УДК 655.3.02

М. С. Шмаков, А. Н. Кошевая
Белорусский государственный технологический университет

РАЗРАБОТКА АЛГОРИТМА ГЕНЕРАЦИИ ВРЕМЕННЫХ КОДОВ ДЛЯ ИДЕНТИФИКАЦИИ ПОДЛИННОСТИ ПРОДУКТА

Разработан алгоритм генерации временных кодов для маркировки упаковки продукции. Осуществлена его программная реализация. С учетом временного фактора, который выражается в том, что код имеет определенный срок действия, разработанный алгоритм является эффективным инструментом для подтверждения подлинности продукта. Оптимальная длина кода (не более 8 разрядов) обеспечивает удобство ввода в специализированную форму покупателем. Программа предусматривает возможность уменьшения числа разрядов при сокращении ассортимента маркируемой продукции. Код формируется из цифр и букв, что в перспективе обеспечивает широкую базу применения для различных ассортиментных единиц, выпускаемых белорусскими предприятиями.

Разработанный алгоритм включает в себя блоки проверки генерируемых кодов на повторяемость и вырожденность. В зависимости от сроков хранения продукции, доступ к идентифицирующей базе (веб-платформа) будет осуществляться с определенным временным лагом, что позволит исключить вероятность ошибки вывода.

Небольшое число разрядов минимизирует расход чернил при последующем нанесении кода на упаковку. Этот фактор также является важным элементом экономической целесообразности внедрения разработки в производственный процесс. С точки зрения поддержания имеющейся высокой репутации белорусских производителей данная разработка обеспечит защиту от подделок со стороны недобросовестных конкурентов. А также повысит лояльность со стороны потребителей к защите собственных интересов с использованием актуальных высокотехнологичных разработок.

Ключевые слова: контрафакт, защита, криптография, подделка, упаковка, печать, программирование, генератор случайных чисел, подлинность.

M. S. Shmakov, A. N. Koshevaya
Belarusian State Technological University

THE DEVELOPMENT OF AN ALGORITHM GENERATING TIME CODE TO IDENTIFY THE AUTHENTICITY OF PRODUCTS

The algorithm was developed to generate time code to the packaging of products. Implemented its software implementation. Taking into account the time factor, expressed that the code has a term of validity, the developed algorithm is an effective tool to confirm the authenticity of the product. Optimal length code (8 bits) provides the convenience of entry to a special form by the buyer. The program provides the possibility of reducing the number of digits in the reduction of the range of labeled products. Code formed from numbers and letters, which in the future provides a broad base of applications for different product units manufactured by the Belarusian enterprises.

The algorithm includes blocks of generated verification codes to the frequency and the degeneracy. Depending on the shelf life of products, access to identified database (web platform) will be carried out with a certain time lag, thus eliminating the probability of error of the output.

A small number of bits minimizes the consumption of ink during the subsequent application of the code on the packaging. This factor is also an important element of economic feasibility of implementing development in the production process. From the point of view of maintaining the existing high reputation of Belarusian manufacturers, the introduction of this innovation will ensure to shield against counterfeiting by unscrupulous competitors. As well as increase loyalty from consumers, showing interest and willingness to protect their own interests, using relevant high-tech developments.

Key words: counterfeit, protection, cryptography, forgery, packaging, printing, programming, random number generator, authenticity.

Введение. Основная идея разрабатываемой системы верификации подлинности продукта состоит в генерации временных идентификационных кодов. Эти коды будут использоваться покупателями для того, чтобы подтвердить подлинность продукта.

Основная часть. Общая схема процесса:

— при упаковывании продукта наносится сгенерированный уникальный временный код (вместе с обязательной информацией о дате выпуска продукта);

— этот код вносится в базу данных специализированного сайта, где покупатель может убедиться в подлинности продукта;

— при покупке продукта покупатель, используя электронное устройство с функцией мобильной передачи данных, может зайти на специализированный сайт и ввести код с упаковки;

— покупателю будет выдана информация о месте производства, дате (временном промежутке) выпуска продукции, а также фото упаковки;

— ограниченное время действия кода (например, 7 дней) исключает возможность его подделки.

Общий вид интерфейса должен быть максимально простым и понятным потребителю. Минимализм в интерфейсе существенно упрощает последующую адаптацию оболочки к различным типам экранов.

Управление идентификационными ключами. Управление ключами — информационный процесс, включающий в себя три элемента:

— генерацию ключей;

— накопление ключей;

— распределение ключей.

Комбинация (идентификационный ключ) должна соответствовать следующим требованиям:

— быть уникальной;

— иметь максимально короткую длину (в символах) для удобства ввода и проверки пользователем;

— иметь случайный характер формирования для исключения подбора.

Системный подход при проектировании представляет собой комплексное, взаимосвязанное, пропорциональное рассмотрение всех факторов, путей и методов решения сложной многофакторной и многовариантной задачи проектирования интерфейса взаимодействия. В отличие от классического инженерно-технического проектирования при использовании системного подхода учитываются все факторы проектируемой системы — функциональные, психологические, социальные и даже эстетические [1].

Автоматизация управления неизбежно влечет за собой осуществление системного подхода, так как она предполагает наличие саморегулирующейся системы, обладающей входами, выходами и механизмом управления. Уже само понятие системы взаимодействия указывает на необходимость рассмотрения окружающей среды, в которой она должна функционировать.

Таким образом, система взаимодействия должна рассматриваться как часть более обширной системы — АСУ реального времени, тогда как последняя — в качестве системы управляемой среды.

В настоящее время можно считать доказанным, что главная задача проектирования интерфейса пользователя заключается не в том, чтобы рационально «вписать» человека в контур управления, а в том, чтобы, исходя из задач воздействия на объект, разработать систему взаимодействия двух равноправных партнеров (человек-оператор и аппаратно-программный комплекс АСУ), рационально регулирующих объект управления.

Интерфейс пользователя можно разделить на пакетный и интерактивные. Пакетные характеризуются тем, что пользователь должен сформировать пакет с заданиями, затем программа эти задания выполняет и выдает результат. Интерактивные отличаются тем, что пользователь в ходе работы программы постоянно с ней взаимодействует. Выделяется еще интерфейс на базе меню, псевдографический и интерактивно-командный интерфейсы. Эти интерфейсы описывать не будем, так как программа, которую необходимо создать по заданию, требует графический интерфейс. Его следует выбирать только для многомодульных программ, которые предположительно будут иметь массовое применение. В соответствии с заданием и для большего удобства и понятия принципа работы программы интерфейс должен содержать наименьшее количество кнопок. Вследствие чего выберем одно поле, в котором будем генерировать сам код, одну кнопку, после нажатия которой будет происходить генерация, и кнопку выхода [2].

Алгоритм генерации идентификационных ключей. Функция генерации ключа реализована следующим образом.

При запуске программы происходит проверка наличия журнала кодов, если он существует, программа выводит путь к нему. При нажатии на кнопку генерировать, программа определяет введенную длину кода (если она не введена, длина автоматически будет составлять 8 символов), выбранные для генерации символы и выбранные проверки.

Программа определяет, какие группы символов выбраны для генерации. Формирует из них строку алфавита ключа и передает ее и длину ключа как входные значения для функции генерации. Формируется случайное число в пределах мощности алфавита ключа, из алфавита выбирается символ, стоящий на позиции, соответствующей случайному числу, и добавляется в строку ключа. Так происходит, пока количество символов в ключе не достигнет значения длины ключа, заданного пользовате-

лем. После достижения длины ключа, равной введенному значению, происходит выход из функции [3].

Функции проверки ключа. Если проверка по журналу не выбрана, то программа переходит к проверке по словарю. Иначе программа построчно считывает ключи, находящиеся в журнале, и сравнивает с проверяемым ключом. Если совпадений не найдено, то программа переходит к проверке по словарю и счетчик увеличивается на 1. Если словарь не указан, то выдается сообщение о необходимости указать словарь или отключить проверку по нему. Если словарь указан, то программа открывает его и построчно сравнивает сгенерированный или введенный ключ с ключами, находящимися в словаре. Если совпадений не найдено, то счетчик увеличивается 1 и программа переходит к проверке на вырожденность. При проверке на вырожденность в программе подсчитывается количество совпадающих символов, и если оно больше 1, то ключ считается вырожденным. Если после завершения проверок счетчик равен 3, выдается сообщение об успешности проверок, иначе выдается сообщение о том, что проверка не прошла [4].

При создании формы программы после ее запуска вызывается функция FormCreate:

```
void __fastcall TForm1::FormCreate(TObject *Sender)
{Char buffer[255];
String s;
GetCurrentDirectory(sizeof(buffer),buffer);
s=buffer;
s=s+"\\jornal.txt";
std::ofstream out("jornal.txt",std::ios::in);
if (out) {
Edit1->Text=s;
}}
```

Функция пытается открыть файл журнала, если он не существует, то отображает в окне программы путь к файлу журнала.

При нажатии в программе на кнопку «Генерировать» вызывается функция Button3Click:

```
void __fastcall TForm1::Button3Click(TObject *Sender)
{String DlP;  String par;
DlP=Edit3->Text;
int dl;
dl=DlP.ToIntDef(8);
if (dl>8) {
    dl=8;
    Application->MessageBox (L"Недопустимая длина ключа! Длина ключа установлена в 8 символов.", L"Генератор кодов", MB_OK|MB_ICONINFORMATION);
    Edit3->Text=dl;}
String sim="";
if (CheckBox1->Checked==true) {
sim+="ASDFGHJKLZXCVBNMQWERTYUIOP";
}
if (CheckBox2->Checked==true) {
  sim+="0123456789";}
if (CheckBox3->Checked==true) {
  sim+="asdfghjklzxcvbnmqwertyuiop";}
if (CheckBox4->Checked==true) {
  sim+="~`!@#$%^&*()_-+={}[]:;<,>.?\/|";}
if (sim.Length()<1) {
    Application->MessageBox(L"Не указаны символы для генерации!", L"Генератор кодов", MB_OK|MB_ICONERROR);
}
else {
    par=Generation(sim,dl);
    Edit4->Text=par;
}
int cheker=0; int prov_num=0;
if (CheckBox5->Checked==true) {
    String path_j;
    prov_num++;
    path_j=Edit1->Text;
    if (path_j.Length()<1) {
      Application->MessageBox(L"Не указан журнал!", L"Проверка по журналу", MB_OK|MB_ICONERROR);
    } else {
            if (Prov_jorn(path_j,par)==0) {
            cheker++;
            Application->MessageBox(L"Нет совпадений с журналом.", L"Проверка по журналу", MB_OK|MB_ICONINFORMATION);
    }}}
    if (Prov_slov(path_s,par)==1) {
      Application->MessageBox(L"Невозможно открыть словарь!", L"Проверка по словарю", MB_OK|MB_ICONERROR);
    }}}
if (CheckBox7->Checked==true) {
            prov_num++;
            if (Prov_vir(par)==0) {
                cheker++;
                Application->MessageBox(L"Ключ не вырожденный", L"Проверка на вырожденность", MB_OK|MB_ICONINFORMATION);
            }
            if (cheker==prov_num) {
    String path_j;
    path_j=Edit1->Text;
    if (path_j.Length()<1) {
```

```
        Application->MessageBox(L"Не указан журнал!", L"Запись в журнал", MB_OK|MB_ICONERROR);
      } else {
        if (Jorn_zap(par,path_j)==true){
        Application->MessageBox(L"Ключ добавлен в журнал", L"Запись в журнал", MB_OK|MB_ICONINFORMATION);
      } else {
            Application->MessageBox(L"Не удалось открыть журнал!", L"Запись в журнал", MB_OK|MB_ICONERROR);
      }}}}
```

Функция считывает введенную пользователем длину ключа, проверяет соответствие введенной длины ключа максимальной и минимальной допустимой длине, определяет выбранные для генерации группы символов. Формирует из выбранных групп символов строку алфавита ключа. Вызывает функцию генератора Generation, предавая как входные значения длину ключа и строку алфавита. После определяется, какие были выбраны проверки, и вызываются соответствующие функции. При успешном прохождении выбранных проверок сгенерированный ключ добавляется в журнал.

Для генерации ключа используется функция Generation:

```
String Generation(AnsiString sim, int dl)
{AnsiString par;
int ln; ln=sim.Length();
for (int i = 0; i < dl; i++) {
par+=sim[random(ln)+1];
}
return par;}
```

В качестве входного значения в функцию передается необходимая длина ключа и строка алфавита ключа. Функция определяет длину строки символов для генерации ключа. В цикле от 1 до длины ключа функция получает случайное число в диапазоне от 1 до длины строки символов для генерации ключа, выбирает из этой строки символ из позиции с соответствующим номером и добавляет символ в строку пароля.

Функция проверки по журналу Prov_jorn:

```
int Prov_jorn(AnsiString path, AnsiString pr)
{
std::ifstream in(path.c_str(),std::ios::in);
      char ch[255];
      String r;   int k=0,d=0;
      if (!in)
      {return 1;}
      else d++;
      if(d!=0)
      {
 while (! in.eof ())
{ in>>ch;
  r=ch;
        if (r==pr)
        {k++;
        break;}
        }
        in.close();
        if (k==0) {
              return 0;
        }  else { return 2;}
}}
```

Функция получает входящим параметром проверяемый пароль и путь к журналу. Из файла журнала построчно считываются ключи и сравниваются с проверяемым ключом. Если совпадений не было функция возвращает 0, если совпадения были — 2, если журнал не найден — 1.

Количественная оценка стойкости парольной защиты. Мерой стойкости паролей традиционно является энтропия — мера неопределенности, измеряемая обычно в битах. Энтропия в 1 бит соответствует неопределенности выбора из двух паролей, в 2 бита — из 4 паролей, в 3 бита — из 8 паролей и т. д. Энтропия в N бит соответствует неопределенности выбора из паролей.

Стойкость того или иного пароля должна рассматриваться только в контексте конкретной системы парольной аутентификации: пароль, являющийся стойким для одной системы, может оказаться совершенно не стойким при использовании другой. Это происходит из-за того, что разные системы в разной степени реализуют (или вовсе не реализуют) механизмы противодействия атакам, направленным на взлом паролей, а также потому, что некоторые системы содержат ошибки или используют ненадежные алгоритмы [5].

Основной способ противодействовать взлому паролей — искусственно замедлить процедуру их проверки. Действительно, займет ли проверка 10 наносекунд или 10 миллисекунд — для пользователя разница будет совершенно незаметной, а с точки зрения взлома скорость упадет очень существенно — со 100 миллионов до 100 паролей в секунду. Замедление обычно достигается за счет многократного вычисления криптографических функций, причем эти вычисления построены таким образом, чтобы атакующая сторона не могла проверить пароль без повторения вычислений (то есть недостаточно просто добавить вызов Sleep в процедуре проверки пароля). Впервые такой вариант был предложен в 1997 г. в работе Secure Applications of Low-Entropy Keys [6].

Заключение. Разработанные методы кодирования предоставляют возможность выбора нескольких вариантов защиты с использованием различной разрядности защитного кода. Разработанный алгоритм позволяет производителю продукции, основываясь на данных о производстве, а также на собственных субъективных требованиях, подобрать подходящий метод кодирования.

Данный алгоритм с учетом временного фактора защищает производителя от подделки и дает потенциальному покупателю возможность легко определить подлинность продукта.

Литература

1. Баричев С. С., Гончаров В. В. Основы современной криптографии. М.: Горячая Линия, 1997. 176 с.

2. Грушо А. А., Тимонина Е. Е. Теоретические основы защиты информации. М.: Яхтсмен, 1996. 112 с.

3. Домашев А. В., Щербаков А. Ю. Программирование алгоритмов защиты информации. М.: Нолидж, 2000. 288 с.

4. Варфоломеев А. А., Жуков А. Е. Блочные криптосистемы. Основные свойства и методы анализа стойкости. М.: ПАИМС, 1998. 224 с.

5. Спесивцев А. В. Защита информации. М.: Радио и связь, 1992. 278 с.

6. Ростовцев А. Г., Матвеев В. А. Защита информации в компьютерных системах. Элементы криптологии. СПб.: СПбГТУ, 1993. 424 с.

References

1. Barichev S. S., Goncharov V. V. *Osnovy sovremennoy kriptografii* [Basics of modern cryptography]. Moscow, Goryachaya Liniya Publ., 1997. 176 p.

2. Grusho A. A., Timonina E. E. *Teoreticheskie osnovy informatsii* [Theoretical foundations of information security]. Moscow, Yachtsmen Publ., 1996. 112 p.

3. Domashov A. V., Scherbakov A. Yu. *Programmirovanie algoritmov zashchity informatsii* [Programming algorithms of information protection]. Moscow, Nolidzh Publ., 2000. 288 p.

4. Varfolomeev A. A., Zhukov A. E. *Blochnye kriptosistemy. Osnovnye svoystva i metody analiza stoykosti* [Block cryptosystems. The basic properties and methods of analysis stand-bone]. Moscow, PAIMS Publ., 1998. 224 p.

5. Spesivtsev A. V. *Zashchita informatsii* [Information protection]. Moscow, Radio i svyaz' Publ., 1992. 278 p.

6. Rostovtsev A. G., Matveev V. A. *Zashchita* informatsii *v komp'yuternykh sistemakh. Elementy kriptologii* [Protection of information in computer systems. Elements of cryptology]. St. Petersburg, SPbGTU Publ., 1993. 424 p.

Информация об авторах

Шмаков Михаил Сергеевич – кандидат технических наук, доцент, заведующий кафедрой полиграфического оборудования и систем обработки информации. Белорусский государственный технологический университет (220006, г. Минск, ул. Свердлова, 13а, Республика Беларусь). E-mail: Shmakov@belstu.by

Кошевая Алеся Николаевна − ассистент кафедры полиграфического оборудования и систем обработки информации. Белорусский государственный технологический университет (220006, г. Минск, ул. Свердлова, 13а, Республика Беларусь). E-mail: Koshevaya@belstu.by

Information about the authors

Shmakov Mikhail Sergeevich – PhD (Engineering), Associate Professor, Head of the Department of Printing Equipment and Information Processing Systems. Belarusian State Technological University (13a, Sverdlova str., 220006, Minsk, Republic of Belarus). E-mail: Shmakov@belstu.by

Koshevaya Alesia Nikolaevna − assistant lecturer, the Department of Printing Equipment and Information Processing Systems. Belarusian State Technological University (13a, Sverdlova str., 220006, Minsk, Republic of Belarus). E-mail: Koshevaya@belstu.by

Поступила 14.08.2017