R2AI

Axelle Apvrille, Fortinet

BlackAlps, Rump Session, November 2024

Who am I?

Officially known as **Axelle** or **cryptax**.
Research on Android and IoT malware at **Fortinet**.
I morph in a *crocodile named Pico le Croco*

I don't like to take risks

=

- r2ai
- r2ai-server
- r2ai-plugin
- **decai**: AI-assisted decompiler

Ghidra on Linux/Devura malware

```
  if (DAT_00134e40 == '\0') {
    sVar3 = readlink("/proc/self/exe",&DAT_00134e40,0xfff);
    if ((DAT_00134e40 == '\0') || (sVar3 == -1)) goto LAB_00104f7c;
    (&DAT_00134e40)[sVar3] = 0;
  }
  pcVar2 = basename(&DAT_00134e40);
  snprintf(local_1018,0x1000,"/lib/udev/%s",pcVar2);
  if (DAT_00134e40 == '\0') {
    sVar3 = readlink("/proc/self/exe",&DAT_00134e40,0xfff);
    if ((DAT_00134e40 == '\0') || (sVar3 == -1)) {
LAB_00104f7c:
                    /* WARNING: Subroutine does not return */
      exit(-1);
    }
    (&DAT_00134e40)[sVar3] = 0;
  }
  pcVar2 = basename(&DAT_00134e40);
  snprintf(acStack_2018,0x1000,"/etc/udev/rules.d/99-%s.rules",pcVar2);
  iVar1 = access(local_1018,0);
  if (iVar1 == 0) {
    printf("%s\n");
  }
  iVar1 = access(acStack_2018,0);
  if (iVar1 != 0) {
    return;
  }
  printf("%s\n",acStack_2018);
```

Decai on Linux/Devura

```
[0x00004e20]> decai -e api=openai
[0x00004e20]> decai -d
```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <libgen.h>
#include <string.h>

char fileName[1000], filePath[1000], udevRuleFilePath[1000];

int main(int argc, char *argv[]) {
    strncpy(fileName, argv[1], sizeof(fileName));
    fileName[sizeof(fileName) - 1] = 0;
    snprintf(filePath, sizeof(filePath), "/lib/udev/%s", fileName);

    readlink(udevRuleFilePath, filePath, 1000);
    snprintf(udevRuleFilePath, sizeof(udevRuleFilePath), "/etc/udev/rules.d/99-%s.ru

    if (access(udevRuleFilePath, F_OK) != -1) {
        printf("%s\n", filePath);
    } else if (access(filePath, F_OK) != -1) {
        printf("%s\n", filePath);
    } else {
        exit(-1);
    }
    return 0;
}
```
```

Decai -x

```
[0x00004e20]> decai -x


This function appears to be chec
rectories related to Ubuntu's de
tempts to read symbolic links to
rate paths to certain udev rule
cks if they exist. It prints out
 process otherwise.
```

Changing the AI prompt

```
[0x00004e20]> decai -e prompt=Explain how you think a malw
[0x00004e20]> decai -d
The pseudocode appears to be part of a malware that levera
nix-type operating systems. The main function is named "ud
lware to ascertain information about the system it is runn
on in a way that benefits the malware.

The function appears to make use of several functions from
of `readlink()` to read the symbolic link "/proc/self/exe"
ting program. It uses `basename()` to get the base name of
ntf()` is used to format and store a series of characters
hecks the file specified's accessibility. If accessible, i
ven contains a call to the `exit()` function which termina

This pseudocode seems like a fragment of larger malware, s
 the purpose of this function is. However, local variables
ted to examining, manipulating or checking the existence o
ev/rules.d/99-%s.rules" or "/lib/udev/%s".
```

Thanks for your attention!

Ready for Raclette!

- https://github.com/cryptax/talks
- @cryptax (X, Mastodon.social)