ЛАБОРАТОРНАЯ РАБОТА №2

МЕТОДЫ И СРЕДСТВА ПРИВЯЗКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ К АППАРАТНОМУ ОКРУЖЕНИЮ И ФИЗИЧЕСКИМ НОСИТЕЛЯМ

1. ЦЕЛЬ РАБОТЫ

Изучить способы взаимодействия программного обеспечения с операционной системой и аппаратурой для реализации привязки ПО к текущей аппаратно-программной конфигурации ЭВМ.

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Для защиты программного обеспечения от несанкционированного копирования и нелегального использования применяется привязка его к аппаратному окружению. Под привязкой понимается некоторая информация, позволяющая уникально идентифицировать пользовательское рабочее место. Под такой информацией обычно понимаются различные параметры системы в зашифрованном виде. Вот лишь некоторые виды привязки:

- привязка к имени компьютера пользователя;
- привязка к имени пользователя;
- привязка к Windows ProductID;
- привязка к аппаратному идентификатору ключа eTokenID;
- привязка к аппаратному обеспечению;
- привязка к идентификатору процессора;
- привязка к идентификатору жесткого диска.
- привязка к версии BIOS

В рамках данной лабораторной работы мы будем рассматривать лишь те характеристики, которые относятся к аппаратному обеспечению. Как же получить данные об аппаратном окружении?

Данные реестра

Информация о многих характеристиках «железа» хранится в реестре Widows. Так, например там можно найти информацию о версии BIOS (**HKEY_LOCAL_MACHINE\\HARDWARE\\DESCRIPTION\\System\\SystemBiosVersion**), серийных номерах жестких дисков и о многом другом. Привязка на основе этих данных не будет надежной, так как их изменение не является затруднительным.

API

Windows API (англ. application programming interfaces) — общее наименование целого набора базовых функций интерфейсов программирования приложений операционных систем семейств Windows и Windows NT корпорации «Майкрософт». Является самым прямым способом взаимодействия приложений с Windows. Для создания программ, использующих Windows API, «Майкрософт» выпускает SDK, который называется Platform SDK и содержит документацию, набор библиотек, утилит и других инструментальных средств.

Windows API является достаточно низкоуровневым средством общения с операционной системой и вполне подходит для получения параметров системы с последующей привязкой ПО к оборудованию.

Для получения сведений о накопителях можно использовать следующую API функцию [7]:

```
BOOL WINAPI GetVolumeInformation(
  __in_opt   LPCTSTR lpRootPathName,
  __out      LPTSTR lpVolumeNameBuffer,
  __in       DWORD nVolumeNameSize,
  __out_opt  LPDWORD lpVolumeSerialNumber,
  __out_opt  LPDWORD lpMaximumComponentLength,
  __out_opt  LPDWORD lpFileSystemFlags,
  __out      LPTSTR lpFileSystemNameBuffer,
  __in       DWORD nFileSystemNameSize
);
```
Ознакомиться с ее описанием можно по адресу

WMI

Windows Management Instrumentation (WMI) в дословном переводе – это инструментарий управления Windows. Если говорить более развернуто, то WMI – это одна из базовых технологий для централизованного управления и слежения за работой различных частей компьютерной инфраструктуры под управлением платформы Windows. Технология WMI – это расширенная и адаптированная под Windows реализация стандарта WBEM, принятого многими компаниями, в основе которого лежит идея создания универсального интерфейса мониторинга и управления различными системами и компонентами распределенной информационной среды предприятия с использованием объектно-ориентированных идеологий и протоколов HTML и XML.

Важной особенностью WMI является то, что хранящиеся в нем объекты соответствуют динамическим ресурсам, то есть параметры этих ресурсов постоянно меняются, поэтому параметры таких объектов не хранятся постоянно, а создаются по запросу потребителя данных. Хранилище свойств объектов WMI называется репозиторием и расположено в системной папке операционной системы Windows:

```
%SystemRoot%\System32\WBEM\Repository\FS
```

Windows Management Instrumentation - это мощный инструмент для администрирования операционных систем семейства Windows с помощью скриптов. С помощью WMI можно управлять устройствами, учетными записями, сервисами, процессами, сетевыми интерфейсами и другими программами, которые расширяют базовую структуру WMI своими классами.

Помимо скриптов WMI может применяться и в полноценных программах, то есть программисты могут внедрять запросы к WMI в исполняемый код, завязывая свою программу с работой WMI, благодаря чему программа становится проще и легче, но зависимой от правильности работы Windows Management Instrumentation.

Покажем, как использовать WMI на примере получения серийных номеров жестких дисков с интерфейсом IDE на языке C#:

```
ManagementObjectSearcher mos = new ManagementObjectSearcher("SELECT * from
Win32_DiskDrive where InterfaceType = 'IDE'");

ManagementObjectCollection moc = mos.Get();

ArrayList IDE_ID = new ArrayList();
```

```
foreach (ManagementObject mo in moc)

{     IDE_ID.Add((mo.Properties["PNPDeviceID"].Value.ToString()));

}
```

Привязка данных

Теперь, когда данные об аппаратном окружении получены, необходимо решить, в каком виде и где их хранить. Хранение этих данных обычно выполняется в шифрованном виде. Для этого используются методы симметричного и ассиметричного шифрования.

Ключевую информацию можно хранить в отдельных файлах. В случае ассиметричного шифрования открытый и закрытый ключи могут быть «вшиты» в программу. Но это опасно возможностью подмены их злоумышленниками.

Чтобы обеспечить некоторую надежность от подмены ключей шифрования лучше всего использовать удаленный сервер, который и будет производить ключевую информацию на основе полученных данных. В качестве HTTP и FTP сервера можно использовать готовое и настроенное решение на базе CentOS Linux [8] – в этом случае программа выполняет запрос на сервер, далее сервер выполняет некоторые заданные преобразования информации (например, с помощью PHP) и возвращает ответ – готовую ключевую информацию. При этом программа при каждом запуске будет заниматься сбором сведений о системе и сравнением ключевой информации.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Получить вариант задания у преподавателя.
2. Разработать программу.
3. Продемонстрировать выполнение программы преподавателю, сравнить полученный результат с ожидаемым.
4. Оформить и защитить отчет.

5. ВАРИАНТЫ ЗАДАНИЙ

1. Привязка программы к конфигурации системы (жесткий диск).
2. Привязка программы к конфигурации системы (материнская плата (серийный номер, название)).
3. Привязка программы к конфигурации системы (видеокарта).
4. Привязка программы к конфигурации системы (полный анализ конфигурации).

4. ТРЕБОВАНИЯ К ОФОРМЛЕНИЮ ОТЧЕТА

Отчет по лабораторной работе должен содержать следующие разделы:

- задание по лабораторной работе;
- описание алгоритма работы программы;
- листинг программы;
- выводы по проделанной работе.