Почему системы ненадежны?

Современные операционные системы имеют две особенности, из-за которых они теряют как в надежности, так и в защищенности. Во-первых, эти ОС огромны по размеру, а, во-вторых, в них очень плохо обеспечена изоляция ошибок. Ядро Linux имеет свыше 2,5 млн. строк кода, а ядро Windows XP как минимум в два раза больше.

Из-за огромного размера современных операционных систем никто в одиночку не может знать их досконально. Действительно, крайне трудно создать хорошую систему, если никто в действительности ее себе полностью не представляет.

К счастью, ситуация не столь бесперспективна. Разработчики стремятся создать более надежные операционные системы. Существует четыре различных подхода, которые применяются для того, чтобы в будущем сделать ОС более надежными и защищенными

Укрепленные операционные системы

Самый консервативный подход, Nooks , был разработан для того, чтобы увеличить надежность существующих операционных систем, таких как Windows и Linux. Технология Nooks поддерживает монолитную структуру ядра, в которой сотни или тысячи процедур связаны вместе в одном адресном пространстве и работают в режиме ядра. Этот подход сосредоточен на том, чтобы сделать драйверы устройств (основная причина всех проблем) менее опасными.

Цели проекта Nooks заключаются в следующем:

* Защитить ядра от ошибок в драйверах;
* Обеспечить автоматическое восстановление в случае сбоя в драйвере;
* Сделать все это путем минимальных изменений в существующих драйверах и ядре

Ограничения

Несмотря на то, что, согласно экспериментам, Nooks может обнаруживать 99% фатальных ошибок драйвера и 55% не фатальных, он далеко не совершенен. Например, драйверы могут выполнять привилегированные команды, которые они выполнять не должны; они могут записывать данные в некорректные порты ввода/вывода и выполнять бесконечные циклы.

Паравиртуальные машины

Второй подход создан на основе концепции виртуальной машины. Эта концепция была разработана в конце 60-х годов. Идея заключается в том, чтобы использовать специальную управляющую программу, называемую монитором виртуальной машины, которая работает непосредственно с аппаратным обеспечением, а не с операционной системой.

Этот метод часто применяется для того, чтобы две или несколько операционных систем, скажем, Linux и Windows, могли работать на одной и той же машине одновременно, причем так, что каждая из ОС считает, что в ее распоряжении находится вся машина.

В принципе, этот подход должен обеспечить более высокую надежность, чем единая операционная система, поскольку в том случае, если в виртуальной машине, содержащей один или несколько драйверов, возникает сбой, то виртуальную машину можно перезапустить – и драйверы вернутся в свое исходное состояние.

Мультисерверные операционные системы

Первые два подхода предусматривают модификацию унаследованных систем. Следующие два посвящены будущим системам.

Один из этих подходов напрямую касается сути проблемы: работы всей операционной системы как единой гигантской бинарной программы в режиме ядра. Вместо этого в данном случае предлагается иметь несколько небольших микроядер, работающих в режиме ядра, в то время как остальная часть операционной системы представляет собой набор полностью изолированных сервера и процессов драйвера, работающих в режиме пользователя.

Защита на базе языка

Самый радикальный подход, что весьма неожиданно, предложили в Microsoft Research, отказавшись от операционной системы как единой программы, выполняющейся в режиме ядра, и некоторого набора пользовательских процессов, функционирующих в режиме пользователя. Вместо этого предлагается система, написанная на совершенно новых, обеспечивающих безопасность типов языках, которые избавлены от всех проблем с указателями и других ошибок, связанных с Си и C++.

Вот список наиболее безопасных дистрибутивов Linux, которые «специально ориентированы на безопасность Linux».

1. Кубес ОС | Qubes Linux
2. Tails Linux
3. Parrot Security OS
4. Кали Линукс
5. Уоникс | Whonix Linux
6. Дискретный | Discreete Linux
7. Линукс Kodachi | Кодачи
8. BlackArch Linux
9. Heads ОС
10. Подграф | Subgraph ОС