# Protocolo de Criptografia OAEP

Bernardo C. Magri

Instituto de Matemática e Estatística
Universidade de São Paulo

Seminário de Segurança de Dados, 2011

# Sumário

# Funções Criptográficas

- São a base da criptografia de chave pública
- Funções *one-way trapdoor*
- RSA, Rabin, Curvas Elípticas, etc

# Protocolos de Criptografia

- O que são?
- Para que servem?

# Definições de Segurança

- São descrições de adversários
- Definem o nível de segurança de um protocolo

# Segurança Semântica

- *"Tudo o que um adversário de tempo polinomial conseguir aprender sobre o texto simples, em posse do texto cifrado, ele deve ser capaz de aprender sem o texto cifrado."*
- Não é prático para se realizar provas
- É equivalente a *Indistinguibilidade de texto Cifrado*

# Indistinguibilidade de texto Cifrado

- Dadas duas mensagens $m_0, m_1$ e um texto cifrado $c$, um adversário (em tempo polinomial) não consegue distinguir se $c = f(m_0)$ ou $c = f(m_1)$
- Um palpite aleatório teria a probabilidade de acerto de $\frac{1}{2}$
- Um adversário tem a probabilidade de acerto de $\frac{1}{2} + adv$
- Dizemos que esse adversário é um distinguidor se $adv$ não for um valor negligenciável

# IND-CPA

## Indistinguishable Chosen-plaintext Attack

1. O adversário e o oráculo $\mathscr{O}$ concordaram previamente com o criptosistema $\mathscr{E}$
2. $\mathscr{O}$ fixou uma chave de encriptação $ke$ para $\mathscr{E}$
3. O adversário ecolhe mensagens distintas $m_0, m_1$ e as envia para $\mathscr{O}$
4. $\mathscr{O}$ joga uma moeda honesta $b \in 0,1$ e calcula

$$c^* = \begin{cases} \mathscr{E}_{ke}(m_0) & \text{if } b = 0 \\ \mathscr{E}_{ke}(m_1) & \text{if } b = 1 \end{cases}$$

   $\mathscr{O}$ envia $c^*$ para o adversário
5. Em posse de $c^*$, o adversário deve responder $b = 0$ ou $b = 1$

# IND-CCA

## Indistinguishable Chosen-ciphertext Attack

1. Considere os passos 1 e 2 da definição de IND-CPA
2. O adversário preparou anteriormente um conjunto de textos cifrados $\mathscr{C}$
3. O adversário envia $c \in \mathscr{C}$ para $\mathscr{O}$
4. $\mathscr{O}$ retorna para o adversário a decriptação de $c$ (este passo pode ser repetido quantas vezes forem necessárias)
5. Após o "treino de criptoanálise" acima ter sido concluído, o adversário executa o IND-CPA

# IND-CCA2

## Indistinguishable Adaptive Chosen-ciphertext Attack

1. Considere os passos 1 e 2 da definição de IND-CPA
2. O adversário executa o IND-CCA
3. Agora, o adversário pode computar $c^{'} \in \mathscr{C}$ e enviar para $\mathscr{O}$, para decriptação
4. É estipulado que $c^{'} \neq c^{*}$, isto é, o adversário não pode enviar $c^{*}$ a $\mathscr{O}$
5. Após o "treino de criptoanálise extendido" acima ter sido concluido, o adversário pode dar seu palpite em relação ao valor de $b$

# NM-CCA2

- *"É impossível modificar um texto simples de uma maneira controlada manipulando apenas o texto cifrado."*
- É equivalente ao IND-CCA2 (se provar um, obtém o outro)

# Modelo do Oráculo Aleatório (ROM)

- Possibilita a criação de "provas de segurança" para protocolos eficientes
- Assume que existem "oráculos aleatórios" a disposição de todos

# Desenvolvimento de Protocolos no modelo ROM

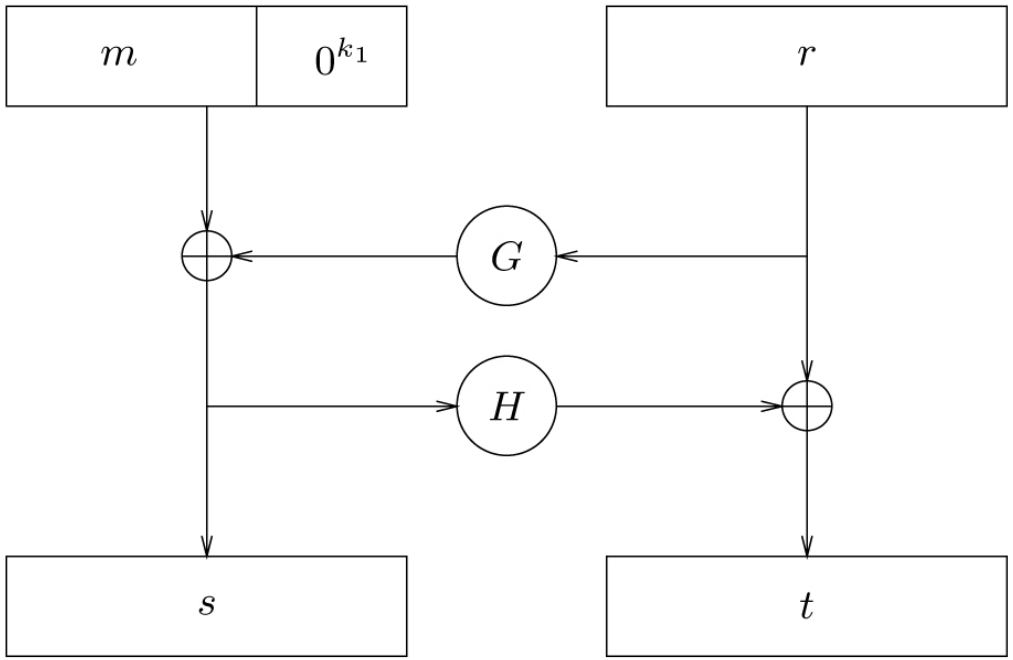
- Econtre uma definição formal para o problema no modelo em que todos compartilham um oráculo aleatório $R$
- Crie um protocolo eficiente $P$ para o problema nesse modelo
- Prove que $P$ satisfaça a definição do problema
- Troque o oráculos aleatório $R$ por uma funçõa de *hash*

# Sumário

# Descrição

- É um protocolo de encriptação probabilístico
- Uma mesma mensagem "nunca" obterá o mesmo texto cifrado (apenas com uma probabilidade de aproximadamente $2^{-160}$)
- É muito eficiente
- É provado IND-CCA2 seguro (se a função $f$ for RSA)

# Esquema



# Esquema

- O tamanho da mensagem a ser encriptada é $|M| = |N| - k_0 - k_1$
- Normalmente, $k_0 = k_1 = 160$
- $s = m^0 \oplus G(r)$ e $t = r \oplus H(s)$
- $c = f(s \parallel t)$

# Sumário

# Idéia

- Reduzir o problema de inverter a função $f$ (o que acredita-se ser intratável), para o problema de quebrar o protocolo
- Suponha que um adversário IND-CCA2 $\mathscr{A}$ consegue quebrar o protocolo em tempo polinomial, com uma vantagem não-negligenciável
- Então conseguimos construir um algoritmo (de tempo polinomial) que inverta a função $f$ utilizando a ajuda de $\mathscr{A}$
- Esse algoritmo será chamado de Simulador.

# Redução

- Simulador recebe uma função $f$ e um ponto aleatório $c^*$ em $f$
- O Simulador quer obter $f^{-1}(c^*)$ utilizando a ajuda de $\mathscr{A}$

1. No "estágio de procurar" de $\mathscr{A}$, Simon receberá textos cifrados para decriptação (para $\mathscr{A}$ construir estes textos cifrados de forma válida, ele deve utilizar os oráculos aleatórios do simulador), e ele deverá decriptá-los simulando o oráculo de decriptação
2. O simulador recebe duas mensagens $m_0, m_1$, e escolhe aleatoriamente o valor de $b \in 0,1$
3. Simon deverá retornar o valor $c^*$ para $\mathscr{A}$, fingindo que $c^*$ é a encriptação de uma das mensagens

# Simulação dos Oráculos G e H

- O Simulador mantém duas listas (G-lista e H-lista) contendo os pares $(g, G(g))$ e $(h, H(h))$ de todas as consultas feitas aos oráculos $G$ e $H$
- Se $g$ ou $h$ existirem nas listas, o valor correspondente é retornado
- Se não existir, o Simulador gera um valor aletório para a consulta e adiciona o par na lista
- Se alguma consulta ocorrer no "estágio de adivinhar" de $\mathscr{A}$, o Simulador tenta calcular $f^{-1}(c^*)$ para todos os pares das listas
- $f(c^*) \stackrel{?}{=} h \,\|\, (g \oplus H(h))$

# Simulação do Oráculo de Decriptação

- O Simulador testa suas listas e checa se $f(c) \stackrel{?}{=} h \,\|\, (g \oplus H(h))$
- Se a igualdade for verdadeira, o Simulador checa se os $k_1$ bits menos significativos de $v = G(g) \oplus h$ são 0
- Se sim, o Simulador devolve os $|M|$ digitos mais significativos de $v$, caso contrário devolve "Texto Inválido"

# Entendendo a Redução

- Existem valores $s^*, H(s^*), r^*, G(r*)$ tal que

$$s^* \parallel r^* \oplus H(s^*) = f^{-1}(c^*)$$

$$m_b = s^* \oplus G(r^*)$$

1. $\mathscr{A}$ só podera descobrir $m_b$ sabendo $G(r^*)$ (se for realmente aleatório)
2. Logo, $\mathscr{A}$ também precisa conhecer $s^*$

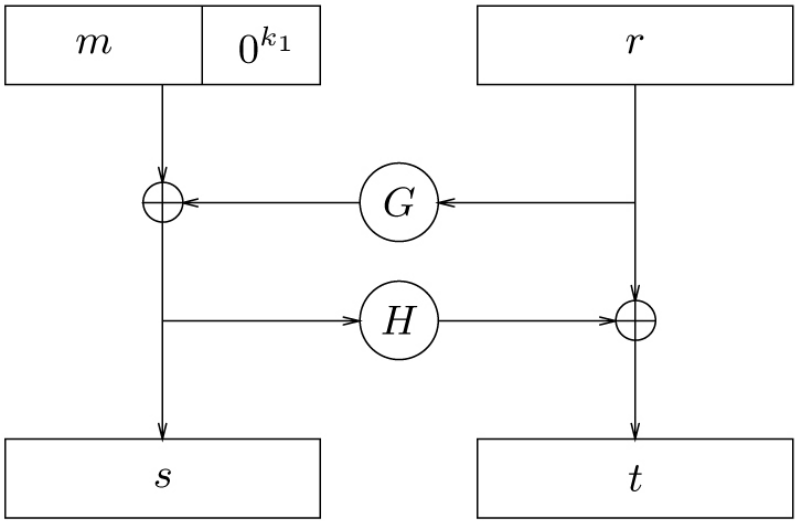
# Sumário

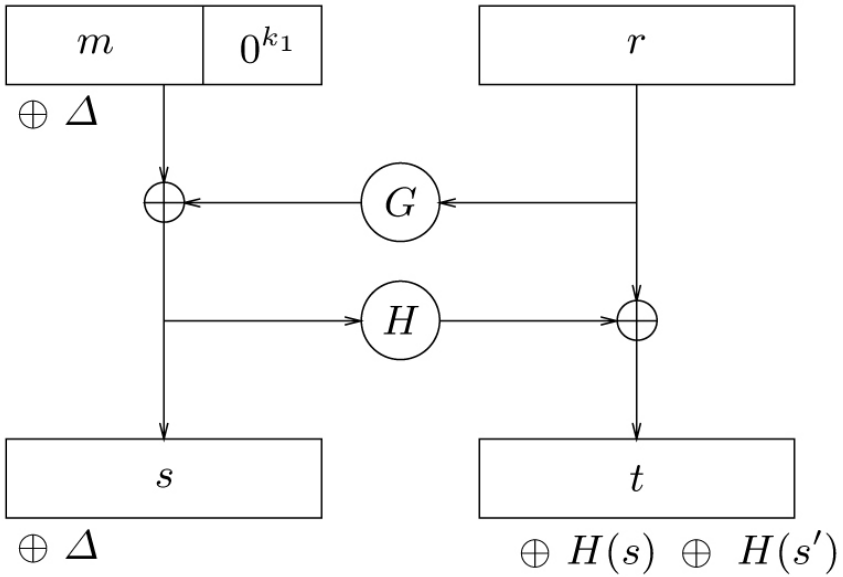
# Observação de Shoup

- Shoup mostrou que se $\mathscr{A}$ possui $s^*$ então ele consegue gerar um texto cifrado válido, baseado $c^*$
- Isso é um ataque de maleabilidade, o que mostra que $f - OAEP$ não é seguro contra NM-CCA2, logo não é seguro contra IND-CCA2

# Ataque de Shoup

O Ataque de Shoup ocorre como descrito a seguir:



# Ataque de Shoup

- Suponha que $f(s \| t) = s \| f_0(t)$ e que $f_0$ seja *xor-malleable*, isto é, dado $f_0(t_1), t_2$ obtém-se $f_0(t_1 \oplus t_2)$
- $s = s^* \oplus (\Delta \| 0^{k_1})$
- $v = f_0(t^* \oplus H(s^*) \oplus H(s))$
- $c = s \| v$
- $r = H(s) \oplus t = H(s^*) \oplus t^* = r^*$
- $\mathscr{A}$ não precisa saber $r^*$

# Sumário

# Recuperação Parcial de Shoup

- Shoup mostrou que se $f$ for o RSA, então Simon ainda consegue obter a inversão de $f$ utilizando a ajuda de $\mathscr{A}$
- $t^* = (X + 2^{k_0} I(s^*))^e \equiv c^* (mod N)$
- Utilizando o algoritmo de Coppersmith (apenas para $e$ pequeno)

# Recuperação Total de Fujisaki et al.

- Técnica de reticulados
- $(2^{k_0} I(s^*) + T)^e \equiv c^* (mod N)$

# Críticas

- O modelo do Oráculo aleatório pode ser considerado válido?
- A redução não é justa em relação a inversão do RSA ( $\mathcal{O}(t^2)$ )

# Conclusão

- Eficiente e prático
- Parece que a vulnerabilidade do protocolo está nas funções de *hash*

# Referências I


Wenbo Mao.
*Modern Cryptography*.
Prentice Hall, 2004.


Menezes, A. Koblitz, N.
Another look at "Provable Security".
2004