УДК 681.3.045

В.В. Гоннов

ОБЕСПЕЧЕНИЕ ЗАЩИТЫ ВЕРХНИХ УРОВНЕЙ ПЕРЕДАЧИ ДАННЫХ В СИСТЕМАХ ИНТЕЛЛЕКТУАЛЬНОГО ЗДАНИЯ*

В настоящее время системы интеллектуальных зданий (ИЗ) приобретают все большую популярность. Данные системы позволяют управлять комплексно различными аспектами функционирования здания, такими как охранная и пожарная сигнализации, микроклимат, освещение и т.д. Однако существуют серьезные проблемы, связанные с безопасностью обмена данными в таких системах. Злонамеренные действия в системе управления ИЗ создают угрозы не только для безопасности информации, но и для имущества и жизни людей. При этом большинство систем управления ИЗ не содержат средств защиты информации. Поэтому особенно актуальной является задача разработки методов защиты данных от несанкционированного просмотра, изменения и удаления. *Цель данной работы состоит в разработке защищенных протоколов передачи данных в сетях интеллектуальных зданий и формальной оценке уровня их безопасности.*

Системы ИЗ имеют различную архитектуру, которая, в существенной мере, определяет методы их защиты. Наиболее распространенной является архитектура, в рамках которой системы ИЗ содержат несколько уровней взаимодействия, в частности уровень сенсорной сети, уровень сети контроллеров, уровень сети станций. На уровне сенсорной сети оконечным оборудованием являются датчики (сенсоры), которые выполняют регистрацию некоторых физических параметров (температуры, задымления и т.д.), считыватели (смарт-карт, клавиатуры и т.д.) и исполнительные устройства (замки, камеры и т.д.). Протоколы взаимодействия таких устройств чаще всего реализуются их производителем и не могут быть изменены для улучшения безопасности. Примерами протоколов сенсорных сетей для проводных систем являются RS-485, LIN, для беспроводных – ZigBee. Таким образом, на данном уровне взаимодействия необходимо полагаться на средства безопасности, уже заложенные в устройствах. Уровни сети контроллеров и сети станций обеспечивают взаимодействие между программируемыми контроллерными устройствами и персональными компьютерами, выполняющими сбор данных, их хранение и обработку. В данном случае для передачи данных используются такие протоколы, как CAN, а также протоколы на основе TCP/IP. CAN (Control Area Network) – последовательная магистраль, обеспечивающая увязку в сеть "интеллектуальных" устройств ввода/вывода, датчиков и исполнительных устройств некоторого механизма или даже предприятия. Характеризуется протоколом, обеспечивающим возможность нахождения на магистрали нескольких ведущих устройств, обеспечивающим передачу данных в реальном масштабе времени и коррекцию ошибок, высокой помехоустойчивостью. Система CAN обеспечена большим количеством микросхем, обеспечивающих работу подключенных к магистрали устройств, разработку которых начинала фирма BOSH для использования в автомобилях, а затем и в других отраслях промышленности. Протокол CAN использует оригинальную систему адресации сообщений. Каждое сообщение снабжается идентификатором, который определяет назначение передаваемых данных,

* Работа выполнена при поддержке грантов РФФИ №07-07-00138-а, №08-07-00117-а.

но не адрес приемника. Любой приемник может реагировать как на один идентификатор, так и на несколько. На один идентификатор могут реагировать несколько приемников. Проблема защиты протоколов данного уровня представляется наиболее актуальной.

В качестве примера системы с подобной архитектурой можно привести систему scalaBACS (scalable Building Automation & Control System) фирмы Datamicro. Комплекс ScalaBACS представляет собой трехуровневую распределенную сеть:

1. 3-й уровень (*серверная сеть)* – серверы, рабочие и операторские станции, объединенные Ethernet сетью. Обеспечивает информационное обеспечение комплекса (базы данных, программы конфигурирования и т.д.) в пределах здания или совокупности зданий.
2. 2-й уровень *(сеть контроллеров)* – контроллеры и пульты, подключаемые по CAM-сети. Обеспечивает связь контроллеров в пределах здания.
3. 1-й уровень *(сенсорная сеть)* – контроллеры, подключаемые по сенсорным сетям (LIN, RS-485, ZigBee и др.). Обеспечивает связь с датчиками в пределах помещения и/или этажа здания.

На рис. 1 приведена общая схема сети scalaBACS.

Из рисунка видно, что CAN-контроллеры взаимодействуют с серверами на основе ПЭВМ для передачи данных и получения управляющих команд. Взаимодействие осуществляется частично по сети CAN (до шлюза), частично по сети TCP/IP. При этом выдвинем следующие требования к безопасности данного взаимодействия:

1. Взаимная аутентификация контроллеров и серверов.
2. Защита от изменения передаваемых данных. Злоумышленники могут использовать изменение информации и команд контроллеров CAN для несанкционированного открытия дверей (например, в целях хищения), блокирования дверей, включения средств пожаротушения и т.д.
3. Защита от несанкционированного просмотра.
4. Защита от атак типа «man-in-the-middle».

Данные требования могут быть реализованы с использованием криптографических протоколов аутентификации, распределения ключей и криптографической защиты передаваемых данных. Однако одной из особенностей системы является низкая производительность и малый объем памяти контроллеров. Кроме того, обработка данных серверами должна происходить в реальном масштабе времени, что также требует использования не ресурсоемких алгоритмов защиты информации.

Для решения описанных проблем предлагается разработка протоколов аутентификации и защищенной передачи данных, на основе направленной модификации алгоритмов защиты информации, используемых в сетях TCP/IP. С учетом требования 1) и 2) для целей взаимной аутентификации устройств и распределения ключей предлагается использовать протокол аутентификации на основе криптосистемы RSA. Каждый контроллер содержит пару – секретный ключ контроллера *Priv_cnt(i)*, где *i* – номер контроллера и открытый ключ сервера *Pub_serv*, сервер системы содержит набор открытых ключей для каждого контроллера *[Priv_cnt(1)… Priv_cnt(N)]*, где *N* – количество контроллеров и собственный секретный ключ *Priv_serv*. MD5(.) – примитив вычисления криптографической контрольной суммы, RSAE(.) – примитив шифрования на алгоритме RSA, RSAD(.) – примитив расшифрования на алгоритме RSA. При включении *i*-го контроллера в сеть выполняется протокол аутентификации, состоящий из четырех этапов.

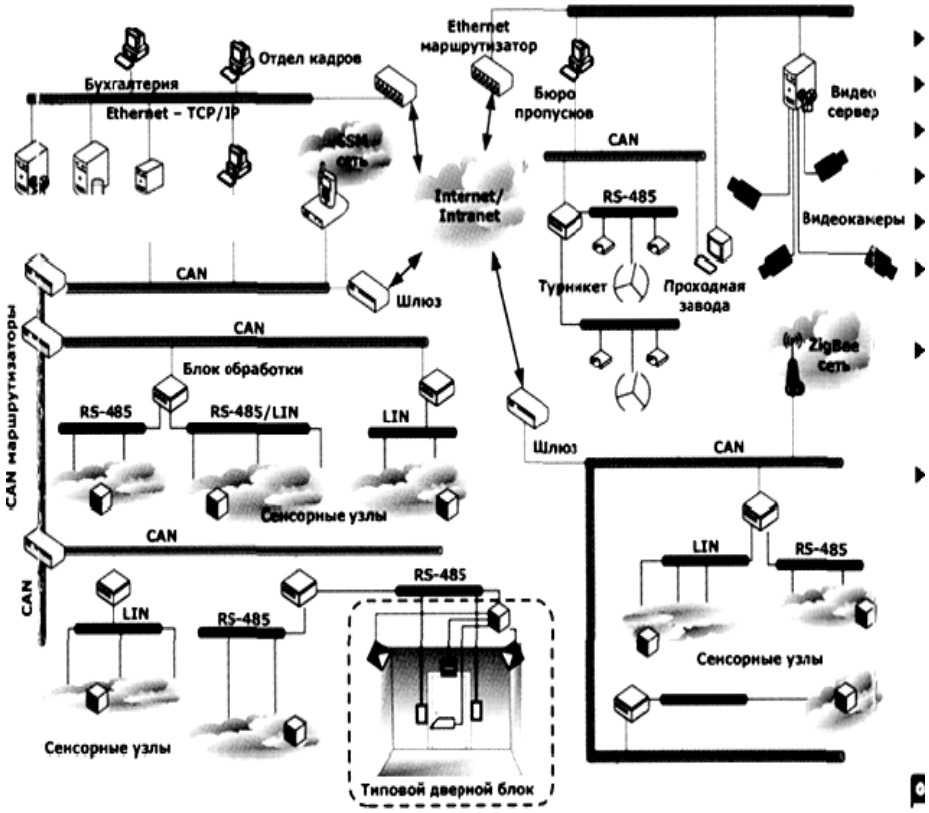

Рис. 1. Схема системы scalaBACS

Запрос доступности

a. Контроллер генерирует команду MSG='AVLBL' и число SALT=rand();
b. PACK=MSG|SALT|MD5(MSG).
c. EPACK=RSAE(PACK, Priv_cnt(i)).
d. Пакет EPACK отправляется серверу.

Инициация протокола Диффи-Хеллмана

1. Сервер получает пакет EPACK и определяет номер отправителя i.
2. PACK=RSAD(EPACK, Pub_cnt(i)).
3. В случае если MSG=='AVLBL' и MD5(MSG) проверена правильно, сервер генерирует команду MSG='KEYNEG' и число DHHKEY – половину ключа Диффи-Хеллмана.
4. PACK=MSG|DHHKEY|SALT|MD5(MSG);
5. EPACK=RSAE(PACK, Priv_serv).
6. Пакет EPACK отправляется контроллеру.

Завершение протокола Диффи-Хеллмана

1. Контроллер выполняет PACK=RSAD(EPACK, Pub_serv).

2. В случае если MSG==’KEYNEG’, SALT равна отправленной ранее и MD5(MSG) верна контроллер формирует MSG==’KEYNEG’ и число DHLKEY – половину ключа Диффи-Хеллмана.

3. PACK=MSG|DHHKEY|SALT|MD5(MSG);

4. EPACK=RSAE(PACK, Priv_cnt(i)).

5. Пакет EPACK отправляется серверу.

6. Сервер обеспечивает получение пакета EPACK и определение идентификатора отправителя.

7. PACK=RSAD(EPACK, Pub_cnt(i)).

8. Если MSG==’KEYNEG’, SALT равна отправленной ранее и MD5(MSG) верна сервер формирует MSG==’KEYDONE’.

9. PACK=MSG||SALT|MD5(MSG);

10. EPACK=RSAE(PACK, Priv_serv).

11. Пакет EPACK отправляется контроллеру.

Формирование общих ключей

1. Контроллер выполняет PACK=RSAD(EPACK, Pub_serv).

2. В случае если MSG==’KEYDONE’, SALT равна отправленной ранее и MD5(MSG) верна контроллер получает из DHHKEY и DHLKEY общий ключ SYMKEY.

3. Одновременно сервер получает общий ключ SYMKEY.

После окончания данного протокола контроллер и сервер будут обладать одним общим криптографическим ключом, который может быть использован для защищенной передачи сообщений. Необходимо учитывать, что ключи RSA в данном случае не хранятся и не передаются открыто, поэтому с учетом низкой производительности контроллеров можно снизить длину ключей RSA до 512 бит.

Для шифрования передаваемой информации предлагается использование алгоритма Blowfish. Данный алгоритм прост в реализации, обладает высокой производительностью. Для ускорения шифрования предлагается использовать Blowfish с уменьшенным количеством раундов и укороченным ключом. Исследования показывают, что для передачи оперативной информации достаточно использования алгоритма с четырьмя раундами шифрования и ключом не более 128 бит.

Приведенные протоколы позволяют решить задачи защиты информации, актуальные для систем ИЗ, в частности, защиты взаимодействия между контроллерами и серверами системы scalaBACS. Поскольку к безопасности алгоритмов в таких системах предъявляются высокие требования, следующим этапом работ станет формальный анализ безопасности разработанных протоколов. В качестве средства верификация может быть использован символьный верификатор системы SMV, который ранее использовался для верификации различных критичных по безопасности протоколов.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Шнайер Б.* Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си. – М.: Изд-во ТРИУМФ, 2003. – 816 с.

2. *Молдовян Н. А.* Криптография: от примитивов к синтезу алгоритмов / Н. А. Молдовян, А. А. Молдовян, М. А. Еремеев. – СПб.: БХВ-Петербург, 2004. – 448 с.