Linenoise

```
\xeb\x1f\x
5e\x89\x76
\x08\x31\x
c0\x88\x46
\x07\x89\x
```

DETECTING TH3 W1LD SH3LLC0D3

Alejandro Barrera (a.k.a. Ergosum)

Linenoise C0N III

17,18,19 de Octubre 2003

```
\x80\x31\x
db\x89\xd8
\x40\xcd\x
80\xe8\xdc
\xff\xff\x
ff/bin/sh
```

DETECTING TH3 W1LD SH3LLC0D3

1. **What is a shellcode?**
2. **Historical Evolution**
3. **IDS Detection technics**
    - 3.1 NOP's sledges
    - 3.2 Dangerous opcodes
    - 3.3 string matching
4. **H4x0r decoying technics**
    - 4.1 NOP substitution
    - 4.2 Alphanumeric shellcodes
    - 4.3 String modifications (inc and others)
5. **Polymorphic shellcodes**
    - 5.1 Polymorphic concept
    - 5.2 Polymorphic shellcode detection
        - 5.2.1 *Spectrum Analysis*
    - 5.3 CLET's polymorphic engine

WHAT IS A SHELLCODE?

- Código ensamblador
- Normalmente ejecuta una shell
  - Su nombre deriva de eso (**shell code**)
- Hoy en día no solo se limita a ejecutar una shell
  - Bind shellcodes
  - Reverse shellcodes
  - Seteuid() shellcodes
  - ...
- Pasamos de representación en asm a opcodes en hex

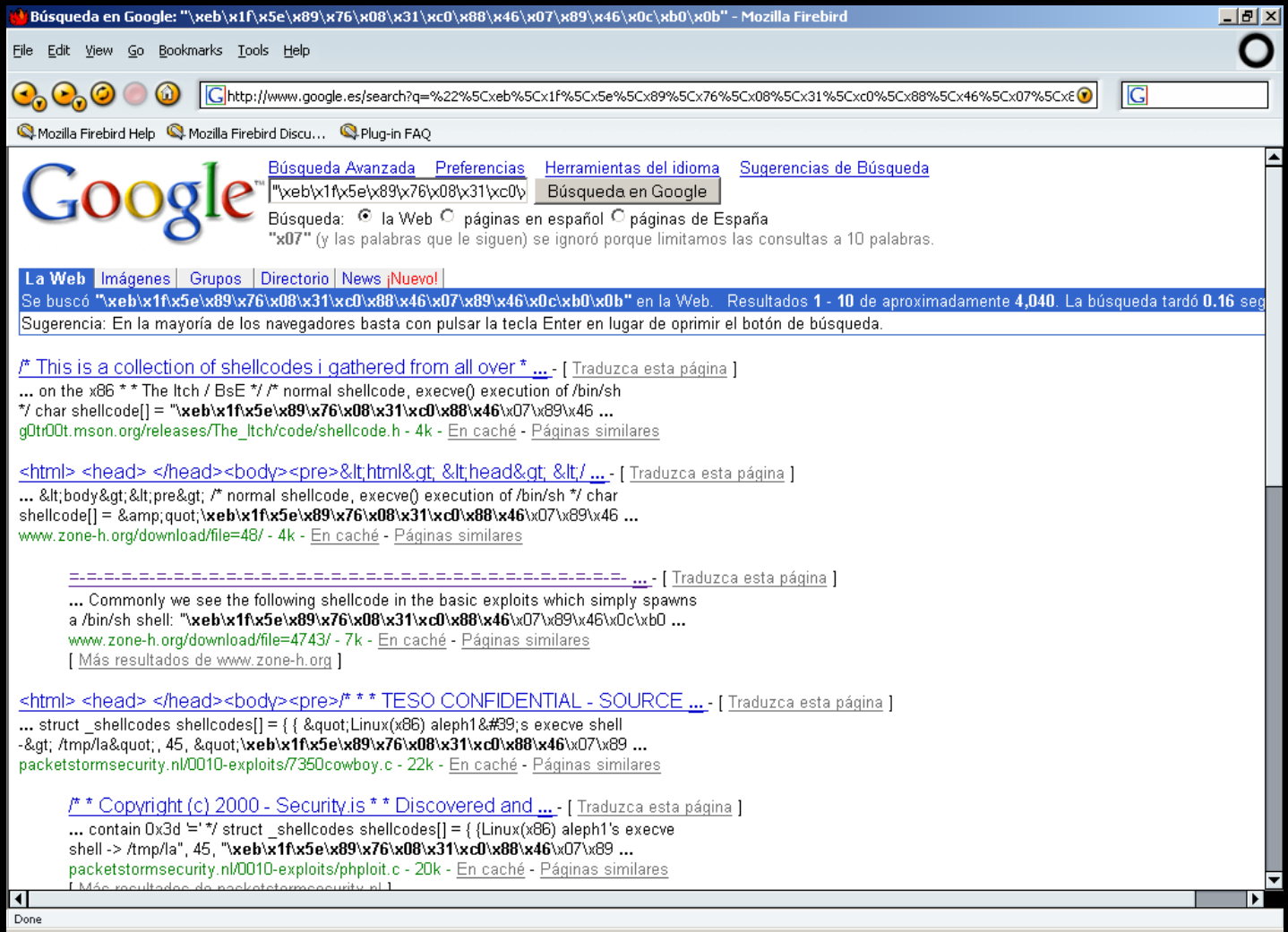
```
80483c1:    31 db           xor    %ebx,%ebx
80483c3:    89 e1           mov    %esp,%ecx
```

HISTORICAL EVOLUTION

- Gusano de Morris **1988**
  - Primero conocido en usar una shellcode
  - 6000 ordenadores infectados en 72h
- Primer artículo donde se explican las shellcodes
  - "*How to write Buffer Overflows*" by Mudge
  - Noviembre de **1995**
- Primer exploit en bugtraq con shellcode
  - 3 de diciembre de **1995**
- "*Smashing the stack for fun and profit*" by Aleph1
  - 8 de noviembre de **1996**
  - Artículo más conocido sobre buffer overflows

HISTORICAL EVOLUTION

- Muy poco avance en sus diseños
  - Usadas una y otra vez (Aleph1's shellcode 4040 hits)



IDS DETECTION TECHNICS

- **IDS** = *Intrusion Detection System*
  - Sistemas que detectan actividades:
    - Inapropiadas
    - Incorrectas
    - Anómalas
- Técnicas de detección más comúnmente implementadas
  - Reconocimiento de patrones
  - Detección de anomalías
  - Decodificación de protocolos
- **HIDS**
  - Host-based IDS
- **NIDS**
  - Network-based IDS

IDS DETECTION TECHNICS

- **NOP Sledges**
  - Detección del opcode de NOP (**N**o **OP**eration)
  - Función:
    - Permitir dirección de retorno no exacta
    - Padding

buffer

NOP NOP NOP NOP ... NOP	SHELLCODE

SHELLCODE	NOP NOP NOP NOP ... NOP

SHELLCODE

IDS DETECTION TECHNICS

- **Dangerous opcodes**
  - Se extiende la búsqueda no solo a NOPs
    - Se analiza instrucción **INT**
      - Linux int $0x80
      - Trap para llamar al SO
    - Otros posibles opcodes sospechosos
- **String Matching**
  - Se hacen búsquedas de cadenas suspechosas
    - /bin/sh (*2f 62 69 6e 2f 73 68*)
    - /bin/bash
    - /etc/passwd
    - /etc/shadow

IDS DETECTION TECHNICS

- **Ejemplo de Reconocimiento de patrones**
  - Snort IDS (www.snort.org)
    - Exploit.rules

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 749 (msg:"EXPLOIT
kadmind buffer overflow attempt"; flow:established,to_server;
cont\ ent:"|00 C0 05 08 00 C0 05 08 00 C0 05 08 00 C0 05 08|";
reference:cve,CAN-2002-1235;
reference:url,www.kb.cert.org/vuls/id/8\ 75073;
classtype:shellcode-detect; sid:1894; rev:1;)
```

H4X0R DECOYING TECHNICS

- **NOP substitution**
  - Sustituimos por instrucciones inocuas
    - No afectan al shellcode
    - Podemos saltar a ellas sin peligro
  - Instrucciones de 1 byte
    - Inc,dec,push,...

```
Opcode 0x40          inc %eax
Opcode 0x91          xchg %ecx,%eax
[...]
```

H4X0R DECOYING TECHNICS

- **NOP substitution**
  - Instrucciones de 2 bytes
    - **Instruction Stacking**
      - Construye instrucciones de multiples bytes con instrucciones de 1 byte

```
Opcode 0xB8 0x41 0x41 0x41 0x41  movl $41414141 %eax

Jmp $offset   -> movl $41414141 %eax
Jmp $offset+1 -> inc %eax
```

- **Alphanumeric shellcodes**
  - Opcodes alfanuméricos
    - 0x30-0x39 (0-9)
    - 0x41-0x5A (A-Z)
    - 0x61-0x7A (a-z)
  - Limita mucho el juego de instrucciones
    - popad, xor, ...
    - push + inc
    - ¡¡NO existe mov!!

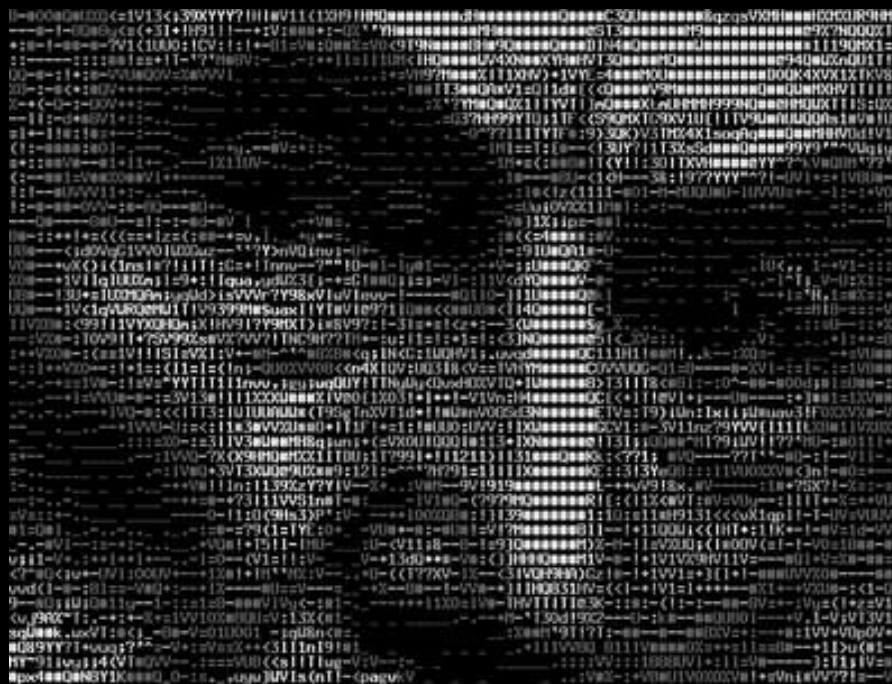
H4X0R DECOYING TECHNICS

mov reg,reg =

```
push eax      ;no change.
push ecx      ;no change.
push edx      ;no change.
push eax      ;EBX will contain EAX after POPAD.
push eax      ;no change (ESP not "poped").
push ebp      ;no change.
push esi      ;no change.
push edi      ;no change.
popad
```

H4X0R DECOYING TECHNICS

- **Alphanumeric shellcodes**
  - Problemas
    - Dirección de retorno
      - Normalmente **NO** *alfanumérica*
      - Posibilidad de detección



H4X0R DECOYING TECHNICS

- **String modifications (inc and others)**
  - Ofuscamos los strings
    - Sobrepasamos filtros (toupper())
    - Pasan por los IDS

```
    "/bin/sh" => \x2f\x62\x69\x6e\x2f\x73\x68
dec 0x21
    "/AHM/RG" => \x2f\x41\x48\x4d\x2f\x52\x47
```

- Añadimos rutina decodificadora
  - Sobrepasamos el filtro
  - Decodificamos en tiempo de ejecución

H4X0R DECOYING TECHNICS

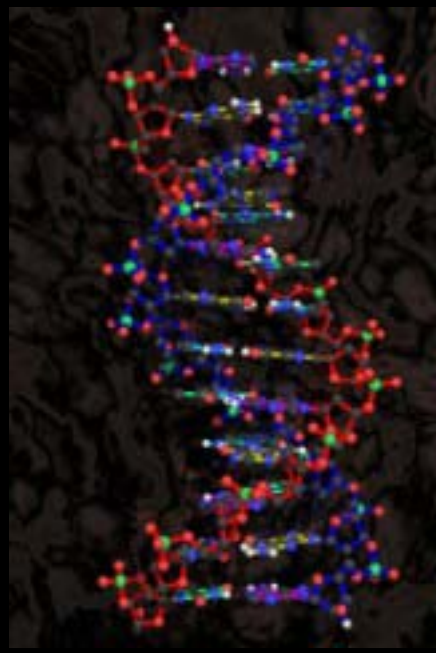
```
"\xeb\x31"              //     jmp    0x31
"\x5b"                   //     popl   %ebx
"\x80\x43\x01\x21"      //     addb   $0x21,0x1(%ebx)
"\x80\x43\x02\x21"      //     addb   $0x21,0x2(%ebx)
"\x80\x43\x03\x21"      //     addb   $0x21,0x3(%ebx)
"\x80\x43\x05\x21"      //     addb   $0x21,0x5(%ebx)
"\x80\x43\x06\x21"      //     addb   $0x21,0x6(%ebx)
[...]
```

- No tendría porque ser el mismo offset

POLYMORPHIC SHELLCODES

- **Concepto de polimorfismo**
  - *"Habilidad de existir en diferentes formas"*
    - Si al aplicar un algoritmo a un fuente obtenemos un objeto con la misma funcionalidad pero con distinta estructura que el fuente.

- Concepto propio de los viruses
  - Evitaba la detección por patrones
  - Portado por **K2** de ADM
    - **ADMmutate**



POLYMORPHIC SHELLCODES

- **Cambios léxicos**
  - Sustituir opcodes por otros equivalentes
- **Cambios sintácticos**
  - Cambiar el orden de las instrucciones
  - *Out-of-order*
- **Cambios morfológicos**
  - Variación de la estructura externa
    - Se mantiene la funcionalidad
    - Se añade basura entre las instrucciones

POLYMORPHIC SHELLCODES

- **Pasos:**
  - Cifrar el shellcode para evitar detección
  - Generar una rutina de decodificación
- **Cifrado del payload**
  - Se realiza normalmente una XOR con el shellcode
    - El cifrado solo es importante para evitar los patrones
    - Otros algoritmos incrementarían el tamaño final
    - Multiples XOR y key variable
  - Se añade una rutina de decodificación al principio
    - Al ejecutarse el cuerpo se descifra y se ejecuta
- **NO CONFUNDIR CIFRADO CON POLIMORFISMO!!**

POLYMORPHIC SHELLCODES

- **Problemas**
  - Si la rutina de decodificación es la misma
    - Será detectado por un patrón
- **Solución**
  - Aplicar polimorfismo a la rutina de decodificación

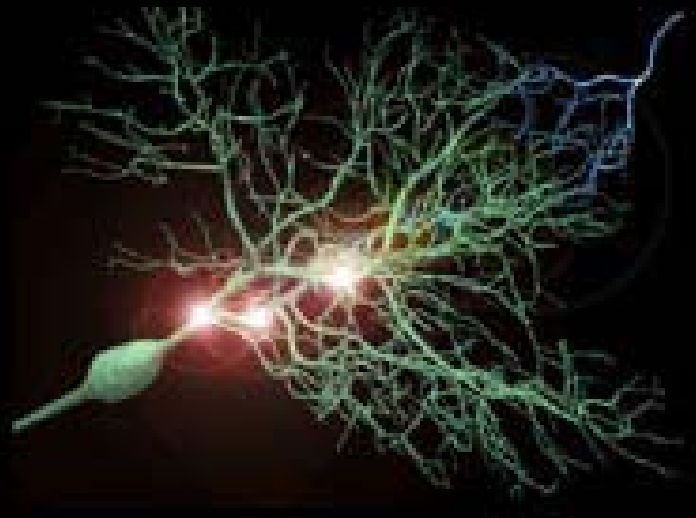
PROJECT DELPHOS

Artificial Neuronal System for Shellcode Detection

PROJECT DELPHOS

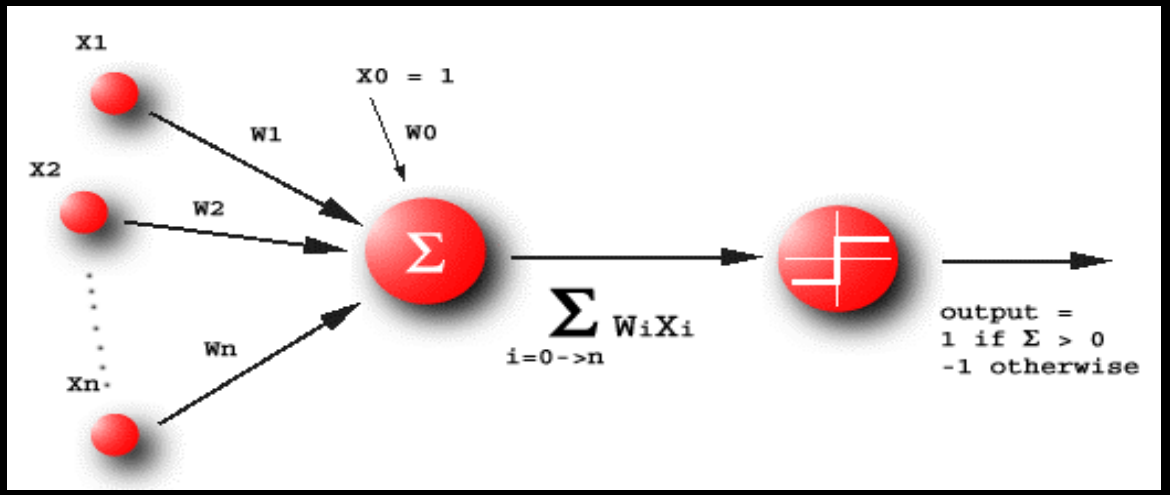
- **Sistemas de Reglas obsoletos**
  - Cada vez mayores bases de datos
  - Cualquier variación deja de ser detectada

- **Presente**
  - Uso de la Inteligencia Artificial (**IA**)
  - Dada una base de hechos
    - Aplicamos motor de inferencias
      - Inferimos que tráfico es sospechoso

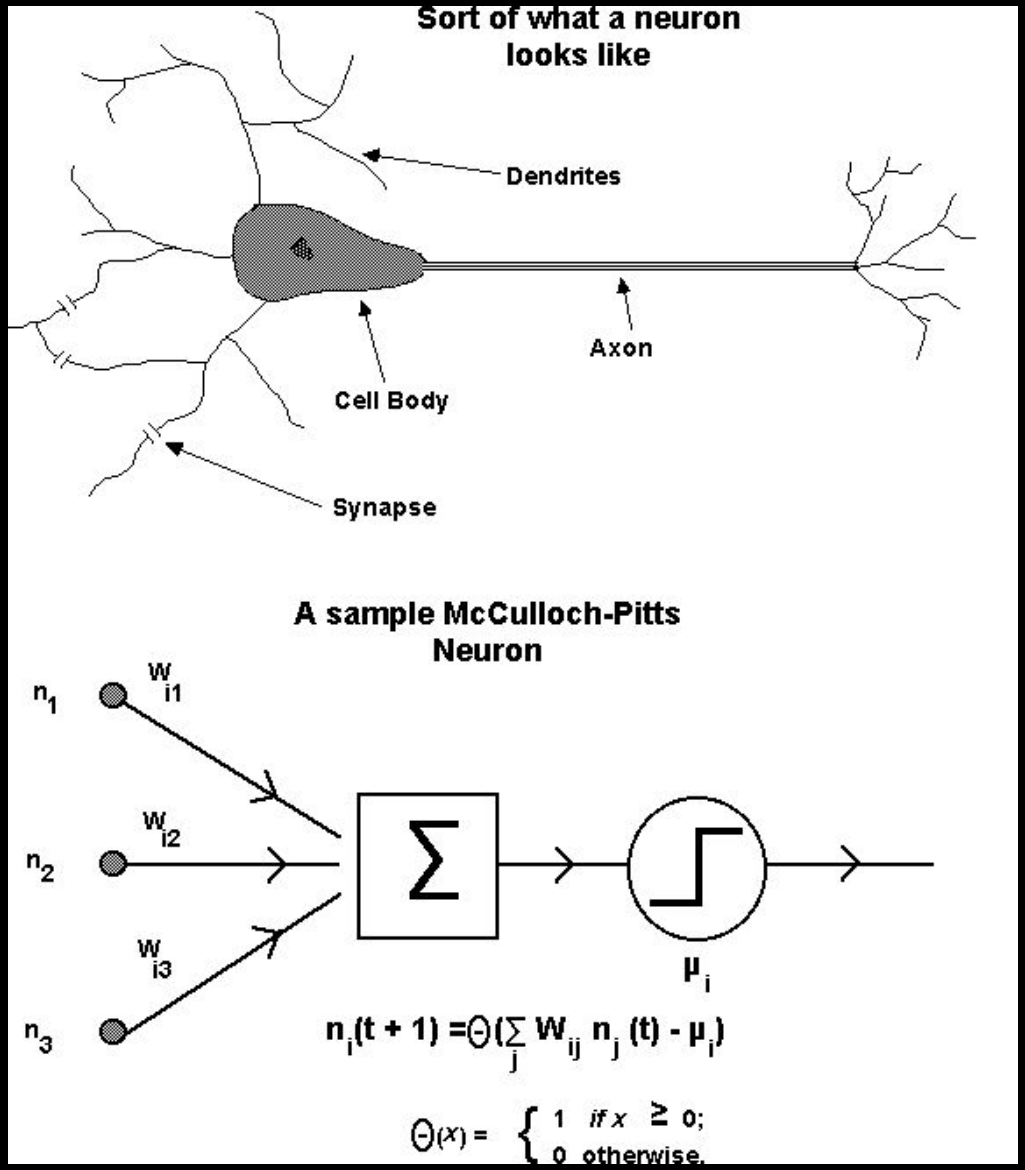


Inteligencia Artificial Conexionista: **Redes de Neuronas**

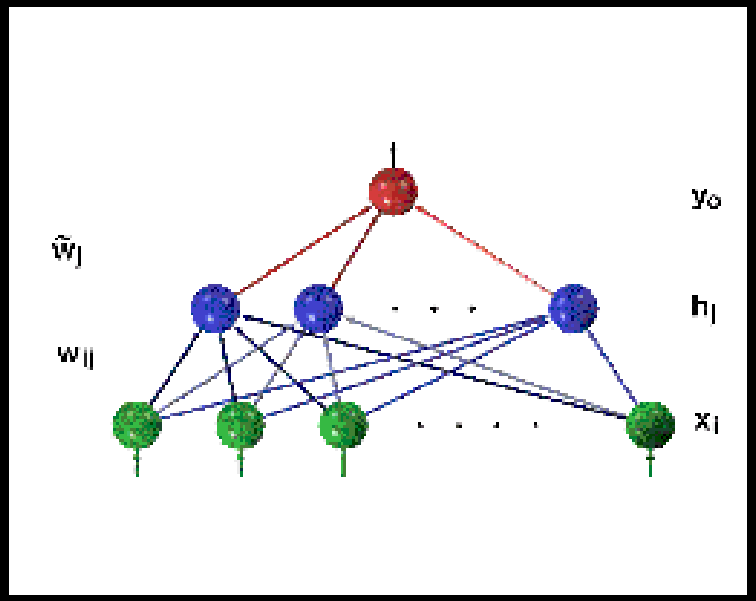
- **Perceptrón**
  - Inputs
  - Pesos
  - Función de activación
    - Función de threshold
    - Función sinusoidal
  - Output = NET de la neurona

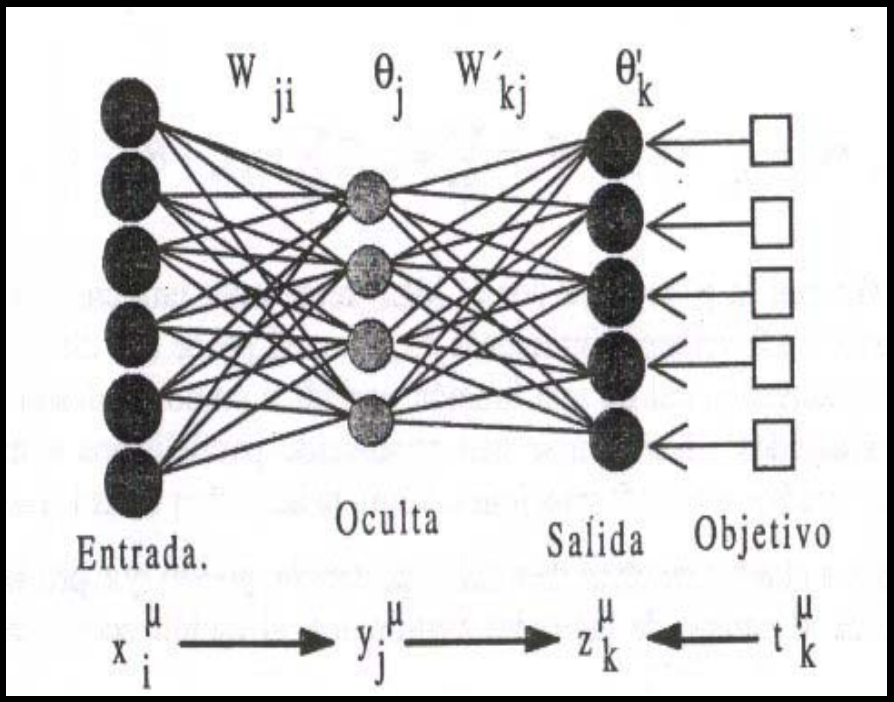


PROJECT DELPHOS



PROJECT DELPHOS





- **Fases**
  - Entrenamiento
    - Conjunto de prueba (training set)
      - Ciclos (Epochs)
    - Primer ciclo
      - Error cuadratico medio
      - Valor esperado
      - Aprendizaje por ***backpropagation***
  - Sistema operacional
    - Reconocimiento de patrones

- **Backpropagation**
  - Propagación del error hacia atrás
    - Modificación de los pesos sinápticos
- **Algoritmos de aprendizaje**
  - 2 Tipos
    - Supervisados
    - No supervisados
  - Muchos y muy distintos
    - Backpropagation el más común y sencillo
    - Algoritmos genéticos
      - Aprendizaje más fiable y rápido

MUCHAS GRACIAS

MUCH4S GR4C14S A T0D05

- http://ergosum.homeftp.org/delphos/delphos.php

- Alejandro Barrera (a.k.a Ergosum)
- ergosum@digitalsec.net

Special thanks too:

- Linenoise staff
- DSR crew (specially to dab ;))


```
\xeb\x1f\x
5e\x89\x76
\x08\x31\x
c0\x88\x46
\x07\x89\x
f3\x8d\x4e
\x08\x8d\x
56\x0c\xcd
\x80\x31\x
db\x89\xd8
\x40\xcd\x
80\xe8\xdc
\xff\xff\x
ff/bin/sh
```