1. Восстановление зашифрованных файлов.
    a. Сохранять открываемые на редактирование файлы в базу данных sqlite.
    b. Структура базы произвольная, однако необходимо уделить внимание вопросам разделения данных на таблицы, минимизации издержек хранения и оптимальному подходу к хранению содержимого файлов.
    c. Файл базы должен лежать в /etc/<project_name>/ -– дерево папок внутри на ваше усмотрение.
    d. Необходимо ограничить доступ к файлу базы данных и к его содержимому всем процессам, кроме доверенного.
    e. Одна (или несколько) из таблиц базы должна хранить white list (список исключений) в виде путей к бинарям, которые не нужно детектить.
    f. Должная быть таблица, в котрой хранятся задетекченные бинари (black list).
    g. Желательно следить за размером базы. Можно использовать параметр, хранящийся в конфиге, определяющий максимальный размер файла базы, который не должен быть превышен.
    h. После принудительного завершения процесса-зловреда испорченные файлы должны быть восстановлены и удалены из базы.
    i. Путь к базе задается в конфиге.
2. Логирование.
    a. Логировать события детекта в файл. Должен логироваться как минимум полный путь к бинарю.
    b. Логировать ошибки работы приложения.
    c. Путь к файлу хранится в конфиге.
    d. Файл лога может лежать /etc/<project_name>/ или в /var/log/<project_name>
    e. Ротацию можно не имлементить.
    f. Можно писать в syslog (если превратите приложение в сервис -– это обязательное требование).
3. Детектор должен учитывать процессы-исключения из white list.
    a. Редактировать список исключений может только доверенное приложение.
    b. Исключениями являются также дочерние процессы оригинального процесса-исключения.
4. Детектор должен блокировать запуск однажды задетеченных процессов.
    a. Обнаружив зловред, необходимо сохранить в black list как минимум хеш сумму бинаря.
    b. Дальнейшие попытки запуска бинарей из black list должны пресекаться.
    c. Если процесс помещается в white list, из black list его необходимо удалить.
    d. Каждую попытку запустить бинарь из black list нужно логировать: полный путь к файлу и хеш сумму.
5. Изменения в детекторе.
    a. Необходимо килять дочерние и родительские (кроме init конечно) процессы задетекченного зловреда.
6. Использовать систему сборки.
    a. Можно использовать любую систему сборки (CMake, Make,...)
    b. Отрефакторить код, разнеся логические части в разные файлы.
    c. Добавить в скрипты сборки инструкции по установке бинарей и конфигов приложения.
7. Доверенное приложение для редактирования списка исключений и доступа к базе -- sqlite3.
    a. Можно использовать «из коробки» тулу sqlite3, которая качается с официального сайта, можно собрать самому (shell.c в поставке исходниками).
    b. Бинарь sqlite3 может лежать где угодно. Приложение должно проверять хэш при попытках доступа этой тулой к базе данных приложения.

    c. Хэш тулы должен быть захардкожен.
8. Все параметры приложения должны быть вынесены в конфиг.
    a. Конфиг должен лежать в /etc/<project_name>/
    b. Формат файла конфига -– произвольный, но распространенный. INI-файл, JSON, XML,...
    c. Парсить конфиг можно любым инструментом:boost.program_options, TinyXML, написать самому паринг ini-формата.
9. Превратить приложение в службу.