моделир. ситуацию с зомби и сиротой
запускаем proc и kill-ом убиваем потомка, увидим
зомби и сироту

ивинки - идентификатор 1

Системный вызов fork (из вакалии):

см (*)

- для различия (родителя и его потомка) fork возвращает потому значение 0 для родителя - идентификатор PID потомка
- чаще всего после вызова fork процесс-потомок сразу же вызывает exec

Порядок выполнения системного вызова fork:

1. резервируется пространство свопинга для данных и стека процесса-потомка
2. назначается новый идентификатор PID и структуру proc потомка
3. инициализируется структуру proc потомка. Некоторые поля этой структуры копируются от процесса-родителя (такие как идентификаторы пользователя и группы, маски сигналов и группа процесса), часть полей устанавливается в 0

время нахождения в резидентном состоянии использование процесса, кроме сна), а остальным присваиваются специфические для потомка значения (поля идентификаторов PID потомка и его родителя, указатель на структуру proc родителя)

4.
5. создаются карты трансляции адресов для процесса-потомка
6. выделяется область и потомка и скопировать в нее содержимое области и процесса-родителя
7. изменяются ссылки области ll на новые карты адресации и пространство свопинга
8. потомок добавляется в набор процессов, разделяющих между собой область кода программы, выполняемой процессом-родителем
9. постранично дублируются области данных и стека родителя и модифицировать карты адресации потомка в соответствии этими новыми страницами
10. потомок получает ссылки на разделяемые ресурсы, наследуемые потомками, такие как открытые файлы и текущий рабочий каталог
11. инициализируется аппаратный контекст потомка посредством копирования текущего состояния регистров его родителя
12. Поместить процесс-потомок в очередь готовых процессов
13. возвращается PID в точку возврата из системного вызова в родительском процессе и 0 - в процессе-потомке

Выполнение действий системным вызовом exec:

1. Разбирает путь к исполняемому файлу и осуществляет доступ к нему
2. Проверяет, имеет ли вызывающий процесс полномочия на выполнение файла
3. Читает заголовок и проверяет, что он действительно исполняемый
4. Если для файла установлены биты SUID или SGID, то эффективные идентификаторы UID и GID вызывающего процесса изменяет на UID и GID, соответствующие владельцу файла
5. Копирует аргументы, передаваемые в exec, а также переменные среды в пространство ядра, после чего пользовательское пространство готово к уничтожению
6. Освобождает пространство свопинга для областей данных и стека
7. Высвобождает старое адресное пространство и связанное с ним пространство свопинга. Если же процесс был создан при вызове vfork, производится возврат старого адресного пространства родительскому процессу
8. Выделяет карты трансляции адресов для нового текста, данных, стека
9. Устанавливает новое адресное пространство. Если область текста активна (какой-то другой процесс уже выполняет ту же программу), то она будет совместно использоваться с этими процессами. В других случаях пространство должно инициализироваться из выполняемого файла
10. Копирует аргументы и переменные среды обратно в новый стек приложения
11. Сбрасывает все обработчики сигналов в действия, определенные по умолчанию, т.к. функции обработчиков сигналов не существуют в новой программе. Сигналы, которые были проигнорированы или заблокированы перед вызовом exec, остаются в тех же состояниях
12. Инициализирует аппаратный контекст. При этом большинство регистров обращаются в 0, а указатель команд получает значение точки входа программы.