы

У вас есть две возможности с помощью **Latteccino** создать исходную программу на C:

- в диалоге к каждой функции вы можете записать в поле **Quellcode** соответствующую исходную программу в промежуточный буфер. Идет речь об исходной программе к указанному в диалоге статусу.
- в поле **API-вызовы** вы можете с помощью нажатия **Copy** записать указанную исходную программу в промежуточный буфер.

Из промежуточного буфера вы можете ввести исходную программу в вашу программу. Учитывайте при этом общие указания по ручному подключению в описании *HL-API*.

8.7 Подключение в помощью RefKey/VerKeys

При подключении с помощью функций `HL_LOGIN` или `HLM_LOGIN` вы можете задать пару кодов **Reference-Key/Verify-Key**. Эти коды вместе с адресом ключа служат для различия ключей разных производителей. Маловероятно, но может быть,

что два производителя создали ключи с одинаковыми адресами. С помощью этих кодов можно перепроверить способ шифрования этих модулей, каждый из которых уникален.

Вы можете вызвать счетчик RefKey/Verkey с помощью нажатия **Tools/RefKey/Verify Key**.

Задайте в диалоговом поле адрес ключа и способ доступа зашифрованного ключа Hardlock и выберите один Reference Key. В качестве альтернативы вы можете случайно создать один Reference Key. С помощью поля **Generate VerKey** вы можете рассчитать через подключенный ключ Hardlock Verify Key. Дополнительно пара кодов будет записана в нижнее поле как код C-Код, так что вы сможете взять данные в свою исходную программу простым нажатием Cut&Paste.

Документация по Hardlock

Документация по **Hardlock** познакомит вас со всеми компонентами продуктовой линейки **Hardlock** и с возможностями защиты вашего программного обеспечения (ПО).

Документация включает в себя следующие разделы.

- **Техническое руководство по Hardlock** описывает принципы защиты ПО с помощью всех компонентов продуктовой линейки. Оно поможет вам выбрать правильную защиту для выполнения ваших задач.
- **Руководство по HL-Bistro** (визуальной среде программирования для Hardlock) описывает этот программный пакет, который позволяет вам проектировать систему защиты в привычной среде Windows.
- **Руководство по HL-Server** описывает применение Hardlock в сетевых приложениях.
- **Руководство по HL-Crypt** описывает применение автоматической защиты вашего ПО.
- **Руководство по HL-API** содержит обзор интерфейса прикладных базовых систем (спецификацию API) для ручной защиты ваших программ.
- **Раздел Понятия** поясняет основные термины и понятия, используемые в документации по Hardlock.

Все разделы дополняются интерактивной помощью (Help) к различным программам. Самые последние изменения и дополнения вы найдете в файлах Readme к программам. В руководствах и файлах интерактивной помощи (Help) применяются следующие типы шрифтовых выделений:

Обозначение	Функция	Пример
жирный	Понятия верхнего уровня Продукты	DatePCTV Hardlock Hardlock Twin
ЗАГЛАВНЫЕ БУКВЫ	Файлы и пути	HLDRV.EXE
Courier	Синтаксис	hlpatchup -m 29809

Содержание

1 Автоматическая установка защиты с HL-Crypt	1
1.1 Система HL-Crypt	1
1.2 Технические основы и предпосылки	3
1.2.1 HL-Crypt и система Hardlock	3
1.2.2 Типы программ	3
1.2.3 Драйвер	4
1.2.4 Тестирование HL-Crypt	5
2 Шифрование и установка защиты	7
2.1 Быстрая установка защиты с помощью HL-Crypt	7
2.2 Процесс шифрования	9
2.3 Шифрование данных с загрузчиком данных	10
2.4 Ограничение времени действия с помощью HL-RUS/LiMaS (W32)	12
2.5 Повышение безопасности	13
2.6 Повышение скорости	13
3 Параметры командной строки	14
3.1 Правила отображения	14
3.2 Обзор параметров	15
3.3 Строение описания параметров	18
3.4 Задание адреса модуля	19
-m:MOD (D,W,W32)	19
3.5 Уровень безопасности	20
-b:FREQU (D,W,W32)	20
-c:CRYPT (D,W,W32)	21
-sec:LEVEL (D,W,W32)	22
-y (D)	23
-lock (D)	24
-qm (D)	25
3.6 Параметры, зависящие от программы	26
-dlx (W32)	26
-exsecnr:SECNR (W32)	27
-!ovl (D,W)	28
-!no:ID (D)	29
-ex:BYTES (D)	30
-o (D)	31
-t (D)	32
-p (D)	33
3.7 Параметры для прозрачного шифрования данных on-line	34
-d:FILE (D,W,W32)	34
-de:FILE (D,W,W32)	35
-k:KEY (D,W,W32)	36
-r:ID (D,W,W32)	37

-w:ID (D,W,W32)	38
-qslren (D,W,W32)	40
3.8 Сеть (HL-Server)	41
-acc:MODE (D,W,W32)	41
-time:MIN (D,W,W32)	42
3.9 Лицензирование с помощью	
HL-RUS (W32)	43
-rusvk:FILE (W32)	43
-rus:MODE (W32)	44
-rusexpwarn:DAYS (W32)	46
-ruscntwarn:UNITS (W32)	47
3.10 Прочее	48
-def:FILE (D,W,W32)	48
-waitbox (W32)	49
-!fcb (D)	50
-!beep (D)	51
-comp (D)	52
-bf (D)	53
-!cf (D)	54
-n (D)	55
-ldtiny (D)	56
-s:STRING (D,W)	57
-!cpu:ID (D)	58
-mem:@FILE (D)	60
4 Приложение	61
4.1 Защита программ в DOS	61
4.1.1 Расширение для DOS	61
4.1.2 Базовая конвейеризация	
Basic-Chaining/КНК-Programme (D)	62
4.1.3 Обработка TSR программ (D)	62
4.1.4 Обработка оверлеев (D)	63
4.1.5 Применение DOS-экстендеров	
(расширителей) программ (D)	63
4.1.6 Quarterdecks DesqView (D)	64
4.1.7 Влияние на скорость	65
4.1.8 Интерфейс пользователя	65
4.2 Перерасчет параметров -le: и -ln: на -sec: (D,W)	66
4.3 Устаревшие параметры	66
4.3.1 sle:SLOTID (D,W,W32)	66
4.3.2 exp (D,W,W32)	67
4.3.3 expwarn:DAYS (D,W,W32)	67
4.3.4 cnt:TIMES (D,W,W32)	68
4.3.5 -cntwarn:TIMES (D,W,W32)	68
4.3.6 -prot:PROT (D)	68

1 Автоматическая установка защиты с **HL-Crypt**

1.1 Система **HL-Crypt**

Использование системы **HL-Crypt** является наиболее безопасным и простым способом защиты вашей программы при помощи ключа **Hardlock**. Такая защита не требует внесения каких-либо изменений в исходный текст программы. Автоматическое выполнение процедуры защиты экономит время и обеспечивает большую степень защиты по сравнению с другими методами.

HL-Crypt модифицирует программы так, что они работают только с одним специально закодированным ключом. **HL-Crypt** встраивает при автоматической установке защиты внутренние модули в программу, которую нужно защитить (так называемые Code-Module). Дополнительно создается бинарный имидж криптографического шифрования. Информация по кодированию ключей **Hardlock** и по основам шифрования вы получите в разделе **Hardlock** - *Техническое руководство*.

Зашифрованная программа объединяет различные подпрограммы с ключами безопасности запроса **Hardlock** и функциональные подпрограммы расширения в единой программе. Дополнительно **HL-Crypt** защищает от манипуляций данные лицензирования, информацию о правах на копии и серийные номера, так как эта информация трансформируется в свою начальную форму только с запуском программы.

С программой **HL-Crypt** вы можете сами устанавливать защитные механизмы и их комбинировать. Например, дополнительно к простому шифрованию вы можете:

- защитить любой файл данных;
- выполнить фоновые запросы к ключу во время выполнения программы;
- предотвратить реинжиниринг и использование отладчиков;
- ограничить время работы счетчиками и таймером времени.

Вы можете использовать систему **HL-Crypt**, выбрав одну из указанных программ командной строки и Windows-программы **Espresso** из **Hardlock Bistro**. Дальнейшую информацию Вы найдете в разделе *Hardlock-Bistro*.

1.2 Технические основы и предпосылки

1.2.1 HL-Crypt и система Hardlock

HL-Crypt – это часть защитной системы **Hardlock** для осуществления программной защиты с помощью аппаратных средств. Обзор всей системы, различные программы и продукты вы найдете в разделе *Hardlock - Техническое описание*.

1.2.2 Типы программ

Система **HL-Crypt** предназначена для защиты следующих программ:

HL-Crypt	Защита для	Операционная система
HL-CRYPT.EXE (HL-EVALU.EXE)	COM EXE	DOS
HLWCrypt.EXE (HLWEVALU.EXE)	EXE DLL (без кодирования данных Online)	Windows 3.x, 16-битные программы
HLCWIN32.EXE (HLEWIN32.EXE)	EXE DLL	Win32 (Windows 95-, Win32s-, NT программы)

Программы **HL-Crypt** сами являются приложениями для DOS и работают с MS- или PC-DOS начиная с версии 3.1 и выше. Сгенерированные программы могут использоваться со всеми операционными системами (если программа и/или система допускают это).

Примечание: обратите внимание, что в DOS .COM-программы всегда превращаются в .EXE-программы.

В скобках указана соответствующая тестовая программа, с помощью которой вы можете опробовать систему **HL-Crypt**, не приобретая ключа.

Использование отдельных параметров обозначается в руководстве следующим образом:

Символ	HL-Crypt
D	HL-Crypt для DOS
W	HL-Crypt Windows
W32	HL-Crypt for Win32

Информацию по шифрованию Вы найдете в *главе 2*. Подробную информацию по защите программ в DOS Вы найдете в *главе 4*.

1.2.3 Драйвер

Для применения ключа **Hardlock** должны быть установлены соответствующие драйверы. Драйверы должны быть установлены и у ваших заказчиков, иначе зашифрованная программа не сможет запрашивать ключи.

Для установки драйверов в разные операционные системы на **Hardlock**-CD имеются файлы EXE- для оболочки Windows и резидентные программы для DOS и терминальных приложений. Вы можете поставлять эти файлы с защищенными программами и ключами вашим заказчикам.

Операционная система и драйверы:

Операционная система	С графическим интерфейсом	Командные строки
Windows 3.x	HLDRV16.EXE	INSTVXD.EXE
Windows 95/98, NT	HLDRV32.EXE	HLINST.EXE

Далее вы можете непосредственно интегрировать установку драйверов в установочные подпрограммы вашей программы. Дальнейшую информацию по драйверам API вы найдете в описании *Hardlock API*.

1.2.4 Тестирование HL-Crypt

Даже если Вы не приобрели полную версию (защищенную ключом в красном корпусе), вы можете проверить работу системы **HL-Crypt**. Для этого используйте тестовые программы.

- HL-EVALU.EXE для DOS-программ.
- HLWEVALU.EXE для 16-Bit Windows-программ.
- HLEWIN32.EXE для 32-Bit Windows-программ

Эти демо-версии защищены тестовым ключом **Hardlock** с адресом 29809. Единственное ограничение состоит в том, что программа может быть защищена в демонстрационной версии только демонстрационным ключом с номером 29809.

Примечание. Учтите, что шифрование с демонстрационным ключом не предоставляет настоящей защиты, так как все демонстрационные ключи имеют одно поведение при шифровании. Настоящую защиту Вы получите, только индивидуально закодировав ключ **Hardlock**.

2 Шифрование и установка защиты

2.1 Быстрая установка защиты с помощью HL-Crypt

Для быстрой защиты вашей программы вам необходимо:

- Ключ защиты в красном корпусе.
- Один из закодированных вами ключей **Hardlock**.
- Сложный программный файл, который вы хотите защитить.

Примечание: Если у вас до сих пор нет ключа **Hardlock**, используйте соответствующие демонстрационные программы **HL-Crypt** и демонстрационный ключ вместо индивидуально закодированного (см *главу 1.2.4*). Однако в этом случае вы не сможете защитить вашу программу по-настоящему.

Подготовка для защиты программного файла

1. Подключите ключ в красном корпусе к порту принтера вашего компьютера. Вы сможете применить систему **HL-Crypt**.
2. Подключите ключ, которым вы хотите защитить вашу программу, к соответствующему порту.
3. Откройте окно данных в DOS и перейдите в директорию **HL-Crypt**, которая была создана при установке программного обеспечения **Hardlock**.

```
Cd\program files\Hardlock\HL-Crypt
```

4. Создайте резервную копию защищаемой программы, которую вы хотите защитить с помощью **HL-Crypt**, так как процесс шифрования необратим.

```
copy d:\develop\original.exe myprg.exe
```

Программа копируется в директорию **HL-Crypt**.

После подготовки вы можете защитить вашу программу, запустив одну из программ **HL-Crypt** с желаемыми параметрами (см.ниже).

Информацию по отдельным параметрам вы найдете в *главе 3*.

Защита программного файла

1. Если вы уже скопировали программу и перешли в директорию **HL-Crypt**, запустите программу **HL-Crypt** с указанием пути и имени защищаемой программы, адресом подсоединенного ключа **Hardlock** и желаемых параметров (см.*главу 3*). Например, для программ Win32:

```
hlcwin32.exe myprg.exe -m:29809
```

Теперь система **HL-Crypt** начинает защищать вашу программу. После ее успешного шифрования на экран будет выдано соответствующее сообщение. Теперь ваша программа защищена и может быть запущена только при наличии соответствующего ключа (который вы запрограммировали).

2. Для проверки успешности шифрования удалите ключ и попробуйте запустить программу.

Пример программы DOS (HL-CRYPT.EXE)

```
HL-Crypt.exe myprg.exe -m:29809 -b:2000 -y
```

Программа шифруется ключом с номером 29809 и только с этим ключом работает. Дополнительно спустя некоторое время запускается фоновый запрос и включается определение отладчиков.

Пример программы Windows 3.x (HLWCrypt.EXE)

```
hlwcrypt.exe myprg.exe -m:29809 -b:2000
```

Программа шифруется ключом с номером 29809 и только с этим ключом работает. Дополнительно каждые 2000 миллисекунды включается фоновый запрос.

Пример программы Win32 (Win NT, Win 95, Win32s), (HLCWIN32.EXE)

```
hlcwin32.exe myprg.exe -m:29809 -b:2000 -acc:lr
```

Программа шифруется ключом с номером 29809 и только с этим ключом работает. Дополнительно каждые 2000 миллисекунд включается фоновый запрос. Кроме этого, активируется дополнительная поддержка **HL-Server**.

2.2 Процесс шифрования

Система **HL-Crypt** шифрует приложение и при необходимости изменяет его структуру. Шифрование всегда выполняется вместе с ключом. Шифрование всегда происходит с индивидуально кодированным ключом **Hardlock**. Различные механизмы защиты и дополнительные функции включаются в программу в форме модулей (эти модули также шифруются.)

Поскольку зашифрованный файл не может быть напрямую загружен операционной системой, **HL-Crypt** создает соответствующую программу загрузки.

Программа теперь представляет собой исполняемый файл и включает:

- зашифрованный исходный файл;
- модули защиты;
- модули загрузки.

Система **HL-Crypt** работает по принципу уникального генерирования. Итак, если вы шифруете программу несколько раз, она будет каждый раз шифроваться по-другому (с помощью случайной генерации). Из-за встроенных модулей увеличивается объем массовой памяти, также как и из-за присоединенных в приложение модулей загрузки (загрузчиков).

Время загрузки и работы программ, обработанных **HL-Crypt**, в любом случае длиннее, чем у незащищенных приложений, так как шифрование и расшифровка занимает время.

Если пользователь запускает защищенную вами программой, происходит следующее:

1. Операционная система запускает не оригинальную программу, а новые модули загрузки.
2. Загрузочные модули расшифровывают ряд модулей и выводят их. Модули или обрабатываются и потом сразу удаляются из памяти, или инсталлируются, чтобы во время работы программы выполнять задания.
3. Последней расшифровывается во время стартовой фазы начальная часть приложения и запускается.

2.3 Шифрование данных с загрузчиком данных

Загрузчик данных работает по следующему принципу: при обращении к файлу система проверяет, зашифрован ли файл.

Если файл данных не зашифрован, доступ к нему осуществляется средствами операционной системы.

Если файл данных имеет защиту, данные файла шифруются при записи и расшифровываются при чтении. Загрузчик остается в памяти после запуска программы.

Шифрование и расшифровка могут происходить только тогда, когда нужный ключ подключен к компьютеру.

Если вы хотите защитить файл данных, вы можете применить **Espresso** из **Hardlock Bistro** . Если вместо этого вы хотите сами установить защиту, то поступите следующим образом:

- Установите при шифровании файла программы, какие файлы данных должны быть зашифрованы, например, с помощью параметра **-d**
- Зашифруйте файл данных, например с помощью **DLCRYPT**.

Информацию по отдельным параметрам для шифрования данных on-line вы найдете в *главе 3.7*.

Самую большую защиту представляет ступень безопасности 6. Она соединяет самую высокую безопасность с самой большой скоростью (параметр **-sec**, установка ступени 6).

Примечание. Если разные программы должны обратиться к одним и тем же защищенным файлам данных, программы должны быть защищены одной ступенью (уровнем) безопасности.

Действия при шифровании данных

Вы шифруете программный файл **MYPROG.EXE** и при этом хотите защитить файл **MYDATA.TXT** . Файл данных **MYDATA.TXT** также только в закодированном виде должен сохраняться на диске.

Используйте для создания защищенного файла данных терминальное приложение **DLCRYPT** (из директории **HL-TOOLS** на диске **Hardlock-CD**):

1. откройте окно **DOS**;
2. перейдите в директорию **HL-TOOLS**;
3. запустите программу **DLCRYPT** с именами файлов, которые надо зашифровать, с именем целевого файла, ключом и, по необходимости, с другими параметрами (см. ниже). Например:

```
dldcrypt orgdata.txt mydata.txt -k:mykey -sec:6 -e
```

4. Твердо определите при шифровании программы, какие файлы данных надо шифровать, например:

```
hlcwin32.exe -m:myprog.exe -m:29809 -d:*.txt -k:mykey
```

Если MYPROG.EXE загружает и использует файл MYDATA.TXT, то этот файл будет зашифрован.

Параметры LCRYPT для шифрования данных

Параметр	Данные
-m:modad	адрес ключа
-k:key	ключ, макс. 8 байтов
-sec:level	степень безопасности, рекомендована 6
-e	зашифровать
-d	расшифровать

2.4 Ограничение времени действия с помощью HL-RUS/LiMaS (W32)

Вы можете ограничить время работы зашифрованной программы с помощью функций **HL-RUS/LiMaS** из системы **HL-Crypt**. Вы можете:

- установить условия, которые надо выполнить, чтобы программа запустилась (с параметром **-rus**);
- давать предупредительные сообщения при приближении окончания работы программы и заканчивающихся счетчиках (с параметром **-rusexpwarn** и **-ruscntwarn**).

Условием этого является применение ключа **Hardlock**, который соответствующим образом закодирован для **HL-RUS**.

Информации по лицензиям защищены ключом **Vendor-Key**, чья внешняя часть встроена в защищенную программу (с параметром **-rusvk**). Дальнейшую информацию по отдельным параметрам вы получите в *главе 3.9*.

2.5 Повышение безопасности

Для повышения безопасности можно установить, как часто должны происходить фоновые запросы (параметр **-b**).

При защите 32-битной программы автоматически интегрируется распознаватель отладок и защита от реинжиниринга. Вам не нужно отдельно это указывать. Дополнительно вы можете установить ключи-коды перед непосредственным программным кодом с помощью параметра **-c**.

Используя модули безопасности, можно повысить безопасность ваших программных файлов в DOS:

- они автоматически распознают присутствие отладчика (D);
- автоматически блокируют отладку (D);
- устанавливают дополнительные модули-коды перед непосредственным программным кодом (D,W).

Применение этих параметров может все же привести к потере скорости. Информация по отдельным параметрам - в *главе 3.5*.

2.6 Повышение скорости

При повышении скорости снижается уровень защиты. Но иногда можно найти компромисс. Вы можете увеличить скорость при загрузке следующим образом:

- уменьшите число установленных защитных модулей (**-c**), (D,W,W32);
- уменьшите частоту фоновых запросов (**-b**), (D,W,W32);
- ограничьте число защищенных файлов данных (**-d**), (D,W,W32);
- для шифрования данных используйте шестой уровень (**-sec:6**), (D,W,W32).

Информацию по отдельным параметрам вы найдете в *главе 3*.

Дальнейшую информацию по ускорению шифрования при защите программ в DOS вы найдете в *главе 4.1.7*.

3 Параметры командной строки

3.1 Правила отображения

Параметрами командной строки можно установить, как **HL-Crypt** должна шифровать вашу программу. У вас большой выбор опций. Перечень всех параметров вы найдете в этой главе или запустив программу **HL-Crypt** с параметром **-?**.

Не все параметры подходят для разных версий. У параметра есть обозначение, которое показывает, что его можно использовать с данной программой. Например, **D** для версии DOS- (**HL-CRYPT**), **W** для версии 16-бит Windows (**HLWCRYPT**) и **W32** для версии 32-бит Windows (**HLCWIN32**).

Чтобы защитить вашу программу, перейдите в директорию **HL-CRYPT** и запустите нужную программу, задав имя программы и нужные параметры. Помните о следующих правилах при указании параметров:

- **порядок следования:** первым параметром идет имя программы, которую вы хотите заащитить. Дальнейшая последовательность любая;
- **символ:** каждый параметр начинайте с символа «-» или «/» без последующего пробела. Разделяйте отдельные параметры пробелом. Соединяйте параметры и аргументы двоеточием (:) без пробела;

Пример:

```
-m:29809 -b:2000
```

- **правописание с большой и маленькой буквы:** если не указано иначе, различий нет;
- **длина командной строки:** помните, что командная строка в среде MS-DOS может иметь длину до 126 символов. Если вы превысили эту длину, строка будет усечена без всякого предупреждения. Это

может привести к нежелательным результатам и ошибкам при выполнении защиты в автоматическом режиме. Этого можно избежать, если подключить файл конфигурации с помощью параметра **-def:FILE** (см. главу 3.10).

3.2 Обзор параметров

В данной таблице различные параметры тематически сгруппированы с указанием их применения и номера страницы.

Назначение	Параметр	Описание	Применимость
выбор ключа	-m:MOD	адрес ключа Hardlock	D,W,W32
безопасность	-b:FREQ	фоновый запрос	D,W,W32
	-c:CRYPT	число закодированных модулей	D,W
	-sec:LEVEL	степень безопасности	D,W,W32
	-y	автоматическое распознавание отладчиков	D
	-lock	отключение прерываний отладчика	D
	-qm	Quiz-Master Trick модуль	D
параметры, зависящие от программы	-dlx	шифрование оверлеев	W32
	-exsecnr:SECNr	исключение секций	W32
	-!ovl	отключение шифрования внутренних оверлеев	D,W

Назначение	Параметр	Описание	Применимость
Online-шифрование	-qslren	быстрая функция переименования	D,W,W32
сеть (HL-Server)	-acc:MODE	тип доступа	D,W,W32
	-time:MIN	управление Таймаутом HL-Server	D,W,W32
сублицензирование с помощью HL-RUS (LiMaS)	-rusvk:FILE	- определить файл Vendor-Key	W32
	-rus:MODE	-определить условия HL-RUS	W32
	-rusexpwar n: DAYS	выдача предупреждающего сообщения с заданного места до окончания работы программы	W32
	-ruscntwarn: UNIT	выдача предупреждающего сообщения с заданного места до окончания работы счетчика	W32
прочее	-def:FILE	файл конфигурации	D,W,W32
	-lfcb	FCB (File Control Block)-отключить поддержку	D
	-!beep	отключить звук	D

Назначение	Параметр	Описание	Применимость
Online-шифрование	-qslren	быстрая функция переименования	D,W,W32
сеть (HL-Server)	-acc:MODE	тип доступа	D,W,W32
	-time:MIN	управление Таймаутом HL-Server	D,W,W32
сублицензирование с помощью HL-RUS (LiMaS)	-rusvk:FILE	- определить файл Vendor-Key	W32
	-rus:MODE	-определить условия HL-RUS	W32
	-rusexpwar n: DAYS	выдача предупреждающего сообщения с заданного места до окончания работы программы	W32
	-ruscntwarn: UNIT	выдача предупреждающего сообщения с заданного места до окончания работы счетчика	W32
прочее	-def:FILE	файл конфигурации	D,W,W32
	-!fcb	FCB (File Control Block)-отключить поддержку	D
	-!beep	отключить звук	D

3.3 Строение описания параметров

Описание параметров строится по следующей схеме:

- Параметр[:Аргумент] (использование)

Краткое описание параметра.

Аргументы

Перечень аргументов, нужных этому параметру

Применение

Подробное описание параметров, указание на их действие.

Пример

```
ïðëíàð è ëðàòëíà ïíëñàíëà æý ïðëíàíàíëý
íàðàíàòðà. Ñòíëò èè íàðáíë ñòðíëà ïðëëàçà
[HL-Crypt], ïíæàò èè ïðíàðáííà çàíàíýòüñý àðóáíë
ïðíàðáííë ñííòààòñòàòðóàáí òëíà.
```

3.4 Задание адреса модуля

-m:MOD (D,W,W32)

Сообщает программе **HL-Crypt** адрес модуля ключа.

Аргументы

MOD # Адрес ключа Hardlock

Применение

При шифровании вашей программы должен быть известен адрес ключа защиты. В этом процессе **HL-CRYPT** «криптографически» знакомится с ключом, указанным при помощи адреса модуля. Этот параметр всегда должен быть указан.

Вы можете указать до 8 адресов модулей (через запятую) при использовании **HL-CRYPT** для DOS. При этом ключи должны быть закодированы идентично.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809
```

Демонстрационный ключ с адресом модуля 29809 соединен с программой.

3.5 Уровень безопасности

-b:FREQU (D,W,W32)

Указывает временные отступы между фоновыми запросами.

Аргументы

FREQU#	0	нет запроса
	1	частые запросы
	2000	стандартные запросы (нормальные)

Использование

Этот параметр активизирует фоновую проверку ключа. Число FREQU - это 32-битное значение, может находиться в области от 0 (=нет фонового запроса) до 4,29 миллиардов (са. все 8 лет). Чем больше выбрано число для TIME, тем реже происходит фоновый запрос. В 16- и 32-битных программах Windows значение соответствует миллисекундам. В DOS-программах время - это функция из Timerticks, DOS-Calls и указание времени клавиатурой. В DOS время точно не указывается.

Чтобы не замедлять запуск приложения, первые две или три минуты не проводятся запросы. Если в программе много расчетов или она часто обращается к различным данным, используйте, если нужно, большее значение, тогда фоновые запросы будут требовать меньшего времени.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -b:3000
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809 и имеет автоматические фоновые запросы.

-c:CRYPT (D,W,W32)

Указывает число использованных модулей защиты

Аргументы

CRYPT	Число модулей, предшествующих программе
--------------	---

Применение

При помощи системы **HL-Crypt** вы можете разместить модули защиты впереди защищаемой программы. При запуске такой программы все модули, предшествующие программе, обрабатываются до начала работы приложения. Это затрудняет хакерам доступ к первоначальному тексту программы, поскольку им придется преодолеть модули защиты.

После запуска приложения модули защиты удаляются из памяти, освобождая место. Помните, что чем больше модулей защиты предшествует программе, тем дольше она будет загружаться. Наличие пяти модулей перед текстом программы (значение по умолчанию) является эффективным компромиссом между быстрым запуском и надежной защитой.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -c:10
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Десять случайных модулей защиты добавляются к приложению.

-sec:LEVEL (D,W,W32)

Определяет силу защиты.

Аргументы

LEVEL (уровень)	6	Быстрое шифрование, высокая степень защиты. (Предварительная установка) Указание: только при этой установке возможно шифрование приложений Win32 в Win9x с помощью HL-Server . На основе имплементированного алгоритма Вы должны в основном применять эту установку.
	5	Медленное шифрование, высокая степень защиты.
	4	
	3	
	2	
	1	Быстрое шифрование, слабая защита.

Применение

Помните, что если несколько программ обращаются к одним и тем же данным, опция должна быть одинаковой для всех них. В случае **HL-Crypt** для Windows параметр имеет отношение только к шифрованию данных **-d**, **-de**, в то время как в **HL-Crypt** для DOS он также влияет на исполняемый код.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -d:*.txt -sec:2
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Шифруются все файлы, имеющие расширение TXT. В угоду высокой скорости ослаблена защита шифрования.

-y**(D)**

Активирует автоматическое распознавание отладчиков.

Аргументы

нет

Применение

Активируется автоматическое распознавание отладчиков (отладчик – это программа, используемая для пошагового выполнения (трассировки) и анализа кода других программ). Модуль автоматически определяет большинство отладчиков и пытается вывести их из строя. Вы можете всегда использовать этот параметр. В программе **HL-Crypt** для Windows (W,W32) этот инструмент уже активирован и не может быть отключен.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -y
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Включено автоматическое определение отладчиков.

-lock (D)

Активирует модуль **Int-Lock**.

Аргументы

нет

Применение

Этот модуль является антиотладочным. Он запрещает использование прерываний 01 и 03 (прерывания отладчика) на все время работы программы (при использовании загрузчика данных). Этот инструмент предотвращает использование отладчиков.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -lock
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Отладчики прерывания блокированы на время выполнения программы.

-qm (D)

Активирует модуль «**Quizmaster**».

Аргументы

нет

Применение

Здесь идет речь об антиотладочном модуле. Он расшифровывает и шифрует случайные данные, относящиеся к ключу, при каждом запуске программы. Если взломщик попытается записать данные, передаваемые между программой и ключом, он будет получать различные результаты при каждом запуске программы. Это делает очень сложным процесс подмены сеанса связи.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -qm
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. В программу вставляется модуль защиты **Quizmaster**.

3.6 Параметры, зависящие от программы

Далее перечислены параметры, которые используются для защиты определенных программ.

-dlx (W32)

Включает шифрование оверлеев.

Аргументы

нет

Применение

В некоторых программах Win32 данные и даже коды присоединяются в конце задокументированного файла Win32-EXE- (Overlay). При шифровании они не распознаются как часть файла. Параметром `-dlx` данные дополнительно шифруются, а при чтении в этой области расшифровываются. Применение параметра увеличивает защиту.

Пример

```
[HL-CRYPT] myprog.exe -m29809 -dlx
```

Оверлей из файла `myprog.exe` зашифровывается, а во время работы расшифровывается.

-exsecnr:SECNR (W32)

Исключает часть программы с заданным номером из шифрования.

Аргументы

SECNR	Номер секции, которая не должна быть зашифрована. Нумерация начинается с 1
--------------	--

Применение

В некоторых случаях **HL-Crypt** не может распознать все области файла EXE. Защищенная программа тогда не может функционировать и **HL-Crypt** выдает предупреждение, называя проблемную часть программы. Эта секция может выпасть при шифровании. Параметр можно использовать много раз.

Пример

```
[HL-Crypt] myprog.exe -m:29809 -exsecnr:1 -  
exsecnr:3
```

При шифровании файла myprog.exe первая и третья часть будут исключены.

!ovl** (D,W)**

Выключает шифрование внутренних оверлеев загрузчиком данных.

Аргументы

нет

Применение

Если производительность программы сильно зависит от частоты загрузки оверлеев, выключите шифрование внутренних оверлеев при помощи этого параметра. Загрузчик данных больше не будет шифровать эти оверлеи при каждом их перезапуске. Это значительно увеличит скорость работы. В версии программы **HL-Crypt** для Windows отладочная информация, присоединяемая в виде оверлеев, отсекается.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -!ovl
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Внутренние оверлеи не шифруются.

!no:ID (D)

Настраивает программу **HL-Crypt** на поддержку DOS-экстендеров.

Аргументы

ID	q	выключает проверку PSP
	m	выключает проверку памяти
	mq	выключает проверку PSP/память (программно)
	mp	выключает проверку PSP/память (аппаратно)

Применение

Некоторые DOS-расширители требуют выключения отдельных механизмов программы **HL-Crypt**. Благодаря способу работы DOS-экстендеров некоторые предложения не являются правильными. При автоматическом применении расширителей DOS используйте эту опцию с параметрами в точности, как указано ниже. В большинстве случаев достаточно использовать **!no:mq**. Не используйте последнюю комбинацию(**!no:mp**), если ваша программа открывает сеанс DOS или вызывает другую программу. Не используйте этот параметр, если ваша программа должна быть защищена как резидентная (TSR).

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -!no:mq
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Приложение защищается с учетом работы под DOS-экстендером.

-ex:BYTES (D)

Число байт исходной программы .EXE, которое не должно шифроваться программой **HL-Crypt**.

Аргументы

BYTES	0	значение по умолчанию
	max.	размер всего файла

Применение

Программа **HL-Crypt** может исключать определенное число байт кода программы из шифрования. Эта часть .EXE-файла не шифруется и может быть прочитана другими программами. Требуемое число не может быть известно заранее, поскольку новый EXE-файл имеет другую структуру и расположение в нем приложения не может быть определено точно.

Этот параметр в основном используется для программ на языке BASIC-Chaining, так как в нем модули могут быть определены и загружены в кодах из образа файла (см. главу 4.1.2).

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -ex:500
```

Программа защищается с помощью демонстрационного ключа с адресом модуля 29809. Первые 500 байт приложения не шифруются.

-o**(D)**

Выбор в явном виде загрузчика оверлеев фирмы Microsoft при выборе стандартного загрузчика (**-n**).

Аргументы

нет

Применение

Обычно программа **HL-Crypt** автоматически определяет все стандартные форматы оверлеев при шифровании с помощью стандартного загрузчика. Однако, если необходимо, вы можете заменить автоматический анализ программы **HL-Crypt** указанием этого параметра и выбором стандартного формата загрузчика Microsoft.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -n -o
```

Программа защищается с помощью демонстрационного ключа с адресом модуля 29809. В приложение включается только стандартный загрузчик. Система заменяет код распознавания оверлея стандартным загрузчиком и использует формат оверлея Microsoft.

-t (D)

Выбор в явном виде загрузчика **Turbo Overlay** при применении стандартного загрузчика (**-n**).

Аргументы

нет

Применение

Обычно система **HL-Crypt** самостоятельно распознает во время шифрования со стандартным загрузчиком все распространенные оверлеи. В исключительных случаях вы можете заменить автоматический анализ программы **HL-Crypt** указанием этого параметра и выбором формата оверлеев **Turbo Overlay-Format** (VROOMM).

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -n -t
```

Программа защищается с помощью демонстрационного ключа с адресом модуля 29809. В приложение включается только стандартный загрузчик. Система заменяет распознавание оверлеев стандартным загрузчиком и использует формат оверлеев **Turbo**.

-p**(D)**

Выбор загрузчика оверлеев при использовании неизвестного или нестандартного типа оверлея при применении стандартного загрузчика (**-n**).

Аргументы

нет

Применение

Обычно система **HL-Crypt** самостоятельно распознает во время шифрования со стандартным загрузчиком все распространенные оверлеи. В исключительных случаях вы можете заменить автоматический анализ программы **HL-Crypt** указанием этого параметра и выбором загрузчика для неизвестного или нестандартного типа оверлея (например, PLINK86, BLINKER, RTLINK, MOVE).

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -n -p
```

Программа защищается с помощью демонстрационного ключа с адресом модуля 29809. В приложение включается только стандартный загрузчик. Система заменяет распознавание оверлеев стандартным загрузчиком и использует формат для нестандартного оверлея.

3.7 Параметры для прозрачного шифрования данных on-line

Все параметры, относящиеся к прозрачному шифрованию. Помните, что в программе **HL-Crypt** для Windows в режиме прозрачного шифрования могут быть зашифрованы данные только .EXE-файлов (не .DLL файлов).

-d:FILE (D,W,W32)

Указания файловой маски определяют, какие файлы должны быть зашифрованы.

Аргументы

FILE	Файловая маска, параметр может быть использован до 8 раз
-------------	--

Применение

В системе **HL-Crypt** можно определить файлы для шифрования, а позже обработать их программой. Файловая маска определяет, какие файлы должны быть зашифрованы. Параметр – **d** может быть использован до 8 раз с различными масками. Вы можете использовать все стандартные шаблоны DOS- (например, . *.DBF, *.?GI или TEXT??.DOC). Программа **HL-Crypt** для Windows поддерживает использование длинных имен в среде Windows 95 и Windows NT.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -d:*.txt -d:foo.dat
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Все файлы с расширением .TXT и файл FOO.DAT должны быть подвергнуты шифрованию. Они должны быть расшифрованы при чтении и зашифрованы при записи.

-de:FILE (D,W,W32)

Через указания файловой маски определяется, какие файлы не должны быть зашифрованы.

Аргументы

FILE	Файловая маска, параметр может быть использован до 8 раз
-------------	--

Применени

В системе **HL-Crypt** можно определить файлы для шифрования, а позже обработать их программой. Файловая маска определяет, какие файлы не должны быть зашифрованы. Параметр может быть использован до 8 раз с различными масками. Вы можете использовать все стандартные шаблоны DOS- (например, *.DBF, *.*?G1 oder TEXT??).DOC). Программа **HL-Crypt** для Windows поддерживает использование длинных имен в среде Windows 95 и Windows NT.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -d:*.txt -  
de:cl1.txt
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Все файлы с расширением .TXT кроме файла CL1.TXT должны быть подвергнуты шифрованию. Они должны быть расшифрованы при чтении и зашифрованы при записи.

-k:KEY (D,W,W32)

Приложению передается ключ для прозрачного шифрования файла. Если многие программы обращаются к одному и тому же набору данных, они должны иметь одинаковый ключ шифрования.

Аргументы

KEY	Ключ состоит из 8 знаков (ASCII). Если указано меньше 8 знаков, система автоматически добавляет пробелы. Прописные и строчные символы отличаются
------------	--

Применение

Все программы, которые обращаются к одному и тому же набору данных, должны иметь одинаковый ключ шифрования (даже разные версии одной программы). Если не указывается специальный ключ, система сама создает случайным образом значение из 8 знаков. Таким образом, программы, имеющие разные ключи, не смогут расшифровать общие данные.

Пример

```
[HL-CRYPT] MYPR1.EXE -m:29809 -d:test.txt  
-k:HARDLOCK  
[HL-CRYPT] MYPR2.EXE -m:29809 -d:test.txt  
-k:HARDLOCK
```

Обе программы защищаются при помощи демонстрационного ключа с адресом модуля 29809. Обе программы используют одинаковый ключ для шифрования данных. Они могут обращаться к файлу TEST.TXT. Если параметр `-k` не был использован, каждый раз создается случайный (различный) ключ.

-r:ID (D,W,W32)

Выключает контроль загрузчиком данных некоторых частей приложения при доступе к чтению.

Аргументы

ID	H	обозначает заголовок EXE-файла приложения
	O	обозначает внутренний оверлей приложения
	R	обозначает исходный код приложения
	D	обозначает файлы, которые должны контролироваться загрузчиком данных (внешние оверлеи тоже являются файлами данных)

Применение

Если указан этот параметр, загрузчик данных не контролирует данные во время чтения. Данные читаются, как они есть, и передаются приложению. Такое поведение может быть задано для различных областей файлов данных приложения. Отключение от приложения ненужных областей повышает скорость и увеличивает защиту.

При **HL-Crypt** для Windows/Windows32 выбирается только параметр **-r:d**.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -r:hr
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Управление доступом не происходит, если приложение пытается прочитать заголовок своего собственного .EXE-файла или исходный код. Поскольку заголовок и исходный код зашифрованы и не могут быть расшифрованы загрузчиком данных при чтении, приложение только определяет зашифрованные данные.

-w:ID (D,W,W32)

Выключает шифрование при доступе к записи во время прозрачного шифрования. Области, которых это касается, могут быть здесь указаны с помощью этого параметра.

Аргументы

ID	H	обозначает заголовок EXE-файла приложения
	O	обозначает внутренний оверлей приложения
	R	обозначает исходный код приложения
	D	обозначает файлы, которые должны контролироваться загрузчиком данных (внешние оверлеи тоже являются файлами данных)

Применение

Если указан этот параметр, загрузчик данных не контролирует данные во время письма. Данные пишутся, как они есть, и передаются приложению. Такое поведение может быть задано для различных областей файлов данных приложения. Отключение от приложения ненужных областей повышает скорость и увеличивает защиту.

При **HL-Crypt** для Windows/Windows32 выбирается только параметр **-w:d**.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -w:hr
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Управление доступом не происходит, если приложение пытается прочитать заголовок своего собственного EXE-файла или исходный код. Поскольку заголовок и исходный код зашифрованы и не могут быть за-

шифрованы загрузчиком данных при письме, приложение пишет незашифрованные (некорректные) данные.

-qslren (D,W,W32)

Активирует функцию быстрого переименования.

Аргументы

нет

Применение

Если файлы данных, помеченные для шифрования, переименовываются так, что они больше не подходят под маску (или наоборот), они должны быть расшифрованы или зашифрованы. В качестве меры предосторожности выполняется резервное копирование файлов, так как если программа остановится (при отключении питания, выключении компьютера и т.д.), файл может остаться наполовину расшифрованным или зашифрованным. Этот механизм требует дополнительного пространства на жестком диске и может быть выключен при помощи параметра **-qslren**. Защита от аппаратных сбоев выходит за рамки рассмотрения данного руководства. Для программ Win16 и Win32 функция переименования уже активирована по умолчанию. Здесь она может быть деактивирована с параметром **!qslren**. Процесс длиннее, но надежнее.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -d:daten.txt  
-qslren
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Доступ с шифрованием применяется к файлу DATEN.TXT. Если приложение переименовывает файл DATEN.TXT в файл DATEN.OLD, загрузчик данных использует быстрый, но менее безопасный метод.

3.8 Сеть (HL-Server)

Описываются все параметры, относящиеся к работе с ключом защиты **HL-Server** в сети.

-acc:MODE (D,W,W32)

Задаёт режим доступа к ключу (локальный/удаленный).

Аргументы

MODE	l	локальный ключ Hardlock (по умолчанию)
	r	сетевой ключ Hardlock через HL-Server (удаленный)
	rl	вначале локальный поиск, потом удаленный

Применение

В зависимости от выбора параметра система включает локальный/удаленный поиск для соответствующего ключа. Когда обе установки активированы, система вначале ищет локальный ключ. Если это не удается, поиск продолжается в сети. Чтобы можно было применить поддержку сети, надо установить программу **HL-Server** в сети. Если выполняются обращения к удаленному ключу, он должен быть доступен во время шифрования.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -acc:r
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Приложение обращается только к сетевому ключу, который является доступным через программу **HL-Server**.

-time:MIN (D,W,W32)

Используется для определения тайм-аута приложения (в минутах), после чего освобождается строка загрузки в **HL-Server**.

Аргументы

MIN	0	использует установки HL-Server (по-умолчанию)
	1-998	новый там-аут в минутах
	999	тайм-аут не используется (нет регистрации)

Применение

Этот параметр позволяет изменить значение тайм-аута программы **HL-Server** для данного приложения. (Значение тайм-аута определяет время, по прошествии которого происходит удаление строки регистрации при отсутствии запросов к ключу). Если значение не указано (эквивалентно -time:0), оно выбирается из установки **HL-Server**. Если **HL-Server** обслуживает более одного приложения, может быть полезно изменить установки для определенного приложения (например, при отсутствии частого выполнения фоновых запросов). Имеет смысл использовать этот параметр при работе с удаленным доступом к ключу.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -acc:r -time:60
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Значение тайм-аута для этой строки регистрации установлено в 60 минут.

3.9 Лицензирование с помощью HL-RUS (W32)

Следующие параметры осуществляют соединение с **HL-RUS** (Remote Update System). Только для защищенных приложений Win32. О подробностях о **HL-RUS** смотрите документацию по **HL-API**.

-rusvk:FILE (W32)

Определяет файл Vendor-Key, с помощью которого защищенная программа обращается к ключу **Hardlock-Modul**. Файл должен быть сначала сгенерирован с помощью менеджера Vendor-Key.

Аргументы

FILE	имя файла ключа Vendor-Key
-------------	----------------------------

Применение

Имя файла может содержать директорию, но должно быть действующим именем файла DOS-8.3. Длинные имена файлов со временем не поддерживаются. Ключ Vendor-Key нужен только в соединении со следующими параметрами HL-RUS. Файл встраивается в защищенную программу.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -rusvk:mykey.svk -rus:E
```

Демонстрационный ключ присоединяется к программе. Применение начинается только тогда, когда ручная подпись данных **HL-RUS** соответствует ключу Vendor-Key mykey.svk. Дополнительно проверяется дата окончания **HL-RUS** (последний параметр).

-rus:MODE (W32)

Определяет условия **HL-RUS**, которые должны быть выполнены, чтобы можно было запустить программу.

Аргументы

MODE	E	Дата окончания (Expiration Date). Ключ HL-RUS Hardlock должен иметь запрограммированное время окончания работы, которое еще не истекло.
	Cn	Счетчик. Счетчик HL-RUS повышается на значение "n". При этом он не должен превышать лимит счетчика. Если "n" не задается, счетчик повышается на единицу.
	Sn	Номер слота. Номер слота "n" должен быть активирован и быть свободным, чтобы можно было запустить защищенную программу.

Основные условия задаются последовательно без пробелов. Если какое-то условие имеет восклицательный знак, то во время работы оно оценивается как И-Не.

Применение

Программа запускается, если выполнены все условия параметра **-rus**. Вы можете указать до 8 условий **HL-RUS** и в течение работы программы они будут обработаны по мере поступления. Первое условие, которое может быть выполнено, будет применяться. Основные условия соединяются друг с другом с помощью «и» внутри одного условия и с помощью «или» между разными условиями.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -rusvk:mykey.svk  
-rus:S16 -rus:ES13 -rus:!EC2S13
```

Программа может быть запущена, если демонстрационный ключ **Hardlock** содержит данные **HL-RUS**, которые согласовываются с ключом Vendor-Key mykey.svk, и дополнительно:

- а) если слот 16 активирован и разрешен;
- в) или если слот 13 активирован и разрешен, и еще не закончился срок работы;
- с) или если слот 13 активирован и разрешен, время работы истекло, и счетчик может иметь добавление 2, не превышая лимита.

Комбинация может быть описана по-другому.

Заказчик получает программу на время тестирования и дополнительно он получает еще два разрешения на свободный запуск программы в качестве скидки. Если он купил программу, активируется слот 16, и тем самым заказчик может использовать свою программу безгранично. Для этого нужно только провести модификацию ключа **Hardlock**. Новые файлы лицензий, которые содержат разрешение, будут считываться.

-rusexpwarn: DAYS (W32)

Передача предупреждающего сообщения об оставшемся числе дней эксплуатации программы.

Аргументы

DAYS Число дней до окончания эксплуатации

Применение:

система выдает предупреждающее сообщение, когда число оставшихся дней становится меньше указанного в аргументе.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -rusvk:mykey.svk -  
rus:E -rusexpwarn:5
```

За пять дней до наступления даты окончания эксплуатации появляется предупреждающее сообщение.

-ruscntwarn:UNITS (W32)

Число единиц счетчиков до лимита, после которого передается предупреждающее сообщение.

Аргументы

UNITS Число единиц счетчиков

Применение

Выдается предупреждающее сообщение, если счетчик **HL-RUS** ближе к лимиту счетчика, чем указание аргументов.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -rusvk:mykey.svk -  
rus:C3 -ruscntwarn:30
```

Если лимит счетчика находится у отметки 300, передается предупреждающее сообщение. Оно передается, когда текущее значение счетчика **HL-RUS** больше чем 270. В примере это соответствует 10 стартам программы. При каждом запуске значение счетчика поднимается на 3 пункта.

3.10 Прочее

Прочие параметры, которые связаны с конфигурацией.

-def:FILE (D,W,W32)

Имя конфигурации файла сообщения.

Аргументы

FILE	Правильное имя и путь файла
-------------	-----------------------------

Применение

Файл конфигурации сообщения является ASCII-файлом и содержит сообщение, которое появляется на экране, когда программа не может быть запущена из-за опции **HL-Crypt** или если не найден ключ **Hardlock**. Для текущего формата файла обратите внимание на пример файла **HL-Crypt.def** на вашем CD.

Пример

```
[HL-CRYPT] MYPROG.EXE -def:config.def
```

Сообщения читаются из файла CONFIG.DEF.

-waitbox (W32)

Активирует во время старта программы сообщение.

Аргументы

нет

Применение

Загрузка защищенной программы длится дольше, чем загрузка незащищенной программы. Поэтому во время фазы загрузки может появиться сообщение о том, что все идет нормально. Сообщение в файле конфигурации (см. параметр **-def**).

Пример

```
[HL-CRYPT] -m:29809 -waitbox
```

Во время загрузки программы появляется сообщение.

-!fcb (D)

Выключает поддержку FCB (File Control Block) - загрузчика данных.

Аргументы

нет

Применение

Интеграция функций FCB в загрузчик данных увеличивает количество резидентной памяти, нужной для защищенного приложения. Если вы уверены, что ваша программа не использует функции FCB, то можете отключить поддержку FCB загрузчиком данных.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -!fcb
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Механизм доступа к файлу FCB не поддерживается. Но если ваша программа использует FCB, это может привести к сбою.

-!beep (D)

Выключается звуковой сигнал, который появляется, когда при фоновых запросах не находится ключ.

Аргументы

нет

Применение

Звуковой сигнал выключается этим параметром. Но это рекомендуется в исключительных случаях, так как когда звуковой сигнал отключен, пользователь может подумать, что компьютер «завис».

Пример:

```
[HL-CRYPT] MYPROG.EXE -m:29809 -b:3000 -!beep
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Фоновые запросы выполняются с частотой 3000. При удалении ключа звуковой сигнал подан не будет.

-comp (D)

Сжимает исходный код приложения.

Аргументы

нет

Применение

Этот параметр предлагает возможность сжать исходный код защищенного приложения. Применение этой опции не уменьшает автоматически размер защищаемой программы, так как другие опции (такие, как число модулей защиты, выбранный загрузчик) тоже влияют на реальные размеры программы.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -comp
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Дополнительно сжимается исходный код приложения.

-bf**(D)**

Разрешает многократное шифрование приложения.

Аргументы

нет

Применение

Данный параметр позволяет вам шифровать приложение несколько раз. Обычно это не нужно, однако может пригодиться, если ваше приложение использует два совершенно разных ключа. При этом для более эффективной защиты лучше использовать параметр **-c**.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809  
[HL-CRYPT] MYPROG.EXE -m:12345 -bf
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Приложение также шифруется на втором ключе с адресом 12345. Программа запускается только тогда, когда оба ключа одновременно являются доступными.

-!cf (D)

Деактивирует поддержку мультизадачной среды.

Аргументы

нет

Применение

Если ваша программа запускается в мультизадачной среде (Desqview, Windows DOS-Box), мультизадачность автоматически выключается при доступе к ключу, чтобы получить доступ к порту. Такое поведение может быть изменено при помощи этого параметра. Система перестает определять и включать мультизадачную среду. Этот параметр может быть использован в исключительных случаях (например, для критичных по времени приложений).

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -!cf
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Для мультизадачных сред не предоставлено возможностей.

-n**(D)**

Присоединяется стандартный загрузчик вместо загрузчика данных.

Аргументы

нет

Применение

По умолчанию в программе **HL-Crypt** используется загрузчик данных. Если вы не укажете параметр **-n**, загрузчик данных автоматически будет включен в вашу программу.

Этот параметр препятствует включению загрузчика данных в вашу программу. Тем самым вы отказываетесь от автоматических фоновых запросов и прозрачного шифрования данных с помощью загрузчика данных.

Преимущество стандартного загрузчика в том, что он не требует дополнительной памяти при работе программы. Загрузчик данных, напротив, требует от 25 KB до 50 KB рабочей памяти в зависимости от выбранной опции.

Стандартный загрузчик не может шифровать внутренние оверлеи. Таким образом, после обработки данных программой **HL-Crypt** они не шифруются и не защищаются.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -n
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. С программой можно использовать только стандартный загрузчик.

-ldtiny (D)

Используется для включения облегченного загрузчика в приложение.

Аргументы

нет

Применение

Облегченный загрузчик - это объединение загрузчика данных и стандартного загрузчика. Дополнительно к стандартному загрузчику он поддерживает фоновые запросы. Но он не шифрует внешние данные. Облегченный загрузчик требует очень мало памяти.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -ldtiny -b:3000
```

Программа защищается с помощью модуля с адресом 29809. Облегченный загрузчик включается в вашу программу. Возможно выполнение фоновых запросов.

-s:STRING (D,W)

Эта строка символов автоматически заменяет указатель места «%%» при выводе сообщения.

Аргументы

STRING Любая строка символов без пробелов длиной до 78 символов

Применение

Строка символов может быть вставлена в сообщение без модификации файла. Она заменяет указатель места «%%» при выводе сообщений (параметр **-x**). Это позволяет вам использовать один и тот же файл для разных продуктов или клиентов (заменяя, например, серийный номер или имя).

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -x:@msg.txt -s:123
```

Программа защищается при помощи демонстрационного ключа с адресом модуля 29809. Сообщения читаются из файла MSG.TXT. Строка символов «%%» заменяется на строку «**123**».

-!cpu:ID (D)

Ограничивает работу программ используемыми процессорами.

Аргументы

ID	CPU	ID	CPU
1	i8088	2048	i486SX RM
2	i8086	4096	i486SX VM
4	i80188	8192	i486DX RM
8	i80186	16384	i486DX VM
16	V20	32768	iPentium P5 RM
32	V30	65536	iPentium P5 VM
64	i286	131072	iPentium P6 RM
128	i386SX RM	262144	iPentium P6 VM
256	i386SX VM	16777216	80x87
512	i386DX RM	33554432	Weitek
1024	i386DX VM		

Применение

Этот параметр позволяет ограничить тип процессоров, с которыми будет работать приложение. По умолчанию значение ID- равно нулю, т.е. все типы процессоров допускаются. Для исключения некоторых типов процессоров введите значение суммы их ID- величин. Сделано различие между режимами Real Mode (RM) и Virtual Mode (VM).

Когда система распознает сопроцессоры, она не различает внутренний и внешний сопроцессоры. Процессор 486DX-CPU соответствует процессору 486SX-CPU с внешним сопроцессором. Только некоторые типы компьютеров распознают процессоры Weitek. Поэтому будьте аккуратнее.

Пример

Применение следующих процессоров следует исключить: i386SX VM, i386DX VM, i486SX VM и i486DX VM. Сумма всех значений ID-256+1024+4096+16384 - составляет параметр 21760.

```
[HL-CRYPT] MYPROG.EXE -m:29809 -!cpu:21760
```

Программа защищается с помощью демонстрационного ключа с адресом модуля 29809.

Программа не может быть использована с процессорами Modus 386/486 CPU в режиме VM.

-mem:@FILE (D)

Указание имени файла, содержащего конфигурацию памяти ключа.

Аргументы

FILE	Правильное имя (и путь) файла
-------------	-------------------------------

Применение

При помощи этого параметра вы можете заставить защищенное приложение выполняться в зависимости от содержимого памяти ключа. Файл конфигурации для проверки содержимого памяти при запуске приложения представляет собой файл ASCII-. Смотрите пример на вашем CD для текущего формата файла.

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -mem:@mem.xmf
```

Программа защищается с помощью демонстрационного модуля с адресом 29809. Информация для проверки содержимого памяти читается из файла с именем MEM.XMF.

4 Приложение

4.1 Защита программ в DOS

4.1.1 Расширение для DOS

Загрузчик данных системы **HL-Crypt** для DOS автоматически загружается в верхнюю часть памяти (если она имеется). Это можно предотвратить запуском защищенного приложения с параметром **-!loadhigh**. Этот параметр отображается на экране защищенным приложением, если оно запускается с параметром **-??**

Загрузчик данных соответствует по умолчанию **HL-Crypt**. При доступе к самому себе приложение выполняет эмуляцию обычного просмотра незашифрованного файла. Загрузчик данных может также эмулировать для приложения доступ к частям EXE файла без шифрования (Superloading). Только с помощью этого механизма возможна безопасная обработка всех внутренних оверлеев в DOS.

Стандартный загрузчик активируется параметром **-n**. В отличие от загрузчика данных, стандартный загрузчик удаляется из памяти после загрузки и, таким образом, не занимает дополнительную память при работе программы. Стандартный загрузчик не допускает прозрачного шифрования, и фоновые запросы не могут проводиться. После запуска программы ключ больше не нужен. Программа не получает доступа к исходному коду или к заголовкам, так как данные зашифрованы или их нельзя найти на ожидаемом месте. Такие программы могут быть выполнены только с загрузчиком данных. Оверлеи являются доступными при помощи стандартного загрузчика, так как они не зашифрованы.

Если бы приложение обратилось к зашифрованному файлу или части самого себя, они не должны быть зашифрованы, так нет такой инстанции, которая могла бы их расшифровать. Загрузчик пропал.

Параметр **-ldtiny** активирует облегченный загрузчик. Это комбинация загрузчика данных и стандартного загрузчика для DOS. Он поддерживает дополнительно к стандартному загрузчику фоновые запросы. Однако он не допускает шифрования внешних данных. Облегченный загрузчик требует очень мало памяти.

4.1.2 Базовая конвейеризация **Basic-Chaining/КНК-Programme (D)**

Если вы получите сообщение «Illegal function call in line xxxx in module xxxx at address xxxx:xxxx» – это значит, что ваша программа скорее всего использует конвейерную связь. Многие приложения DOS КНК используют этот метод для переключения между модулями.

Попробуйте использовать параметр **-ex:Byte** для поиска значения, с которым ваша программа будет работать. Этот параметр может быть использован для исключения некоторого количества байт начального кода из шифрования. Модули затем могут быть распознаны и загружены снова через код образа файла.

4.1.3 Обработка TSR программ (D)

Защита программ «Terminate and Stay Resident» (TSR) стандартным загрузчиком не вызывает никаких проблем.

При применении загрузчика данных надо учитывать следующее:

- никогда не используйте параметры **-!no:p** или **-!no:q**. Они могут привести к ошибкам приложения при работе с резидентными программами;
- программа TSR может быть удалена из памяти только при помощи горячей клавиши, если она уже загружена в память. Удаление из системы невозможно без перезапуска программы.

4.1.4 Обработка оверлеев (D)

Внешние оверлеи обрабатываются как файлы данных и подлежат процессу прозрачного шифрования, описанному выше (*раздел 2.4*).

При применении внутренних оверлеев надо учитывать следующее:

Загрузчик данных

- Шифрование внутренних оверлеев не является проблемой.
- Шифрование внутренних оверлеев может быть деактивировано параметром **-!ovl** для повышения производительности.
- Параметры **-w:o** и **-r:o** нельзя использовать при использовании шифрования внутренних оверлеев.

Стандартный загрузчик

- Стандартный загрузчик автоматически распознает большинство методов оверлеев. Автоматическое распознавание может быть изменено при помощи параметров **-o**, **-t**, **-p**.
- Внутренние оверлеи обычно не зашифрованы.

4.1.5 Применение DOS-экстендеров (расширителей) программ (D)

Программа DOS-экстендер выполняет приложение DOS в защищенном режиме. Для защиты в системе **HL-Crypt** действуют некоторые особенности. Существует два основных типа программ DOS-экстендер.

Программа с прилинкованным DOS-экстендером (EXE)

Файл может быть зашифрован системой **HL-Crypt**. Добавочный параметр **-!no:mq**. Он необходим.

```
HL-CRYPT MY386.EXE -m:29809 -!no:mq
```

DOS-экстендер и отдельное приложение DOS

Система **HL-Crypt** также может поддерживать эту программную структуру, но процедура намного более сложная. Поскольку программа, например, запускается через командную строку: `XM MYCOB.EXE`, система DOS запускает необходимый DOS-экстендер, который в свою очередь загружает и запускает отдельный файл `MYCOB.EXE`.

В этом случае защищайте вашу программу следующим образом:

1. Зашифруйте ваш программный файл `MYCOB.EXE` как файл данных, например, `DLCRYPT` (см. главу 2.3).
2. Защитите DOS-экстендер системой **HL-Crypt** и задайте шифрование программного файла. Кроме этого, задайте параметр **-!no:mq**

```
HL-CRYPT xm.exe -m:29809 -d:mycob.exe -k:42  
-!no:mq  
HL-CRYPT cc.exe -m:29809 -d:*. * -k:42  
-r:drename mycob.exe myprg.org  
dlcrypt.exe mycob.org mycob.exe -e -k:42  
-m:29809
```

Примечание: помните, что система обрабатывает файл `MYCOB.EXE` как файл данных. Он может быть запущен теперь только с помощью программы `XM.EXE` и только ей расшифрован. Он не может быть запущен сам по себе.

4.1.6 Quarterdecks DesqView (D)

Запуск приложения, защищенного при помощи **HL-Crypt** для DOS в среде DV не вызывает проблем, если значение параметра Protection Level для DV (уровень защиты) установлен в 1 или 0.

4.1.7 Влияние на скорость

Повышение скорости - это всегда снижение защиты. Но чаще всего можно найти компромисс.

- Уменьшите число защитных модулей по умолчанию (-s:CRYPT), (D,W).
- Используйте параметры **-w:hr** и **-r:hr** (D).

Следующие установки позволят увеличить скорость во время работы программы:

- Уменьшите частоту фоновых запросов (**-b:FREQ**) (D,W,W32).
- Ограничьте число защищенных файлов данных (**-d:FILES**) (D,W,W32).
- Используйте ступень безопасности 6 (-sec:6) (D,W,W32).
- Активируйте параметр **-qslren** (D,W).
- Активируйте параметр **-n** (D).

4.1.8 Интерфейс пользователя

Система **HL-Crypt** предоставляет вам наравне с командной строкой еще и графический интерфейс. Он может обслуживаться мышью, если установлен ее драйвер.

Но из-за того, что не все опции можно получить через интерфейс, и не была произведена интеграция всех программ в интерфейс пользователя, далее графический интерфейс не рассматривается.

Интерфейс активируется, когда **HL-Crypt** для DOS запускается без следующих параметров. Как вариант, Вы можете использовать **Espresso** из **Hardlock-Bistro**.

4.2 Перерасчет параметров **-le:** и **-ln:** на **-sec:** (D,W)

Для упрощения параметры **-le:LEVEL** и **-ln:TABLE** были объединены для создания одного параметра **-sec:LEVEL**. Но не все возможные значения могут быть скомбинированы. Еще может быть использовано старое преобразование параметров.

-le:ID	-ln:ID	-sec:LEVEL
0	(n/a)	1
2	(n/a)	2
3	14	3
3	9	4
4	8	5

4.3 Устаревшие параметры

4.3.1 sle:SLOTID (D,W,W32)

Параметр определяет для приложения идентификационный номер, на основе которого ключ проверяет действенность логинов. Значение может лежать в интервале [1..65530].

Использование параметра не рекомендуется. Используйте вместо этого новую функцию **HL-RUS** (см. главу 3.9).

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -acc:r -sle:111
```

4.3.2 exp (D,W,W32)

Этим параметром вы можете ввести в память ключа **Hardlock** дату окончания программы.

Для этого закодируйте ключ с соответствующим содержанием (см. *описание HL-Bistro, глава Cappuccino*). Запишите дату окончания в форме Eхuuуuуmmdd (EX год месяц день) в памяти ROM ключа **Hardlock** и отметьте оба последних регистра памяти значением, бинарным нулю.

Если модуль окончания работы во время запуска приложения распознает попытку обойти эту дату, все содержание ключа сотрется (на 0xFF).

Возможность снабдить программы датой остановки программы ограничена локальными ключами. Доступ к **HL-Server** не разрешается.

Использование этого параметра не рекомендуется. Вместо этого используйте новую функцию **HL-RUS** (см. главу 3.9)

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -exp
```

4.3.3 expwarn:DAYS (D,W,W32)

Передается предупреждающее сообщение, если оставшиеся дни превышают дату, заданную в аргументах. Этот параметр имеет смысл только вместе с **-exp**.

Использование этого параметра не рекомендуется. Вместо этого используйте новую функцию **HL-RUS** (см. главу 3.9).

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -expwarn:15
```

4.3.4 cnt:TIMES (D,W,W32)

Ограничивается число стартов программы.

Использование этого параметра не рекомендуется. Вместо этого используйте новую функцию **HL-RUS** (см. главу 3.9).

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -cnt:15
```

4.3.5 -cntwarn:TIMES (D,W,W32)

Передается предупреждающее сообщение, если оставшееся число возможных вызовов программы превышает значение аргумента. Этот параметр имеет значение только в соединении с **-cnt**.

Использование этого параметра не рекомендуется. Вместо этого используйте новую функцию **HL-RUS** (см. главу 3.9).

Пример

```
[HL-CRYPT] MYPROG.EXE -m:29809 -cnt:15  
-cntwarn:5
```

4.3.6 -prot:PROT (D)

С помощью этого параметра поиск предоставленного **HL-Server** модуля ограничивается определенными протоколами.

- prot:1** Поиск через протокол IPX.
- prot:2** Поиск через протокол r NetBios.
- prot:3** Оба протокола могут использоваться (по умолчанию).

Этот параметр может использоваться значительно гибче с использованием переменной среды HL_SEARCH. Он еще имеется, но не документируется.

Документация по Hardlock

Документация по **Hardlock** познакомит вас со всеми компонентами продуктовой линейки **Hardlock** и с возможностями защиты вашего программного обеспечения (ПО).

Документация включает в себя следующие разделы.

- **Техническое руководство по Hardlock** описывает принципы защиты ПО с помощью всех компонентов продуктовой линейки. Оно поможет вам выбрать правильную защиту для выполнения ваших задач.
- **Руководство по HL-Bistro** (визуальной среде программирования для Hardlock) описывает этот программный пакет, который позволяет вам проектировать систему защиты в привычной среде Windows.
- **Руководство по HL-Server** описывает применение Hardlock в сетевых приложениях.
- **Руководство по HL-Crypt** описывает применение автоматической защиты вашего ПО.
- **Руководство по HL-API** содержит обзор интерфейса прикладных базовых систем (спецификацию API) для ручной защиты ваших программ.
- **Раздел Понятия** поясняет основные термины и понятия, используемые в документации по Hardlock.

Все разделы дополняются интерактивной помощью (Help) к различным программам. Самые последние изменения и дополнения вы найдете в файлах Readme к программам. В руководствах и файлах интерактивной помощи (Help) применяются следующие типы шрифтовых выделений:

Обозначение	Функция	Пример
жирный	Понятия верхнего уровня Продукты	DatePCTV Hardlock Hardlock Twin
ЗАГЛАВНЫЕ БУКВЫ	Файлы и пути	HLDRV.EXE
Courier	Синтаксис	hlpatchup -m 29809

Содержание

1 Hardlock-сервер:

Hardlock в сети	1
1.1 Общие положения	1
1.2 Режим Demo	1
1.3 Принцип работы и предпосылки	2
1.3.1 Для пользователя	2
1.3.2 Для сервера	2
1.3.3 Поддерживаемые локальные сети	3
1.4 Типы HL-Server	4
1.5 Проверка установки HL-Server	4

2 Установка защиты и лицензирование	7
2.1 Лицензирование HL-Server и HL-LiMas	7
2.2 Лицензирование без HL-LiMaS	8
2.3 Автоматическая установка защиты с программой Espresso/HL-Crypt	9
2.4 Ручная установка защиты с помощью программы Hardlock API	10

3 HL-Server как 32-разрядное приложение WIN32 (HLS32)	13
3.1 Общие положения	13
3.2 Установка	14
3.3 Параметры командной строки	15
3.4 Интерфейс HL-Server (HLS32.EXE)	17
3.5 Программа HL-Server Administrator (HLSADMIN)	17
3.5.1 Меню File	19
3.5.2 Меню Edit	19
3.5.3 Меню Service	20
3.5.4 Меню Help	21

4 Программа HL-Server как NT-Service (HLS32SVC) ...	23
4.1 Общие положения	23
4.2 Установка	23
4.3 Параметры командной строки для установки программы	25
4.4 Параметры командной строки для управления программой (HLS32CMD.EXE)	26
4.5 Управление с помощью программы HL-Server-Administrator	26

5 Программа HL-Server как приложение Netware NLM	27
5.1 Общие положения	27
5.2 Установка	27
5.3 Параметры командной строки	28

6	Программа HL-Server для DOS TSR	31
6.1	Общие положения	31
6.2	Установка	32
6.3	Командные строки для DOS TSR	33
6.4	Интерфейс для DOS TSR	35
6.4.1	Обслуживание интерфейса	35
6.4.2	Описание меню и его функций	36
6.4.3	Меню Status	36
6.4.4	Меню Install	37
6.4.5	Меню System	38
6.4.6	Меню Local	38
6.4.7	Меню Remote	42
6.4.8	Меню Exit	43
7	Параметры командной строки	45
	-Параметр:[Аргументы] (Использование)	48
	-auto (N)	50
	-boxsize:NUMBER (D)	51
	-comm:PROT[,PROT] (D,W,N)	52
	-display (D)	53
	-enable:PROT (W)	54
	-forceNT (W)	55
	-install (D,W)	56
	-logindisable (D,N,W)	57
	-module:MOD[,LIZENZ] (D,N,W)	58
	-name:SNAME (D,N)	59
	-password:PASS (D,N,W)	60
	-quickoff (D)	61
	-remove (WSVC)	62
	-search:PORT[,PORT] (N)	63
	-start (WSVC)	64
	-stop (WSVC)	65
	-timeout:MIN (D,N,W)	66
	-uninstall (D,N)	67
8	Вспомогательные программы	68
8.1	HLOGIN	68
8.2	HLOGOUT	68
8.3	EDITEEP	69
8.4	HHT	69
9	Толкование терминов	70
10	Сообщения об ошибках	72
11	Проблемы, советы и замечания	75
11.1	Порты, сокет и фильтры	75
11.2	Помощь при зависании компьютера	75

11.3 HL-Server TSR и другие резидентные программы	76
11.4 Мультизадачность, Windows & OS/2 и DOS TSR	76
11.4.1 Windows 3.x и DOS TSR	76
11.4.2 Windows для рабочих групп (Workgroups) и DOS TSR	77
11.4.3 Windows 9x, Windows NT	78
11.4.4 OS/2 DOS-Box, LAN-Server и DOS TSR	78
11.5 Несколько систем HL-Server в сети	79
11.5.1 Дополнительная лицензия	80
11.5.2 Эмуляция Novell NetBios	80
11.5.3 Чрезмерная загрузка сети	81
11.5.4 Время в DOS	81

1 Hardlock-сервер: Hardlock в сети

1.1 Общие положения

HL-сервер позволяет использовать систему защиты **Hardlock** в сети. При этом собственно ключ, который делает возможным работу ваших защищенных прикладных программ, предоставляется для общего пользования на центральном сервере. Защищенные программы обращаются через сеть к этому ключу, причем Вы сможете установить, сколько возможно одновременных обращений к ключу (без **HL-LiMas**), или соответственно к отдельным модулям (с **HL-LiMas**).

Применение **HL-Server** у Ваших покупателей требует:

- наличия функционирующей сети;
- наличия самого ключа **HL-Server**;
- установки программного обеспечения **HL-Server**, которое управляет доступом к **HL-Server** (в DOS TSR, NLM, Win 32-приложении или Windows NT сервис).

Информацию о возможных рабочих (эксплуатационных) системах и протоколах вы получите в *главе 1.3*, связанные термины поясняются в *главе 9*.

Обращение по сети к модулю **HL-Server** вручную или автоматически должно быть предусмотрено при защите. При этом вы не должны выбирать один из способов, а можете использовать оба способа. Тогда модуль-**HL** будет искаться сначала локально а потом в сети. Дальнейшую информацию по применению **HL-Server** и возможностях лицензирования как с, так и без **LiMas** вы получите во второй главе.

1.2 Режим Demo

Многие функции **HL-Server** могут быть проверены в режиме Demo. Этот режим (для 10 загрузок логинов) включается автоматически из программы **HL-Server**, если при установке было задано применение тестового ключа **HL-Server**. Необходимый для этого модуль имеет адрес 29809. К этому модулю относятся все приведенные в этом руководстве примеры.

1.3 Принцип работы и предпосылки

1.3.1 Для пользователя

Прикладные программы, которые используют предоставленный сервером ключ, обращаются к **HL-Server** (через логин) и затем могут использовать сетевой ключ аналогично использованию локального на своем собственном компьютере. Библиотеки для доступа к сети встраиваются в программу либо автоматически с помощью **HL-Crypt** (см. руководство *HL-Crypt*), либо вручную (см. руководство *Hardlock API*).

Рабочие станции в сети, которым необходим один из защитных модулей **HL-Server**, используют только соответствующий сетевой протокол. В прикладных программах возможно использование следующих сетевых протоколов:

Возможные протоколы для пользователя

Протокол	DOS	W16	W32
IPX	x	x	x
Netbios	x	x	
TCP/IP		x	x

Когда защищенная программа получит доступ к модулю **HL**, она будет занесена в логин-таблицу сервера, которая обеспечивает столько позиций, сколько лицензий вы передали для вашего программного обеспечения. Когда достигается максимальное число одновременно работающих пользователей, не может быть запущена ни одна копия прикладной программы до тех пор, пока один из работающих пользователей не закроет своей копии.

Если прикладная программа определенное время не обращается к модулю **Hardlock**, ввод данных может быть автоматически деблокирован. Период времени ожидания устанавливается при установлении соединения с защитным модулем.

1.3.2 Для сервера

HL-Server работает с прямым доступом к уровню сетевого протокола. Отсюда следует, что для эксплуатации сервера необходимо подключение к функционирующей сети и должен быть загружен протокол (IPX, NetBios или TCP/IP).

Возможные протоколы на сервере

Протокол	DOS	W16	W32	NML
IPX	x	x	x	x
Netbios	x		x	
TCP/IP		x	x	x

Модули **HL-Server** имеются в виде внешних ключей (для параллельных портов) и платы расширения (для ISA и MCA-компьютеров) - см. *Техническое руководство*.

Одна программа **HL-Server** может поддерживать одновременно до десяти (в DOS - до трех) ключей **HL-Server** на одном компьютере. Для каждого из этих модулей закладывается своя собственная логин-таблица. Число возможных входов ограничено максимально 250 лицензиями на один модуль. В процессе инициализации своей программы и ключа вы можете сами установить допустимое число одновременно работающих пользователей от одного до максимального числа (250).

HL-Server может быть загружен на каждый компьютер только один раз.

1.3.3 Поддерживаемые локальные сети

Из-за почти бесконечного количества возможных комбинаций из типов сетей, протоколов и конфигураций мы сознательно отказываемся от перечисления всех подходящих вариантов.

Если ваша сеть не будет соответствовать вышеперечисленным критериям, позвоните нашему представителю, который даст вам индивидуальные советы или объяснит, поддерживает ли **HL-Server** вашу сеть.

1.4 Типы HL-Server

HL-Server может работать в различных рабочих средах. Вы можете воспользоваться следующей таблицей для выбора типа **HL-Server** и места его описания:

Система управления	HL-Server	Протоколы	Описание
DOS	HLServer, EXE	IPX,NetBios	глава 6
OS/2	HLServer EXE	IPX,NetBios	глава 6
Netware3.1x,4xx	HLServer,NML	IPX,TCP,IP	глава 5
Windows 3.1x	HLServer, EXE	IPX,NetBios	глава 6
Windows 95	HLS32,EXE	IPX,TCP,IP,NetBios	глава 3
Windows NT3.51,4.0	HLS32SVC,EXE	IPX,TCP,IP,NetBios	глава 4

Установка различных типов программы **HL-Server** описывается в данной главе. В качестве альтернативы к описанным в этой главе шагам вы можете интегрировать подпрограммы для установки программы **HL-Server** и необходимые драйверы с вашей программой, с установкой драйвера API и установкой программы **HL-Server API**. Подробное описание функций вы получите в файле HINSTALL.HLP (установка драйвера API) и в файле HSIAPI.PDF (установка сервера API); см. также в описании главу *Интерфейс прикладного уровня HL-API*.

1.5 Проверка установки HL-Server

После установки, которая описывается в *главах 3-6* для различных рабочих сред, программа **HL-Server** загружается. Заявленный модуль, (в примерах тест-модуль с адресом модуля 29809) может быть использован всеми компьютерами в сети.

Доступ к сетевому модулю может быть проверен с отдельных компьютеров с помощью программ DOS HLOGIN.EXE, HLOGOUT.EXE. Для этого запустите программу HLOGIN.EXE с указанием адреса модуля.

```
HLOGIN 29809
```

Корректное выполнение команды подтверждается сообщением:

```
Login to HL-Server Hardlock 29809 successful.
```

Доступ соответствующей станции к ключу сохраняется до тех пор, пока не последует команда отключения.

HLLOGOUT 29809

Правильное выполнение команды подтверждается следующим сообщением:

Logout from HL-Server Hardlock 29809 successful.

Подробное описание обеих вспомогательных программ вы найдете в *главе 8*.

2 Установка защиты и лицензирование

2.1 Лицензирование HL-Server и HL-LiMas

При кодировании ключей **HL-Server** вы имеете возможность используя **HL-LiMas** провести дифференцированное лицензирование модулей (см. также приложение к HL-LiMas). Вы имеете возможность использовать до 32768 модулей и выдать 32767 лицензий на каждый модуль.

Лицензирование пользователей по модулям заменяет лицензирование пользователей по ключам **HL-Server** и позволяет увеличить количество лицензий. С использованием **HL-LiMas** вы устанавливаете защиту каждого программного файла, и при каких условиях она начинает действовать. При этом вы можете использовать отдельно и в комбинации счетчик, модули времени действия программы. При запуске программы проверяется выполнение хотя бы одного благоприятного условия. Информация по лицензированию сохраняется как в памяти ключа **Hardlock**, так и в отдельных файлах с расширением ALF (Aladdin License File).

Для лицензирования необходимо выполнить следующие шаги.

- При защите вашей программы четко определите, при каких условиях запускается программа. При этом запуск программы вы можете сделать зависимым от лицензии на модуль, от времени действия и от глобального счетчика (с помощью **Espresso**, с помощью автоматической защиты с использованием **Hardlock**, или при ручной защите с использованием **HL-API**).
- Четко определите для вашего покупателя, сколько лицензий должно быть на один модуль, какая дата запуска распространяется на отдельные модули и всю программу и какую верхнюю границу имеет счетчик (например, с программой **Cappuccino**).

Из-за высокого количества лицензий лицензирование при применении **HL-LiMas** становится относительно сложным. Поэтому будьте особенно тщательны при подборе условий и четко проиграйте различные возможности, которые вытекают из вашего лицензирования.

Пример использования моделей

Условия старта программы

- Программа TEXT действует, если свободна одна лицензия для модуля 1 или для модуля 2.
- Программа TABELLE действует, если свободна одна лицензия для модуля 2.

Кодирование **Hardlock** для заказчика

- Модуль 1 имеет одну лицензию.
- Модуль 2 имеет три лицензии.

Применение

1. Пользователь запускает TEXT и TABELLE.

Назначается лицензия для модуля 1 (первое условие) и одна лицензия для модуля 2.

2. Второй пользователь запускает TABELLE.

Назначается одна лицензия для модуля 2.

3. Третий пользователь запускает TEXT.

Так как не осталось свободной лицензии для модуля 1, вызывается второе условие: назначается одна лицензия для модуля 2.

2.2 Лицензирование без HL-LiMaS

Без **HL-LiMaS** вы имеете возможность установить при применении программы **HL-Server** число лицензий для доступа к ключу **Hardlock**. Возможно до 250 лицензий.

Вы можете установить ограничение лицензий с помощью сигнатуры в домене RAM или в домене ROM памяти **Hardlock**.

Сигнатура имеет форму **Logins:nnnn**, четырехзначное число дается в десятичном формате.

домене RAM сигнатура небезопасна, так как она может быть изменена программным обеспечением. Единственное преимущество состоит в том, что лицензия может быть изменена без нового кодирования. Если вы хотите часто использовать эту возможность для себя, мы советуем вам применение **HL-LiMaS**: такая возможность модификации удобнее и надежнее.

Сигнатура в домене ROM устанавливается при кодировании ключа **Hardlock**. Это лицензирование не может быть изменено изменением программного обеспечения. Содержание памяти вы можете установить при кодировании с помощью **Sappuccino** (см. документацию *HL-Bistro*) или с помощью вспомогательной программы EDITEEP (см. главу 8.3).

Примечание. Если вы при использовании ключа для вашего приложения работаете с нормальным загрузчиком (HL-Crypt для программ в DOS, опция *-l*), никакое лицензирование не сможет быть проведено. Нормальный загрузчик используется только в фазе загрузки приложения, затем сразу отключается и полностью исчезает из памяти. При работе программы ни один запуск приложения не занимает **HL-Server** надолго. Это означает, что сколько бы ни запускались защищенные приложения, всегда будет найден **HL-Server** с корректным адресом. Эту особенность можно использовать, если вы хотите лицензировать ваше приложение без ограничения количества пользователей на одну сеть.

2.3 Автоматическая установка защиты с программой Espresso/HL-Crypt

При автоматической установке защиты с помощью программы **HL-Crypt** (см. раздел HL-Crypt) или программы **Espresso** (см. раздел HL-Bisto) вы должны сделать возможным доступ к ключу **Hardlock** по сети. Вы можете сделать возможными оба способа доступа (локальный и через сеть). Тогда подходящий ключ будет искаться сначала локально, а потом через сеть.

Espresso: активируйте при запуске программы **Hardlock** опцию поиска **HL-Server** в сети. Здесь же вы можете установить время окончания программы. Эта установка времени четко определяет, когда будет отключена загрузка в таблицу логинов, если клиент больше не обратился к ключу **Hardlock**.

HL-Crypt: Используйте **-acc:lr** (локально или через сеть) или **-acc:r** (исключительно через сеть), например:

```
[HL-Crypt] myprogram.exe -m 29809 -acc:lr
```

Если используется **LiMaS**, установите при автоматическом выполнении защиты условия, которые должны быть выполнены, для того чтобы включился защищенный файл программы (см. главу 2.1).

2.4 Ручная установка защиты с помощью программы Hardlock API

Активизация доступа через сеть

При активизации доступа с помощью команд **HL LOGIN** или **HLM LOGIN** задайте **DONT_CARE** (локально или через сеть) или **NET_DEVICE** (только через сеть) как вид доступа, например:

```
hlresult = HL_LOGIN(29809,NET_DEVICE,RefKey,VerKey) ;
```

Проверка числа лицензий (без HL-LiMaS)

Дополнительно при доступе вы можете во время работы программы опросить, сколько пользователей допустимо в таблице, и сколько пользователей реально работают в данный момент. Для этого используйте функции **HL MAXUSER** (допустимое количество пользователей) и **HL USERINF** (количество реально подключенных пользователей). Это имеет смысл, если вы не используете **HL-LiMaS**, так как с использованием **HL-LiMaS** лицензии выдаются только на каждый доступ, а не на ключ.

Распределение модулей и проверка числа лицензий с помощью HL-LiMaS

Если вы используете **HL-LiMaS**, вы можете сделать работу программ зависимой от определенных условий. Используйте для этого функции **HLM OCCUPYSLOT**, **HLM FREESLOT**, **HLM CHECKSLOT**, **HLM CHECKCOUNTER**, **HLM CHECKEXPDAT** и т.д., чтобы распределить лицензии по модулям, освободить и проверить текущее состояние.

Задание поведения

Есть возможность задать программе способ ее поведения в тех случаях, когда недоступен **HL-Server**, например при обрыве сети. Ваша программа может, например, выдавать сообщение пользователю или произвести нормальное завершение.

Дальнейшую информацию по ручной установке защиты вы получите в *главе HL-API*.

3 HL-Server как 32-разрядное приложение WIN32 (HLS32)

3.1 Общие положения

HL-Server для 32-битной программы Windows состоит из трех компонентов: сервиса, который работает в Windows NT (см. главу 4), приложения для Windows 9x и программы **HL-Server Administrator**, который может быть использован как в Windows 9x, так и в Windows NT. Имейте в виду, что ни одна из этих программ не должна работать под Win32s, 32-битном приложении Windows3.x. Программа HL-Server может быть загружена на один компьютер только один раз. Одной программой **HL-Server** поддерживаются до десяти ключей (модулей) **Hardlock**.

32-разрядное приложение программы **HL-Server** поддерживает протоколы IPX, Netbios и TCP/IP. Страница IPX может быть запрошена всеми IPX-клиентами, поддержку IP требуют клиенты Win16 и Win32. При этом TCP/IP должен быть доступен в сокетах Windows (WINSOCK.DLL или WSOCK32.DLL). Например, в случае с Windows NT, Windows 9x и WfW 3.11 с MS-TCP/IP. Другие реализации WINSOCK должны также действовать. Успешно были протестированы, например, CompuServe Internet Dialer und Trumpet WinSock.

Программа **HL-Server** может управляться непосредственно из командной строки. Этим самым становится возможным ввести функции программы **HL-Server** в пакетный файл. Тогда при вызове программы различные функции будут передаваться в форме параметров. Перечень возможных параметров вы найдете в *главе 3.3*.

HL-Server 32 может обслуживаться как из меню, так и с командной строки. Функции по большей части идентичны. Интерфейс Win32 располагает целым рядом дополнительных функций, например для показа системных данных и для тестирования модулей безопасности. **HL-Server-Administrator** может обслуживаться через интерфейс Windows.

3.2 Установка

Файлы для программы **HL-Server** в формате 32 разрядных приложений вы найдете в каталоге *HARDLOCK/HLSERVER/NT_95* на *Hardlock-CD*. Для установки у заказчика потребуются следующие файлы:

- HLS32.EXE для собственно программного обеспечения сервера.
- HLSADMIN.EXE для **HL-Server-Administrator**.
- HLSADMIN.HLP для помощи программе **Administrator**.
- HLSADMIN.CPL для удобного старта программы **Administrator** из управления системой (control panel).

От заказчиков потребуются следующие шаги.

- Установить драйвера **Hardlock** с помощью соответствующих Setup-файлов (см. *Техническую документацию*)
- Загрузить драйверы протокола.
- Установить внешний ключ **Hardlock** в принтерный интерфейс компьютера (LPT-Port) или внутренний ключ **HL-Server** в соответствующий слот (см. *Техническую документацию*).
- Скопировать и запустить HLS32.EXE.
- Скопировать HLSADMIN- файлы в системный каталог.

Затем программу **HL-Server** можно запускать.

1. Перейдите в каталог, куда вы загрузили программное обеспечение **HL-Server**.
2. Запустите программное обеспечение с указанием названий программ и адресов модулей (в примере - адрес тестирующего модуля).

```
HLS32 -m:29809
```

3. Если вы хотите использовать программу **HL-Server Administrator** для простого обслуживания программы **HL-Server 32**, запустите ее с указанием:

```
HLSADMIN
```

В качестве варианта вы можете запустить программу **HL-Server-Administrator** путем нажатия символа запуска в панели управления, если вы скопировали файл HLSERVER.CPL вместе с файлом HLSADMIN.EXE в системный каталог.

В заключение установка может быть протестирована с компьютера-клиента (см. гл. *Установка различных типов сервера*). Также имеется возможность встроить инсталляцию **HL-Server** и необходимых драйверов в вашу программу с помощью API-установки **Hardlock-Server**.

Подробное описание функций вы найдете в файлах HINSTALL.HLP и HSIAPI.PDF, см. *раздел HL-API*.

3.3 Параметры командной строки

Стандартно **HL-Server** как 32 разрядное приложение может быть вызван:

HLS32

Указание адреса модуля при этом не обязательно, адреса модулей, найденных ключей **Hardlock** ищутся при запросе и добавляются (autoadd).

Таблица включает следующие параметры, которые могут быть записаны или использованы в краткой форме. Подробное описание отдельных параметров вы найдете в *главе 7*.

Параметр	Значение	Примечание	Краткая форма
-?	Показывает справку по допустимым параметрам программы HL-Server		-?
-install	установка HL-Server. При этом должен быть указан адрес модуля	необязательный параметр	-i

Параметр	Значение	Примечание	Краткая форма
module:n[,m]	Адрес модуля ключа, который должен быть сохранен (может быть задан несколько раз). "n" обозначает десятичный адрес ключа, "m" - опциональное ограничение загрузки входа на данное число (другие ограничения, например, полученная от «Аладдина» лицензия, не могут быть превышены).	Необязательный параметр, HLS автоматически ищет ключи при запросе.	-m:n[m]
-comm:s[,s]	Выбирает один или несколько сетевых протоколов. Если этот параметр не задан, программа HL-Server поддерживает все имеющиеся в компьютере протоколы одновременно. "s" обозначает протоколы IPX или NetBios.		-c:s,[s]
-timeout:n	Устанавливает временные рамки (в минутах), по которым освобождается загрузка входов от зависших станций. Предварительно выставленное значение 15, возможные значения лежат между 0 (позиция timeout деактивирована) и 9999.		-t:n
-password:s	определяет пароль для погашения отдаленных входов. Этот пароль запрашивается при погашении загрузок входов через интерфейс.		-p:s
-forceNT	Вызывает старт HLS32 под Windows NT	Если не задано, вызывает под Nt указание на сервис	- f

3.4 Интерфейс HL-Server (HLS32.EXE)

Интерфейс программы **HL-Server** предлагает обзор модулей **HL-Server** и их особенностей:

- Module: адреса модулей ключа **HL-Server**;
- Logins: имеющееся число подключенных пользователей;
- Peak: пиковое число (с момента старта программы) подключенных пользователей;
- Limit: число разрешенных лицензий;
- Req.: число запросов к серверу;
- Errors: число ошибочных запросов к серверу (программе).

Если вы два раза щелкните мышью на соответствующий адрес ключа, откроется окно, в котором вы сможете увидеть все предоставленные вам для подключения рабочие станции с их текущим значением (узел или IP-адрес клиента , а также идентификатор задачи – ID), значение счетчика Timeout , а также дату подключения.

Если вы хотите иметь больше информации по текущему статусу вашего ключа, нажмите дважды на символ **Hardlock** в нижнем левом углу окна интерфейса **HL-Server**. Откроется следующее окно, в котором вы увидите детальную информацию о событиях в системе. Так, вы найдете здесь информацию о каждом единичном вызове API, запросившем ее объекте (адресе узла или соответствующем адресе IP), о текущем рабочем состоянии, а также об используемом протоколе передачи.

3.5 Программа HL-Server Administrator (HLSADMIN)

Интерфейс программы состоит в принципе из трех частей:

- из строки меню и панели инструментов с иконками (чья функция будет описана в следующей главе);
- списка всех активных программ **HL-Server** и подключенных ключей **HL-Server** в левой части экрана;
- окна диалога в правой части экрана.

В зависимости от того, выбирается в левом окне программа **HL-Server** или устанавливается связь с ключом, на правой стороне экрана появляется окно Server-диалога или **Hardlock**-диалога.

Информация о серверном компьютере

Если вы щелкнете мышью по вызванному серверу, вы получите следующую информацию:

- имя программы **HL-Server**, а если оно не известно, то имя компьютера;
- номер версии программы **HL-Server** (только с версии 3.00 могут быть добавлены или удалены ключи **HL-Server**, или протоколы активированы, деактивированы);
- активный адрес сети (зависимый от примененных протоколов);
- статистику опрарвленных и полученных пакетов данных для каждого протокола;
- управляющий элемент для используемых сетевых протоколов (эта установка действует только на стороне клиента), программа **HL-Server Administrator** может найти и другие программы **HL-Server**, независимо от того, где они установлены.

Информация по ключу HL-Server

Если вы щелкнете мышью на ключ **Hardlock**, вы получите следующую информацию:

- назначение входов/число подключенных пользователей;
- адреса портов подключенных клиентов (пожалуйста, обратите внимание: адреса протокола TCP/IP даются не в его оригинальном формате, например, 1.12.123.44, а в формате 001012123044, чтобы они были совместимы с программами DOS TSR, соответственно NLM-**HL-Server**);
- идентификатор задачи Task ID;
- значение счетчика времени Timeout;
- дата подключения;
- время подключения.

3.5.1 Меню File

Находящиеся здесь подменю позволят вам выполнить следующие действия.

Rescan позволяет произвести перезапуск процесса сканирования сети, при этом в соответствии с протоколом в сети ищутся и показываются подходящие программы **HL-Server**. Тем самым актуализируются внутренние таблицы программы **HL-Server Services**. Аналогичный процесс может быть начат при активации иконки с увеличительным стеклом.

Exit заканчивает работу программы **HL-Server Administrator**. Программа **HL-Server Service**, напротив, остается активной.

3.5.2 Меню Edit

Здесь вы найдете следующие подменю, которые облегчат вам управление имеющимися сетевыми ключами.

Add Hardlock добавляет новый сетевой ключ к только что запущенной программе **HL-Server**. При этом у вас попросят ввести соответствующий адрес модуля и пароль для замены подключенных к сетевому ключу пользователей сети. То же самое произойдет при щелчке мыши на иконку с символом **Hardlock** «зеленый плюс» или с помощью команды в командной строке: `hls32cmd add xxxxx`. При добавлении еще одного нового ключа автоматически запускается Rescan.

Remove Hardlock удаляет ключ из программы **HL-Server**. У вас попросят ввести соответствующий пароль. Эти действия могут быть проведены путем щелчка мыши на иконку с перечеркнутым красным символом **Hardlock**. После удаления ключа автоматически запускается программа сканирования сети **Rescan**.

Delete Login Entry удаляет текущее выбранное подключение к сетевому ключу. При этом вы должны ввести пароль, выбранный при установке ключа, т.е. старый пароль. То же самое можно достичь при щелчке мыши на иконку с перечеркнутым красным символом пользователя.

Change Password: изменяет действующий пароль для определенного ключа. Сначала вы должны ввести старый пароль. Эта команда может быть активирована только при помощи данного пункта меню.

Примечание: если удален сетевой ключ или строка загрузки, ключ с этого момента становится недоступным для соответствующей станции или приложения. Как это в дальнейшем отразится на активной программе, зависит от того, какой модуль защиты был встроен (например, от запроса ключа только при запуске или путем фоновых запросов). Ваше приложение должно в этом случае постараться снова восстановить контакт с HL-Server, но в любом случае действовать так, как было определено (см. главу 2).

3.5.3 Меню Service

Имеющиеся здесь подфункции облегчают вам работу с программой **HL-Server Services** в NT:

Start запускает программу, инсталлированную с помощью Install на локальном компьютере. Подтвердите запрос, должна ли быть запущена программа. После запуска программы начинается автоматическое сканирование всей сети. Эта функция может быть запущена командной строкой **hls32svc -start** или с помощью зеленой иконки светофора.

Stop останавливает работу программы **HL-Server Service** на локальном компьютере. Полное удаление этой программы возможно с помощью функции remove. После остановки программы проводится автоматическое сканирование сети. Аналогичная функция выполняется командной строкой **hls32svc -stop**, либо графически с помощью иконки с символом красного светофора.

Install устанавливает программу **HL-Server Service** на локальном компьютере. После этого программа должна быть запущена с помощью команды **Start**. Если Вы используете в качестве альтернативы зеленый символ-светофор, программа **HL-Server Service** будет установлена и запущена сразу. Командная строка для установки программы: **hls32svc -install**. После инсталляции программы происходит автоматический перезапуск сканирования сети.

Remove удаляет программу **HL-Server Service** с локального компьютера. После удаления начинается автоматическое сканирование всей сети. В качестве альтернативы здесь также можно использовать командную строку **hls32svc -remove** или иконку с красным символом светофора.

3.5.4 Меню Help

В этом меню вы найдете информацию и помощь по всем вопросам работы программы **HL-Server Administrator**.

Index активирует функцию помощи программы **HL-Server Administrator**. Эта функция может быть вызвана с помощью символа вопросительный знак на интерфейсе или нажатием функциональной кнопки F1.

About даст вам информацию о версии и т.д.

4 Программа **HL-Server** как NT-Service (**HLS32SVC**)

4.1 Общие положения

Программа **HL-Server** для 32-битного Windows состоит из трех частей: сервиса, который работает в Windows NT, приложения для Windows 9x, и программы **HL-Server Administrator**, которая работает как в Windows 9x, так и в Windows NT (см. главу 3). Обратите внимание, что ни одна из этих программ не работает в Win32s, 32-битном приложении Windows 3.x. На один компьютер программа **HL-Server** может быть загружена только один раз. Программой **HL-Server** поддерживается до трех ключей-**Hardlock**.

На уровне протоколов 32-битный **HL-Server** поддерживает протоколы IPX, Netbios и TCP/IP. Страница IPX может быть запрошена всеми клиентами, поддержку IP предусмотрена у клиентов Win16 и Win32. Протокол TCP/IP должен быть при этом доступен через Windows Sockets (WINSOCK.DLL соответственно WSOCK32.DLL). Это, например, характерно для Windows NT, Windows 9x и WfW 3.11 с MS-TCP/IP. Другие разработки WINSOCK должны также работать. Успешно были протестированы, например, CompuServe Internet Dialer и Trumpet WinSock.

Программа **HL-Server** может напрямую управляться командной строкой. При этом возможно связать функции программ **HL-Server** и **Batch**. Различные функции будут передаваться в форме параметров при вызове программы. Перечень возможных параметров вы получите в главе 4.3 и 4.4.

4.2 Установка

Файлы для программы **HL-Server** в виде 32 разрядного приложения вы найдете в перечне HARDLOCK/HLSERVER/NT_95 на *Hardlock-CD*. Для установки у заказчика потребуются следующие файлы:

- HLS32SVC.EXE — программа **HL-Server-Service**;
- HLS32CMD.EXE — управляющий модуль NT-Service с помощью командной строки;
- HLSADMIN.EXE — программа **HL-Server-Administrator**;

- HLSADMIN.HLP — помощь программы Administrator;
- HLSADMIN.CPL — контрольная панель для программы Administrators.

От заказчика потребуются следующие шаги:

- установить драйвер **Hardlock** с помощью инсталляционных файлов Setup (см. Техническую документацию);
- загрузить драйверы протокола;
- установить ключ **Hardlock** (внешний — в параллельный порт, а внутренний — в соответствующий слот);
- скопировать и запустить HLSVC.EXE;
- скопировать HLSADMIN- файлы в системный каталог.

После этого программу **HL-Server** можно запустить.

Для этого перейдите в каталог, куда вы загрузили программное обеспечение **HL-Server**, и подайте команду

HLSADMIN

В другом варианте запуск программы **HL-Server-Administrator** производится путем нажатия символа запуска в системном управлении, если вы скопировали файл HLSERVER.CPL вместе с файлом HLSADMIN.EXE в системный каталог.

На возникшем меню интерфейса нажмите зеленый символ-светофора. Тем самым будет инсталлирована и запущена программа **HL-Server Service**. Добавьте ключ **Hardlock** к локальному компьютеру путем нажатия иконки с символом ключа. При этом дайте по требованию адрес модуля (при использования тестового модуля – 29809).

Программа **HL-Server Service** может быть запущена из командной строки со следующими параметрами:

```
HLS32SVC -install  
HLS32SVC -start  
HLS32CMD -add 29809
```

В заключении установка может быть протестирована с компьютера-клиента (см. главу *Установка различных типов сервера*). Также имеется возможность встроить инсталляцию **HL-Server**

и необходимых драйверов в вашу программу с помощью API-установки **Hardlock-Server**. Подробное описание функций вы найдете в файлах HINSTALL.HLP и HSI-API.PDF, см. *раздел HL-API*.

4.3 Параметры командной строки для установки программы

Стандартным образом программа NT-Service может быть следующим образом установлена и запущена:

```
hls32svc -install
hls32svc -start
```

Указание адреса ключа при этом необязательно, адреса найденных ключей **Hardlock** будут при запросе найдены и добавлены (autoadd).

Следующая таблица содержит параметры. Подробное описание отдельных параметров вы найдете в *главе 7*.

Параметры	Значение	Краткая форма
-?	Показывает справку по допустимым параметрам программы	-?
-install	Устанавливает программу HL-Server Service	(нет)
-start	Запускает HL-Server Service	(нет)
-stop	Останавливает HL-Server Service	(нет)
-remove	Удаляет программу HL-Server Service	(нет)

4.4 Параметры командной строки для управления программой (HLS32CMD.EXE)

Если конфигурировать NT-Service не через программу **HL-ServerAdministrator**, а с помощью командной строки, то можно для этого использовать HLS32CMD.EXE.

Параметры	Значение
-add [moduleaddr]	Добавляет Hardlock-modul к действующему серверу
-remove [moduleaddr]	Отзывает ключ Hardlock от действующей программы
-timeout [minutes]	Устанавливает время Server-Timeout.
-enable [protocol]	Активирует протоколы (TCP, IPX, NetBios)
-disable [protocol]	Деактивирует протоколы (TCP, IPX, NetBios), NetBios будет только маркирован и после нового запуска программы деактивирован

4.5 Управление с помощью программы HL-Server-Administrator

Вы можете установить **HL-Server-Service** с помощью программы **HL-Server-Administrator**, удалить ее, запустить и остановить. Используйте для этого пункты меню Service. Подробное описание различных пунктов меню вы найдете в *главе 3.5*.

5 Программа **HL-Server** как приложение Netware NLM

5.1 Общие положения

Программа **NLM-HLSERVER.NLM** может быть напрямую установлена на сервер Novell File Server и функционирует в NetWare > 3.1x. Установка программы под Netware SFT III невозможна. На один компьютер программа **HL-Server** может быть загружена только один раз.

Для работы без проблем требуется современная версия файлов CLIB.NLM STREAMS.NLM. Файлы должны быть как минимум 1993 года. Обновления можно бесплатно скачать с Интернет-странички производителя. Проследите, чтобы использовались файлы из одного (последнего) пакета обновлений.

К сожалению, программа **HL-Server** не может всегда перед использованием CLIB.NLM определить, насколько она современна. Если вы получите сообщение на отсутствующие public Symbols (обычно «IsColorMonitor»), это скорее всего свидетельствует о том, что установлена очень старая версия CLIB.NLM.

Программой **HL-Server** можно управлять непосредственно из командной строки. Тем самым становится возможно внести функции программы **HL-Server** в пакетный файл (NCF). Различные функции при этом будут передаваться в форме параметров при его вызове. Параметры представлены в *главе 5.3*.

Программа **HL-Server** для NLM может, кроме того, обслуживаться из стандартного для Novell NetWare интерфейса.

5.2 Установка

Файлы для программы **HL-Server** в формате NLM вы найдете в перечне HARDLOCK/HLSERVER/ NLM на *Hardlock-CD*. Для установки у заказчика потребуются следующие файлы: HLSERVER.NLM и опционально HLSERVER.CFG и HLSERVER.NFC.

Для пользования командной строкой нужен только файл HLSERVER.NLM. Если вы хотите запустить программу **HL-Server** из стандартного интерфейса, рекомендуется дополнительно использовать файлы HLSERVER.CFG (файл конфигурации) и HLSERVER.NCF (простой Batch-файл).

Заказчику потребуется следующее.

- Для установки NLM вам необходим доступ к серверной консоли. Убедитесь, что вы обладаете соответствующими правами.
- Для запуска программы NLM файл HLSERVER.NLM должен находиться на дисковом сете. Если вы копируете файл в каталог [Servername]/SYS:SYSTEM вашего сервера NetWare, вы можете отказаться от указания директории при запуске NLM.
- Ключ **Hardlock** необходимо установить в принтерный порт компьютера (внешний) или в соответствующий слот (плата расширения), см. *Техническую документацию*.

Программа **HL-Server** может быть инсталлирована и запущена с системной консоли Novell NetWare командой

LOAD HLSERVER

Кроме того, инсталляция может быть тестирована с компьютера-клиента (см. главу *Установка различных типов сервера*). Также имеется возможность встроить инсталляцию **HL-Server** и необходимых драйверов в вашу программу с помощью API-установки **Hardlock-Server**. Подробное описание функций вы найдете в файлах HISTALL.HLP и HSI-API.PDF, см. *раздел HL-API*.

5.3 Параметры командной строки

Стандартно программа **HL-Server** как NLM может быть вызвана следующим образом:

load HLSERVER

Указание адреса ключа при этом необязательно, адреса найденных ключей будут определяться при запросе и добавляться (autoadd).

Деинсталляция производится с помощью команды:

unload HLSERVER

Таблица содержит следующие параметры, которые могут быть записаны или использованы в краткой форме. Подробное описание отдельных параметров вы найдете в *главе 7*.

Параметр	Значение	Замечание	Краткая форма
-?	Показывает справку по допустимым параметрам программы		-?
-install	Устанавливает HL-Server. При этом должен быть указан адрес модуля	необязательный параметр	-i
module:n[,m]	Адрес модуля ключа, который должен быть сохранен(может быть дан несколько раз). "n" обозначает десятичный адрес ключа, "m" опциональное ограничение загрузки входа на данное число (другие ограничения, например, полученная от «Аладдина» лицензия, не могут быть превышены)	необязательный параметр, HLS автоматически ищет ключи при запросе	-m:n[,m]
-comm:s[,s]	Выбирает один или несколько сетевых протоколов. Если этот параметр не задан, программа HL-Server поддерживает все имеющиеся в компьютере протоколы одновременно. "s" обозначает протоколы IPX или TCP		-c:s[,s]
-name:s	Определяет имя для программы HL-Server (не допускаются пробелы, максимально 12 знаков). Служит только для лучшего различия администрации разных HL-Server	имя по умолчанию - это имя процессора NetWare	-n:s
-logindisable	Предупреждает дальнейшие загрузки входов в программу HL-Server		-l
-timeout:n	Устанавливает временные рамки (в минутах), по которым освобождается загрузка входов от зависших узлов. Предварительно выставленное значение 15, возможные значения лежат между 0 (позиция timeout деактивирована) и 9999		-t:n

Параметр	Значение	Замечание	Краткая форма
-password:s	Определяет пароль для погашения отдаленных входов. Этот пароль запрашивается при погашении загрузок входов через интерфейс		-p:s
-auto	После инсталляции автоматически возвращается к экрану консоли		-a
-search:n[,n]	Утверждает адреса портов для последовательности поиска ключей Hardlock		-s:n[,n]

6 Программа **HL-Server** для DOS TSR

6.1 Общие положения

Программное обеспечение **HL-Server** работает в MS- или PC-DOS, начиная с версии 3.0. Резидентная программа SERVER.EXE требует в зависимости от числа управляемых ключей и загрузок входов, а также активных протоколов, примерно 70 KB основной памяти. При этом программа может быть загружена в область High Memory (DOS 5.0+ UMBs, QEMM, и т.д.). На один компьютер может быть загружена программа **HL-Server** только один раз.

Перед применением в сети программа **HL-Server** с желаемыми параметрами должна быть загружена резидентно. На компьютере с программой **HL-Server** должны применяться только стабильные приложения, которые не приведут к зависанию компьютера своими сбоями. В таком случае приложения, которые уже были в этой программе, тоже не будут работать корректно.

Программы для обслуживания сети, работающие в фоновом режиме, выполняются компьютером в первую очередь. Затрата необходимого для этого времени приводит к тому, что у программ, работающих на первом плане (в зависимости от объема сети и частоты запросов через сеть) возникают частичные простои.

Программой **HL-Server** можно управлять непосредственно из командной строки. Тем самым становится возможно записать функции программы **HL-Server** в пакетном файле. Различные функции при этом будут передаваться в форме параметров при его вызове. Параметры представлены в *главе 6.3*.

Программа **HL-Server** может обслуживаться и через интерфейс в виде меню. Функции большей частью идентичны. Интерфейс DOS в виде меню имеет ряд дополнительных функций (которые для собственного функционирования программы **HL-Server** не необходимы), это такие функции, как просмотр системных данных и тестирование защитных модулей. NLM могут также управляться через этот интерфейс.

6.2 Установка

Файлы для программы **HL-Server** для DOS вы найдете на *Hardlock-CD* в каталоге HLSERVER/DOS. Для установки у заказчика потребуются файлы HLSERVER.EXE, HLSERVER.OVL и HLSERVER.HLP.

Для использования командной строки обязательно нужен файл HLSERVER.EXE. Если вы дополнительно хотите использовать меню интерфейса и имеющуюся там помощь, тогда вы должны использовать еще и файлы HLSERVER.OVL и HLSERVER.HLP.

От заказчика потребуются следующие шаги:

- скопировать HLSERVER.EXE и HLSERVER.OVL;
- загрузить драйвер протокола (например, IPX и/или NetBios);
- установить ключ **Hardlock** на параллельный порт (внешний) или в соответствующий слот (внутренняя плата расширения) компьютера (см. *Техническую документацию*).

Программа **HL-Server** может быть запущена, например в режиме командной строки.

1. Перейдите в каталог, где вы инсталировали программное обеспечение **HL-Server**.
2. Запустите программное обеспечение с указанием имени программы и адреса ключа (в примере - адрес тест-модуля):

```
HLSERVER -i -m:29809
```

В заключение установка программы может быть протестирована с компьютера-клиента (см. главу *Установка различных типов сервера*). Также имеется возможность встроить инсталляцию **HL-Server** и необходимых драйверов в вашу программу с помощью API-установки **Hardlock-Server**. Подробное описание функций вы найдете в файлах HISTALL.HLP и HSI-API.PDF, см. *раздел HL-API*.

6.3 Командные строки для DOS TSR

Стандартно программа **HL-Server** как NLM может быть вызвана

HLSERVER -module:29809

Таблица содержит следующие параметры, которые могут быть записаны или использованы в краткой форме. Подробное описание отдельных параметров вы найдете в *главе 7*.

Параметр	Значение	Замечание	Краткая форма
-?	Показывает справку по допустимым параметрам программы		-?
-install	Устанавливает HL-Server. При этом должен быть указан адрес модуля	может быть выброшен	-i
module:n[m]	Адрес модуля ключа, который должен быть сохранен(может быть дан до трех раз). "n" обозначает десятичный адрес ключа, "m" опциональное ограничение загрузки входа на данное число (другие ограничения, например полученная от Аладдина лицензия, не могут быть превышены.	программа HL-Server для DOS не поддерживает автоматически добавление ключей!	-m:n[m]
-uninstall	Завершает работу программы HL-Server и удаляет ее из рабочей памяти. Если еще имеются подключения в таблицу входов, происходит запрос. Специально для разработчика предусмотрен параметр -uf, который отменяет этот защитный запрос.		-u,-uf (без подтверждения)

Параметр	Значение	Замечание	Краткая форма
display	Включает и выключает информацию по статусу о зарезервированных ключах, входах, пакетах данных в первую строку экрана в DOS (перед стартом нужно отключиться от Windows 3.x)		-d
-comm:s[s]	Выбирает один или несколько сетевых протоколов. Если этот параметр не задан, поддерживает программа HL-Server все имеющиеся в компьютере протоколы одновременно. "s" обозначает протоколы, IPX или NetBios		-c:s,[s]
-name:s	Определяет имя для программы HL-Server (не допускаются пробелы, максимально 12 знаков). Служит только для лучшего различия администрации разных HL-Server.	Имя по умолчанию "HLServer"	-n:s
-logindisable	Предупреждает дальнейшие загрузки входов в программу HL-Server		-l
-timeout:n	Устанавливает временные рамки (в минутах), по которым освобождается загрузка входов от зависших узлов. Предварительно выставленное значение 15, возможные значения лежат между 0 (позиция timeout деактивирована) и 9999		-t:n
-password:s	Определяет пароль для погашения отдаленных входов. Этот пароль запрашивается при погашении загрузок входов через интерфейс.		-p:s

Параметр	Значение	Замечание	Краткая форма
boxsize:n	Определяет число использованных приемных буферов	Необходимо только в экстремальных случаях	-b:n
-quickoff	Выключает сверхоперативную память (кэш) при доступе к памяти ключа Hardlock		-q

6.4 Интерфейс для DOS TSR

6.4.1 Обслуживание интерфейса

Интерфейс резидентной программы предлагает много функций тестирования для администратора программы **HL-Server**. Управлять им могут также установленные NLM. Также может быть проведена полная установка программы через интерфейс. Интерфейс программы TSR имеет следующую схему.

- Отдельные меню расположены одно под другим в левой части экрана (в поле меню), при этом активное в данный момент меню помечено маленькими треугольниками и черной тенью.
- Отдельные функции меню (опции), которые могут быть выбраны из активного меню, представлены в верхней строке экрана (в функциональной строке). Активный пункт отображается на желтом фоне.
- Интерфейс меню может обслуживаться как клавиатурой, так и мышью.
- Выбор меню и функций происходит с помощью комбинации клавиш на клавиатуре CTRL+клавиша со стрелкой.
- Отдельные опции внутри окна функций выбираются клавишей курсора.
- Функциональная клавиша F1 или правая кнопка мыши дает вам экранную подсказку по текущим функциям меню. Нажав клавишу ESC, вы возвратитесь к активным функциям меню.

6.4.2 Описание меню и его функций

Многие пункты меню понятны по названию, многие совпадают с названиями описанных параметров командной строки, поэтому их дальше не стоит объяснять.

Следующие ниже опции разделены в соответствии с меню программы **HL-Server** (на вертикальную панель) и их подменю (на горизонтальную строку).

6.4.3 Меню Status

Пункты этого меню обеспечат вас информацией о текущем состоянии локальной программы **HL-Server**. Вам предлагаются следующие функции.

About предоставляет информацию о текущей версии программы **HL-Server**, версии, использованной в программе **Hardlock API**, а также о статусе программы (резидентная или нет, что может быть определено с помощью картинки).

Main предоставляет информацию о локальных ключах (модулях защиты, если они установлены), о степени использования ключа **HL-Server**, а также другие сведения. Важной здесь является информация в строке «# of Overloads», которая показывает вам, как часто была перегрузка ключа (независимо от продолжительности перегрузки).

Если такие перегрузки наступают часто при нормальной эксплуатации, стоит перенести программу **HL-Server** на более мощный компьютер, который будет быстрее отвечать на поступающие запросы.

Tasks предоставляет информацию о внутренней мультизадачной системе **HL-Server**. Эта информация служит в основном для сервисных целей. Значения отдельных задач следующие:

Задача	Функция
Main	Установка. Инициализация. Снятие
Hardlock	Доступ к ключу
Supervisor	Административные функции HL-Server
IPX	Сетевой драйвер для IPX
NetBios	Сетевой драйвер для NetBios
Listen	Прерывание задачи для приема запросов через сеть (зависит от протокола)

Network предоставляет информацию о текущем состоянии сети, такую, как число использованных сокетов, размеры буфера и счетчик для принятых и переданных пакетов.

6.4.4 Меню Install

Это меню предлагает вам возможность инсталлировать программу **HL-Server** напрямую из интерфейса и выбрать необходимые для этого опции.

Install. При нажатии клавиши ENTER проводится инсталляция программы **HL-Server** с указанными командной строкой `iaðaiãððàiè. Äëÿ îîëñaiëÿ iaðaiãððîâ îðî+ðèððà` главу *Параметры командной строки*. здесь может быть выбрано использованное имя сервера. Если вы хотите найти другие модули защиты, вы можете выбрать в функции меню Local – Select до трех плат расширения ключей, которые, конечно, также должны иметься.

Uninstall. Эта функция меню дает вам возможность снова удалить резидентную программу **HL-Server** из рабочей памяти. Пожалуйста, учитывайте замечания в описании параметров к – *uninstall*.

Options. Предоставляет вам на выбор дополнительные функции программы. При этом учитывайте следующие разделы *главы 7*.

```
iaðaiãðð -display
iaðaiãðð -password
iaðaiãðð -logindisable
iaðaiãðð -comm
```

Опция **verbose debug display** служит для внутреннего обслуживания и может быть вызвана только с помощью пароля нашего обеспечения. Скорее всего, вам эта функция не потребуется.

Change Timeout. Используется для определения промежутка времени, по истечении которого запись регистрации зависшей станции будет удалена. Описание параметра **timeout** вы найдете в *главе 7*.

6.4.5 Меню System

Это меню предлагает вам возможность показать размещение памяти локального компьютера и отключение этого компьютера с установленной программой **HL-Server**.

Memory Map. Отображает размещение основной памяти. Показываются области, занимаемые программой **HL-Server**. Эта функция доступна, если вы работаете с MS-DOS версия 4.0 и выше.

Lock Console. Эта функция меню защищает компьютер, на котором вы установили программу **HL-Server**, от несанкционированного использования. В этом случае после ввода пароля клавиатура запирается и экран гаснет (чтобы избежать эффекта прожигания). Но на экране двигается змейка, которая сигнализирует, что компьютер включен и программа **HL-Server** активна.

Ввод пароля происходит сразу же после выбора опции **Lock console**. Для ввода пароля допускаются алфавитно-цифровые символы (большие и маленькие буквы и цифры от 0 до 9).

С новым вводом пароля вы можете снова активировать клавиатуру.

Примечание: не перепутайте этот пароль с паролем параметра —**password**, используемого для удаления строк регистрации.

6.4.6 Меню Local

Здесь помещены использованные адреса модуля ключа, важные функции тестирования для локальных и сетевых ключей. Поэтому описание будет подробным.

Select. Эта функция меню используется для трех задач:

- для выбора от 1 до 3 локальных ключей, которые становятся доступными для программы **HL-Server** ко времени инсталляции при использовании их в сети;
- для выбора одного из трех активных локальных ключей для опций **Local - EYE-Test** и **Local – Logins**;
- для ограничения числа подключений к определенным ключам.

Ввод адресов ключей. По умолчанию первым в списке ключей указывается адрес тест-ключа 29809. Для получения других адресов ключей подведите курсор к одной из трех строк (или нажмите на одну из строк мышью) и нажмите клавишу ENTER. Теперь вы можете изменить адрес ключа или задать его заново. Ввод подтверждается клавишей ENTER.

Опционально можно ограничить допустимое число строк в таблице регистрации. «0» означает отсутствие ограничения (см. описание параметра – **module** в главе 7).

Выбор ключа. Если у вас активированы несколько локальных ключей, только один может быть выбран для выполнения функций меню **EyeTest** и **Logins**.

Примечание. Компьютер с программой **HL-Server** не может поддерживать несколько ключей с идентичными адресами модулей, но разными субкодами. В этом случае лицензии складываются вместе, но только первый из найденных в последовательности портов LPT1 - LPT2 - LPT3 ключ используется для шифрования. Сложение лицензий **HL-Server** может быть использовано специально для расширения возможностей **HL-Server**.

На различных компьютерах сети возможна комбинация ключей с одинаковыми адресами модулей (см. главу *Термины*). Обратите внимание, что для различия ключей с одним адресом модуля, но разными кодировками (код компании, субкод) необходимы для распознавания нужного ключа (см. руководство *HL-API*).

Если в одной сети установлены несколько программ **HL-Server**, которые обслуживают одинаковые ключи, то их лицензии будут складываться вместе. Конкретно запросы на регистрацию обрабатываются самым быстрым компьютером с программой **HL-Server**, где еще имеются свободные строки в таблице регистрации. Компьютеры с заполненной таблицей регистрации на запросы не отвечают.

EyeTest. Эта функция используется для тестирования локального ключа, выбранного с помощью функции меню Select. Этот тест идентифицирует и отражает соответствующий адрес порта, выполняет тестовое шифрование (отображаются начальная строка символов base string и результат шифрования encrypted), и содержание областей ROM и RAM показывается.

Может быть выбрано отображение содержания памяти EEPROM и RAM между Big Endian (старший байт слева, младший байт справа) или в формате Little Endian (стандарт Intel: младший байт слева, старший байт справа). Переключение между этими режимами отображения происходит с помощью клавиши TAB или щелчка мыши в поле, помеченном квадратными скобками. При активации формата Big Endian появляется символ «галочка».

Переключение на другой адрес модуля вы можете провести напрямую, не покидая опцию меню **Eye-Test**. Нажмите мышью на поле ввода адреса модуля или нажмите клавишу ENTER. Тем самым будет погашен показываемый адрес модуля, и новый, для этого теста нужный адрес, может быть введен.

Logins. В окне этой функции меню показываются параметры работы ключа, выбранного при помощи функции меню Select.

Login Count. В этой строке отображаются следующие значения из таблицы регистрации.

- **active:** текущее число строк регистрации.
- **peak:** наибольшее число строк регистрации, которые к любому моменту времени были загружены или имеются в наличии (начиная с момента загрузки программы в резидентном режиме).
- **max:** максимальное число строк регистрации, которое допускается для данного ключа.

E-Y-E Requests. В этой строке указывается число обращений к ключу с момента загрузки программы **HL-Server**.

Примечание. Счетчик доступа к модулю защиты фиксирует также обращения в администраторских целях (таких как LOGIN и LOGOUT), так как они также выполняют проверку ключа.

В нижней части функционального окна представлена таблица регистраций модуля защиты со всеми активными строками регистрации. Любая строка регистрации характеризуется следующим.

- **No.:** Номер, используемый в таблице.
- **Address.:** 12-символьный шестнадцатеричный аппаратный адрес сетевого адаптера подключенного клиента.

- **Task ID:** В этой графе возникает идентификатор задачи подключенного клиента. Этот идентификатор либо генерируется автоматически программой **HL-Crypt**, либо указывается создателем приложения при ручной работе с **Hardlock API**. Идентификатор задачи разрешает использовать многим программам с одного компьютера одновременно программу **HL-Server** без взаимного влияния одной на другую (например, Windows 95, Windows NT, OS/2-или DESQview DOS-панелях).
- **Timeout:** В этой графе отображено время, оставшееся до истечения периода таймаута. По прошествии этого времени строки регистрации автоматически удаляются.
- **DATE/TIME:** В этих графах находятся дата и время помещения отдельных строк в таблицу регистрации.

Удаление строки из таблицы регистрации

Вы можете использовать эту функцию, чтобы убрать существующие строки из таблицы регистрации. Это делается так:

1. поместите строку, которую вы хотите удалить, с помощью щелчка мыши или перемещения курсора на желаемую строку таблицы;
2. нажмите клавишу ENTER или щелкните на поле [del] в нижней строке окна для удаления строки таблицы;
3. если строки таблицы защищены паролем при помощи пункта setup options, возникнет сообщение о просьбе ввести пароль (этот шаг применяется только для удаленных строк таблицы);
4. строка таблицы будет удалена только тогда, когда вы подтвердите удаление нажатием клавиши Y после предупреждающего сообщения. Если вы нажмете любую другую кнопку, удаление будет прервано, и строка не будет удалена.

Примечание. Как только вы удалите строку регистрации для данной станции, ключ для нее перестанет быть доступным. Реакция приложения будет зависеть от способа использования ключа. (Например, запрос модуля только при старте программы или фоновый запрос). В любом случае ваше приложение должно попытаться восстановить связь с **HL-Server** и вести себя определенным образом.

Stress. Эта функция меню подвергает локальные ключи, которые активированы опцией Select, особому тесту на стресс.

Система выполняет случайные операции чтения и шифрования для выбранных ключей; и успешные, и ошибочные результаты собираются и отображаются (успешные – желтыми цифрами, ошибочные – красными). Дополнительно общее число процедур доступа и скорость доступа (в пакетах/в секунду) отображаются в нижней части функционального окна.

Примечание. Указанное значение для скорости доступа не является корректным, так как включает время, необходимое для отображения текущего значения счетчика в графическом интерфейсе. Максимальная скорость доступа, достигаемая без использования интерфейса в виде меню, значительно выше отображаемого значения.

Нормальным является отсутствие ошибочных событий, это значит, что красное число должно быть нулем. Если все же будут зарегистрированы ошибки, причины их должны быть определены и устранены. Иначе приложения, имеющие доступ к ключу **HL-Server**, могут работать неверно. Исключением является случай, когда вы подвергаете один ключ тесту на стресс одновременно с нескольких станций (с помощью пункта remote – stress). Это приводит к частоте запросов, намного превышающей нормальный режим, что может перегрузить сеть. (При работе приложения в нормальных условиях дополнительная нагрузка сети, создаваемая программой **HL-Server**, незначительна).

6.4.7 Меню Remote

Пункты этого меню касаются не локальных, а удаленных ключей.

Select / EyeTest / Logins / Stress. Обслуживание и работа этого меню соответствуют почти полностью описанным выше в предыдущих разделах локальным функциям меню. Здесь учитываются многие защитные модули, которые обслуживаются программой **HL-Server** на всех станциях внутри сети. (При тесте на стресс используются только первые десять найденных ключей).

При первом входе в меню **Remote** после активизации графического интерфейса происходит сканирование сети на предмет определения доступных ключей. Так как эта процедура может занять определенное время, в зависимости от размеров сети, она проводится только при первом запросе меню.

Все доступные удаленные ключи отображаются на экране и могут быть выбраны для функций меню **Eye-Test** или **Stress**. Для прокрутки таблиц, которые не помещаются на экране, используйте клавиши со стрелками или щелкните мышью на левой стороне таблицы.

Примечание. Функции меню **Temote/Eyetest** и **Remote/Logins** могут использоваться только тогда, когда в таблицах регистрации соответствующих ключей есть свободное место, так как и этот тест должен быть сначала зарегистрирован для получения доступа к определенному ключу. При выполнении всех подключений это приведет к пиковому значению.

Если в таблице нет свободного места, никакие подключения не будут проводиться, и на экране появится соответствующее сообщение.

Rescan. Функция меню **Remote/rescan** проводит поиск всех доступных в сети удаленных ключей. Она обновляет все внутренние таблицы интерфейса меню **HL-Server**.

Этот пункт меню позволяет вам обновить таблицу доступных ключей при необходимости.

Будут отображены адреса модулей, опциональные имена программ **HL-Server**, сетевые адреса и (как сокращение) сетевые протоколы, к которым имеет доступ интерфейс в виде меню программы **HL-Server**.

Для поиска будут использоваться все сетевые протоколы, указанные в пункте **Setup/Options**.

Используйте эту функцию, если вы считаете, что с момента первого обращения к этому меню в сети или в конфигурации системы **HL-Server** произошли некоторые изменения. (Например, появились новые ключи, снят ключ, завис компьютер, выключен сетевой мост и т.д.).

6.4.8 Меню Exit

Есть много возможностей покинуть интерфейс программы **HL-Server** и вернуться в DOS.

Работа с меню с помощью клавиатуры или мыши.

- Выберите меню Exit (комбинацией клавиш CTRL и клавиша со стрелкой), и затем нажмите ESC.

Только с помощью клавиатуры.

Вы можете в любое время покинуть программу **HL-Server** напрямую из любого меню, нажав комбинацию клавиш ALT+X или просто нажав два раза подряд клавишу ESC.

7 Параметры командной строки

В этой главе описываются параметры командной строки, с помощью которых может быть запущена программа **HL-Server**. Не все параметры могут использоваться с разными версиями (TSR, NLM, Win32). Перечень когда-либо используемых параметров вы получите при описании типов **HL-Server** в главах 3-6.

Применяемость параметров обозначена в следующем описании:

Параметр	Применение
D	HL-Server как DOS TSR
N	HL-Server как Netware NLM
W	HL-Server как 32-bit Windows-Applikation
WSVC	HL-Server как NT-Service (Win32)

Начинайте все параметры со знака «-» или «/» без последующего пробела. Отдельные параметры отделяйте пробелом. Соединяйте параметр и аргумент двоеточием (:) без пробела.

Обозначение отдельных параметров может быть полным или (у некоторых параметров) только первыми буквами. Две командные строки в следующем примере являются идентичными в своих функциях:

```
HLSERVER -install -module:29809,5 -timeout:10
HLSERVER -i -m:29809,5 -t:10
```

Параметры командной строки

Параметр	Значение	Использование	Краткая форма
-?	Показывает возможные параметры HL-Server	D, N, W, WSV	-?
-auto	Возвращает после установки программы к экрану консоли назад.	N	-a
boxsize:n	Определяет число использованных приемных буферов	D	-b:n
-comm:s[s]	Выбирает один или несколько сетевых протоколов. Если этот параметр не задан, поддерживает программа HL-Server все имеющиеся в компьютере протоколы одновременно. "s" обозначает протоколы, IPX или NetBios	D, W, N	-c:s,[s]
display	Включает и выключает информацию по статусу о зарезервированных ключах, входах, пакетах данных в первую строку экрана в DOS (перед стартом нужно отключиться от Windows 3.x)	D	-d
-enable:s	Активирует один протокол	W	-e:s
-forceNT	Обеспечивает старт HL32 под Windows NT	W	-f
-install	Установка HL-Server При этом должен быть указан адрес модуля [-module:n[m]]	D, W, WSV D, N, W, WSV	-i
-logindisable	Предупреждает дальнейшие загрузки строк регистрации в программу HL-Server	D, N	-l

Параметр	Значение	Использование	Краткая форма
module:n[,m]	Адрес модуля ключа, который должен быть обслужен (может быть дан до трех раз). "n" обозначает десятичный адрес ключа, "m" - опциональное ограничение строк регистрации на данное число (другие ограничения, например, полученная от «Аладина» лицензия, не могут быть превышены	D, N, W	-m:n[,m]
-name:s	Определяет имя для программы HL-Server (не допускаются пробелы, максимально 12 знаков). Служит только для лучшего различия администрации разных HL-Server	D, N	-n:s
-password:s	Определяет пароль для погашения отдаленных входов. Этот пароль запрашивается при погашении строк регистрации через интерфейс	D, N, W	-p:s
-quickoff	Выключает сверхоперативную память, (кэш) при доступе к памяти ключа Hardlock	D	-q
-remove	Удаляет программу HL-Server Service	WSVC	нет
-search:n[,n]	Устанавливает адреса портов ключей для последовательности поиска	N	-s:n[,n]
-start	Запускает программу HL-Server Service	WSVC	нет
-stop	Останавливает программу HL-Server Service	WSVC	нет

Параметр	Значение	Использо- вание	Краткая форма
-timeout:n	Устанавливает временные рамки (в минутах), по которым освобождается строки регистрации от зависших узлов. Предварительно выставленное значение 15, возможные значения лежат между 0 (позиция timeout деактивирована) и 9999	D,N,W	-t:n
-uninstall	Заканчивает работу программы HL-Server и удаляет ее из рабочей памяти. Если еще имеются подключения в таблицу входов, происходит запрос. Специально для разработчика предусмотрен параметр -uf , который отменяет этот защитный запрос	D	-u,-uf (без запроса)

Описание параметров

Описание параметров построено по следующей схеме:

–Параметр:[Аргументы] (Использование)

Краткое описание параметра.

Аргументы

Список аргументов, которые нуждаются в этом параметре

Использование

Детальное описание параметра. Указания по его использованию.

Пример

İðëiað è ëðàòëia iienàieà ii ëñiieüçiaâieð
iaðàiaòðà. Çààaiiûa iaðàiaòðû áóáóò çääñû
ëñiieüçiaâòñý â èò ëðàòëie ôiðia.

Ðaçëë-iûâ iðiaðàliû HL-Server íaiçia-àðòñý
[HLSERVER]. Çaiâieòâ ýòè àâiiûa äëý iðiaðàli ia
ñëääópùèâ:

D	HLSERVER
N	load HLSERVER
Win32-	HLS32
WinNT-Service	HLS32SVC

-auto (N)

Автоматически возвращает после инсталляции к экрану консоли.

Аргументы

нет

Применение

Используется для автоматического возврата на экран консоли после установки программы HL-Server.

Пример

```
LOAD HLSERVER -i -m:29809 -a
```

-boxsize:NUMBER (D)

Определяет число использованных принимающих буферов.

Аргументы

NUMBER	6	минимум
	16	по умолчанию
	128	максимум

Применение

Полученные пакеты данных для увеличения скорости работы должны быть помещены в буфер. Если не много рабочих станций обращаются к программе **HL-Server**, вы можете освобождать резидентную память (около 400 байт на буфер) путем уменьшения числа приемных буферов. При нагрузке число буферов может быть увеличено. Установленного по умолчанию числа буферов обычно бывает достаточно.

Пример

```
HLSERVER -i -m:29809 -b:80
```

Повышает число использованных буферов до 80.

-comm:PROT[,PROT] (D,W,N)

Выбирает один или несколько сетевых протоколов.

Аргументы

PROT действующие параметры IPX и NETBIOS

Использование

Определите драйвер протокола, который будет «слушать» программа **HL-Server**. Если этот параметр не будет выбран, система будет поддерживать все имеющиеся сетевые протоколы.

Пример

```
[HLSERVER] -i -m:29809 -c:ipx
```

HL-Server будет поддерживать только протокол IPX.

-display (D)

Показывает состояние в первой строке экрана.

Аргументы

Нет

Использование

Этим параметром вы можете отобразить и убрать строку состояния как при инсталляции программы **HL-Server**, так и позднее в среде DOS. Не используете опцию, если вы работаете в Windows 3.x при загруженной программе **HL-Server**. Это может привести к искажению на экране Windows.

Пример

```
HLSERVER -d
```

Включает, выключает строку состояния, когда программа **HL-Server** только что загружена в резидентном режиме.

```
HLSERVER -i -m:29809 -d
```

Включает строку состояния при установке программы.

Строка состояния программы **HL-Server**.

-enable:PROT (W)

Активирует сетевые протоколы из системы.

Аргументы

PROT действующие параметры IPX и NETBIOS

Применение

Этим протоколом активируется NETBIOS для 32-х битной версии **HL-Server**. Обратите внимание, что применение протокола NETBIOS в Win 95 приведет к большим задержкам при запуске программы **HL-Server**.

Пример

```
HLs32 -e:netbios
```

-forceNT (W)

Заставляет Windows 95 Server (HLS32) запускаться в Windows NT.

Аргументы

Нет

Использование

С этим параметром HLS32 может быть запущена и в Windows NT для тестовых целей. Такая процедура не желательна для нормальной работы. Используйте для Windows NT соответствующую программу **HL Server Service**.

Пример

```
HLS32 -i -m:29809 -f
```

-install (D,W)

Устанавливает программное обеспечение **HL-Server**.

Аргументы:

Нет

Использование

Устанавливает программу **HL-Server**. В зависимости от используемой топологии сети и протоколов установка программы может занять несколько минут. В это время производится сканирование сети на предмет поиска всех адаптеров и идентификация всех сегментов сети для дальнейшей работы.

Пример

```
[HLSERVER] -i
```

Устанавливает программу **HL-Server**.

-logindisable (D,N,W)

Используется для запрета подключения к таблице регистрации других станций.

Аргументы

Нет

Применение

Этим параметром разрешается или запрещается подключение к таблице регистрации. Этот параметр может быть использован для тестирования и обслуживания отключения пользователей от регистрации.

Пример

```
[HLSERVER] -1
```

Для программы **HL-Server NLM** команда может отдаваться только в интерфейсе NLM.

-module:MOD[,LIZENZ] (D,N,W)

Передаёт программе **HL-Server** адреса модуля ключа.

Аргументы

MOD	адрес модуля ключа Hardlock
LIZENZ	опциональное число лицензий, зарезервированных для ключа

Применение

Адрес модуля используемого ключа должен быть указан при установке программы **HL-Server**. Чтобы установить программу для многих ключей, нужно указать до десяти (в DOS - трех) адресов. Опционально можно ограничить число записей в таблицу регистрации на один ключ (например, чтобы распределить все используемые лицензии по многим адресам модулей). Из соображений безопасности этот вариант не рекомендуется и используется только в целях тестирования.

Пример

```
[HLSERVER] -i -m:29809,4 -m:29822,6
```

Устанавливает программу **HL-Server** и предоставляет сети два ключа с заданными адресами. При этом резервируются четыре позиции в таблице регистрации для первого ключа и шесть – для второго. Таким образом, можно распределить 10 лицензий программы **HL-Server** между двумя адресами ключей. Если будет отсутствовать число строк, все доступные строки позиции 0 будут назначены сначала для первого ключа, а для второго может не остаться свободных строк (это действует только при условии, что в EEPROM памяти первого ключа не было указано ограничений на число лицензий).

-name:SNAME (D,N)

Определяет имя программы **HL-Server**.

Аргументы

SNAME	Не допускается пробелов, можно использовать до 12 символов
--------------	--

Применение

В одном сегменте цепи программа HL-Server может быть установлена не более чем на 250 компьютерах. Чтобы однозначно идентифицировать отдельные программы **HL-Server** в сети (например, при администрировании через интерфейс **HL-Server**), каждой программе при запуске может быть дано имя. Если не будет дано имя, система по умолчанию использует имя HLSERVER в DOS, в NetWare имя программы - NetWare-Server.

Пример

```
[LOAD] HLSERVER -i -m:29809 -n:PRODUCTION
```

-password:PASS (D,N,W)

Определяет пароль для погашения отдаленных строк регистрации.

Аргументы

PASS	Разрешено до 8 алфавитно-цифровых символов строчными и заглавными буквами и цифры от 0 до 9
-------------	---

Применение

При установке программы **HL-Server** может быть назначен пароль для предотвращения несанкционированного удаления строк регистрации. Если такой пароль установлен, то «удаленные» строки регистрации могут быть вычеркнуты из таблицы только при применении этого пароля.

Пример

```
[HLSERVER] -i -m:29809 -p:Secret
```

-quickoff (D)

Выключает кэш при обращении к памяти ключа.

Аргументы

Нет

Использование

Чтобы получить быстрый доступ к памяти ключа **Hardlock**, все операции чтения выполняются с использованием кэш-памяти. Кэш-память может быть отключена этим параметром. Но по соображениям скорости это не рекомендуется. Параметр предназначен для тестовых целей и специальных применений.

Пример

```
HLSERVER -i -m:29809 -q
```

-remove (WSVC)

Удаляет программу **HL-Server Service** из системы.

Аргументы

Нет

Применение

Этим параметром программа **HL-Server Service** окончательно удаляется из системы. Чтобы снова применить программу **HL-Server 32** на компьютере с Windows NT, нужно заново установить программу **HL-Server Service**.

Пример

```
(W)  HLS32SVC -remove
```

Обратите внимание, что этот параметр не должен быть сокращен. Прежде чем вы полностью удалите программу **HL-Server Service**, ее удаление должно быть сначала остановлено параметром **-stop**.

-search:PORT[,PORT] (N)

Устанавливает последовательность поиска ключей.

Аргументы

PORT	Адрес порта в шестнадцатеричном представлении
-------------	---

Применение

Этим параметром подробно задается последовательность поиска. Это предотвратит конфликт при возникновении проблем из-за автоматического поиска адресов параллельных портов (например, если на них сконфигурированы сетевые адаптеры).

Пример

```
LOAD HLSERVER -i -m:29809 -s:378,278
```

При инициализации программы **HL-Server** ключ **Hardlock** будет искаться только на параллельных портах с адресами 378 и 278 (оба адреса в шестнадцатеричном виде).

-start (WSVC)

Запускает программу **Hardlock Service**.

Аргументы

Нет

Применение

Этим параметром запускается программа **HL-Server Service** при использовании интерфейса в виде командной строки.

Пример

```
HLS32SVC -start
```

Обратите внимание, что этот параметр не может быть сокращен. Перед тем, как может быть запущена программа **HL-Server Service**, она должна быть установлена параметром **-install**.

-stop (WSVC)

Закрывает программу **HL-Server Service**.

Аргументы

Нет

Применение

Этим параметром программа **HL-Server Service** закрывается, но не удаляется.

Пример

```
HLS32SVC -stop
```

Обратите внимание, что этот параметр не может быть сокращен. Если вы полностью хотите удалить программу **HL-Server Service**, вы должны дополнительно применить параметр **-remove**.

-timeout:MIN (D,N,W)

Устанавливает время, после которого строки таблицы регистрации, к которым больше не обращаются, будут удалены.

Аргументы

MIN	0	timeout отключен
	10-9999	тайм-аут в минутах
	15	значение по умолчанию

Применение

Из опыта известно, что как минимум при разработке программы компьютер может зависнуть и будет восстановлен только путем «теплой» перезагрузки или сброса аппаратных средств. В этом случае строка регистрации соответствующей станции будет занята в таблице программы **HL-Server**. Таблица регистрации будет заполнена неиспользованными строками регистрации. Чтобы избежать этого эффекта, этим параметром устанавливается время. Если за это время не произойдет обращение с рабочей станции к ключу, то строка регистрации будет автоматически удалена из таблицы. Освобождается место для других приложений.

Обратите внимание, чтобы при защите программ с ручным подключением было достаточно времени для того, чтобы ваше приложение успело сделать запрос к программе **HL-Server**. Если этого не произойдет, строка регистрации будет просто удалена по истечении заданного времени. (Эта ситуация наступает, если программа ожидает ввода без запроса к ключу.) Поэтому выберите как можно больше времени с учетом установки ручной защиты, в крайнем случае деактивируйте повторную проверку.

Пример

```
[HLSERVER] -i -m:29809 -t:45
```

Эта команда устанавливает программу **HL-Server** с тайм-аутом в 45 минут. Тайм-аут может быть дополнительно изменен в DOS TSR (при NLM только через интерфейс меню). В программе **HL-Server 32** тайм-аут не может быть дополнительно изменен. Программы **HL-Server Administrator** и **HL-Server Service** в настоящее время не имеют опции тайм-аут.

-uninstall (D,N)

Удаляет программу **HL-Server** из рабочей памяти.

Аргументы

Нет

Применение

Удаляет загруженную программу **HL-Server** из памяти. Вы не должны удалять программу из памяти, пока другие рабочие станции в сети подключены к одному из ключей. Защитные модули не смогут служить программам, и эти программы не смогут работать из-за невозможности доступа к ключам. Поэтому, если таблица регистрации не пуста, при попытке удаления программы HL-Server система задаст вопрос:

```
Login table not empty, uninstall anyway (y/n) ?
```

Если вы ответите положительно, программа, несмотря на имеющиеся строки в таблице регистрации, будет удалена.

Пример

```
[HLSERVER] -u
```

Удаляет программу **HL-Server**. Если таблица регистрации не пуста, то возникает соответствующий вопрос.

```
[HLSERVER] -uf
```

Принудительно удаляет программу **HL-Server**. В этом случае не возникает никакого вопроса.

8 Вспомогательные программы

8.1 HLOGIN

HLOGIN.EXE используется для ручного подключения к программе **HL-Server** при тестировании. Для этого нужно только указать адрес модуля желаемого ключа и опционально идентификатор задачи Task-ID. В качестве идентификатора задачи допускается максимально восьмизначное шестнадцатеричное число. Ввод может быть как в десятичной (например, 12340), так и в шестнадцатеричной форме (например, \$6af3). В интерфейсе меню идентификатор задачи представлен в шестнадцатеричной форме.

HLOGIN 29809	Подключает к демо-версии ключа без указания идентификатора задачи (внутри используется идентификатор задачи "0")
HLOGIN 29809 1234	Подключается с Task-ID "1234"
HLOGIN 29809 FILL	Заполняет таблицы регистрации всех доступных в сети ключей с адресом 29809

Вы можете использовать эту программу, чтобы заполнить в тестовых целях таблицу регистрации программы **HL-Server** (или чтобы просто проверить функцию программы).

Для получения краткого описания работы программы введите HLOGIN без всяких параметров.

8.2 HLOGOUT

HLOGOUT.EXE позволяет вручную отключиться от программы HL-Server. Здесь снова представлен адрес ключа и опциональный идентификатор задачи.

HLOGOUT 29809	Отключается без Task-ID от ключа (внутренне используется Task-ID "0")
HLOGOUT 29809 \$1234	Отключается с Task-ID "\$1234" (шестнадцатеричный)

Если вы в качестве идентификатора задачи укажете звездочку (*), все записи станции, с которой вы запустили HLOGOUT, будут удалены с заданным адресом ключа независимо от идентификатора задачи.

HLOGOUT 29809* отключает все идентификаторы задачи этой станции от всех ключей.

Запуск программы HLOGOUT без заданных параметров дает краткую справку о работе программы.

Отключение чужих станций (с другими адресами адаптеров) при помощи программы HLOGOUT невозможно.

8.3 EDITEEP

Программа EDITEEP позволяет удобно изменить файлы конфигурации **Hardlock**-EEPROM. Обработанные EEP-файлы могут читаться напрямую из программы Crypto-Programmer-Software CP-EYE.

Пример:

EDITEEP APPLI004

На этом примере мы видим одновременное лицензирование четырех пользователей. Это происходит путем «зажигания» сигнатуры «Logins:nnnn» в EEPROM памяти ключа, причем «nnnn» указывает число лицензий в десятичной форме (четыре цифры отделяются нулями слева). Так вы можете изменить число лицензий простой заменой, «перезажиганием» ключа, без изменения программного обеспечения.

Использование EDITEEP просто:

- используйте стрелки для подвода курсора к полю ввода;
- используйте клавишу табулятора для перехода с ASCII на HEX;
- используйте клавишу ESC для прерывания редактирования без сохранения;
- используйте клавишу F1 для сохранения изменений в активных файлах и выхода из программы (содержание EEPROM ключа не может быть изменено напрямую программой EDITEEP).

8.4 ННТ

С помощью этой программы ищутся все локальные адреса на предмет наличия ключей и отображаются допустимые подключения лицензий.

Ключ **HL-Server** в виде платы расширения не распознается, если он подключен к кабелю программирования.

9 Толкование терминов

- **Client/Server.** Компьютерная система, в которой службы и ресурсы одной рабочей станции предоставлены многим другим станциям сети.
- **Client.** Термин Client используется для программ, которые используют программу **HL-Server**, для обращения к ключу и лицензированию с любого компьютера сети.
- **Server.** Термин **Server** или **HL-Server** обозначает компьютер, на котором работает программа **HL-Server** и чьи функции широко используются в сети.
- **Serven.** Это значит предоставлять сервис в сети. Программа **HL-Server** предоставляет сети функции ключа **Hardlock** - мы говорим о том, что программа **HL-Server** обслуживает модули.
- **Protokolltreiber.** Программа, которая регулирует общение с сетью и таким образом может состояться обмен пакетами данных (например, IPX, NetBios). Она напрямую связана с драйвером сетевой карты.
- **Netzwerkshell.** Программа, которая распознает запросы к сетевому драйверу/службам сети и передает их дальше на драйвер протоколов (например, NETX).
- **remote** (удаленный). Имеется в виду доступ к службе/устройству, которое доступно через сеть. Устройство при этом не является локальным для компьютера.
- **Login.** Идентификация пользователя в сети или на сервере перед использованием их функций.
- **Login Table.** Список, где зарегистрированы пользователи. Часто число одновременных подключений ограничено (лицензировано).
- **Logout.** Окончание работы в сети или сервере. Функции сети больше не нужны. Ресурсы могут быть освобождены.
- **TSR.** Программа (Terminate and Stay Resident) в DOS, которая устанавливается в оперативную память компьютера и снова передает управление операционной системе, в то время как сама продолжает работать в фоновом режиме.

- **NLM** (NetWare Loadable Module). Загруженный модуль для NetWare. Что для DOS является файлом .EXE, для операционной системы NetWare 386 фирмы Novell является .NLM.
- **Node**. Устройство в сети, например, сетевой адаптер компьютера, которое обозначается однозначно своим собственным адресом.

В этом руководстве мы исходим из того, что сервер и клиент не идентичны, а связаны друг с другом сетью. Через сеть они обмениваются информацией.

10 Сообщения об ошибках

Сообщение об ошибках	Причина/устранение	Применимо
Resident HL-Server is damaged Повреждена резидентная программа	Удаление программы HL-Server. Была переписана ее область памяти другой программой (возможно, другой программой TSR). Программа HL-Server не может быть удалена. Компьютер нужно как можно быстрее перезагрузить	TSR
Remove TSR loaded after HL-Server, then retry Удалите TSR после программы HL Server и повторите операцию	Были загружены другие резидентные программы после программы HL Server. Эта программа может быть только тогда удалена, если она является последней загруженной резидентной программой	TSR
More than x Modules specified Больше, чем x ключей было определено	Программа HL-Server может обслуживать только до 10 ключей (в резидентном варианте - 3), а было задано больше при помощи -m:nnnn	TSR/NLM
Invalid Option: xxxxx Неправильная опция	Заданная опция неизвестна. Введите HLSERVER -?, чтобы получить список доступных опций.	TSR/NLM
Keep failed Внутренняя ошибка	Резидентная программа не может быть запущена из-за несовместимости с DOS-версией	TSR
HL-Server is not installed Программа не установлена	Программа HL-Server не установлена. Указанные опции могут быть применены только тогда, когда программа HL-Server загружена резидентно или будет загружена с этим вызовом	TSR
Network Driver not found Сетевой драйвер не найден	Заданный сетевой драйвер не загружен, вообще никакой сетевой драйвер не загружен	TSR

Сообщение об ошибках	Причина/ устранение	Применимо
HL-Server is already installed Программа HL-Server уже установлена	Программа HL-Server уже установлена	TSR/NLM
No HL-Server Hardlock found Не найден ключ Hardlock	Не найден ключ Hardlock	TSR
No Hardlocks specified Ни один ключ не определен.	При загрузке программы HL-Server должен быть указан хотя бы один адрес модуля	TSR
Illegal timeout value Неверный параметр времени	Значение параметра времени должно быть равно нулю (= нулевому тайм-ауту) или быть от 10 до 9999.	TSR/NLM
TSR was not started, please use HLS32 under Windows 95. Please use HLS32SVC under Windows NT Резидент не был запущен, пожалуйста используйте программу HLS32 в Windows 95 или программу HLS32SVC в Windows NT	Программа HL-Server для DOS была запущена в Windows 95 или в Windows NT. Используйте вместо нее 32-битную программу HL-Server (HLS32 в Windows 95, соотв. HLS32SVC в Windows NT)	TSR

Коды возврата

Резидентная программа **HL-Server** и ее служебные программы дают следующий перечень кодов возврата Error Level Codes, которые могут быть вызваны соответствующей пакетной программой. (Пожалуйста, обратите внимание, что при тестировании коды возврата должны быть в убывающем порядке).

Код возврата	Описание ошибки
0	нет ошибок
1	ошибки из-за неправильных параметров
4	функция не поддерживается
5	обнаружен тайм-аут сети
6	нет регистрации
7	не обнаружен ключ
8	ошибка сети
9	доступ не возможен
10	ошибка распределения памяти
11	отсутствует клиент/серверная версия

11 Проблемы, советы и замечания

В этой главе вы найдете советы, где найти помощь, указания к определенным конфигурациям, а также информацию об использовании программы **HL-Server** с другими программами. Обратите также внимание на различные файлы для прочтения (README) в программном обеспечении ключа **Hardlock**, где содержится информация по темам.

11.1 Порты, сокеты и фильтры

Чтобы получить доступ в программу **HL-Server** через IPX или TCP/IP, клиенты посылают запрос на определенный порт или соответствующий сокет. Эти запросы только в том случае достигнут программы HL-Server, если эти порты или сокеты не заблокированы Router, Firewalls или чем-то похожим.

- Под IPX используются сокеты 6666h и 7777h.
- Под TCP/IP используются порты 3047/udp и 3047/tcp.

Убедитесь, что через эти порты/сокеты возможна коммуникация в сети, если вы хотите использовать соответствующие протоколы. Если вы используете функцию защиты в TCP/IP под Windows NT, обратите также внимание на свободное подключение портов 3047/udp и 3047/tcp.

11.2 Помощь при зависании компьютера

При зависании клиентов может случиться, что заблокируется таблица регистрации с неиспользованными строками. Программное обеспечение предлагает три возможности удалить строки из таблицы.

1. В соответствии с заданным временем для функции тайм-аут каждая станция, которая за это время не провела ни одного обращения к сетевому ключу, будет автоматически отключена (см. также описание параметра **-timeout.**)
2. С помощью тестирующей программы HLOGOUT могут быть снова удалены строки одной станции в таблицы регистрации. Ее вызов может быть включен в файле

AUTOEXEC.BAT, чтобы при перезагрузке компьютера занятые этой станцией строки автоматически были удалены. (Укажите для этого в качестве идентификатора задачи звездочку (*). В случае, если станция не подключена, ничего не произойдет).

3. В интерфейсе меню строки регистрации могут быть удалены по отдельности с помощью функций **local** – **logins** соответственно **remote** – **logins**.

Возможность 3 может использоваться для защиты паролем от неавторизованных обращений.

11.3 HL-Server TSR и другие резидентные программы

Если вы используете резидентные программы, которые размещают часть своего кода или своих данных на жестком диске или в EMS/XMS, загружайте эти программы после загрузки программы **HL-Server**. Эти программы помещают часть оперативной памяти на жесткий диск и загружают свой собственный код. Если бы программы **HL-Server** лежала в выгружаемом на диск разделе памяти программы, она была бы переписана при выгрузке, и были бы нарушены сетевые соединения.

Примеры таких программ - QTSR и SideKick PLUS.

11.4 Мультизадачность, Windows & OS/2 и DOS TSR

Мультизадачные программы, такие как Carusell, DESQview, NetWare Access Server, Windows и т.д. должны очень осторожно применяться на компьютере, где установлен **HL-Server**, т.к. в зависимости от их конфигурации существенно меняется функция программы **HL-Server**. В любом случае программа **HL-Server** должна быть загружена до запуска мультизадачной программы. Особенно тщательно перепроверьте вашу установку, прежде чем вы используете компьютер в сети.

11.4.1 Windows 3.x и DOS TSR

Под Windows 3.x программа **HL-Server** работает без проблем. Но и здесь проследите, чтобы программа **HL-Server** была загружена до запуска Windows. Строка статуса программы

HL-Server не должна быть активной при запуске Windows. интерфейс меню может быть вызван из панели DOS, но при этом, пожалуйста, не проводите операции install или uninstall. Установка программы **HL-Server** на DOS-панели в Windows возможна без проблем, после закрытия панели программа **HL-Server** больше недоступна для приложений.

11.4.2 Windows для рабочих групп (Workgroups) и DOS TSR

При применении WfW и программы **HL-Server** могут возникнуть проблемы, которые повлияют как на функции самой программы **HL-Server**, так и используемых приложений.

Сетевые функции программы WfW работают нестабильно и в простых операциях типа Peer-to-Peer, но особенно в комбинации с драйверами NetWare-ODI. Далее описание возможностей.

- WfW без функций Peer-to-Peer-Funktionen (выбрана опция «Windows –поддержку установить только для следующей сети»): программа WFW работает в этом режиме как Windows 3.1. **HL-Server** TSR и доступные приложения могут применяться неограниченно.
- WfW в качестве сети Peer-to-Peer (при настройке сети установлены опции «Microsoft Windows Netzwerk» и «Kein zusatzliches Netzwerk» (никакой дополнительной сети). Здесь в Windows обычно устанавливается протокол NetBIOS, который может использоваться программой **HL-Server**. HLSERVER.EXE вызывается в группе автостарта. Так как идет речь о резидентной программе TSR, проследите за тем , что панель DOS не закрывается после окончания программы.
- Комбинация WfW-Netz и Novell NetWare. Здесь установлены протоколы NetBEUI и Novell IPX на драйвере ODI сетевого адаптера. Во многих случаях здесь наступает следующий эффект: определенные информационные пакеты маршрутизатора, которые могут быть переключены системой доступа программы **HL-Server**, передаются с ошибками. Возможный диапазон проблем при этом - от потери доступа к программе **HL-Server** до потери доступа к серверу сети (может произойти только случайно). Для обхода этой проблемы потребуется очень много времени и усилия разных сторон.

Мы можем порекомендовать применение только первого варианта конфигурации программы **HL-Server**, вторая версия должна представлять промежуточное решение в экстремальных случаях.

Так как программа WfW пока далее не развивается, проблема будет существовать до выхода новой версии (сетевые программы новой версии должны быть полностью переделаны, тогда не возникнут проблемы).

11.4.3 Windows 9x, Windows NT

Установка резидентной программы TSR **HL-Server** в панели DOS в Windows NT или Windows 95 в принципе возможна. Тем не менее эта установка больше не поддерживается из-за различных недостатков применения DOS в Windows (закрытие панели равнозначно окончанию работы программы **HL-Server**). Здесь следует использовать 32-битную программу **HL-Server** или **HL-Server Service**.

Примечание. Если у вас появятся проблемы при установке программы **HL-Server** в Windows 95, вы можете заставить систему создать файл протокола при помощи команды `HLS32 - m:xxxxx`. Файл протокола `hls32.log` находится в текущей директории сервера. Из него вы можете извлечь необходимую вам информацию об успешных шагах при инициализации протокола.

11.4.4 OS/2 DOS-Box, LAN-Server и DOS TSR

Установка **HL-Server** на панели DOS в OS/2 возможна, но после закрытия окна приложения не могут использовать программу HL-Server.

В некоторых версиях OS/2 Lan-Server службы имен протокола NetBios выключены для панели Dos по умолчанию. Без этих служб имен не может быть найдена ни одна программа **HL-Server** (это касается и других программных обеспечений, например, Stomper oder Exac).

Для активации окна сервиса имен укажите сразу после запуска панели DOS (во всяком случае до первого обращения в сеть) следующую команду:

```
LTSVCFG s=10 c=10 n=10 n1=1
```

Программа LTSVCFG.COM является частью программы LAN-Servers и автоматически устанавливается вместе с ней.

Значение параметров:

s	число сессий NetBios
c	число комад NetBios
n	число имен NetBios
n1	0 служба имен выключена
n 2	1 служба имен включена

Здесь существенно включение сервиса имен с параметром **n1=1**. Другие параметры s,c,n могут быть подстроены специальной конфигурацией. Если программы **HL-Server** или **HL-Server-Client** являются единственными, которые используют NetBios в панели Dos, указанных выше параметров хватает.

11.5 Несколько систем HL-Server в сети

Вы можете использовать до 250 систем **HL-Server**. Ключи с одинаковыми адресами модулей, но с разными кодировками различаются только тогда, когда программа-клиент использует идентификацию через функцию HL_LOGIN(...) (через параметр REFKEY и VERKEY). См. *Описание ручного подключения программы HL-Server*. **HL-Crypt** проводит этот вид идентификации всегда автоматически.

Само собой разумеется, что вы можете использовать сколько угодно локальных ключей дополнительно к системе HL-Server.

Обычно ключи с одинаковым адресом модуля, но с разным программированием базовых субкодов не могут комбинироваться на одном компьютере с программой **HL-Server**. Вы должны использовать различные адреса для ключей с различным шифром (кодировкой), так как иначе вы потеряете возможность комбинирования в локальной сети.

11.5.1 Дополнительная лицензия

Если вы примените защищенное приложение для использования локального и удаленного ключа, оно будет локально обращаться к ключу при использовании компьютера, то есть не будет происходить регистрации ключа в таблице регистрации. Таким образом приложение можно запустить на один раз больше, чем мест в таблице регистрации.

Вы можете этого избежать, если вы зададите разные адреса модулей для локального и удаленного ключей (это значит, что приложение использует или локальный ключ 12345, или удаленный ключ 12346). В ином случае учитывайте это при лицензировании.

11.5.2 Эмуляция Novell NetBios

Эмуляция Novell NetBios имеет некоторые особенности, которые могут повлиять на работу с программой **HL-Server**.

- Указание имени группы при установке программы **HL-Server** может длиться долго, особенно в сети со множеством сегментов.
- При первом обращении к NetBios после загрузки NETBIOS.EXE, а также позже, эмуляция проводит через нерегулярные промежутки времени процедуры инициализации, которые занимают некоторое время. Это может привести к тому, что функции `remote - select` и `remote - rescan` длятся существенно дольше, чем в настоящих сетях NetBios или при IPX.
- Если на один компьютер сначала устанавливается программа **HL-Server** с протоколом IPX, потом загружается NetBios, наконец запускается графический интерфейс программы **HL-Server** и проводится **remote - rescan** через IPX и NetBios, то, скорее всего, NetBios будет в зависимости от величины сети работать неверно.

Эти проблемы возникает с эмуляцией Novell NetBios-Emulation, другие сети NetBios не показывает такой нерегулярности в работе.

11.5.3 Чрезмерная загрузка сети

При чрезмерной загрузке сети (например, если содержимое файлового сервера копируется на рабочую станцию или нагрузочный тест выполняется на очень большом числе рабочих станций) может случиться, что какой-либо клиент **HL-Server** будет не в состоянии найти «свою» программу HL-Server. Сама программа **HL-Server** и подпрограммы клиента имеют сложные механизмы таймаута и повтора, но всегда может произойти ситуация, когда из-за нагрузки сети не сможет состояться коммуникация. (Эта проблема существует и для Novell NetWare: при перегрузке сети может быть не найден сервер.) В этом случае подпрограммы клиента **HL-Server** действуют так, как будто запрашиваемого модуля просто нет.

Этот эффект возникнет на медленном компьютере с программой **HL-Server** скорее, чем на быстром, так как более быстрый компьютер может ответить на большее количество запросов до наступления таймаута, чем медленный.

Улучшить положение можно только путем разгрузки всей сети. Меры в каждом конкретном случае специфичны и зависят от проблемы, поэтому не могут быть освещены в рамках этой книги.

11.5.4 Время в DOS

Болезненная тема времени в компьютерах затронула и **HL-Server**. Наверняка вы один раз уже выяснили, что после долгого копирования через сеть время программного обеспечения вашего компьютера (не время технического обеспечения) отстает. Это происходит оттого, что при сильной нагрузке сетевого адаптера иногда пропускается прерывание от таймера. Этот таймер отвечает именно за то, что программа управления часами компьютера работает дальше. (Аппаратные часы нормально читаются только при перегрузке системы).

У подпрограмм клиента **HL-Server** эти проблемы не возникают. Но, во всяком случае, компьютер, на котором резидентно установлена программа **HL-Server**, может этим страдать. Если вы часами проводите удаленные тесты на «стресс», время на счетчике времени вашего компьютера может несколько отставать от реального времени.

Эту проблему можно обойти так: регулярно читать часы аппаратные и заново ставить программные. Но это отрицательно повлияет на программу времени **HL-Server**.

Из-за того, что эффект отстающих часов при нормальной работе невозможно определить, от постоянной сверки времени программных часов отказываются.

Если это приводит в специальных конфигурациях при длительной работе программы **HL-Server** к большим различиям во времени, вы можете провести сверку программных часов по аппаратным в полночь параметром **-mtc** (Midnight Time Correction) при установке резидентной программы.

```
hlserver -i -m:29809 -mtc
```


Документация по Hardlock

Документация по **Hardlock** познакомит вас со всеми компонентами продуктовой линейки **Hardlock** и с возможностями защиты вашего программного обеспечения (ПО).

Документация включает в себя следующие разделы.

- **Техническое руководство по Hardlock** описывает принципы защиты ПО с помощью всех компонентов продуктовой линейки. Оно поможет вам выбрать правильную защиту для выполнения ваших задач.
- **Руководство по HL-Bistro** (визуальной среде программирования для Hardlock) описывает этот программный пакет, который позволяет вам проектировать систему защиты в привычной среде Windows.
- **Руководство по HL-Server** описывает применение Hardlock в сетевых приложениях.
- **Руководство по HL-Crypt** описывает применение автоматической защиты вашего ПО.
- **Руководство по HL-API** содержит обзор интерфейса прикладных базовых систем (спецификацию API) для ручной защиты ваших программ.
- **Раздел Понятия** поясняет основные термины и понятия, используемые в документации по Hardlock.

Все разделы дополняются интерактивной помощью (Help) к различным программам. Самые последние изменения и дополнения вы найдете в файлах Readme к программам. В руководствах и файлах интерактивной помощи (Help) применяются следующие типы шрифтовых выделений:

Обозначение	Функция	Пример
жирный	Понятия верхнего уровня Продукты	DatePCTV Hardlock Hardlock Twin
ЗАГЛАВНЫЕ БУКВЫ	Файлы и пути	HLDRV.EXE
Courier	Синтаксис	hlpatchup -m 29809

Содержание

1 Ручная установка защиты	1
1.1 Принцип защиты	1
1.2 Необходимые условия	1
1.3 Поставка защищенной программы	2
1.4 Библиотеки функций	2
1.5 Защита различных типов программ	4
1.5.1 DOS	4
1.5.2 Windows 3.x	4
1.5.3 Windows NT/2000	4
1.5.4 Windows 9x	5
1.5.5 DOS-экстендеры	5
1.5.6 UNIX	6
1.5.7 OS/2	6
2 Работа с API	9
2.1 Общие положения	9
2.2 Меры безопасности	10
2.2.1 Нападение и защита	10
2.2.2 Применение функций верхнего уровня	11
2.2.3 Шифрование	
с помощью Hardlock-API	12
2.2.4 Инициализация ключа	
к запуску программы	13
2.2.5 Меры по защите во время	
выполнения программы	13
2.2.6 Установки для завершения программы	16
2.2.7 Использование ключа Hardlock	
с опцией памяти	16
2.2.8 Меры после распознавания атаки	17
2.3 LiMaS-функции программы Hardlock API	18
2.4 Доступ к HL-Server	19
2.4.1 Изменение типа доступа	19
2.4.2 Поведение приложения при тайм-ауте	20
3 Функции Hardlock-API (верхнего уровня)	21
HL_ABORT()	24
HL_ACCINF()	25
HL_AVAIL()	26
HL_CODE(DATAPTR,BCNT)	27
HL_HLSVERS()	29
HL_LOGIN(MOD,ACCESS,REFKEY,VERKEY)	30
HL_LOGOUT()	32
HL_MAXUSER()	33
HL_MEMINF()	34

HL_PORTINF()	35
HL_READ(REG,VALUE)	36
HL_READBL(DATAPTR)	37
HL_READID(IDLow,IDHigh)	38
HL_SELECT(DATA_AREA)	39
HL_USERINF()	40
HL_VERSION()	41
HL_WRITE(REG,VALUE)	42
HL_WRITEBL(DATAPTR)	43
HLM_CHECKCOUNTER(INCVAL,	44
MAXCOUNTER, CURRENTCOUNTER)	44
HLM_CHECKEXPCODE(SLOT,YEAR,	45
MONTH,DAY)	45
HLM_CHECKSLOT(SLOT,MAXUSER,	46
CURRENTUSER)	46
HLM_FREESLOT(SLOT)	47
HLM_GETRUSINFO(BUFLEN,	48
RTBBUFFER,BASE64)	48
HLM_ISRUSHL(ID)	50
HLM_LOGIN(MOD,ACCESS,REFKEY,	51
VERKEY,VKEY, OPTIONS, SEARCHSTR)	51
HLM_OCCUPYSLOT(SLOT)	53
HLM_WRITELICENSE(BUFLEN,	54
RTBBUFFER,ACCESS,SEARCHSTR,	54
OPTIONS)	54
4 Приложение	55
4.1 Таблица значений статуса	55
4.2 Поиск файла лицензии	
HL-RUS (ALF)	58
4.3 Определение последовательности поиска API	58
4.3.1 Подготовка	58
4.3.2 Синтаксис	58
4.3.3 HL-Server Client для TCP/IP	60
4.3.4 Стратегия поиска	62
4.3.5 Замечания	63
4.4 Совместимые соглашения о вызовах	64
4.5 Модули Hardlock в режиме совместимости	65

1 Ручная установка защиты

1.1 Принцип защиты

При ручной защите программ защитный модуль встраивается в саму программу. Целью является создание единого целого из программы и защитного модуля. Защищенная программа не работоспособна без подходящего ключа.

С этой целью в текст программы вводят различные вызовы функций, которые опрашивают и управляют ключом **Hardlock**. Именно эти функции и находятся в библиотеке функций **Hardlock API**.

Затраты времени при ручной установке выше, чем при автоматической, но зато защита настраивается индивидуально.

Для еще более высокой безопасности вы можете комбинировать оба вида установки защиты. Если вы хотите избежать больших затрат времени, то можно воспользоваться автоматической защитой с помощью программы **Espresso** из пакета **Hardlock Bistro**.

Если вы все же захотите произвести защиту ручным способом, то необходимые инструменты для тестирования ее функциональности вы найдете в **Latteccino** из пакета **Hardlock Bistro**.

Информация по основам ручной установки защиты, процедуре и о различных защитных механизмах содержится в *главе 1*.

1.2 Необходимые условия

Необходимые условия для ручной установки следующие:

- установка **Hardlock API**;
- установка драйвера ключа **Hardlock** (см. раздел *Техническое описание*);
- доступ к исходному коду приложения, которое вы хотите защитить.

Информацию по основам ручной установки защиты, процедуре и различным защитным механизмам вы найдете в *главе 1*.

1.3 Поставка защищенной программы

Поставляйте свою защищенную программу вместе с закодированным ключом и соответствующим драйвером для ключа **Hardlock**.

Вы можете поставлять отдельную программу для установки драйвера или интегрировать установку драйвера в вашу собственную установочную программу (см. главу 1.4).

1.4 Библиотеки функций

Hardlock API - это интерфейс между приложением, требующим защиты, и защитным модулем. Эта программа содержит функции управления и запроса ключа **Hardlock**. Она содержит два уровня:

- **API низкого уровня (Low-Level API)** – это подпрограммы драйвера, зависящие от технического обеспечения и системы эксплуатации, управляющие защитным модулем. Подпрограммы низкого уровня предоставляются в виде библиотек или объектных файлов;
- **API высокого уровня (High-Level API)** – это сегмент для простого и удобного обслуживания API. Они используются для компиляторов самых распространенных языков высокого уровня. В данном описании указанные функции API высокого уровня обслуживают сегменты программы API низкого уровня. Встраивание API верхнего уровня происходит на уровне исходного кода программы.

Существенное преимущество функций высокого уровня состоит в том, что при их вызове будут определены и переданы только те параметры, которые необходимы для приложения. Все остальные необходимые для корректного управления модулем безопасности внутренние параметры управляются API высокого уровня автоматически и передаются вместе с другими, нужными для приложения, как структура на API низкого уровня.

Для большинства прикладных программ достаточно использовать функции высокого уровня. Если вы хотите самостоятельно использовать сегменты программы низкого уровня, мы вышлем вам документацию по подпрограммам низкого уровня.

Информацию по основам ручной установки, процедуре и различным защитным механизмам вы получите в главе 1. Параметры описываются в *главе 3*.

Примечание. Ручная реализация защиты ПО при помощи **Hardlock** API используется во всех продуктах **Hardlock** фирмы «Аладдин» (ручная установка защиты, программа HL-Server, HL-Crypt). Чтобы осуществить слаженную работу всех продуктов, нужно проследить за тем, чтобы совместно использовались только программы с одинаковым номером основной версии API. Так, программа HL-Server с версией API 2.xx не может применяться совместно с приложением, у которого ручная установка защиты проходила с применением API с номером версии 3.xx. А совместное использование программных модулей версий 3.xx, напротив, вполне возможно.

Установка драйвера API

Если вы не хотите использовать вспомогательные программы HLDV32.EXE или HLDINST.EXE для установки драйвера, то вы можете встроить в вашу установочную программу API установки драйвера для статической и динамической компиляции. Необходимые файлы вы найдете на **Hardlock**-CD в каталоге **\HARDLOCK\DRIVER\API**. Более подробное описание вы найдете в файле HINSTALL.HLP, который находится в том же каталоге.

API установки Hardlock-Server

Чтобы установить **Hardlock**-Server из своей собственной программы в Windows 9x или Windows NT, нужно использовать файлы для статической и динамической компиляции HSI-API, находящиеся на **Hardlock**-CD в каталоге **\HARDLOCK\HLSERVER\NT_95\INSTALL**. Подробное описание функций вы найдете в файле HSI-API.PDF этого же каталога.

Программирование Hardlock API

Если вы захотите включить кодирование ключа в свое собственное программное обеспечение, например, использовать базу данных для управления ключами, то такую возможность предоставляет HLPROG.DLL. Вы найдете ее на **Hardlock**-CD в каталоге **\HARDLOCK\CP\WIN32**. Полная документация по функциям представлена в файле HLPROG.PDF в каталоге **\HARDLOCK\CP\WIN32\SAMPLES**.

1.5 Защита различных типов программ

1.5.1 DOS

В среде DOS интерфейс API включается в приложение через объектные (.OBJ) или библиотечные (.LIB) файлы. Интерфейс API становится частью самого приложения и может обращаться к аппаратным средствам компьютера без использования дополнительных драйверов.

1.5.2 Windows 3.x

Для поддержки обращений к ключу **Hardlock** в Windows 3.x необходима установка драйвера. Установка этого драйвера может проходить с помощью устанавливающей программы (см. раздел *Техническое описание*). В качестве альтернативы вы можете включить установку драйвера с API в устанавливающие подпрограммы вашей программы (см. главу 1.4).

При программировании в интерфейсе Windows 3.x API присоединяется к приложению в виде объектных и библиотечных файлов. В основном различают два способа компиляции программ:

- **статическая компиляция:** требуемые объектные и библиотечные файлы напрямую включаются в программу. Это увеличивает размер EXE-файла;
- **динамическая компиляция:** библиотеки становятся доступными в виде DLL-файлов. Требуемые функции динамически включаются в приложение только при его запуске.

API может использовать общий файл .DLL. Файлы .DLL поддерживают работу в мультизадачной среде. Они могут одновременно запустить несколько заданий печати на принтере в Windows и несколько задач по **Hardlock** так, что они не будут влиять друг на друга.

1.5.3 Windows NT/2000

Установка необходимого драйвера может проходить с помощью установочной программы (см. раздел *Техническое описание*). В качестве альтернативы вы можете включить установку драйвера в подпрограммы по установке вашей программы (см. главу 1.4).

Чтобы **Hardlock** работал в Windows NT, необходим драйвер **HARDLOCK.SYS**. Для поддержки интерфейса USB (возможно только в Win2000) предоставляются файлы AKSUSB.SYS и Plug'n'Play конфигурации AKSUSB.INF. Применение **Hardlock** PCMCIA требует файла Plug'n'Play конфигурации HLPCCMIA.INF.

Начиная с версии 3.20 интерфейс **Hardlock-API** полностью поддерживает программы DOS и WIN16 в среде Windows NT. Программы, скомпилированные с версией API ниже 3.20, также могут быть использованы, но они исполняются с меньшей скоростью. Это возможно благодаря применению драйвера виртуального устройства Virtual Device Driver HLVD.DLL. Этот файл может быть использован вашим приложением как 32-битный .DLL.

1.5.4 Windows 9x

Установка необходимого драйвера может проходить с помощью установочной программы (см. раздел *Техническое описание*). В качестве альтернативы вы можете включить установку драйвера в подпрограммы по установке вашей программы (см. главу 1.4).

В Windows 9x ключу **Hardlock** требуется драйвер виртуального устройства **HARDLOCK.VxD**. Этот драйвер позволяет системе обращаться к ключу без участия Windows 9x. Для поддержки интерфейса USB имеются файлы AKSUSB95.SYS (Windows 95) или AKSUSB.SYS (Windows 98), а также Plug'n'Play файл конфигурации AKSUSB.INF. Применение **Hardlock** PCMCIA требует файла Plug'n'Play конфигурации HLPCCMIA.INF.

1.5.5 DOS-экстендеры

Интерфейс **Hardlock** API поддерживает несколько DOS-экстендеров (как 16-битных, так и 32-битных).

При конфигурации вашего приложения или экстендера убедитесь, что 9Кб оперативной памяти доступны API для выполнения сетевых программ, так как в этой области создаются буферы для доступа к сети. Большинство DOS-экстендеров настроено на 64 Кб оперативной памяти. Дополнительную информацию смотрите в документации по вашему DOS-экстендеру.

16-битный DOS-экстендер

Текущая версия API поддерживает все 16-битные DOS-экстендеры, совместимые с DPML 0.9 и выше (например, Windows 3.x DOS Box, языки Borland).

32-битные DOS-экстендеры

Текущая версия API поддерживает все совместимые с DPMI 32-битные DOS-экстендеры (например, Rational DOS/4GW, PharLap 386|DOS).

AutoCAD12 применяет DOS-экстендер PharLap 386| и поэтому тоже совместима с **Hardlock** API. Для получения дополнительной информации по применению смотрите соответствующий пример программы. По умолчанию AutoCAD не резервирует область в оперативной памяти, так что вы должны сами настроить установку для использования сетевых подпрограмм API.

```
cfg386 acad.exe -minr 8192 -maxr 8192
```

1.5.6 UNIX

В среде UNIX доступ ключа **Hardlock** производится через модуль ядра, который может динамически загружаться и выгружаться.

Применение API для версии SCO PC-UNIX требует компиляции ядра UNIX (установки драйвера API).

Применение API (включая драйвер) для SCO PC-Unix, LINUX x86 и SOLARIS x86 находится на CD. Для получения дополнительной информации свяжитесь с нашей группой поддержки.

1.5.7 OS/2

Здесь применима информация для Windows-DLL. Но нужно учитывать следующее.

Код DLL имеет **статус IOPL**, тем самым вы должны указать параметр **IOPL=YES** в файле CONFIG.SYS. Эта установка является значением по умолчанию для OS/2 версии выше 2.x. Система должна иметь доступ к OS/2 **Hardlock** DLL через переменную среды LIBPATH. Это можно сделать по умолчанию, если LIBPATH получит значение «.» (обозначение текущей директории), а .DLL лежит в той же директории, что и OS/2.

Для OS/2 (начиная с версии 1.3) доступны три файла .DLL.

- HLOS2LOW.DLL всегда требуется, так как только через него возможен доступ к аппаратному средству, ключу **Hardlock**. Он содержит часть API низкого уровня. Файл DLL записан для OS2 в 16-битном режиме (IOPL!). Биб-

лиотечные файлы .OBJ и .LIB не могут применяться в OS2, так как прямой (без .DLL) доступ программ OS2 к порту принтера не всегда можно гарантировать.

- HLOS2_32.DLL предоставляет 32-битной программе интерфейс верхнего уровня High-Level в форме .DLL и использует низкоуровневую часть Low-Level в файле HLOS2LOW.DLL.
- HLOS2_16.DLL предоставляет 16-битной программе интерфейс верхнего уровня High-Level в форме .DLL и использует низкоуровневую часть Low-Level в файле HLOS2LOW.DLL.

Как и при работе в Windows, API верхнего уровня может быть скомпилирована статически в приложение.

Файлы .DLL мультизадачны. Вы можете одновременно запустить без всяких проблем несколько задач печати на принтере в OS2 и несколько задач **Hardlock** без их взаимного влияния друг на друга. Но на основе приоритета доступ к ключу будет проведен раньше, так что выполнение других задач будет несколько замедленно.

Если несколько приложений в (DOS/Win) используют **Hardlock** и/или принтер, статус порта должен быть изменен на **Shared access**. При этом поступайте следующим образом:

1. откройте контекстное меню иконки принтера правой кнопкой мыши;
2. выберите **Open/Settings**;
3. выберите **Output**;
4. выберите используемый интерфейс (например, **LPT1**);
5. выберите **Share access**.

Эти установки требуются только для защищенных программ DOS/Windows. Они не требуются для программы OS/2, которые обращаются к ключу через OS/22-DLL.

2 Работа с API

2.1 Общие положения

Для надежной ручной защиты нет готовых рецептов. Но мы дадим вам советы, исходя из нашего опыта. Мы предоставляем вам все вспомогательные средства и при необходимости с удовольствием ответим на ваши вопросы. Границы осуществления ручной защиты зависят только от вашей фантазии и готовности потратить время на повышение качества защиты. Информацию о возможной защите вы получите в *главе 2.2*.

При установке защиты вы можете ввести функцию HL-RUS (LiMaS), которая сделает возможным дифференцированное лицензирование и модификацию. Вы можете установить строки регистрации для различных частей программы, установить глобальный счетчик и данные по выполнению команды. Дальнейшую информацию вы получите в *главе 2.3*.

Информацию о доступе к модулю ключа в сети вы получите в *главе 2.4*.

Общие замечания по безопасности

- Не скрывайте запросы высокого уровня путем неаккуратного программирования. Многие компиляторы оптимизируют код так, что даже в этом случае генерируется «чистый» ассемблерный код.
- Не используйте в вашей программе никаких опций для отключения запросов ко внутренним проверкам. Этим самым система защиты может быть отключена простым изменением программы.
- Удалите из поставочной версии вашего программного обеспечения отладочную информацию.

Установка компиляторов

Необходимые для программы установки для компиляторов, объектов и библиотек вы найдете в пакетных (Batch) файлах и проектах. Они ориентированы чаще всего на минимальную конфигурацию.

В зависимости от вида компилятора не все описанные методы могут быть напрямую введены в ваш язык программирования. Но многие компиляторы предоставляют возможность получить

прямой доступ к вашему программному коду с помощью указаний на встроенные ассемблерные или объектные модули программы.

Обратите внимание на программы примеров для разных компиляторов, которые находятся на **Hardlock-CD**.

Программы примеров

Сначала попробуйте оттранслировать программу примера вашим компилятором, прежде чем вы начнете интегрировать ключ **Hardlock** в вашу программу.

Имейте в виду, что приведенные образцы лишь объясняют применение различных вызовов функций и их работу. Поэтому они такие простые и краткие и не являются хорошим примером ручной защиты. Поэтому недостаточно перенести уже имеющиеся примеры в вашу программу. Подумайте о том, что хакер тоже может знать о наших примерах программ.

2.2 Меры безопасности

2.2.1 Нападение и защита

Любая защита программы может быть распознана и проанализирована, и тогда ее можно будет обойти. Те, кто предлагает абсолютные системы защиты, не должны восприниматься всерьез. Но механизмы защиты **Hardlock** позволяют вам увеличить время анализа защищенной программы.

Если вы хотите действительно защитить вашу программу, то должны представлять, как потенциальный хакер пытается взломать программу. Ваша цель при установке ручной защиты должна заключаться в том, чтобы хакер потратил как можно больше времени и усилий на взлом.

Инструмент хакера – это отладочная программа. Исходите из того, что у хакера имеются самые лучшие средства. К ним относятся в первую очередь отладчики, которые используют процессор в защищенном режиме или специальное оборудование для выполнения процесса отладки. Такие средства могут проанализировать программу шаг за шагом.

Хакер ищет машинный код, генерируемый вашим компилятором. Он попытается найти все запросы к защитному модулю. После того, как хакер нашел механизм защиты, он должен так

изменить программу, чтобы она работала нормально со снятой защитой. Отсюда делается вывод, что не одно «хитрое» решение, а масса небольших защитных мер может существенно затруднить работу хакера.

С помощью **Hardlock** вы можете легко защитить программу так, что фактически невозможно будет обойти систему защиты, не имея нужного ключа Hardlock, так как параметры сегментов кода и данных, необходимые для запуска программы, не могут быть использованы, пока они не будут расшифрованы при помощи ключа Hardlock. Для запуска программы без ключа защиты хакер должен заменить все зашифрованные данные на расшифрованные. Поскольку хакер не сможет отгадать требуемую информацию, он должен получить доступ к ключу Hardlock.

Даже если хакер имеет подходящий защитный ключ, а исходя из этого, время на протоколирование всех правильных результатов и включение их в программу очень велико, если будет применяться метод распределения данных и параметров шифрования по всему телу программы. Эта защитная мера может быть усилена с помощью «неправильных маневров» и тестов на корректность, на которые придется затратить еще больше времени.

2.2.2 Применение функций верхнего уровня

Вы можете индивидуально создавать защиту вашего программного обеспечения с помощью языков программирования высокого уровня. Далее следует список мер, которые вы можете или должны применить.

- Установить ключ **Hardlock** и установить время.
- Перепроверить наличие ключа перед расшифровкой большого количества данных.
- Перепроверить наличие ключа во время работы программы.
- Зашифровать части программы, внешние и внутренние данные.
- Срочная зашифровка после расшифровки и использования программы (метод временного окна).
- Физическое разделение и специализация мер.
- Использование памяти ключа.

- Принятие специальных мер при распознавании атаки (скрытие данных, замедление выполнения).
- Тестирование состояния (контрольных сумм, перепроверка контрольного механизма и времени выполнения).

Дальнейшую информацию по отдельным мерам вы получите в следующих главах. Информацию по отдельным вызовам языков высокого уровня вы получите в *главе 3*.

2.2.3 Шифрование с помощью Hardlock-API

Ключ работает как алгоритмическая защита. Именно таким образом в аппаратные средства включается функция. Поведение этой функции зависит от того, как был закодирован ключ. Модули с одинаковыми параметрами, но с разным кодированием дают разные результаты.

Вместе с используемой функцией **Hardlock** работает как генератор ключа. Отпадает необходимость его тайного хранения, сохранения в памяти или переноса, так как он скрыт в аппаратных средствах. С его помощью могут быть зашифрованы и снова расшифрованы во время работы приложения, константы, данные и участки кода программы.

Функция HL_CODE содержит алгоритм преобразования данных K-EYE, который вместе с ключом **Hardlock** шифрует данные 64-битными (8-байтными) блоками (блочный шифр). Шифр работает в прямом и обратном направлениях. Это значит, что при повторном использовании одного ключа из зашифрованного текста получают нормальный. Функция HL_CODE должна использоваться с указателем на определенную область данных и число зашифрованных 8-байтных блоков. Можно зашифровать до $64\text{Kb}=81192$ блоков за один вызов функции HL_CODE.

Шифрование нескольких блоков данных (по 8 байт) за один вызов функции HL_CODE (см. пример 1) приводит к результату, отличающемуся от результата при шифровании за несколько вызовов функции (например, вызова функции на каждый блок или на группу из нескольких блоков, см. пример 2).

Пример 1	Пример 2
Byte Block [16]; HL_CODE(Block,2);	Byte Block [16]; HL_CODE(Block,1); HL_CODE(Block+8,1);

В обоих случаях шифруются два блока по 8 байт. Но результаты шифрования различны.

2.2.4 Инициализация ключа к запуску программы

Включите в начале программы простой запрос к ключу для его инициализации. Тем самым проверяется, что:

- имеется ключ с правильным адресом;
- найденный ключ с этим адресом модуля правильно закодирован.

Обе задачи по проверке наличия ключа и его идентификации выполняются при помощи функций `HL_LOGIN` или `HLM_LOGIN`.

Эти первые шаги служат для инициализации доступа к ключу и правильному его распознаванию. Но они не защищают программное обеспечение. Однако система получает информацию, что правильный ключ с ожидаемым кодом найден и является доступным.

Для хакера очень легко обойти такую систему защиты. Она также не обеспечивает никакой защиты от использования приложения на разных рабочих станциях. В любом случае дополните защиту при старте программы другими мерами.

2.2.5 Меры по защите во время выполнения программы

Зашифруйте для работы программы необходимые ей параметры, данные или части кода и расшифруйте их только во время выполнения программы (будьте внимательны при использовании Protected Mode). Возможность зашифровать нужную информацию позволяет представить всю зашифрованную информацию в виде нормального текста только на короткое время, пока она действительно нужна (метод временного окна). Используйте расшифрованные данные, чтобы установить ловушки или передать параметры функции.

Перенос расшифрованных значений как локальных переменных в стек дополнительно усложняет применение отладчиков.

Используйте возможность шифрования данных для внешних файлов, которые нужны вашей программе и которые только через определенное время будут расшифрованы как, например, файлы конфигурации.

Перед расшифровыванием большого объема данных система должна проверить наличие ключа с вашим кодированием. Это предохранит любые важные данные от их уничтожения. Проверка доступности ключа проходит с помощью функции `HL_AVAIL()`.

Примечание. Не сохраняйте копию защищенных данных в вашей программе, чтобы нельзя было сравнить расшифрованные данные с оригиналом.

Метод временного окна

У вас имеется возможность расшифровать нужную информацию только на необходимое для ее использования время. Она хранится в программе в зашифрованном виде и расшифровывается только на короткое время (когда она действительно нужна), и то только частично – никогда одновременно вся она не появляется в виде нормального текста.

Выбор стратегии защитных мер

При выборе стратегии защиты вы должны стремиться к созданию разветвленной структуры защитных мер. Запросы ключа должны находиться на самом глубоком уровне структуры меню.

Физическое разделение

Участки программы с запросами к ключу не должны программироваться связано, поскольку это почти всегда приводит к приблизительно одинаковому коду. Тем самым хакеру легче найти запросы ключа. Поэтому физически разделите эти последовательности в вашей исходной программе (насколько это возможно).

Необходимость изучения программы

Обычно хакер не знает и сам не использует программу. Представьте, сколько времени должен потратить начинающий хакер, чтобы изучить, например, сложную программу бухгалтерского учета вплоть до последнего диалога. Для него всегда трудной задачей будет сделать правильный вывод и убедиться в правильности работы программы. Защитная программа всегда будет психологическим барьером для хакера, так как он никогда не будет уверен в том, что действительно устранил все ловушки.

Включение шума

Добавьте шум в зашифрованных файлах. Для этого в любом месте программы подойдут запросы к ключам, шифрование и дешифровка данных с помощью программы HL_CODE. Избыточные значения будут генераторами случайных чисел, значениями времени, промежуточными результатами вычислений и так далее. Конечно, эти действия не должны давать никаких значимых результатов.

Но в этом есть смысл, так как хакеру невозможно отследить реальные данные. Этим шумом хакер отвлекается от действительно необходимых данных.

Проверки последовательности

Хакер должен модифицировать вашу программу так, чтобы она работала без ключа защиты. Это значит, что он должен заменить определенные последовательности байтов в вашей программе на другие. Одно из средств увеличить время взлома вашей программы – проведение проверок на целостность. Но эта мера не может применяться достаточно часто. Перепроверка должна проводиться всегда по-разному:

- внесите в программу вычисление и проверку различных контрольных сумм важных участков программы. Сами контрольные суммы должны быть защищены от модификации, например, путем деления на многие переменные;
- вычислите и введите перепроверку контрольных сумм определенных участков EXE-файлов, оверлеев и т.д.;
- введите перепроверку изменения отдельных важных команд.

Используйте также различные алгоритмы для расчета различных контрольных сумм. Например, вы можете поместить константы в память ключа (если она доступна), которые включаются в расчет как компоненты контрольных сумм.

Проверка функций контроля

Выполнение запросов должно быть многоуровневым и комплексным. Так, подпрограммы должны запрашивать, имеется ли ключ. Другие подпрограммы должны определять доступность и работоспособность запросов. Взаимосвязь между этими проверками может быть расширена до бесконечности. При этом

одна подпрограмма, которая только что была расшифрована, проверяет другую, которая должна быть зашифрована, является ли она зашифрованной.

Сделав одну программу зависимой от другой, вы значительно усиливаете защиту.

Контроль времени выполнения программы

Сравните время фактического выполнения защищенной программы с обычно требующимся для этого временем. Программа в пошаговом режиме или в режиме трассировки отладчика работает значительно медленнее. Это можно использовать, чтобы определить, работает ли программа в данный момент под отладчиком. Если да, то нужно предпринять массу тонких мер, чтобы не показать хакеру, что он был обнаружен.

2.2.6 Установки для завершения программы

Проследите, чтобы после выхода из программы ключ был деинициализирован. Только после вызова функции `HL_LOGOUT()` будет проведена правильная установка времени окончания работы программы. Это будет означать освобождение занятой памяти и строки в таблице регистрации при обращении к сетевому ключу.

2.2.7 Использование ключа Hardlock с опцией памяти

Если у вас есть ключ с опцией памяти, вы можете включить отдельные регистры памяти в запросы защиты. Так, в ROM область могут быть помещены имя пользователя и серийные номера (см. *раздел Термины*) и отображены во время работы программы.

Неиспользованные регистры памяти могут быть заполнены случайными числами, которые ваше приложение запросит, но не будет применять дальше. Но имейте в виду, что ROM область (см. *раздел Термины*) может быть описана любым приложением.

2.2.8 Меры после распознавания атаки

Если при помощи одного из описанных выше методов или проверок на последовательность вы определили, что ваша программа атакуется или была изменена, есть масса возможностей сделать ее непригодной.

При этом предпринятые меры должны быть как можно дальше от истинной причины распознавания атаки, чтобы хакер не знал, как он был обнаружен. Для этого вы можете использовать несколько приведенных ниже способов.

Задержка

Задержка мер во времени. Только через некоторое время (от секунд до дней при использовании системных данных) что-либо произойдет.

Скрытие

Скрытие причинной связи. Маленькое изменение приводит ко многим изменениям в других величинах, среди которых только одно осуществляет действительно реальную меру.

Искажение

Пусть ваша программа продолжает работать, но неверно: допускает ошибки при вычислениях, выдает не совсем верные результаты, которые не сразу будут обнаружены.

Ограничения

Маскировка трудно распознаваемых ограничений: хакер слишком поздно замечает, что программа работает только в демонстрационном режиме (например, с ограниченным набором данных или без печати).

2.3 LiMaS-функции программы Hardlock API

Функции **LiMaS** в программе **HL-API** позволяют осуществить ручное подключение HL-RUS к 32-х разрядным приложениям Windows.

Система предоставляет реальную возможность управлять опциями программного обеспечения и ключом **Hardlock**, производить их подключение электронным путем. Для этого каждой опции будет соответствовать одна сублицензия (один слот). Каждой программе будет соответствовать один счетчик (в программе **Hardlock Server** – каждому слоту) и одни данные по выполнению. Эти лицензии могут быть назначены по электронной почте.

В качестве аппаратного средства вам нужен ключ с памятью, для применения в сети — ключ **Hardlock-Server**. Необходимым условием применения HL-RUS является соответствующее кодирование ключей, так, чтобы часть памяти осталась зарезервированной для HL-RUS. Данные занимают 14 байтов в ROM и все 32-байтное пространство RAM.

По отдельности вам предоставляются следующие возможности с использованием разных ключей Hardlock.

	Локальный ключ	Сетевой ключ (сервер)
информация по лицензиям	в памяти Hardlock	в памяти Hardlock и в одном лицензионном файле
максимальное число слотов	до 96 слотов	до 32.768 слотов
статус слота	AN/AUS	число одновременных пользователей слотами
счетчик	1 глобальный счетчик	1 глобальный счетчик
дата окончания действия	1 глобальная дата	1 глобальная дата на каждый слот (проверка проходит автоматически при применении слотов)

Данные по лицензированию защищаются персональным ключом Vendor-Key от манипуляций. Он проверяет лицензии и модификации. Этот Vendor-Key создается вами с помощью программы **Vendor-Key-Manager** (см. *раздел HL-Bistro*) и де-

лится на сигнатурную часть (защищается паролем) и исполняемую (используемую при кодировании модулей и их модификации), которая включается в ваше программное обеспечение и используется для проверки лицензий.

API предоставляет необходимые команды, чтобы ввести все эти функции в ваше приложение. Вы можете запросить и увеличить счетчик числа запусков программы, проверить последовательность (глобально и при применении **Hardlock Server** на каждый слот) и занять отдельные слоты. Также легко можно произвести активацию информации по лицензиям. Необходимым условием для этого является то, что используемый ключ **Hardlock** при кодировании программой **Cappuccino** был инициализирован непосредственно для HL-RUS. HL-RUS-API может использоваться в Win32 и для Linux.

2.4 Доступ к HL-Server

При осуществлении защиты ручным методом с использованием API легко отличить запросы к локальному и сетевому ключу, предоставляемому программой HL-Server. В зависимости от выбранного режима при помощи функции HL_LOGIN(...) или HLM_LOGIN(...) система будет пытаться получить доступ к локальному и/или сетевому ключу. Для доступа к сетевому ключу в сети должна быть установлена программа HL-Server.

Итак, у вас есть возможность, запрограммировав систему, использовать локальный или сетевой ключ для одной и той же версии программы.

Более подробную информацию о применении ключа **Hardlock** в сети вы найдете в разделе *HL-Server*.

2.4.1 Изменение типа доступа

После установления связи с ключом при помощи функции HL_LOGIN(...) связь поддерживается до выполнения функции HL_LOGOUT(), то есть имеется возможность перехода от локального к сетевому ключу следующим путем:

```
HL_LOGOUT()
HL_LOGIN(..., REMOTE_DEVICE, ..., ...)
```

Тем самым становится возможным при удалении локального ключа **Hardlock** автоматически переключиться на ключ, доступный через HL-Server.

С другой стороны, в программе должен быть автоматический переключатель на локальный ключ защиты, чтобы система продолжала работать правильно, если по каким-либо причинам программа HL-Server (или сетевой ключ) перестанут быть доступными.

2.4.2 Поведение приложения при тайм-ауте

При использовании программы HL-Server нужно обратить внимание на опциональное применение тайм-аута (см. раздел *HL-Server*). Если в течение установленного времени не будет запроса к ключу у программы HL-Server (например, программа находится в цикле ожидания ввода с клавиатуры и не выполняет запрос), соответствующая строка таблицы регистрации будет снова удалена. Тем самым прерывается связь защищенной программы с ключом. Связь может быть легко установлена вновь с помощью указанной выше последовательности. Однако во время перерыва другая рабочая станция может зарегистрироваться на HL-Server, и тем самым будет достигнут лимит лицензий.

Поэтому используйте достаточно большое значение параметра тайм-аута при установке программы или отключите использование тайм-аута при использовании программ с ручной защитой. При защите программы автоматическим путем с HL-Crypt могут быть выполнены периодические запросы к сетевому ключу в фоновом режиме (см. раздел *HL-Crypt*).

3 Функции Hardlock-API (верхнего уровня)

С помощью описанных ниже функций вы сможете сами выполнить вызовы **API** на языке верхнего уровня. Обратите внимание, что применение отдельных функций в вашем языке программирования может немного отличаться от описанных ниже функций. Все приведенные здесь примеры написаны в псевдокоде.

Все функции основываются на функциях **API** нижнего уровня. Для многих приложений является достаточным применение вызовов функций верхнего уровня. Если вы заинтересованы в том, чтобы самостоятельно использовать интерфейс нижнего уровня, мы вышлем вам по вашему требованию документацию по программам нижнего уровня. Часть этой документации вы найдете **Help к программе Latteccino** (см. раздел *HL-Bistro*).

Для специальных требований конфигурации **API** верхнего уровня также эмулирует функции старого ключа **Hardlock** (перед вводом **API**). Подробнее см. в *разделе 4.4*.

Пояснения к отдельным функциям высокоуровневого API имеют следующую структуру:

Название функции ([аргумент 1 [, аргумент 2 [, ...]]])

Краткое описание функции

Аргументы

Перечень всех параметров, которые передаются этой функцией.

Возвращает (Значение)

Перечень всех значений, которая эта функция передает приложению.

Применение

Описание действия функции.

Примеры

```
ïðëääääáííüâ â ýòìì íáçíðâ ìðëíáðû íàìëñáíü â
íñääääíëíää è ñëóæàð òíëüëí äëý äáíííñòðàöëë
ìðëíáíáíëý òóíëöëë.
```

Функции приведены в алфавитном порядке.

Обзор функций высокого уровня

Функция высокого уровня	Применение
HL_ABORT()	Освобождает структуру API. Эта функция может быть использована только в аварийных случаях (например, в обработчике критических ошибок)
HL_ACCINF()	Устанавливает, был ли ключ установлен локально или "удаленно"
HL_AVAIL()	Устанавливает, требуется ли ключ с определенным кодированием
HL_CODE (DATA_POINTER, BCNT)	Зашифровывает и расшифровывает данные при помощи ключа
HL_HLSVERS()	Выдает номер версии используемого HL-Server
HL_LOGIN(MOD, ACCESS, REFKEY, VERKEY)	Инициализирует структуру API и определяет режим доступа. Если необходимо приложение регистрируется на HL-Server
HL_LOGOUT()	Освобождает структуру API. Если нужно, приложение отключается от HL-Server
HL_MAXUSER()	Определяет число максимально возможных регистраций
HL_MEMINF()	Выясняет, имеет ли активированный ключ опцию памяти
HL_PORTINF()	Выдает адрес порта ключа
HL_READ (REGISTER, VALUE)	Читает содержание регистра памяти ключа
HL_READBL (DATAPTR)	Считывает все содержание памяти ключа в данную область
HL_READID (IDLow, IDHigh)	Считывает номера серий Hardlock USB
HL_SELECT (DATA_AREA)	Переключает между различными областями применения функций Hardlock
HL_USERINF()	Показывает число строк регистрации в таблице HL-Server
HL_VERSION()	Выдает номер версии локальных подпрограмм
HL_WRITE(REGISTER, VALUE)	Описывает регистр Hardlock в области RAM

Функция высокого уровня	Применение
HL_WRITEBL (DATAPTR)	Записывает передаваемую область данных в область RAM ключа
HLM_CHECKCOUNTER (INCVAL, MAXCOUNTER, CURRENTCOUNTER)	Запрашивает глобальный счетчик Hardlock системы RUS
HLM_CHECKEXPPDATE (SLOT, YEAR, MONTH, DAY)	Запрашивает последовательность HL-RUS. Номер слота обозначает глобальную последовательность
HLM_CHECKSLOT (SLOT, MAXUSER, CURRENTUSER)	Перепроверяет слот в HL-RUS
HLM_FREESLOT (SLOT)	Освобождает один слот в HL-RUS
HLM_GETRUSINFO (BUFLFN, RTBBUFFER, BASE64)	Считывает информацию по лицензиям в HL-RUS, чтобы ее отправить производителю.
HLM_ISRUSHL (ID)	Выясняет, инициализирован ли ключ в системе RUS
HLM_LOGIN (MOD, ACCESS, REFKEY, VERKEY, VKEY, RUSOPTIONS, SEARCHSTR)	Инициализирует структуру API и определяет режим доступа. Если необходимо, приложения регистрируются на HL-Server. Одновременно инициализируется система RUS
HLM_OCCUPYSLOT (SLOT)	Устанавливает, свободен ли слот HL-RUS
HLM_WRITELICENSE (BUFLFN, RTBBUFFER, ACCESS, SEARCHSTR, OPTIONS)	Записывает активированные данные лицензий в HL-RUS.

HL_ABORT()

Снова освобождает структуру API.

Аргументы

Нет

Значение

Не определено.

Применение

Эта функция применяется только в исключительном случае (например, в обработчике критических ошибок). Все задействованные прерывания восстанавливаются. После вызова этой функции API больше не может выполнять ни одной функции, не разрешается проводить никаких новых инициализаций. Приложение должно быть закрыто как можно скорее.

Пример

```
.  
.  
HL_ABORT ( ) ;  
QUIT;
```

HL_ACCINF()

Определяет, является ли ключ локальным или инициализирован в сети.

Аргументы

Нет

Значение

Возможны следующие выходные значения:

- | | | |
|-----------|---------------------|--------------------------|
| 1 | LOCAL_DEVICE | ключ установлен локально |
| 2 | NET_DEVICE | ключ установлен в сети |
| -1 | | произошла ошибка |

Применение

Если был определен параметр DONT_CARE для функции HL_LOGIN(...), функция HL_ACCINF() может быть использована для определения инициализации локального или удаленного ключа. Значение DONT_CARE не может быть выдано после инициализации.

Пример

```
result = HL_LOGIN
(29809,DONT_CARE,»HARDLOCK»,»@0=/&#s3");
IF (result == STATUS_OK)
    IF (HL_ACCINF() == NET_DEVICE)
        PRINT «Hardlock is remote»;
    ELSE
        PRINT «Hardlock is local»;
    ENDIF;
    result = HL_LOGOUT();
ENDIF;
```

HL_AVAIL()

Выясняет, имеется ли ключ с ожидаемым кодированием.

Аргументы

Нет

Значение

Возвращаемое значение получает статус API (см. таблицу значений статуса API в Приложении).

Применение

Эту функцию вызывают, чтобы выяснить, есть ли ключ с ожидаемым кодированием (например, до того как данные будут расшифрованы). Если пара значений параметров REFKEY и VERKEY не равны нулю, они используются для идентификации ключа. Дополнительно при параллельном обращении проверяется статус выделенной линии порта принтера, так как у некоторых принтеров в выключенном состоянии линии становятся заземленными, и ключ не может быть определен.

Пример

```
result = HL_LOGIN
(29809,DONT_CARE,»HARDLOCK»,»@0=/{s3");
IF (result == STATUS_OK)
    .
    IF (HL_AVAIL() == STATUS_OK)
    .
    .
    (Hardlock Funktion)
    .
    .
ENDIF;
HL_LOGOUT();
ENDIF;
```

HL_CODE(DATAPTR,BCNT)

Зашифровывает и расшифровывает данные при помощи ключа.

Аргументы

DATAPTR	указывает на область данных, которая должна быть зашифрована. Эта переменная содержит 32-битный указатель реального режима (сегмент смещение) для некоторой области данных. При использовании 32-битного интерфейса API должен использоваться 32-битный (near) указатель защищенного режима (смещение).
BCNT	Число блоков данных, требующих зашифровки 8-байтными блоками. Значение может быть от 0 (но это на самом деле не имеет смысла) до 8192 (соответствует 64 KB).

Значение

Значение выхода содержит статус API (см. таблицу значений статуса API в *Приложении*).

Применение

Эта функция зашифровывает или расшифровывает область данных. Заданная область данных будет напрямую модифицирована, т.е. переписана со значениями, полученными при вызове функции. Вызовы с неправильно установленными показателями могут привести к нежелательным результатам. В случае ошибки указанная область данных может быть разрушена. Так как здесь применяется блок-шифр, нужно указать число шифруемых блоков данных (по 64 бит каждый). Если вы хотите зашифровать данные длиной менее 8 байт, нужно дополнить их до этого значения. Приложение должно быть уверено, что весь блок данных не выходит за границу сегмента (указатели должны быть нормализованы), чтобы избежать отсечения (Wrap around).

Пример

```
result = HL_LOGIN
(29809,DONT_CARE,»HARDLOCK»,»@0=/{s3");
IF (result == STATUS_OK)
    text = «Hello Hardlock !»;
    IF (HL_CODE(text, 2) == STATUS_OK))
        PRINT "The encrypted data is: "
            + text;
    ENDIF;
result = HL_LOGOUT();
ENDIF;
```

HL_HLSVERS()

Выдает номер использованной версии **HL_HLSVERS()**.

Аргументы

Нет

Значение

Система выдает номер используемой версии HL-Server в виде целого числа (например, 210 соответствует версия 2.10). Это не API-версия HL-Server. При локальном ключе (или в случае ошибки) выдается значение 0.

Применение

Функция информирует вас об использованной версии HL-Server. Эта информация важна, если некоторая функция является доступной только для определенной версии.

Пример

```
result = HL_LOGIN
(29809,NET_DEVICE,»HARDLOCK»,»@0=/{&#s3");
IF (result == STATUS_OK)
    IF (HL_HLSVERS() < 210)
        PRINT «Application needs HL-Server >=
                2.10»;
        result = HL_LOGOUT();
        QUIT;
    ENDIF;
.
.
result = HL_LOGOUT();
ENDIF;
```

HL_LOGIN(MOD,ACCESS,REFKEY,VERKEY)

Инициализирует структуру API и определяет режим доступа. В случае необходимости приложение регистрируется на HL-Server.

Аргументы

MOD	Адрес модуля ключа Hardlock.
ACCESS	Определяет режим доступа
1 LOCAL_DEVICE	локальный ключ
2 NET_DEVICE	Доступ через HL-Server.
3 DONT_CARE	Искать локально, потом через сеть.
REFKEY	Если эта строка (8 байт) не равна нулю(двоичному), она сравнивается с зашифрованным значением VERKEY для идентификации ключа.
VERKEY	(8 байт) содержит зашифрованное значение REFKEY . Корректное значение для REFKEY (Это значит зашифрованное Вашим ключом значение VERKEY) может быть легко найдено с помощью Latteccino.

Значение

Значение выхода содержит статус API (см. таблицу значений статуса API в *Приложении*).

Применение

Эта функция передает информацию о ключе в структуру API при ее инициализации. Если определено, приложение регистрируется на HL-Server. Все другие доступы должны быть выполнены между HL_LOGIN(...) и HL_LOGOUT(). Вот почему важно выполнить функцию HL_LOGIN(...) при запуске приложения. Если введен параметр DONT_CARE, интерфейс API будет выполнять поиск вначале локального, потом сетевого ключа.

Интерфейс API автоматически ищет ключ по всем адресам параллельных портов. По умолчанию последовательность поиска \$378, \$278 и \$3BC. Последовательность поиска на последовательных портах может быть активирована при помощи переменной среды (см. раздел 4).

Если используются параметры REFKEY и VERKEY, они будут автоматически привлечены для анализа алгорит-

ма работы ключа. Используемый для этой цели внутренний алгоритм не совместим с K-EYE.

Для отключения проверки REFKEY/VERKEY должен использоваться двоичный ноль (не NULL, как в языке Си). Обратите обязательно внимание, что параметры REFKEY/VERKEY относятся во всех демонстрационных программах только к демонстрационному ключу (адрес модуля 29809). Для применения API с вашими собственными ключами вы должны сгенерировать набор значений при помощи программы Latteccino.

Пример

```
result = HL_LOGIN(29809,DONT_CARE,»HARDLOCK»,»@0=/  
&#s3");  
IF (result == STATUS_OK)  
    .  
    .  
    (Hardlock Funktionen)  
    .  
    .  
    result = HL_LOGOUT();  
ENDIF;
```


HL_LOGOUT()

Освобождает структуру API. Если определено, то приложение отключается от HL-Server.

Аргументы

Нет

Значение

Значение функции содержит статус API. Смотри таблицу значений статуса в *Приложении*.

Применение

Эта функция снова освобождает структуру API. Если необходимо, система сначала отключается от HL-Server. После выполнения этой функции ключ перестает быть доступен. Следовательно, она должна вызываться в конце приложения.

Пример

```
result = HL_LOGIN
(29809,DONT_CARE,»HARDLOCK»,»@0=/&#s3");
IF (result == STATUS_OK)
.
(Hardlock Funktionen)
.
result = HL_LOGOUT();
QUIT;
ENDIF;
```

HL_MAXUSER()

Определяет число максимально возможных (эффективных) строк регистрации.

Аргументы

Нет

Значение

Значение выхода содержит число максимально возможных строк регистрации, если будут инициализирован сетевой ключ. В случае локального ключа выдается значение 1, в случае ошибки – значение –1.

Применение

Вы можете использовать эту функцию при работе с ключом в сети для указания разрешенного числа подключений пользователей к HL-Server. Поскольку сеть может поддерживать несколько ключей защиты с одинаковым адресом модуля, то на разных компьютерах с программой HL-Server интерфейс API группирует вместе все строки для ключей с одинаковым адресом модуля и кодированием. Для запроса максимального числа пользователей с отдельных позиций используется функция HLM_CHECKSLOT.

Пример

```
result = HL_LOGIN
(29809,DONT_CARE,»HARDLOCK»,»@0=/{&#s3");
IF (result == STATUS_OK)
    IF (HL_ACCINF() == NET_DEVICE)
        max = HL_MAXUSER();
        PRINT «HL-Server is licensed for: «
            + max;
    ENDIF;
ENDIF;
result = HL_LOGOUT();
```

HL_MEMINF()

Устанавливает, имеет ли инициализированный ключ память.

Аргументы

Нет

Значение

Значение выхода содержит статус API. Смотри таблицу значений статуса в *Приложении*.

Применение

Этой функцией можно проверить наличие у ключа опции памяти. Функция пытается записать и прочитать значение регистра. Из сравнения результатов делается вывод о наличии опции памяти. Далее восстанавливается первоначальное состояние.

Пример

```
result = HL_LOGIN
(29809,DONT_CARE,»HARDLOCK»,»@0=/&#s3");
IF (result == STATUS_OK)
    IF (HL_MEMINF() == STATUS_OK)
        PRINT «Hardlock with memory found.»;
    ENDIF;
    result = HL_LOGOUT();
ENDIF;
```

HL_PORTINF()

Выдает информацию об адресе порта с установленным ключом.

Аргументы

Нет

Значение

Система выдает адрес порта инициализированного ключа. При **Hardlock** USB значение 0, при ошибке -1.

Применение

Эта функция определяет адрес порта с ключом, инициализированным при помощи функции HL_LOGIN(...). При локальном ключе выдаваемое значение – это локальный порт, при HL-Server – использованный порт на компьютере с сервером. Но этой функцией нельзя определить, инициализирован ключ локально или в сети. Для этого используйте функцию HL_ACCINF.

Пример

```
result = HL_LOGIN 29809,DONT_CARE,»HARDLOCK»,»@0=/  
&#s3");  
IF (result == STATUS_OK)  
.  
.  
    Port = HL_PORTINF();  
.  
.  
    result = HL_LOGOUT();  
ENDIF;
```

HL_READ(REG,VALUE)

Читает содержание регистра памяти ключа **Hardlock**.

Аргументы

REG	номер регистра должен находиться между 0 и 63.
VALUE	переменная, которой должно быть присвоено значение 16-битного регистра и которая должна быть передана с помощью ссылки

Значение

Система выдает статус ключа. Если указан недействительный регистр, система выдает сообщение об ошибке **INVALID_PARAM** (см. таблицу значений статуса в *Приложении*).

Применение

Функция читает только один регистр памяти ключа, если ключ имеет опцию памяти. Содержание регистра передается в виде 16-битного значения переменной **VALUE** (формат Intel).

Пример

```
result = HL_LOGIN
(29809,DONT_CARE,»HARDLOCK»,»@0=/&#s3");
IF (result == STATUS_OK)
    IF (HL_AVAIL() == STATUS_OK)
        value = 0;
        result = HL_READ (48, value);
    ENDIF;
    .
    result = HL_LOGOUT();
ENDIF;
```

HL_READBL(DATAPTR)

Читает целый блок памяти ключа.

Аргументы

DATAPTR указатель области данных

Значение

Значение выхода содержит статус API. Смотри таблицу значений статуса в *Приложении*.

Применение

С помощью этой функции можно читать всю память (регистр от 0 до 63). Область данных, определяемая при помощи указателя DATAPTR, должна иметь размер 128 байт.

Пример

```
result = HL_LOGIN
(29809,DONT_CARE,»HARDLOCK»,»@0=/{#s3");
IF (result == STATUS_OK)
    IF (HL_AVAIL() == STATUS_OK)
        eeprom = SPACE(128);
        HL_READBL (eeprom);
    ENDIF;
    result = HL_LOGOUT();
ENDIF;
```

HL_READID(IDLow, IDHigh)

Читает внутренний номер ключа **Hardlock USB**.

Аргументы

IDLow	слово низкого уровня (Low-Word) номера серии ключа Hardlock USB
IDHigh	слово высокого уровня (High-Word) номера серии ключа Hardlock USB

Значение

Система выдает номер серии (Low-/High-Word). Из-за требуемой конфигурации к высокоуровневым языкам программирования ID передается как результат через два слова. Общий ID рассчитывается следующим образом: $\text{Serial ID} = \text{IDHigh} \times 65536 + \text{IDLow}$.

Применение

Эта функция читает номер серии ID с ключа **Hardlock USB**. Она может быть потом применена, если на основе ID должно быть разрешено одноразовое выделение защищенного программного обеспечения или для распределения заказчик – ключ.

Внимание. Эта функция выдает только при ключе **Hardlock USB** действующий ID.

Пример

```
result = HL_READID( &LowWord, &HighWord );
IF (result == STATUS_OK)
    longbuffer = Highword * 65536 + LowWord;
    PRINT «The serial number is: «
                                +longbuffer;
ELSE
    PRINT «No valid serial ID found!»
ENDIF;
```

HL_SELECT(DATA_AREA)

Переключение между различными рабочими областями для применения функций ключа.

Аргументы

DATA_AREA указатель на область с диапазоном в 256 байт

Значение

Возвращаемое значение содержит статус API (см. таблицу значений статуса в *Приложении*).

Применение

В одной программе может потребоваться переключение между несколькими рабочими областями для многократной инициализации ключа. Для этого должны быть зарезервированы соответствующие области данных для управления структурой API. Область данных должна быть инициализирована нулями. С помощью этой функции можно подключиться к нескольким модулям ключа одновременно.

Выбор в области 0 устанавливает значение по умолчанию и использует только внутреннюю зарезервированную область данных.

Пример

```
area1 = SPACE(256);
area2 = SPACE(256);
memset(area1, 0, 256)

HL_SELECT(area1);
HL_LOGIN(29809, DONT_CARE, »HARDLOCK», »@0=/&#s3");
HL_SELECT(area2);
HL_LOGIN(18328, DONT_CARE, »HARDLOCK», »#:f;)?@0");
...
...
HL_SELECT(area1);
HL_LOGOUT();
HL_SELECT(area2);
HL_LOGOUT();
```


HL_USERINF()

Показывает число строк в таблице регистрации HL-Server.

Аргументы

Нет

Значение

Система выдает данные о числе строк в таблице регистрации (включая ваше собственное подключение) для удаленного ключа. При локальном ключе указывается значение 1, при ошибке –1.

Применение

Эта функция информирует вас о числе подключенных пользователей к HL-Server. С помощью этой функции вы можете сами проверить лицензии на доступ к сети. Так как возможно, что многие ключи с одинаковым адресом модуля, но с разных компьютеров с программой HL-Server одновременно представлены для пользования в сети, интерфейс API группирует вместе все строки, относящиеся к одному и тому же ключу. Используйте для опроса активных пользователей с различных рабочих станций функцию HLM_CHECKSLOT.

Пример

```
result = HL_LOGIN
(29809,NET_DEVICE,»HARDLOCK»,»@0=/&#s3");
IF (result == STATUS_OK)
    IF (HL_USERINF() > 5)
        PRINT «Too many users»;
        result = HL_LOGOUT();
        QUIT;
    ENDIF;
..
    result = HL_LOGOUT();
ENDIF;
```

HL_VERSION()

Определяет номер версии программы API.

Аргументы

Нет

Значение

Номер версии API выдается как целое число (например, 350 соответствует версии 3.50). При ошибке выдается значение -1.

Применение

Этой функцией может быть определена версия API.

Это особенно важно при работе с программой HL-Server, так как только версии API с одним и тем же основным номером работают верно. Так, HL-Server, созданный с использованием API версии 2.xx, не может использоваться с приложением API версии 3.xx. Смешивание версий 3.xx, напротив, возможно.

Версия API активной HL-Server не может быть определена через сеть при помощи вызова API на верхнего уровня. Но с помощью кода состояния VERSION_MISMATCH можно определить несовместимые версии API.

Пример

```
result = HL_LOGIN
(29809,DONT_CARE,»HARDLOCK»,»@0=/&#s3");
IF (result == STATUS_OK)
    version = HL_VERSION();
    result = HL_LOGOUT();
ENDIF;
```

HL_WRITE(REG,VALUE)

Описывает регистр ключа **Hardlock** области RAM.

Аргументы

REG	номер описываемого регистра
VALUE	значение, которое должно быть записано в регистр

Значение

Возвращаемое значение содержит статус API (см. таблицу значений статуса в *Приложении*).

Применение

Этой функцией может быть описан только единственный регистр ключа **Hardlock** области RAM (см. раздел *Термины*). Номер регистра должен располагаться между 48 и 63 (область RAM). Содержание области сохраняется при отключенном ключе (EEPROM).

Внимание. Эта функция может разрушить информацию по лицензиям в ключе **RUS-Hardlock**.

Пример

```
result = HL_LOGIN
(29809,DONT_CARE,»HARDLOCK»,»@0=/&#s3");
IF (result == STATUS_OK)
    result = HL_WRITE(48, 147);
.
.
    result = HL_LOGOUT();
ENDIF;
```

HL_WRITEBL(DATAPTR)

Записывает переданную область данных в область RAM ключа.

Аргументы

DATAPTR указатель области данных

Значение

Возвращаемое значение содержит статус API (см. таблицу значений статуса в *Приложении*).

Применение

Этой функцией можно описать данные в области RAM (см. раздел *Термины*) памяти ключа (регистры от 48 до 63). Область данных должна иметь размер 32 байта. Содержание области сохраняется при отключенном ключе (EEPROM).

Внимание. Эта функция может разрушить информацию по лицензиям в ключе RUS-Hardlock.

Пример

```
result = HL_LOGIN
(29809,DONT_CARE,»HARDLOCK»,»@0=/&#s3");
IF (result == STATUS_OK)
    IF (HL_AVAIL() == STATUS_OK)
        eeprom = SPACE(32);
        HL_WRITEBL (eeprom);
    ENDIF;
    result = HL_LOGOUT();
ENDIF;
```

HLM_CHECKCOUNTER(INCVAL, MAXCOUNTER, CURRENTCOUNTER)

Запрос глобального счетчика (HL-RUS/LiMaS).

Аргументы

INCVAL	значение, на которое должен быть повышен активный счетчик (16 бит)
MAXCOUNTER	указатель переменной, которая отображает максимальное значение счетчика (32 бита)
CURRENT COUNTER	указатель на 32 бита, к оторый показывает текущее состояние счетчика

Значение

Возвращаемое значение содержит статус API (см. таблицу значений статуса в *Приложении*).

Применение

Если необходимо использовать возможности счетчика, то можно этой функцией запросить текущее состояние счетчика. Это состояние может быть повышено на заданное число (INCVAL).

Пример

```

result =
HLM_LOGIN(29809,DONT_CARE,»HARDLOCK»,»@0=/&#s3",
          RUS_VK, 0,»378p,IPX»);
IF (result == STATUS_OK)
    result =
        HLM_CHECKCOUNTER(1,&MAXCOUNT,&CURRENT);
    if (MAXCOUNT-CURRENT<10)
        PRINT(«ïñòàäëîñü òîëüëî ñòîëüëî òî
              %ñòàððòà»,MAXCOUNT-CURRENT);
    .
    .
    result = HL_LOGOUT();
ENDIF;
```

HLM_CHECKEXPDATE(SLOT, YEAR, MONTH, DAY)

Запрашивает данные (HL-RUS/LiMaS).

Аргументы

SLOT	этой командой передается номер позиции. Номер 0 обозначает глобальные данные. действующие значения для локальных модулей: от 0 до 96 и от 1 до 32768 для Hardlock Server (32 Bit)
YEAR	указывает на переменную, в которой отображается сообщение о времени (годе) выполнения программы (16 бит)
MONTH	указывает на переменную, отражающую истекший месяц работы программы (16 бит)
DAY	указывает на переменную, отражающую истекший день работы программы (16 бит)

Значение

Возвращаемое значение содержит статус API (см. таблицу значений статуса в *Приложении*).

Применение

При использовании опции даты проверяется, когда истекает лицензия. При использовании ключа **Hardlock Server** для каждой позиции это может быть по-разному. С помощью позиции 0 запрашивается глобальная дата (локально и на сервере).

Пример

```
result =
HLM_LOGIN(29809,DONT_CARE,»HARDLOCK»,»@0=/#s3",
          RUS_VK, 0,»378p,IPX»);
IF (result == STATUS_OK)
    result = HLM_CHECKEXPDATE(0, &year, &month,
                              &day);

    IF (result == RUS_DATE_EXPIRED)
        PRINT(«ïðáâûðáíà äéíáâëüíäý ààòà»);
        .
        .
    result = HL_LOGOUT();
ENDIF;
```

HLM_CHECKSLOT(SLOT,MAXUSER,CURRENTUSER)

Проверяет одну позицию (HL-RUS/LiMaS).

Аргументы

SLOT	этой функцией передается номер слота. Подходящие значения для локального ключа от 1 до 96 и от 1 до 32768 для Hardlock Server (32 Bit)
MAXUSER	показывает переменную с максимально возможным числом пользователей на один слот (32 бит)
CURRENT USER	указатель на переменную, где отображается текущее число активных пользователей (32 бит)

Значение

Возвращаемое значение содержит статус API (см. таблицу значений статуса в *Приложении*).

Применение

При применении сублицензирования запрашивается актуальный статус одного слота. Передается номер слота и получается информация для этого слота, сколько лицензий используются в данный момент и сколько максимально разрешено.

Пример

```
result =  
HLM_LOGIN(29809,DONT_CARE,»HARDLOCK»,»@0=/&#s3",  
          RUS_VK, 0,»378p,IPX»);  
IF (result == STATUS_OK)  
    result = HLM_CHECKSLOT(3, &MAXUSER,  
                          &CURRENT);  
.  
.  
    result = HL_LOGOUT();  
ENDIF;
```

HLM_FREESLOT(SLOT)

Снова освобождает одну позицию (слот) в системе **Hardlock** **RUS**.

Аргументы

SLOT Этой функцией передается номер слота. Подходящие значения для локального ключа от 1 до 96 и от 1 до 32768 для Hardlock Server (32 бит)

Значение

Возвращаемое значение содержит статус API (см. таблицу значений статуса в *Приложении*).

Применение

Этой функцией освобождается одна позиция (слот) или одна лицензия на один слот, который до этого был занят. См. также HLM_OCCUPYSLOT.

Пример

```
result =
HLM_LOGIN(29809,DONT_CARE,»HARDLOCK»,»@0=/&#s3",
          RUS_VK, 0,»378p,IPX»);
IF (result == STATUS_OK)
    HLM_OCCUPY(3)
    .
    .
    HLM_FREESLOT(3);
    HL_LOGOUT();
ENDIF;
```


HLM_GETRUSINFO(BUFLEN, RTBBUFFER, BASE64)

Читает информацию по лицензиям (HL-RUS/LiMaS). Эту информацию можно послать производителю.

Аргументы

BUFLEN	Указатель	Указывает на длину буфера, которая применяется для запоминания RTB (RUS Транспортного Блока). После выполнения функции этой переменной возвращается действительная длина буфера. Если этот параметр 0, то функция отображает необходимую длину буфера (32 бита).
RTB BUFFER BASE64	Указатель	Указывает на первый байт буфера RTB (8 бит).
	0	RTB отражается в бинарном формате.
	1	RTB отражается в кодировке BASE64

Все остальные значения зарезервированы для будущего использования (16 бит).

Значение

Возвращаемое значение содержит статус API (см. таблицу значений статуса в *Приложении*).

Применение

Чтобы сообщить об изменении потребности в количестве лицензий, сначала пользователь должен передать соответствующую информацию о своей текущей лицензии поставщику программного обеспечения. Чтобы прочитать эту информацию из ключа, и применяется эта функция (может не работать, если необходимые данные ключа используются из **HL-DB/LiMaS**).

Пример

```
result = HLM_LOGIN(29809,DONT_CARE,»HARDLOCK»,
                  «@0=/&#s3»,RUS_VK,0,»378p,IPX»);
IF (result == STATUS_OK)
    BUFLen = 0;
    result = HLM_GETRUSINFO(&BUFLen, NULL,1)
    BUFFER = SPACE(BUFLen);
    result = HLM_GETRUSINFO(&BUFLen, BUFFER,1);
    .
    result = HL_LOGOUT();
ENDIF;
```

HLM_ISRUSHL(ID)

Определяет, инициализирован ли ключ **Hardlock** для системы HL-RUS.

Аргументы

ID	Указатель на переменную, отображающую серийный номер ключа (32 бита)
-----------	--

Значение

Возвращаемое значение содержит статус API (см. таблицу значений статуса в *Приложении*).

Применение

Этой функцией можно проверить, сконфигурирован ли ключ **Hardlock**, к которому только что подключились, для системы RUS. Если да, то передается номер серии RUS.

Примечание. Серийный номер ключа **Hardlock** USB, который читается функцией HL_READID(...), от этой функции независим.

Пример

```
result =
HLM_LOGIN(29809,DONT_CARE,»HARDLOCK»,»@0=/&#s3",
          RUS_VK, 0,»378p,IPX»);
IF (result == STATUS_OK)
    result = HLM_ISRUSHL(&SERIAL);
    IF (result == STATUS_OK)
        PRINT(«RUS HARDLOCK found»);
    .
    .
    result = HL_LOGOUT();
ENDIF;
```

HLM_LOGIN(MOD,ACCESS,REFKEY,VERKEY,VKEY, OPTIONS, SEARCHSTR)

Инициализирует структуру API и определяет режим доступа. Если определено, приложение будет зарегистрировано на HL-Server. Одновременно будет инициализирована система RUS.

Аргументы

MOD, ACCESS, REFKEY, VERKEY VKEY	Смотри описание функции HLM_LOGIN(...). Указатель на структуру данных ключа Vendor-Key. Вы можете создать ваш собственный Vendor-Key с менеджером Vendor-Key. Если VKEY = 0, проводится нормальное подключение, после чего возможно указание последовательности поиска (SEARCHSTR).
OPTIONS=1	Этой функцией API инициализируется в режиме RUS без указания VKEY. При этом возможен запрос данных по лицензиям без ключа Vendor-Key. Применение лицензий, например, занять строку регистрации, этой функцией невозможно. Опция будет игнорирована, если будет задан ключ Vendor-Key. Другие значения зарезервированы для будущих расширений (32 бита).
SEARCHSTR	Указатель на строку поиска, определенную программой. Смотрите главу 4, последовательность поиска для API.

Значение

Возвращаемое значение содержит статус API (см. таблицу значений статуса в *Приложении*).

Применение

Эта функция расширяет функцию HLM_LOGIN (см. описание этой функции). Дополнительно проводится инициализация функций системы RUS и сублицензирование для функций API. Обратите внимание на главу 2.3.

Дополнительно есть возможность напрямую вести последовательность поиска **Hardlock-API** в программе. Для этого функция HL_SEARCH должна пропустить параметры для работы функции согласно переменным среды, смотрите также главу 4.2.

Примечание. Если вызывается функция с параметрами HLM_LOGIN (MODAD, ACCESS, REFKEY, VERKEY, NULL, 0, NULL), она полностью совместима с функцией HL_LOGIN.

Пример

```
result = HLM_LOGIN(29809,DONT_CARE,»HARDLOCK»,
                  «@0=/&#s3»,RUS_VK,0,»378p,IPX»);
IF (result == STATUS_OK)
    .
    .
    (Hardlock RUS Funktion)
    .
    .
    result = HL_LOGOUT();
ENDIF;
```

HLM_OCCUPYSLOT(SLOT)

Занимает одну позицию (слот) (HL-RUS/LiMaS).

Аргументы

SLOT	Передается номер слота. Действующие значения от 1 до 96 для локальных модулей и от 1 до 32768 для Hardlock Server (32 Bit)
-------------	--

Значение

Возвращаемое значение содержит статус API (см. таблицу значений статуса в *Приложении*).

Применение

Система сублицензирования **Hardlock** основана на расположении лицензий на один модуль программного обеспечения. Это реализуется применением позиций (слотов). При локальных ключах одному модулю соответствует одна позиция (один слот) и потом устанавливается, может ли эта позиция использоваться или нет. При **Hardlock Server** на одну позицию может быть задано несколько лицензий и одна дата выполнения. С HLM_OCCUPYSLOT занимается один заданный слот или лицензия.

Пример

```
result =
HLM_LOGIN(29809,DONT_CARE,»HARDLOCK»,»@0=/&#s3",
          RUS_VK, 0,»378p,IPX»);
IF (result == STATUS_OK)
    result = HLM_OCCUPYSLOT(3);
    if (result == TOO_MANY_USERS)
        PRINT(«Too many users»);
    .
    .
    result = HL_LOGOUT();
ENDIF;
```

HLM_WRITELICENSE(BUFLEN, RTBBUFFER, ACCESS, SEARCHSTR, OPTIONS)

Записывает актуализированные данные по лицензиям назад в систему **Hardlock** RUS.

Аргументы

BUFLEN	указывает длину буфера, который используется для сохранения транспортного блока RUS (32 бит)
RTBBUFFER	указывает на первый байт буфера RTB (8бит)
ACCESS	смотрите описание функции HL_LOGIN
SEARCH	смотрите описание функции HLM_LOGIN
STRING	
OPTIONS	если это значение равно 1 (FORCE_ALF_CREATE), в любом случае будет создан файл лицензии Aladdin (ALF), даже если в стандартной директории поиска не будут найдены файлы ALF. В остальных случаях равно 0 (16 бит)

Значение

Возвращаемое значение содержит статус API (см. таблицу значений статуса в *Приложении*).

Применение

Этой командой активируется информация по лицензиям в ключе **Hardlock** или в файле лицензий.

Пример

```
result = HLM_WRITELICENSE(BUFLEN, BUFFER,
                           DONT_CARE, «378p», 0);
if (result == RUS_LICFILE_NOT_FOUND)
    PRINT(«Forcing file creation»);
    HLM_WRITELICENSE(BUFLEN, BUFFER, DONT_CARE,
                     «378p», FORCE_ALF_CREATE);
```

4 Приложение

4.1 Таблица значений статуса

№.	Обозначение	Значение
0	STATUS_OK	Функция выполнена без ошибок.
1	NOT_INIT	Структура API не инициализирована. Функция не может быть выполнена. сначала должен быть проведен успешный HL_LOGIN(...).
2	ALREADY_INIT	Структура API уже инициализирована. Была проведена попытка повторной инициализации без ее освобождения.
3	UNKNOWN_DONGLE	Неизвестный ID модуль. Этот ID не поддерживается API. Обычно это результат ошибки в программе (неправильный указатель).
4	UNKNOWN_FUNC	Неизвестный номер функции API. Использован неверный номер функции. Обычно это результат ошибки в программе (неправильный указатель).
6	HLS_FULL	Таблица регистрации заполнена.
7	NO_DONGLE	Не подключен ключ. При работе в сети причины ошибок могут быть следующие: Timeout, станция с HL-Server не доступны через сеть.
8	NETWORK_ERROR	Ошибка в сетевой операции (удаленной). Причиной может быть отсутствие загруженных протоколов, таких как IPX или NetBios на локальной машине.
9	NO_ACCESS	Неправильно выбранный режим доступа (например, ACCESS = 0). Обычно это результат ошибки в программировании (неправильный указатель).
10	INVALID_PARAM	При вызове функции был выбран недействительный или неправильный параметр.

№.	Обозначение	Значение
11	VERSION_MISMATCH	При установлении связи система определила, что HL-Server или установленный драйвер ключа Hardlock несовместимы с используемой версией API.
12	DOS_ALLOC_ERROR	Обнаружена ошибка при выполнении программы размещения памяти.
13	(reserved)	
14	CANNOT_OPEN_DRIVER	Не может быть открыт драйвер. Обычно причина в незагрузке драйвера.
15	INVALID_ENV	Указанная переменная среды для последовательности поиска API содержит только неправильные строки.
16	DYNALINK_FAILED	Не найдена функция в DLL.
17	INVALID_LIC	Не найдена текущая информация по лицензиям.
18	NO_LICENSE	Слот/лицензия не освобождены.
20	RUS_NO_DEVICE	Нет ключа HL-RUS.
21	RUS_INVALID_LIC	Неверная лицензия RUS.
22	RUS_SYNC_ERR	FIB в ключе и структура API не подходят друг другу.
23	NOT_IMPLEMENTED	Еще не реализованы (не разработаны).
24	BUFFER_TOO_SMALL	Буфер слишком мал для функции.
25	UNKNOWN_HW_TYPE	Неизвестный ответ программного обеспечения.
26	RUS_INV_FBPOS	Ошибка в блоке лицензий (Fixed Block).
27	RUS_INVALIDSLOT	Указанный номер слота не существует.
28	RUS_DATE_FAKE	Установлена манипуляция с датой RUS.
29	RUS_COUNT_DOWN	Достигнут лимит счетчика RUS.
30	RUS_INVALID_VK	Ключ RUS-Vendor-Key не работает.
31	RUS_NO_LIC_FILE	Не найден файл лицензии RUS.
32	RUS_INV_VBLOCK	Некорректный блок лицензий RUS (Var Block).

№.	Обозначение	Значение
33	RUS_LIC_FILE_WRITE_ERR	Ошибка при записи активированного файла лицензий.
34	RUS_NO_INFO_AVAILABLE	Нет информации по аппаратным средствам RUS-Hardware .
35	RUS_INFO_PACK_ERR	Ошибка в кодировании информации аппаратных средств RUS-Hardware (TLV Block).
36	RUS_LIC_WRITE_ERR	Ошибка при записи лицензий.
37	RUS_DATE_EXPIRED	Достигнута дата окончания работы RUS.
38	TS_DETECTED	Найден терминал Server/Citrix Winframe.
39	RUS_INVALID_RTБ	Неверная информация по модификации (RTB).
40	RUS_RTБ_EXPIRED	Данные модификации RTB больше не действительны.
41	RUS_SERIAL_MISMATCH	Серийный номер данных модификации не соответствует ключу.
256	TOO_MANY_USERS	Достигнуто максимальное число пользователей.
257	SELECT_DOWN	Если ключ не получает питания, он не работает и, следовательно не может быть определен. Интерфейс API может определить эту ситуацию измерением уровня напряжения ключа на выбранной цепи порта принтера. Если эта цепь соединена с корпусом, это означает, что и другие цепи (с которых ключ получает питание) также закорочены. Например, это может быть когда некоторые принтеры находятся в режиме офлайн или выключены. Здесь программист может дать сигнал пользователю включить принтер он-лайн или удалить его от интерфейса. Эта ошибка относится только к локальной операции.
258	NO_SERIALID	Номер серии не читается или не имеется вообще.

4.2 Поиск файла лицензии HL-RUS (ALF)

Приложение ищет информацию по лицензиям в каталоге, в котором само находится. Это действие может быть изменено заданием переменной среды. Задайте переменную следующим образом:

```
SET HL_LICENSEDIR=[directory]
```

Пример

```
SET HL_LICENSEDIR=C:\MYAPP\LICENSE
```

Приложение ищет в заданном каталоге файлы с расширением ALF (Aladdin License File) и распознает нужную лицензию по номеру серии RUS.

4.3 Определение последовательности поиска API

4.3.1 Подготовка

В **Hardlock** API версии 3.25 и выше можно точно указать последовательность поиска с помощью переменной среды. При указании последовательности поиска можно избежать конфликта, если система автоматически ищет адреса LPT-портов (например, когда сетевая карта настроена на адреса LPT-портов).

Поиск ключа при помощи интерфейса API на последовательном порту может быть выполнен только при помощи переменной среды.

4.3.2 Синтаксис

Переменная среды имеет следующий синтаксис:

```
HL_SEARCH=[Port] , . . . , [Protocol] , . . .
```

[Port]- представляет адрес ввода/вывода в шестнадцатеричной форме и идентификатор (ID) порта.

Идентификатор порта	Значение
p = параллельный	нормальный параллельный порт
s = последовательный	нормальный последовательный порт
e = ECP	параллельный порт в режиме ECP
n = NEC (Japan)	У японских моделей NEC другое назначение порта. Этот параметр активирует специальный обработчик, поэтому отдельный NEC API не требуется
C = Compaq Contura Dockingbase	Мультиплексор модуля (используемого для переключения между параллельным портом и адаптером Ethernet) переключается на параллельный порт для запроса к ключу Hardlock
i = IBM PS/2	IBM PS/2 исключает ошибку при перепрограммировании порта некоторых видеодрайверов в среде Windows. (Hardlock не находится после того, как запускается Windows). Раньше это выполнялось при помощи Hardlock API. Теперь эта функция может быть выполнена только при помощи переменной среды

[Protocol] – определяет протокол для доступа к HL-Server. На данный момент поддерживаются следующие ключевые слова:

Протокол	Значение
IPX	HL-Server ищется через IPX, или. SAP.
IP	HL-Server ищется через TCP/IP
NETBIOS	HL-Server ищется через NETBIOS.

Пример 1

```
SET HL_SEARCH=378p
```

Ключ ищется на параллельном порту с адресом 0x378.

Пример 2

```
SET HL_SEARCH=378e,2f8s
```

Ключ ищется на локальном параллельном порту с адресом 0x378. Во время доступа к ключу порт переключается

из режима ECP в нормальный режим. Если система не найдет ключ, поиск продолжается на последовательном порту с адресом 0x2f8.

Пример 3

```
SET HL_SEARCH=IPX,278p
```

Сначала система ищет ключ **Hardlock** при помощи HL-Server с помощью IPX/SAP. Если подключиться к HL-Server не удастся, поиск продолжается на локальном параллельном порту с адресом 0x278.

Пример 4

```
SET HL_SEARCH=378p,278p,ЗВср,IPX,NETBIOS,IP
```

Это соответствует автоматической последовательности поиска (HL_LOGIN с DONT_CARE), если переменная среды не определена. Таким образом, эта последовательность является лишней.

Примечание. Последовательность поиска может быть определена непосредственно программированием на низком уровне API.

4.3.3 HL-Server Client для TCP/IP

При использовании 32-битной версии HL-Server для Win95 или Windows NT нужно обратить внимание, что последовательность поиска протоколов (если вы не используете HL_Search0) зависит от клиента. Таким образом:

- 16-битная последовательность поиска: IPX, NetBios, IP.

Последним ищется IP, чтобы как можно меньше изменить поведение.

- 32-битная последовательность поиска: IP, IPX.

Поиск IP выполняется первым, так как он выполняется быстрее, чем по IPX или NetBios.

Последовательность поиска адресов IP

1. Сначала через переменную среды HLS_IPADDR (см. ниже).

2. Если переменная среды не определена, поиск идет через DNS или HOSTS для станции HLSERVER.
3. Если адрес снова не найден, производится широкополосный поиск (255.255.255.255) в локальной субсети.

Для переноса IP пакетов Winsock обращается к соответствующему (16- или 32-битному) файлу WINSOCK.DLL. Обратите внимание, что во время установки Интернета многие клиенты используют свои собственные WINSOCK.DLL (CompuServe, AOL, T-online). В этом случае доступ к IP может привести к набору провайдера Интернета, если HL-Server не будет найден через IPX и NetBios. Вы должны тогда исключить IP из поиска следующим образом:

```
SET HL_SEARCH=IPX,NetBios
```

Для того, чтобы улучшить поиск через TCP/IP, вводится переменная среды HLS_IPADDR. При ее помощи может быть определено один или несколько адресов IP или имен. Однако указав несколько адресов одновременно, в отличие от HL_SEARCH, вы не сможете предсказать, какой из определенных HL-Server в конечном итоге используется.

Пример

```
SET HLS_IPADDR=192.9.209.17,luzie.aladdin.de
```

При желании HLS_IPADDR может использоваться для определения широкоэмитательных адресов:

```
SET HLS_IPADDR=192.9.209.255,192.9.201.255
```

Поскольку IP-сети, в основном, имеют значительно большие различия по времени распространения пакетов, чем сети IPX (например, WAN-маршруты), тайм-ауты и повторы для клиента должны находиться в широких пределах. Значения по умолчанию установлены так, чтобы HL-Server мог быть найден внутри существующего 64 Кбит соединения.

- **SET HLS_WAIT=**

Устанавливает время ожидания между двумя повторами в миллисекундах.

default	TCP/IP:	1000,
	IPX:	200 (*)
min		200
max		30000

- **SET HLS_RETRIES=**

Устанавливает число повторов до выдачи сообщения DONGLE_NOT_FOUND.

default	5
min	2
max	30

Примечание. Значения по умолчанию различны, SET HLS_WAIT изменяет значения для IPX und TCP/IP.

4.3.4 Стратегия поиска

Следующие пункты надо учитывать при указании доступа через HL_LOGIN:

- **HL_LOGIN(MODAD, LOCAL_DEVICE, ...)**

Все локальные порты ищутся без переменной среды. При указании переменной среды может идти поиск только локальных адресов (параллельного и последовательно-го). Расширение поиска через сеть невозможно.

- **SET HL_SEARCH=IPX,278p**

При этом используется только адрес 0x278. IPX игнорируется.

- **SET HL_SEARCH=IPX**

Ключ ни в коем случае не будет найден, так как строка заменяется типом доступа, указанным в HL_LOGIN, и HL_LOGIN выдает код ошибки 15 (INVALID_ENV).

- **HL_LOGIN(MODAD, NET_DEVICE,)**

Система ищет HL-Server по всем поддерживаемым протоколам без переменной среды. При помощи переменной среды вы можете только ограничить используемые для поиска протоколы. Дополнительное расширение поиска на локальных портах невозможно.

- **SET HL_SEARCH=IPX,2f8s**

Используется только протокол IPX. Адрес 2f8s игнорируется.

- **SET HL_SEARCH=278p**

Ключ ни в коем случае не будет найден, так как строка заменяется типом доступа, указанным в HL_LOGIN, и HL_LOGIN выдает код ошибки 15 (INVALID_ENV).

- **HL_LOGIN(MODAD, DONT_CARE,)**

Без переменной среды система сначала ищет на всех локальных параллельных портах. Затем поиск HL-Server продолжается по всем поддерживаемым протоколам. Указанием переменной среды вы можете ограничить поиск по своему желанию.

4.3.5 Замечания

- Если переменная среды не содержит ни одной правильной строки, функция HL_LOGIN возвращает код ошибки 15 (INVALID_ENV).
- Не имеет значения, указана ли переменная среды большой или маленькой буквой.
- Обратите внимание, что переменная среды должна быть введена до начала работы Windows для программ Windows 9x. Последующие изменения в DOS Box не влияют на программы Windows.
- В среде Windows NT вы можете поместить переменные среды в управление системой, вызвав команду **Система/Среда**. Не нужен новый запуск.
- Чтобы осуществить в среде Windows NT корректное использование порта ECP, должна быть установлена строка HL_SEARCH с идентификатором порта **e**, так как аппаратный драйвер ключа запрашивает в NT режим порта не так, как в 9x.

- Указание адреса порта гарантирует поддержку определенного ключа. Например, при указании строки SET HL_SEARCH=320p система будет искать ключ **Hardlock** по адресу порта 0x320. Указание неправильного адреса порта может привести к конфликту.
- Программы, зашифрованные при помощи HL-Crypt (начиная с версии 5.64), HLWCrypt (начиная с версии 4.06) и HLCWin32 (начиная с версии 1.03 и выше), ищут ключ по правилам, описанным выше.
- Порядок поиска на последовательных портах поддерживается только версией API 3.50 и выше.

4.4 Совместимые соглашения о вызовах

Следующие функции старых версий (до API) заново были разработаны из соображений совместимости.

- HL_ON (адрес порта, адрес модуля)
- HL_OFF (адрес порта)
- HL_RD (адрес порта, регистр)
- HL_WR (адрес порта, регистр, значение)
- INT_ON ()
- INT_OFF ()
- K_EYE (порт, указатель данных, число блоков)

Тем самым библиотеки API позволяют использовать расширенные возможности API. Применение возможно только для доступа к локальному ключу.

Пример команд

```
INT_OFF()  
    HL_ON(...)  
        HL_WR(...)  
        HL_RD(...)  
        K_EYE(...)  
    HL_OFF(...)  
INT_ON()
```

Здесь они непосредственно включены в API. При этом различные параметры и функции игнорируются, так как обработка производится на низкоуровневом API.

Если вы разрабатываете новую программу или расширяете существующую систему и хотите использовать все преимущества API, не используйте эти функции. Эти функции были разработаны для исключительных случаев, когда совместимость является очень важной деталью. Не смешивайте новые функции API со старыми функциями.

Примечание. Обратите внимание, что при программировании с использованием нового интерфейса **Hardlock** API функции HL_LOGIN(...) и HL_LOGOUT() не являются аналогичными функциям HL_ON(), HL_OFF(), так как реальная активация ключа происходит автоматически внутри API.

4.5 Модули Hardlock в режиме совместимости

Параллельное вычисление было также применено в API для ключей, которые работают в старом режиме параллельного вычисления. Соответствующая функция ожидает 4 раза по 8 бит в качестве 32-битного аргумента и выдает 8-битное значение.

Пример

```
result = HL_LOGIN (29809,DONT_CARE,»HARDLOCK»,»@0=/  
&#s3");  
IF (result == STATUS_OK)  
    value = HL_CALC (i1, i2, i3, i4);  
ENDIF;  
result = HL_LOGOUT();
```


Время окончания

В ключе Hardlock определяется, до какого времени может использоваться защищенная программа (только с **HL-LiMaS**). Для локальных ключей устанавливается глобальное время выполнения программы, для сетевых – для каждой отдельной позиции. Время окончания работы может быть изменено файлом модификации.

Aladdin License File (ALF) Лицензионный файл Алладина

Файлы .ALF создаются при кодировании сетевых ключей Hardlock для применения в сети и содержат информацию по лицензиям (например, количество лицензий).

API Интерфейс прикладного программирования

Application Programming Interface API – это интерфейс между приложением, которое вы хотите защитить, и Hardlock. Он содержит все функции, необходимые для управления и выполнения запросов к ключу (например, библиотеку Hardlock). При этом преимущество API заключается в том, что запросы для определенного набора функций являются одинаковыми для всех операционных систем. Имена функций идентичны для всех систем (если только они не различны из-за языка программирования).

API разделен на два уровня: **API низкого уровня** и **API высокого уровня**.

API низкого уровня содержит все функции для управления ключом Hardlock. Программные драйверы, зависящие от аппаратных средств и операционной системы, отвечают за установление связи с используемым ключом Hardlock. Программы низкого уровня API доступны как объектные файлы.

API высокого уровня был разработан для более удобной работы с **API**. Он доступен для трансляторов наиболее популярных языков программирования высокого уровня. Функции API высокого уровня, описанные в руководстве, также поддерживают интерфейс программирования API низкого уровня. API высокого уровня представлены в виде исходных тестов.

ASIC Чип

ASIC – это чип, помещенный во все ключи Hardlock, кроме Hardlock SE и Hardlock USB, который проводит шифрование ключа и управляет доступом к памяти Hardlock.

Базовый код

См. *Код*.

Базовый адрес модуля

См. *Адрес модуля*.

Блок-шифр

Шифр кодирования, данные шифруются блоками.

Код

Слово «Код» относится к параметрам, сохраненным в ячейках памяти EEPROM чипа Hardlock, что позволяет алгоритмически отличить один ключ Hardlock от другого. Число различных комбинаций кода Hardlock – 2^{48} . Код состоит из базового кода и субкода.

Базовый код

При программировании ключа с использованием базовой сигнатуры Криптокарты (или Мастер-ключа) может быть создано всего 232 различных комбинаций базового ключа. Каждая базовая сигнатура может использоваться только для создания одного базового ключа.

Субкод

Под Субкодом мы понимаем часть кода, программируемого самим пользователем при использовании Криптокарты. Для каждого из 232 основных кодов доступно 43.680 субкодов (чисто математически получается 65.536 субкодов, но «Аладдин» зарезервировал оставшиеся субкоды для собственных нужд).

Каждый из этих различных субкодов позволяет вам создавать полностью отличающуюся модель шифрования ключа с одним и тем же базовым кодом.

Данные CTV

Данные Customer-To-Vendor или блок данных с информацией по ключу Hardlock, который заказчик считывает из своего ключа и который производитель программного обеспечения передает, чтобы можно было производить файлы модификации для этого ключа.

EEPROM

(Electrically erasable programmable read-only memory) – энергонезависимая электрически перезаписываемая память в ключе Hardlock.

Ключ Hardlock

Защитный модуль, с помощью которого данные зашифровываются и расшифровываются.

High-Level-API – Интерфейс прикладного программирования высокого уровня

См. *API*.

HL-API

См. *API*.

HL-Crypt

Система автоматической защиты уже скомпилированных программных файлов (см. описание *HL-Crypt*). Интерфейс Windows предоставляет программу Espresso для HL-Crypt (см. описание *HL-Bistro*).

HL-DB

Функция для переноса данных, для сохранения в памяти и обработки данных по ключу с распределением заказчиков в банке данных Hardlock, для присоединения некоторых банков данных и т.д. Интерфейс для HL-DB – это Gazzetta (См. *HL-LiMaS*).

HL-LiMaS

Система лицензионного менеджмента для Hardlock. Составляет из HL-RUS (для сублицензирования и создания модификаций) и из HL-DB (для переноса данных). См. *HL-LiMaS*.

HL-LiMaS Modul – Ключ системы лицензионного менеджмента

Красный ключ Hardlock, которой делает возможным применение защитной системы (ранее – ключ HL-Crypt). Возможности зависят от индивидуального лицензирования.

HL-RUS

Remote Update System – система для дифференцированного освобождения отдельных программных файлов и производству файлов модификации для дальнейшего изменения лицензирования. Предпосылкой использования этой функции является наличие ключа Hardlock с памятью. См. HL-LiMaS.

HL-Server

Программа HL-Server позволяет использовать ключ в сети. При этом один ключ на одном сервере делает возможным применение защищенных программ на многих компьютерах-клиентах. Для этого нужно иметь аппаратные средства и установить программу HL-Server на компьютере-сервере. См. описание *HL-Server*.

Кодирование

Перенос сигнатуры, заложенной в аппаратных средствах (Криптокарта, Мастер-Hardlock), в ключ Hardlock. См. *Техническое описание* и описание *HL-Bistro*.

Адрес модуля

Пятизначная последовательность чисел для идентификации ключа Hardlock. Все закодированные «Аладдином» ключи имеют тестовый код с адресом модуля 29809, который меняется при индивидуальном кодировании с помощью криптокарты.

Адрес модуля (15 бит) состоит из базового адреса модуля (10 бит) и из адреса модуля пользователя (5 бит).

При каждом кодировании ключа указывается его адрес. Он не имеет никакого влияния на шифрование защитного модуля и никакого значения для безопасности.

Имеется 2^{15} различных адресов ключей (модулей). Вероятность того, что два ключа двух разных производителей программного обеспечения не смогут быть каскадированы, составляет 1 к 3000. Адрес модуля для

однозначной идентификации может быть дополнен парой ключей Reference-Key/Verify-Key, которые дополнительно проверяют шифрование.

Базисный адрес модуля

Базисный адрес модуля ключа генерируется из базисной сигнатуры Криптокарты (или Мастер-ключа) при программировании ключа и не может быть изменен.

Адрес модуля пользователя

При кодировании можно свободно выбрать массу адресов модулей. Пользователь может назначить до 30 адресов модуля. Оставшиеся комбинации зарезервированы. Адрес модуля пользователя не влияет на поведение ключа при шифровании и, таким образом, не имеет отношения к безопасности.

Модификация	Влияние
Субкод	Одинаковый адрес модуля, разный алгоритм поведения
Адрес модуля пользователя	Разные адреса модуля, одинаковый алгоритм поведения
Адрес модуля пользователя и субкод	Разные адреса модуля, разный алгоритм поведения

См. *Техническое описание* и описание *HL-Bistro*.

Внесение изменений

Внесение части программы в двоичном коде в файл без изменения исходного текста программы.

RAM (Hardlock)

Random Access Memory – область памяти с произвольным доступом, в которую может быть произведена запись. Любая программа, которая использует функции Hardlock-API, может использовать эту область памяти. Эта память является энергонезависимой. См. *Техническое описание*.

Reference Key - Verify Key

Две строки символов для проверки поведения ключа при шифровании. Дополнительно служит для адреса модуля для однозначной идентификации ключа. См. описание *HL-Bistro*, главы *Latteccino*.

ROM (Hardlock)

Read Only Memory. Эта область памяти Hardlock читается любой программой, имеющей функции Hardlock API. Изменение содержимого этой области памяти может быть произведено только при использовании Криптокарты. Эта память является также энергонезависимой. См. *Техническое описание*.

RTB

RUS Transport Block. Блок данных с информацией по модификациям, который является частью файлов VTC CTV, с помощью которых передается информация по модификациям.

RUS

См. *HL-RUS*.

Серийный номер (HL-RUS).

Ключ Hardlock, в котором используется функция HL-RUS по сублицензированию и генерированию модификаций, получает автоматически генерированный серийный номер. Этот номер позволяет однозначно идентифицировать ключ и применить модификацию только на этот ключ.

Примечание. Серийный номер (HL-RUS) независим от серийного номера модуля Hardlock-USB.

Серийный номер (USB)

Ключи Hardlock-USB имеют однозначно понимаемые номера. У этого номера другая функция, отличная от функции HL-RUS.

Сигнатура

Сигнатура – специальная защищенная последовательность чисел, используемая при программировании Hardlock. Она помещается в чип ключа при программировании его с помощью Криптокарты. Логическая матрица чипа ключа проверяет корректность сигнатуры перед модификацией чипа с ее использованием. Сигнатура нужна только для программирования ключа. Она не со-

держится в запрограммированном ключе и не может быть извлечена из данных, хранящихся в ключе. Производитель «Аладдин» использует каждую сигнатуру только один раз. Сигнатура определяет алгоритм поведения ключа Hardlock. Следовательно, очень важно, чтобы Криптокарта хранилась в безопасном месте.

Slot – Позиция

Слоты служат для распределения сублицензий. Они закладываются при кодировании ключей Hardlock и при шифровании программ распределяются по разным программным компонентам. С ними возможно дифференцированное сублицензирование, предоставление отдельных функций программного пакета.

Поточный шифр

Шифр кодирования, при котором данные шифруются последовательно, бит за битом.

Subcode Субкод

См. Код.

Сублицензирование

Дифференцированное предоставление функций программного пакета, с указанием времени окончания работы, счетчиков и распределения слотов.

Включить

Параметр, с помощью которого запускается программа командной строки, например HL-Crypt.

Ключ для тестирования

Ключ Hardlock с адресом модуля 29809. Модуль для тестирования шифрует программы с целью тестирования. Кроме того, этот ключ тестирует функцию модификации (HL-LiMaS).

Файл модификации

Файл, который может быть применен для ключа, чтобы изменить лицензирование (только с HL-LiMaS).

Vendor-Key

Личный ключ, который защищает модификации и информацию по сублицензиям. Он генерируется менеджером Vendor-Key и защищен паролем (см. описание *HL-Bistro*). Эти ключи генерируются, если приобретен HL-LiMaS.

Verify Key

См. Reference Key.

VTC

Vendor-To-Customer. Файл или блок данных с информацией по модификациям, которые производитель программного обеспечения передает заказчику, чтобы провести модификацию ключа.

Счетчик

При шифровании программы устанавливается, сколько раз может быть запущена программа, при кодировании ключа устанавливается высшая граница допустимого числа запусков (только с HL-LiMaS).