西安交通大學
XI'AN JIAOTONG UNIVERSITY

第 9 章：数字签名
Digital Signature Schemes

赵俊舟

西安交通大学网安学院
junzhou.zhao@xjtu.edu.cn

2025 年 12 月 20 日

目录

目录

软件发布的完整性验证问题

Linux Mint 21 "Vanessa"

Linux Mint 21

Cinnamon Edition

On this page you can download Linux Mint either directly or via torrent as an ISO image.
Make sure to verify your image after downloading it.

Information

- Size: 2.4GB
- Installation Guide
- Release Announcement
- Release Notes
- Torrent Download: 64-bit

Integrity & Authenticity

Anyone can produce fake ISO images, it is your responsibility to check you are downloading the official ones.
Download the ISO image, right-click->"Save Link As..." on the sha256sum.txt and sha256sum.txt.gpg buttons to save these files locally, then follow the instructions to verify your downloaded files.

sha256sum.txt | sha256sum.txt.gpg | Verify

软件发布的完整性验证问题

PCWorld NEWS BEST PICKS REVIEWS HOW-TO DEALS LAPTOPS WINDOWS SECURITY MORE

Linux Mint website hacked, ISO downloads replaced with backdoored operating system

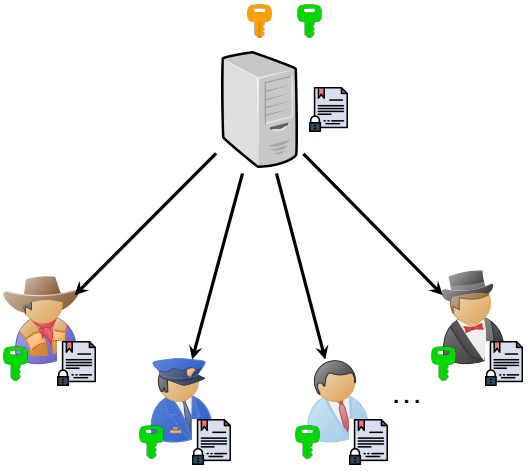
If you downloaded Linux Mint on Saturday, February 20th, you may have grabbed a hacked version that includes a backdoor. Here's what you need to know.

By Nick Mediati
PCWorld | FEB 21, 2016 10:02 AM PST

If you downloaded Linux Mint on Saturday, February 20th, you may have unknowingly downloaded a hacked version of the operating system.

According to a blog post on the Linux Mint site, hackers broke into the Linux Mint website at some point on Saturday and made changes in order to direct users toward downloading "a modified Linux Mint ISO, with a backdoor in it." Using the hacked version could allow hackers to steal your private information. According to Linux Mint, the hack only affects those who downloaded the Linux Mint 17.3 Cinnamon edition from the Linux Mint website on Saturday.

软件发布的完整性验证与数字签名

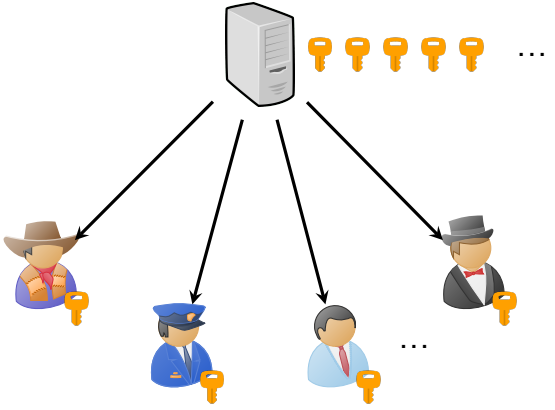


☞ 数字签名可以看作是公钥密码中的消息认证码。

- 软件开发商持有一对公私钥 (pk, sk)；
- 发布软件（更新）时，使用私钥 sk 对软件 m 签名得到 (m, σ)；
- 用户使用公钥 pk 验证软件的签名 σ 是合法的，从而相信软件 m 未经篡改；
- 安全的数字签名方案可以保证敌手无法对篡改后的软件 m' 生成合法签名 σ'。

消息认证码的问题：密钥管理问题

- 消息认证码可以对信源的正确性和消息的完整性进行验证。
- 消息认证码是基于对称密码实现的，如果有 n 个接收方，那么需要在发送方和接收方之间管理 n 个密钥。



消息认证码的问题：假冒和否认问题

- 消息认证可以保护信息交换双方不受第三方攻击，但是它不能处理通信双方自身发生的攻击。
- 例如，当 Alice 给 Bob 发送一条认证消息时：
- 假冒问题：Bob 可以伪造一条消息并声称该消息发自 Alice。Bob 只需要生成一条消息，并用 Alice 和 Bob 共享的密钥产生 MAC 码，并将 MAC 码附于消息之后。
- 否认问题：Alice 可以否认曾经给 Bob 发送过消息。因为 Bob 可以伪造消息，所以无法证明 Alice 确实发送过该消息。
- 在收发双方不能完全信任的情况下，需要其他方法来解决这些问题——数字签名（Digital Signature）。

数字签名的定义

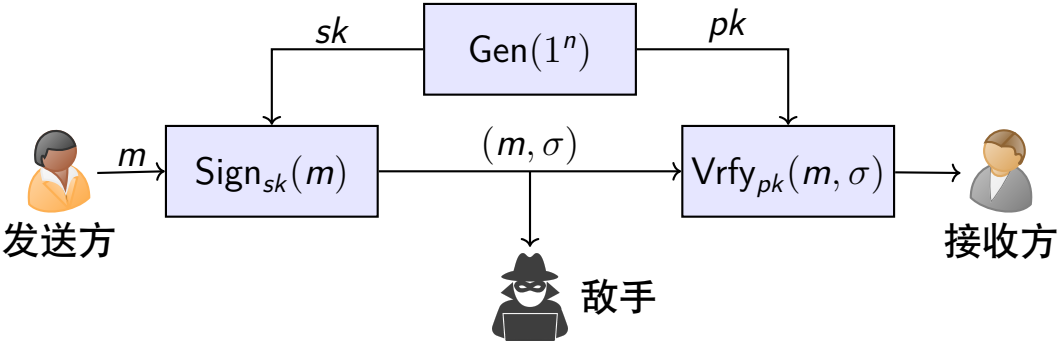
定义 (数字签名)

数字签名由三个 PPT 算法 $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ 组成，满足：

- 密钥生成算法 Gen：输入安全参数 1^n，输出密钥对 (pk, sk) 且 $|pk|, |sk| \geq n$；
- 签名算法 Sign：输入私钥 sk 和消息 m，输出签名 $\sigma \leftarrow \mathsf{Sign}_{sk}(m)$；
- 验证算法 Vrfy：输入公钥 pk，消息 m 和签名 σ，输出比特 $b = \mathsf{Vrfy}_{pk}(m, \sigma)$，$b = 1$ 表示签名有效，否则无效。

- 正确性要求：$\mathsf{Vrfy}_{pk}(m, \mathsf{Sign}_{sk}(m)) = 1$。
- 如果消息 $m \in \{0,1\}^{\ell(n)}$，则称 Π 为定长数字签名。

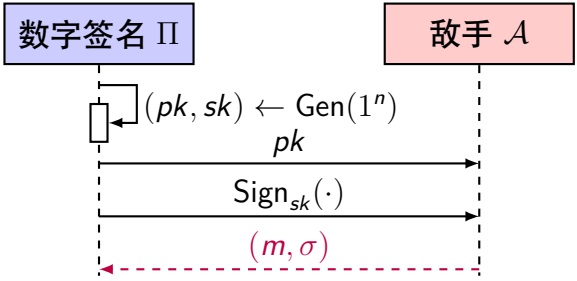
数字签名



- **可以公开验证**（Publicly Verifiable）：如果一个接收方验证签名合法，那么其他接收方验证该签名也必然合法。
- **可以传递签名**（Transferable）：因为签名可以公开验证，所以把合法的签名交给第三方，仍可以保证签名的合法性。

安全性定义

- 安全的数字签名不应让敌手能够对一条新发送的消息生成正确的签名。
- 选择消息攻击假设：敌手可以观察 (m, t) 对，并欺骗发送方生成 m' 的签名 σ。



- 敌手具有神谕 $\mathrm{Sign}_{sk}(\cdot)$。
- 如果敌手能生成消息 m 的签名 σ 且之前未对 m 使用过神谕，则称敌手攻击成功，记为

$$\text{Sig-forge}_{\mathcal{A},\Pi}(n) = 1$$

- 如果对任意 PPT 敌手，都存在可忽略函数 negl，满足

$$\Pr[\text{Sig-forge}_{\mathcal{A},\Pi}(n) = 1] \leq \text{negl}(n)$$

称消息认证码 $\Pi = (\mathrm{Gen}, \mathrm{Mac}, \mathrm{Vrfy})$ 是安全的。

目录

目录

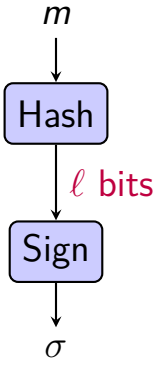
Hash-and-Sign

- 先对消息哈希，得到定长哈希码，然后对定长哈希码签名。
- Hash-and-Sign 可以被证明是安全的。

设计 (Hash-and-MAC)

令 $\Pi' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Vrfy}')$ 为消息长度为 $\ell(n)$ 的定长数字签名，令 $\mathcal{H} = (\mathsf{Gen}_H, H)$ 为输出长度为 $\ell(n)$ 的哈希函数。如下构造数字签名方案 $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$：

- Gen: 输入 1^n，$(pk, sk) \leftarrow \mathsf{Gen}'(1^n)$，$s \leftarrow \mathsf{Gen}_H(1^n)$，输出公钥 $\langle pk, s\rangle$ 和私钥 $\langle sk, s\rangle$；
- Sign: 输入私钥 sk 和消息 $m \in \{0,1\}^*$，输出 $\sigma \leftarrow \mathsf{Sign}'_{sk}(H_s(m))$；
- Vrfy: 输入公钥 pk，消息 $m \in \{0,1\}^*$ 和签名 σ，输出 $\mathsf{Vrfy}'_{pk}(H_s(m), \sigma)$。



目录

RSA-FDH

设计 (RSA Full Domain Hash)

- Gen: 输入 1^n，$(N, e, d) \leftarrow \mathrm{GenRSA}(1^n)$，输出公钥 $\langle N, e\rangle$ 和私钥 $\langle N, d\rangle$；同时，确定哈希函数 $H\colon \{0,1\}^* \mapsto \mathbb{Z}_N^*$。
- Sign: 输入私钥 $\langle N, d\rangle$ 和消息 $m \in \{0,1\}^*$，输出
$$\sigma = H(m)^d \bmod N$$
- Vrfy: 输入公钥 $\langle N, e\rangle$，消息 $m \in \{0,1\}^*$ 和签名 σ，输出
$$\sigma^e \stackrel{?}{=} H(m) \bmod N$$

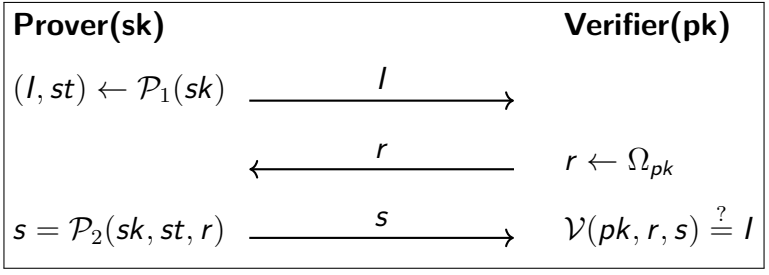
定理

基于 *RSA* 问题困难性假设，*RSA-FDH* 签名方案是安全的。

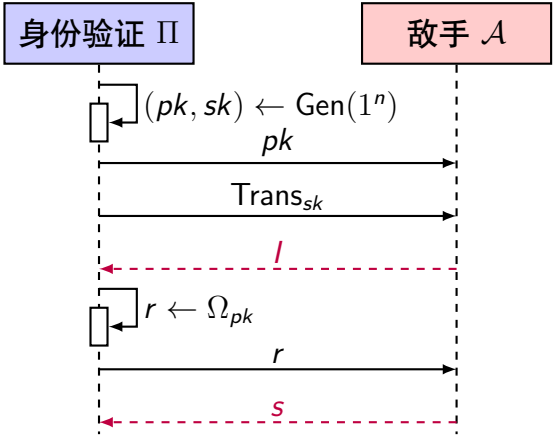
目录

身份验证方案（Identification Schemes）

- 身份验证方案是一种交互式证明协议，允许证明方向验证方证明自己的合法身份。
- 成功的身份验证使验证方相信证明方是预期的通信方，而非冒名顶替者。
- 三轮交互式身份证明：证明方包含算法 $\mathcal{P}_1$ $\mathcal{P}_2$，验证方包含算法 $\mathcal{V}$。其中 r 称为挑战（challenge），s 称为响应（response）。



对身份验证方案的安全性要求



- 敌手如果不知道证明方的私钥 sk，则不能欺骗验证方使验证通过。
- 即使敌手可以窃听证明方和验证方的所有交互 (I, r, s)：向敌手提供神谕 Trans_{sk}。
- 如果敌手成功欺骗验证方，记为

$$\mathrm{Ident}_{\mathcal{A},\Pi}(n) = 1$$

- 如果对于任意 PPT 敌手，存在可忽略函数 negl，使

$$\Pr[\mathrm{Ident}_{\mathcal{A},\Pi}(n) = 1] \leq negl(n)$$

则称身份验证方案 $\Pi = (\mathrm{Gen}, \mathcal{P}_1, \mathcal{P}_2, \mathcal{V})$ 是**安全**的。

Fiat-Shamir Transform：身份验证 $\rightarrow$ 数字签名

- 将要签名的消息 m 和初始消息 I 通过哈希函数 H 哈希为挑战 r，并生成响应 s。(r, s) 与消息 m 绑定，即构成签名。
- 安全的哈希函数 H 产生的 r 是 Ω_{pk} 中的均匀分布的随机数，故数字签名的安全性等价于交互式身份验证的安全性。

设计 (Fiat-Shamir Transform)

令 $\Pi = (\mathsf{Gen}_{id}, \mathcal{P}_1, \mathcal{P}_2, \mathcal{V})$ 为交互式身份验证方案。

- Gen: 输入 1^n，$(pk, sk) \leftarrow \mathsf{Gen}_{id}(1^n)$；同时，确定挑战集合 Ω_{pk} 和哈希函数 $H\colon \{0,1\}^* \mapsto \Omega_{pk}$。
- Sign: 输入私钥 sk 和消息 $m \in \{0,1\}^*$，输出签名 (r, s)。

$$(I, st) \leftarrow \mathcal{P}_1(sk) \quad r = H(I, m) \quad s = \mathcal{P}_2(sk, st, r)$$

- Vrfy: 输入公钥 pk，消息 m 和签名 (r, s)，计算 $I = \mathcal{V}(pk, r, s)$，输出 $H(I, m) \stackrel{?}{=} r$。

Schnorr 身份验证方案

- 给定循环群 $\mathbb{G}$，阶为素数 q $(\|q\| = n)$，生成元为 g。选择随机数 $x \in \mathbb{Z}_q$ 作为私钥，计算 $y = g^x$ 作为公钥。

Prover(x)		**Verifier**$(\mathbb{G}, q, g, y)$
$k \leftarrow \mathbb{Z}_q$		
$I = g^k$	$\xrightarrow{I}$	
	$\xleftarrow{r}$	$r \leftarrow \mathbb{Z}_q$
$s = xr + k \bmod q$	$\xrightarrow{s}$	$g^s y^{-r} \stackrel{?}{=} I$

- **正确性**：$g^s y^{-r} = g^{xr+k} g^{-xr} = g^k = I$
- **安全性**：假设敌手在不知道私钥 x 时能对挑战 r 产生正确的响应 s，则

$$g^s y^{-r} = I \Rightarrow s = \log_g(I y^r) = k + r \log_g y$$

即敌手可以求解离散对数问题。

Schnorr 数字签名方案

设计 (Schnorr 数字签名)

- Gen: 输入 1^n，确定群 $(\mathbb{G}, q, g)$；选择随机数 $x \in \mathbb{Z}_q$ 作为私钥，计算 $y = g^x$ 作为公钥。确定哈希函数 $H: \{0,1\}^* \mapsto \mathbb{Z}_q$。
- Sign: 输入私钥 x 和消息 $m \in \{0,1\}^*$，输出签名 (r, s)。

  $$k \leftarrow \mathbb{Z}_q, I = g^k \quad r = H(I, m) \quad s = xr + k \bmod q$$

- Vrfy: 输入公钥 y，消息 m 和签名 (r, s)，计算 $I = g^s y^{-r}$，输出 $H(I, m) \stackrel{?}{=} r$。

DSA 和 ECDSA

- DSA 和 ECDSA 都属于基于 DLP 的数字签名方案，定义在不同的群上，由相同的身份验证方案转换而来。
- 给定循环群 $\mathbb{G}$，阶为素数 q，生成元为 g。选择随机数 $x \in \mathbb{Z}_q$ 作为私钥，计算 $y = g^x$ 作为公钥。

Prover(x)		**Verifier**$(\mathbb{G}, q, g, y)$
$k \leftarrow \mathbb{Z}_q$		
$I = g^k$	$\xrightarrow{I}$	
	$\xleftarrow{\alpha, r}$	$\alpha, r \leftarrow \mathbb{Z}_q$
$s = k^{-1}(xr + \alpha) \bmod q$	$\xrightarrow{s}$	$g^{\alpha s^{-1}} y^{rs^{-1}} \stackrel{?}{=} I$

- **正确性**：

$$g^{\alpha s^{-1}} y^{rs^{-1}} = g^{\alpha s^{-1}} g^{xrs^{-1}} = g^{(\alpha+xr)s^{-1}} = g^{(\alpha+xr)k(\alpha+xr)^{-1}} = I$$

DSA 和 ECDSA

设计 (DSA 和 ECDSA 数字签名)

- Gen: 输入 1^n，确定群 $(\mathbb{G}, q, g)$；选择随机数 $x \in \mathbb{Z}_q$ 作为私钥，计算 $y = g^x$ 作为公钥。同时，确定哈希函数 $H\colon \{0,1\}^* \mapsto \mathbb{Z}_q$ 和函数 $F\colon \mathbb{G} \mapsto \mathbb{Z}_q$。
- Sign: 输入私钥 x 和消息 $m \in \{0,1\}^*$，输出签名 (r, s)。

$$k \leftarrow \mathbb{Z}_q \quad r = F(g^k) \quad s = k^{-1}(xr + \alpha) \bmod q$$

- Vrfy: 输入公钥 y，消息 m 和签名 (r, s)，计算 $I = g^s y^{-r}$，输出 $F(g^{H(m)s^{-1}} y^{rs^{-1}}) \stackrel{?}{=} r$。

小结

1. 基本概念
2. 数字签名方案