Módulo 04 · Fundamentos

Da Linguagem C ao Assembly, Registradores & Criptografia

Portal de Computação — Alinhamento CS50 Harvard

Gestão da Tecnologia da Informação — Fatec Assis

Agenda do Módulo 04 — 50 minutos

Bloco	Tempo	Tópico	Alinhamento CS50
1	12 min	Da Linguagem C ao Código de Máquina	CS50 Week 1 & 2 (Compilação C)
2	13 min	Registradores da CPU & Ciclo de Instrução	CS50 Week 4 & Hardware
3	13 min	Funções Hash Criptográficas (SHA-256)	CS50 Week 10 (Cybersecurity)
4	12 min	Criptografia Simétrica (AES) & Assimétrica (RSA)	CS50 Week 10 & HTTPS

Bloco 1 · Da Linguagem C ao Assembly

A Tradução pelo Compilador

O Pipeline de Tradução do Compilador

Exemplo em C:

```c
int somar(int a, int b) {
    return a + b;
}
```

Instrução Assembly Traduzida (x86_64 Intel):

```
somar:
    mov eax, edi    ; Copia o 1º parâmetro para o registrador EAX
    add eax, esi    ; Soma o 2º parâmetro ao registrador EAX
    ret             ; Retorna o resultado contido em EAX
```

As 4 Fases da Compilação — O Que Acontece "Debaixo do Capô"

1. **Análise Léxica (Scanning):** quebra o texto em **Tokens** (palavras-chave, variáveis, operadores).
2. **Análise Sintática (Parsing):** monta a **Árvore Sintática (AST)** e valida a gramática da linguagem.
3. **Análise Semântica:** garante que as operações fazem sentido (ex.: impede somar `string` com `número`).
4. **Geração de Código:** converte a estrutura validada em instruções de máquina para o processador alvo.

Compilado vs. Interpretado — Duas Estratégias de Tradução

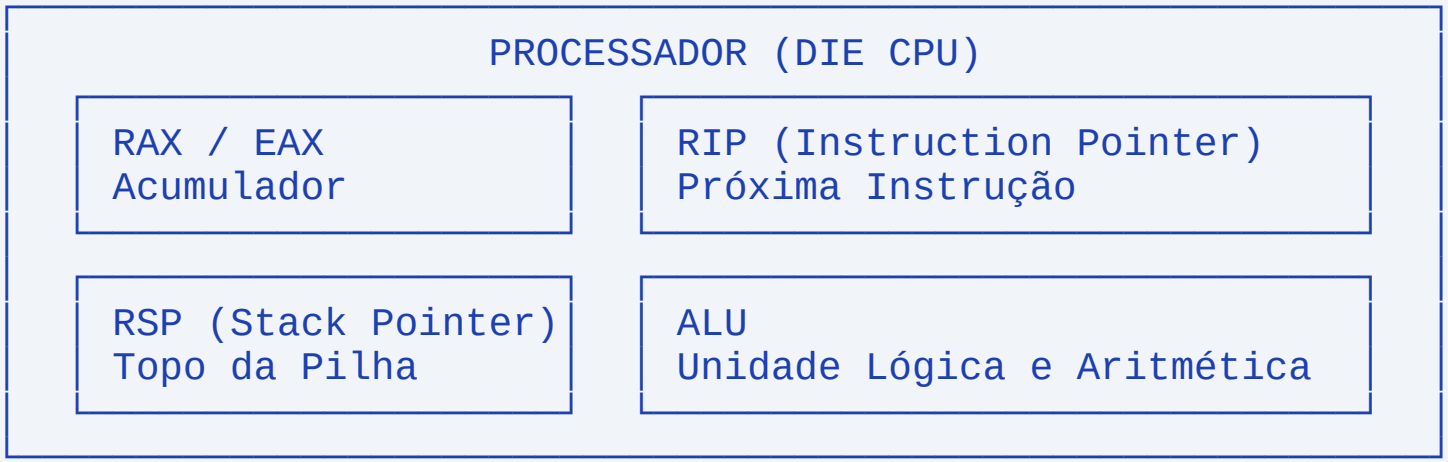
Característica	Compilado (C, Rust)	Interpretado (Python, JS)
Momento da tradução	Antes de rodar (gera `.exe`)	Durante a execução (linha por linha)
Velocidade de execução	Ultra-rápido (otimizado)	Mais lento (sobrecarga de tradução)
Portabilidade	Exige recompilar por sistema	Roda onde houver interpretador
Detecção de erros	Encontra a maioria antes de rodar	Só ao passar pela linha com erro

⚙️ **Modelo híbrido (Java, C#):** compila para **Bytecode** intermediário; uma Máquina Virtual o executa, e o compilador **JIT** otimiza os trechos mais usados direto para binário em tempo real.

Bloco 2 · Registradores da CPU

Execução de Instruções na CPU Real

Registradores — As Memórias de Silício da CPU

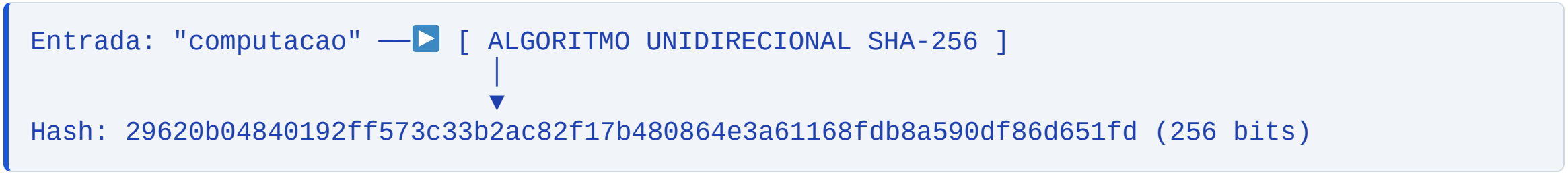



```
Ciclo Fetch-Decode-Execute:
1. FETCH ──▶ UC busca a instrução apontada por RIP na RAM/Cache
2. DECODE ──▶ Decodifica os opcodes binários
3. EXECUTE ──▶ ALU executa a soma nos registradores RAX/EDI
```

Bloco 3 · Funções Hash Criptográficas

SHA-256 & Integridade de Dados (CS50 Week 10)

Funções Hash (SHA-256)



Três Propriedades Fundamentais de Segurança:

1. **Inversibilidade Impossível (One-Way):** Não há fórmula matemática para reverter o hash e obter a senha/texto original.
2. **Efeito Avalanche:** Alterar apenas 1 bit no texto original muda completamente todos os 64 dígitos hex.
3. **Resistência a Colisões:** Dois textos diferentes nunca geram a mesma hash.

Bloco 4 · Criptografia & HTTPS

Protegendo Transmissões Digitais

Da Cifra de César ao HTTPS — Uma Ideia Antiga, Sempre Atual

- **Cifra de César (~50 a.C.):** desloca cada letra do alfabeto em N posições. Introduz os conceitos de **Texto Limpo**, **Chave** e **Texto Cifrado**.
- **Máquina Enigma (2ª Guerra):** a cifragem vira mecânica/elétrica — e a corrida para "quebrar" o código motiva os primeiros computadores elétricos (Turing incluído).
- **O salto para hoje:** a ideia de "chave" continua — só a matemática por trás ficou muito mais forte (RSA/AES em vez de deslocar letras).

Criptografia Assimétrica (Par de Chaves RSA)

- **Criptografia Simétrica (AES-256):** Utiliza a **mesma chave secreta** para cifrar e decifrar os dados. Altíssima velocidade para grandes volumes de dados.
- **Criptografia Assimétrica (RSA / ECC):** Utiliza um par de chaves relacionadas matematicamente (Pública e Privada).

Resumo do Módulo 04

1. **Compilação:** Traduz C para Assembly x86_64 e binário executável.
2. **Registradores:** RAX, RIP e RSP executam os ciclos da CPU na escala de nanosegundos.
3. **SHA-256:** Função hash unidirecional para verificação de senhas e integridade.
4. **HTTPS / TLS:** Combina criptografia assimétrica (troca de chaves) com simétrica (dados rápidos).

> 💡 **Próximo Módulo:** Módulo 05 — Arquitetura da Placa-Mãe, Fatores de Forma, Sockets e Memória RAM.