SafeFunction

COLLABORATORS			
	TITLE : SafeFunction		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		August 19, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

1 SafeFunction **1**

1.1 SafeFunction 1.5 par F.Delacroix . . . 1
1.2 distribution . . . 1
1.3 Configuration minimale . . . 2
1.4 introduction . . . 2
1.5 Installation . . . 3
1.6 Enlèvement . . . 3
1.7 Remerciements . . . 3
1.8 Contacter l'auteur . . . 3
1.9 Quelques informations techniques . . . 3

Chapter 1

SafeFunction

1.1 SafeFunction 1.5 par F.Delacroix

Bienvenue à SafeFunction 1.5 par Frédéric Delacroix.

Lisez tout ce document avant d'utiliser SafeFunction car vous devez savoir certaines choses très importantes.

Distribution

Configuration nécessaire

Introduction

Installation

Enlèvement

Informations techniques
<=- Programmeurs lisez ceci!

Remerciements

Auteur

1.2 distribution

SafeFunction 1.5 a été écrit par
moi-même
(F.Delacroix) avec le merveilleux assembleur Devpac 3 d'HiSoft sur un vieil Amiga 500 avec le Kickstart 3.1. Cependant, mon travail est essentiellement basé sur le programme SaferPatches, qui a été écrit par Martin Adrian et placé dans le domaine public par lui. Vous pouvez le trouver sur Aminet.

SafeFunction 1.5, PatchList 1.0 et le code source sont placés dans le domaine public. N'importe qui peut les utiliser de façon commerciale ou

non. Cependant, je vous demande de laisser la distribution complète (ou au moins les exécutables et le fichier guide) pour ne pas troubler l'utilisateur; et j'apprécierais grandement que vous mentionniez mon nom quand vous redistribuez le programme.

Mais souvenez-vous:

Ce logiciel est fourni "TEL QUEL", sans garantie d'aucune sorte, explicite ou implicite. L'auteur ne pourra être tenu pour responsable pour les dommages directs ou indirects causés par ce programme. Toute utilisation se fait à vos risques et périls.

SafeFunction est un programme assez dangereux. Tout a été fait pour le rendre le plus sûr possible et votre amiga sera sans doute plus stable lorsqu'il sera installé, mais il y a des problèmes qui ne peuvent être évités, et certains programmes n'apprécieront pas la présence de SafeFunction (très probablement à cause d'une programmation illégale). Jusqu'ici, je n'ai eu de gourou immédiat qu'avec ReqChange 3.2, mais il y aura surement d'autres cas. N'oubliez donc pas la décharge ci-dessus.

Continuez à lire, certains sujets importants doivent encore être mentionnés.

1.3 Configuration minimale

Contrairement au SaferPatches de Martin Adrian, SafeFunction requiert au moins le Kickstart 3.0. C'est normalement la seule condition.

1.4 introduction

Ayant beaucoup de programmes de patch installés dans mon système, et sachant combien SetFunction() peut être dangereuse (des programmes qui quittent dans le mauvais ordre peuvent provoquer des accès à de la mémoire libre), j'ai récemment eu l'idée d'écrire un programme qui patcherait cette fonction pour que les autres patchs du système puissent être rendus plus surs. Alors j'ai commencé à coder ce patch, en assembleur, avec plusieurs structures de données pour les bibliothèques, fonctions et patches.

Puis, dans un meeting (mai 95, expo CIEV, Vendin-Le-Vieil), je suis tombé sur une version CD-ROM des archives Aminet, sur laquelle j'ai trouvé le programme SaferPatches 2.0, écrit par Martin Adrian.

Ce programme avait exactement le même but que celui que j'avais commencé, mais il utilisait des structures et routines plus petites. J'ai donc abandonné mon propre projet, et commencé à utiliser SaferPatches.

Mais, en examinant le source (en C) de SaferPatches, j'ai trouvé qu'il avait quelques faiblesses, et SafeFunction est ma tentative pour les corriger, rendant les patches encore plus surs et le programme encore plus efficace par l'utilisation des allocations de mémoire en bassins et de l'assembleur. La contre-partie est que le programme requiert maintenant le Kickstart 3.0, alors que SaferPatches n'avait besoin que de la version 1.2.

Si vous n'êtes pas sous 3.0 ou 3.1 (ou plus ?!), gardez SaferPatches.

1.5 Installation

SafeFunction doit être lancé assez tôt dans votre startup-sequence, alors suivez ces instructions:
- Assurez-vous d'utiliser au moins le kickstart 3.0
- Copiez SafeFunction dans C:
- éditez votre Startup-Sequence et insérez la ligne suivante juste après SetPatch:

```
SafeFunction >NIL:
```

-redemarrez l'Amiga.

1.6 Enlèvement

Vous ne pouvez pas enlever SafeFunction une fois qu'il est chargé. Vous pouvez bien sûr l'enlever pour le prochain redémarrage en l'effaçant simplement de votre startup-sequence.

1.7 Remerciements

Je remercie Martin Adrian, l'auteur de SaferPatches, pour son bon travail; ProMédia, et tous mes amis... Et ESCOM !!!!

1.8 Contacter l'auteur

N'hésitez pas à m'écrire à:
Frédéric Delacroix
5, rue d'Artres
59269 Quérénaing
FRANCE

1.9 Quelques informations techniques

Ce programme tente de rendre votre amiga plus sûr, mais il y a un inconvénient. En effet, SetFunction(), en théorie, n'a pas le droit d'échouer. Mais dans le patch, je dois faire des allocations de mémoire. Tout a été fait pour s'assurer du succès de ces allocations: elles sont petites (environ 14 octets par patch, et un peu plus pour chaque bibliothèque) et faite avec les fonctions de bassin, et un gestionnaire de pénurie de mémoire est installé pour jeter un block d'urgence dans le système en cas d'échec.

MAIS le risque d'échec existe toujours. C'est pourquoi je voudrais vous demander à vous, programmeurs, de toujours vérifier le résultat de SetFunction(): NULL signifie un échec, et la fonction n'est pas patchée. Ceci sera encore plus de sécurité pour l'utilisateur (être capable d'enlever les patchs dans n'importe quel ordre est déjà une bonne contrepartie pour ces petits ennuis supplémentaires).

De plus, tout a été fait pour s'assurer que les caches du processeur sont effacés correctement après avoir modifié le code (j'ai dû construire des instructions JMP manuellement) et les parties critiques (où, par exemple, JMP 0 aurait pu être exécuté par une autre tâche ou interruption) sont entourées par Disable()/Enable().

Finalement, je voudrais que vous tous, programmeurs, qui aimez faire des patches, de tester si SafeFunction est installé avant de dire à l'utilisateur qu'il est trop dangereux d'enlever le patch. Tout au moins, comme MagicMenu, vérifiez la présence d'un tooltype que l'utilisateur expérimenté peut régler pour désactiver les vérifications de sureté. En effet, si vous épiez directement dans la table de saut de la bibliothèque, vous trouverez que l'adresse est différente de celle que vous avez utilisé quand SafeFunction est installé, mais cela ne veut pas dire qu'un autre programme a patché la fonction.

Tester si SafeFunction est installé se fait en cherchant (avec FindSemaphore()) un sémaphore nommé "SafeFunction.Semaphore" (attention aux maj/minuscules). Un retour NULL indiquera que SafeFunction n'est pas installé.

Il y a un programme supplémentaire: PatchList. Ce programme est adapté de ShowPatch, par Martin Adrian, fourni avec SaferPatches. Pourvu que SafeFunction soit installé, il liste tous les patchs faits au système (cette liste pourrait être plus impressionante que vous pensez !).

Pour chaque bibliothèque, il imprime sur le canal standard de sortie le nom de la bibliothèque et son adresse, et la liste des patchs. Pour chaque patch, il imprime l'offset de la fonction, les ancienne (avant patch) et nouvelle (après patch) fonctions et, s'il est disponible, le nom de la fonction. Pour que cette possibilité marche, vous aurez besoin des fichiers FD. Le répertoire par défaut où PatchList peut les trouver est "FD:", mais ceci peut être modifié sur la ligne de commande avec le mot-clé FDDir. "NOFD" désactivera la recherche. PatchList suppose que les fichiers FD sont nommés selon la règle suivante: ".library" (ou device, resource) remplacé par "_lib.fd". Vous devrez donc peut-être renommer "wb_lib.fd" en "workbench_lib.fd".

Finalement, il y a une limitation mineure: les fonctions avec des offsets entre -6 et -24 ont des noms réservés: LIB_OPEN, LIB_CLOSE, LIB_EXPUNGE et LIB_RESERVED. C'est pourquoi certaines fonctions de ressources (qui commencent à -6) pourront ne pas avoir le nom correct à l'écran. De même, les devices devraient avoir les fonctions -30 et -36 nommées DEV_BEGINIO et DEV_ABORTIO, mais je n'ai pas pris la peine de faire les tests nécessaires. Faites-moi savoir si cela vous ennuie.

J'ajoute aussi que PatchList ne doit être utilisé que pour des opérations de débogage: il accède aux fonctions du DOS (qui vont en Wait()) tout en détenant le sémaphore (mode partagé), empêchant ainsi d'autres patchs d'être faits. Vous pouvez donc avoir un blocage si le processus de console

a une idée bizarre et effectue un SetFunction() sur quelque chose...

Ah oui, une dernière (promis!) chose: comme j'utilise ObtainSemaphore(), il est possible qu'une tâche soit mise en attente par Wait() en tentant de patcher une fonction par SetFunction(). Ainsi, vous ne devriez pas tenter de faire un patch dans un partie critique (mode Forbid() nécessaire) de votre code...