# 基于错误检验故障的RSA-CRT差分故障分析

陈财森 王 韬 田军舰 张金中
(军械工程学院计算机工程系 石家庄 050003)

**摘 要** 原有基于模幂运算故障的RSA-CRT故障攻击算法,因添加了错误检验操作而失效。为寻找新的故障攻击方法,以BOS防御算法为攻击分析对象,针对在检错步骤产生故障的情况进行分析,建立了基于错误检验故障的攻击模型,提出了能够完整推算出RSA密钥的差分故障攻击算法。进行了推导论证和实验仿真,结果表明原有防御措施并不能有效地抵御故障攻击,新的攻击算法具有良好的可行性,在算法复杂度上,比Wagner的攻击算法需要更少的搜索空间,单个字节故障最多只需要256个样本空间。最后分析了原有防御算法的问题所在,同时给出了相应的防御建议。

**关键词** 旁路攻击,故障模型,差分故障分析,错误检验,中国剩余定理,RSA密码算法
**中图法分类号** TP393.08 **文献标识码** A

## Differential Fault Analysis on RSA-CRT Based on Fault in Error Checking Operation

CHEN Cai-sen WANG Tao TIAN Jun-jian ZHANG Jin-zhong
(Department of Computer Engineering, Ordnance Engineering College, Shijiazhuang 050003, China)

**Abstract** The former fault analysis can't attack on RSA-CRT with corresponding countermeasure. In order to find the new vulnerability to fault analysis, this paper took BOS countermeasure as the analyzed object. An attack model based on fault in error checking operation was advanced, and a differential fault analysis algorithm was given that can completely recover the RSA key. The fact that the previous countermeasures can't effectively resist the differential fault analysis was demonstrated, and the complexity of our attack was estimated both by a theoretical analysis and software simulations. Experiment results show that the new fault analysis algorithm has well feasibility, it requires less faulty signature samples than Wagner's attack algorithm, almost need 256 samples for single byte fault. Finally, a corresponding advice on countermea-sure to differential fault analysis was given by analyzing the problem of previous countermeasures.

**Keywords** Side channel attack, Fault model, Differential fault analysis, Error checking, Chinese remainder theorem, RSA

## 1 引言

自从Kochor提出旁路攻击思想以来[1],出现了各种各样针对嵌入式系统(例如智能卡的)的旁路攻击方法,其中最为强大的是Boneh等人提出的故障攻击[2]。故障攻击注入故障的方式有多种,如电压变化、改变时钟周期、温度变化、激光和X射线、微探测技术等等。这些方式都是为了使攻击对象在执行计算时出错,导致输出错误结果。这些错误信息可能被攻击者用来破解密钥。业已证明该方法可以成功获取RSA和DSA的密钥。

第一个遭受故障攻击的是利用中国剩余定理(CRT)的RSA算法,即RSA-CRT,通过故障攻击可以完全破解密钥。抵御此类攻击最简单的方式是执行两次签名并进行对比。但是这需要消耗两倍的计算时间,而且该方式并不能有效防御参数出错。另外一种常用的防御方式是采用公钥$e$进行签名验证,令设备只有在$S^e=m(\mathrm{mod})N$时才输出结果,但是此方式在$e$值较大时代价较高。此外,一些应用中(例如java卡),在签名过程中并不能访问公钥$e$。后来以Shamir[3]和Johannes Blomer等人[4]为代表,提出了增加错误检验步骤的防御算法,其基本思想都是在最后输出签名结果时进行正确性检验,以保证不输出错误的签名。实验结果表明,采用他们的防御算法能够有效抵御原有故障分析算法的攻击。

本文通过研究现有攻击算法,结合对带错误检验的防御算法的分析,发现现有的防御算法并没有考虑错误检验步骤出故障的情况,仍然存在故障攻击的隐患。在此基础上,建立了新的基于错误检验故障的RSA-CRT故障攻击模型,通过算法分析和推理论证,利用在模幂运算和错误检验步骤中分别注入故障的方法,对故障出现的不同情况进行分析讨论,并

到稿日期:2010-04-21 返修日期:2010-08-01 本文受国家自然科学基金(60772082),河北省自然科学基金(08M010)资助。
陈财森(1983—),男,博士生,主要研究方向为信息安全和公钥密码旁路分析,E-mail:caisenchen@163.com;王 韬(1964—),男,教授,博士生导师,主要研究方向为网络安全与对抗;田军舰(1985—),男,硕士生,主要研究方向为信息安全和公钥密码旁路分析;张金中(1985—),男,硕士生,主要研究方向为信息安全与网络对抗。

给出相应的攻击算法，通过仿真实验，证明了算法的可行性，分析了算法的执行效率，最后提出防御建议，以提高 RSA 算法的安全性。

## 2 RSA-CRT 算法与故障攻击模型

### 2.1 RSA-CRT 算法

RSA 密码算法的安全性是基于大整数因式分解的困难性问题。因式分解 $n=pq$ 等价于计算私钥 $d\in\varphi(n)$，$\varphi(*)$ 表示欧拉函数。相应于私钥 $d$ 的公钥为：$e=d^{-1} \bmod \varphi(n)$。RSA 签名消息 $m$ 的计算结果为 $S=m^d \bmod n$，通过比较计算 $\tilde{m}=S^e \bmod n$ 和 $m$ 进行验证。为提高 RSA 执行效率，采用了中国剩余定理实现的 RSA-CRT 算法，如图 1 所示。

| 输入：p，q，d，n和消息M |
| --- |
| 1、$S_p = M^{d \bmod(p-1)} \bmod p$　$S_q = M^{d \bmod(q-1)} \bmod q$<br>2、$c_p = q(q^{-1} \bmod p)$　$c_q = p(p^{-1} \bmod q)$ |
| 3、$S = CRT_{(p,q)\to n}(S_p, S_q) = c_p.S_p + c_q.S_q \bmod n$ |

图 1　RSA-CRT 算法

由图 1 可知通过 $M^d = CRT_{(p,q)\to n}(S_p, S_q)$ 可以快速实现 RSA 签名。CRT 算法能够提高 RSA 大约 4 倍的执行速度。

### 2.2 RSA-CRT 故障攻击模型

RSA-CRT 算法主要分为模幂运算和组合运算两个阶段，原有的基于模幂运算产生故障的攻击模型如图 2 所示，虚线框中给出模幂运算可能发生故障的两个位置。

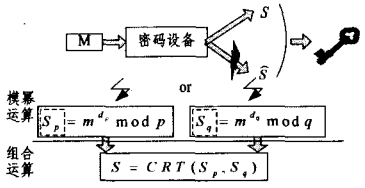


图 2　基于模幂运算的 RSA-CRT 故障攻击模型

Dan Boneh 等人[2]指出 RSA-CRT 算法中只要其中的一个幂运算（$S_p$ 或 $S_q$）发生错误，则模数 $N$ 就可能通过故障签名 $\hat{S}$ 分解。假设计算 $S_p$ 时发生错误，$\hat{S}_p$ 为故障中间值，那么我们可以利用错误签名 $\hat{S}=\mathrm{CRT}(\hat{S}_p, S_p)$ 和正确的签名 $S$，通过计算 $\mathrm{GCD}(S-\hat{S}, N)$ 获取素数 $q$。

Arjen Lenstra 扩展了上面的攻击算法，提出只需要一个故障的签名即可通过 $\mathrm{GCD}(\hat{S}^e-m, N)$ 获取素数 $q$，成功地因式分解模数 $n$，最终推算出密钥 $d$[5]。

## 3 基于错误检验故障的攻击模型

### 3.1 带错误检验的故障攻击防御算法

除了在签名输出之前对同一消息进行两次签名验证和采用公钥对签名进行验证两个简单的防御算法外，Shamir 针对故障攻击算法，给出另一种防御算法，主要思想是选择一个大约 32 位的随机小素数 $r$，通过计算验证模幂运算的正确性，避免输出错误签名[3]。Johannes Blomer 等人分析了 Shamir 的小素数验证方法存在的不足，提出带错误检验的故障攻击防御算法，该算法将防御措施扩展到整个 RSA-CRT 算法当中，包括其中的组合运算步骤，同时能够估计检验步骤的单点故障问题[4]，算法描述为：

**算法 1**　BOS 防御算法[4]

输入：消息 $m$

参量：$p.t_1, q.t_2, N, N.t_1.t_2, d_p, d_q, t_1, t_2, e_1, e_2$

输出：$Sig$

1. $S_p = m^{d_p} \bmod p.t_1$
2. $S_q = m^{d_q} \bmod q.t_2$
3. $S = CRT(S_p, S_q) \bmod N.t_1.t_2$
4. $c_1 = (m - S^{e_1} + 1) \bmod t_1$
5. $c_2 = (m - S^{e_2} + 1) \bmod t_2$
6. $Sig = S^{c_1 \cdot c_2} \bmod N$

其中，$t_1$ 和 $t_2$ 为两个接近 $k$ 位的随机整数，如果没有错误发生，$C_1$ 和 $C_2$ 的计算结果都为 1，最终返回正确的签名。算法在 Shamir 防御算法的基础上，修改了其中 3 个变量，即 $p=p\cdot t_1$，$q=q\cdot t_2$，$N=N\cdot t_1\cdot t_2$，这样的话，在运算过程中任何一个步骤出故障，都将导致一个不可预测的错误签名，使得原先的攻击算法失效。

### 3.2 基于错误检验故障的攻击模型

早在 2004 年，Wagner 等人已经针对算法 1 提出一种攻击算法[6]，算法假设在模幂运算阶段计算 $S_p$ 时一个随机暂时故障修改了 $m$ 的值，然后又恢复正常，导致 $C_1\neq 1, C_2=1$。攻击者通过猜测 $C_1$ 的值，计算 $\gcd(m^{c_1}-\hat{S}^e, N)$ 获取其中一个素数。但是由于 $c_1=(m-S^{e_1}+1) \bmod t_1$，其所要猜测的空间很大。

本文在图 2 攻击模型的基础上，建立针对算法 1 在错误检验步骤注入故障的新的攻击模型，如图 3 所示。

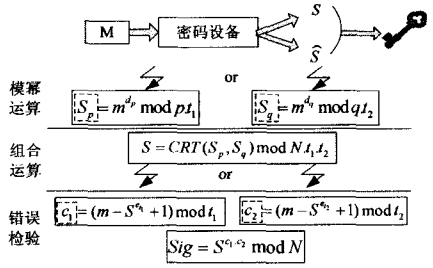


图 3　基于错误检验故障的 RSA-CRT 差分故障攻击模型

带错误检验的防御算法，主要由模幂运算、组合运算和错误检验 3 部分组成。图 3 中虚线框中给出模幂运算和错误检验阶段可能发生故障的 4 个位置，每个阶段的故障各为一个，因此属于二阶故障攻击模型。故障类型可分为永久故障、暂时故障和组合故障 3 种[7]，本文的模型属于组合故障，即一个为永久故障，一个为暂时故障。下面依据图 3 模型，做攻击算法的进一步分析。

### 3.3 故障注入

Johannes Blomer 等人已经论证了 BOS 防御算法能够抵御在 CRT 组合运算阶段产生的故障的攻击[4]，但是依据图 3 建立的攻击模型，通过分析发现 BOS 防御算法仍然存在遭受基于错误检验故障的组合故障攻击的可能性，即注入一个永久故障和一个暂时故障。假设其中受永久故障影响的变量为 $\{pt_1, qt_2\}$ 中的一个，从算法 1 可看出，出错的变量会导致 $S_p$ 或 $S_q$ 出错。假设 $S_p$ 引入的永久故障为 $\delta$，与 $S_q$ 引入故障的情况分析是一样的，因此 $\tilde{S}=0 \bmod q$ 且 $\tilde{S}\neq 0 \bmod p$。另外假设暂时故障在计算 $c_1$ 时发生，引入的故障为 $\delta_t$，$c_2$ 对应于 $S_q$ 出错时发生的故障。当无法判断 $S_p$ 还是 $S_q$ 发生错误时，可通过分别对 $c_1$ 或 $c_2$ 持续注入故障来加以猜测。同时假设引入永久故障 $\delta$ 和暂时故障 $\delta_t$ 时，在 BOS 防御算法计算过程

中产生的中间故障变量有 $\Delta,\tilde{\delta}$和$\delta_2$。

### 3.4 攻击算法

根据注入的故障，由 BOS 算法可以得到：

$$\tilde{S}_p=S_p+\delta=m^{d_p}+\delta \bmod pt_1$$

$$S_q=m^{d_q} \bmod qt_2$$

$$\begin{aligned}\tilde{S}&=\mathrm{CRT}_{(pt_1,pt_2)\to nt_1t_2}(\tilde{S}_p,S_q)\\&=\mathrm{CRT}_{(pt_1,pt_2)\to nt_1t_2}(S_p,S_q)+\delta t_2q((t_2q)^{-1} \bmod pt_1) \bmod nt_1t_2\\&=S+\tilde{\delta}q \bmod nt_1t_2\end{aligned}$$

$$c_1=m-\tilde{S}^{e_1}+1+\delta_t \bmod t_1=\delta_2+\delta_t \bmod t_1$$

$$c_2=m-\tilde{S}^{e_2}+1 \bmod t_2=1 \bmod t_2$$

$$\widetilde{Si}_g=\tilde{S}^{\delta_2+\delta_t} \bmod n=(S+\tilde{\delta}q)^{\delta_2+\delta_t} \bmod n=S^{\delta_2+\delta_t}+q\Delta \bmod n$$

式中，$q\Delta$ 表示计算$(S+\tilde{\delta}q)^{\delta_2+\delta_t}$时，除 $S^{\delta_2+\delta_t}$一项之外的其它项，因此 $\delta_t$ 的一个小小的变化将明显改变 $\Delta$，而且由于 $\delta_2$，$\delta_t$，$\Delta$ 都是未知的，$q$ 不能直接通过$\widetilde{Si}_g$ 恢复。结合图 3 建立的攻击模型，在一个永久故障 $\delta$ 的基础上，可以通过两次注入暂时故障 $\delta_t^1$ 和 $\delta_t^2$的方法获得同一个消息的两个错误签名$\widetilde{Si}_{g1}$和$\widetilde{Si}_{g2}$，于是有：

$$\widetilde{Si}_{g1}(\widetilde{Si}_{g2})^{-1}=\frac{(S^{\delta_2+\delta_t^1}+\Delta^1q)}{(S^{\delta_2+\delta_t^2}+\Delta^2q)} \bmod n$$

$$=S^{\delta_t^1-\delta_t^2}\left(\frac{S^{\delta_2+\delta_t}{}^2}{S^{\delta_2+\delta_t}{}^2+\Delta^2q}\right)+q\left(\frac{\Delta^1}{S^{\delta_2+\delta_t^2}+\Delta^2q}\right) \bmod n$$

$$=S^{\delta_t^1-\delta_t}{}^2\left(\frac{S^{\delta_2+\delta_t}{}^2+\Delta^2q-\Delta^2q}{S^{\delta_2+\delta_t^2}+\Delta^2q}\right)+q\left(\frac{\Delta^1}{S^{\delta_2+\delta_t^2}+\Delta^2q}\right) \bmod n$$

$$=S^{\delta_t^1-\delta_t^2}+q\left(\frac{-S^{\delta_t^1-\delta_t^2}\Delta^2+\Delta^1}{S^{\delta_2+\delta_t^2}+\Delta^2q}\right) \bmod n$$

假如知道 $\delta_t^1-\delta_t^2$的值，那么可以通过计算公式：

$$\gcd(\widetilde{Si}_{g1}(\widetilde{Si}_{g2})^{-1}-S^{\delta_t^1-\delta_t^2},n)=q \tag{1}$$

因式分解模数 $n$，也可以修改为如下的表达公式：

$$\gcd((\widetilde{Si}_{g1}(\widetilde{Si}_{g2})^{-1})^e-m^{\delta_t{}^1-\delta_t{}^2},n)=q \tag{2}$$

那么只要满足$\frac{-S^{\delta_t^1-\delta_t^2}\Delta^2+\Delta^1}{S^{\delta_2+\delta_t^1}+\Delta^2q}\neq 0 \bmod p$ 的条件，剩下的工作就是处理 $\delta_t^1-\delta_t^2$的差值。

假设注入的暂时故障使得一个未知的字节发生错误，即 $\delta_t^1$与 $\delta_t^2$之间的差异是一个字节。可建立一个包含存储所有错误签名$\widetilde{Si}_{gl}$的列表，当通过式(2)未能因式分解 $n$ 时，产生一个新的错误签名$\widetilde{Si}_{gk}$，对于列表中的所有签名$\widetilde{Si}_{gl}$分别计算 $\gcd(\widetilde{Si}_{gl}(\widetilde{Si}_{gk})^{-1}-S^{\delta_t^l-\delta_t^k},n)$和 $\gcd(\widetilde{Si}_{gk}(\widetilde{Si}_{gl})^{-1}-S^{\delta_t^k-\delta_t^l},n)$，尝试因式分解 $n$，假如没有成功则将$\widetilde{Si}_{gk}$添加到列表中，并产生新的错误签名，循环前面的操作。具体攻击流程如图 4 所示。

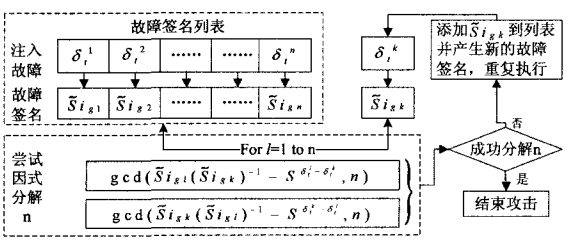


图 4 故障攻击算法执行流程图

## 4 实验结果及讨论分析

### 4.1 仿真实验结果及对比分析

根据本文前面的攻击算法，在 Windows xp 环境下，在普通 PC 机(CPU 为 AMD Athlon 64 3000+，1G 内存)上使用 VC++ 6.0 结合 OpenSSL 密码库实现了 BOS 防御算法，设计并仿真实现了本文提出的故障攻击算法。利用软件模拟故障诱导过程，通过在模幂运算阶段注入永久故障，在错误检验阶段注入随机单字节暂时故障，获取对应的故障签名$\hat{S}$，依据攻击算法尝试因式分解模数 $N$，进而得到私钥 $d$。假设 RSA 密钥长度为 1024 位，算法 1 中 $t_1$ 和 $t_2$ 为 $k$ 位的两个随机整数，那么和 Wagner[6] 提出的算法相比，因式分解 $N$ 所需的搜索空间如表 1 所列。

表 1 两种攻击算法的搜索空间

| 攻击算法 | 故障类型 | 条件 | 搜索空间 |
|---|---|---|---|
| 文献[6] | 暂时故障 $S_p$ 或 $S_q$ | $C_1$ 的值未知 | $2^k$ |
| 本文 | 永久故障{pt1，qt2} | $\delta_t^1-\delta_t^2$已知 | 1 |
| | 和暂时故障{c1，c2} | $\delta_t^1-\delta_t^2$未知 | 256 |

Wagner 提出的攻击算法，为了成功实现攻击，必须对 $C_1$ 值依据 $c_1=(m-S^{e_1}+1) \bmod t_1$ 进行遍历，而 $C_1$ 是由故障值$\hat{S}$扩散的，因此其搜索空间与 $t_1$ 的位数相关。当 $t_1$ 为 80 位长度的素数时，对于 1024 的 RSA 密钥，大约只有 4%的故障是对攻击有用的。而本文的攻击算法，当 $\delta_t^1-\delta_t^2$已知时，只需要两个不同暂时故障对应的签名和一个正确签名即可；当 $\delta_t^1-\delta_t^2$未知时，理论推导所需故障数目为 256。但实际仿真实验时，采用了图 4 的循环操作算法，实验结果表明在注入 25 个故障时攻击成功率已经达到 93%，通过重复对比原有错误签名与新的错误签名猜测 $\delta_t^1-\delta_t^2$的差值，使得注入 $C_1$ 或 $C_2$ 的每个暂时故障都可用来参与计算，通过优化计算减少注入的故障次数，提高攻击效率。

### 4.2 故障攻击算法的进一步讨论分析

基于错误检验的 RSA 攻击算法，是建立在组合故障模型的基础上，其中永久故障 $\delta$ 注入在 $S_p$ 或 $S_q$ 中，暂时故障 $\delta_t$ 注入在 $c_1$ 或 $c_2$ 中，并且两个故障是对应发生的。假如把永久故障注入在 $c_1$ 或 $c_2$ 中，暂时故障注入在 $S_p$ 或 $S_q$ 中，同样可以采用新给出的算法执行攻击。但是无论同时引入两个永久故障还是两个暂时故障，都会使本文的攻击算法失效。假设 $S_p$ 引入的故障为 $\delta_1$，$S_q$ 引入的故障为 $\delta_2$，其它的$\tilde{\delta}'$，$\Delta'$和 $\Delta$ 为计算过程中的中间变量，那么依据算法 1 可以得到：

$$\tilde{S}_p=S_p+\delta_1=m^{d_p}+\delta \bmod pt_1$$

$$\tilde{S}_q=S_q+\delta_2=m^{d_q}+\delta_2 \bmod qt_2$$

$$\begin{aligned}\tilde{S}&=\mathrm{CRT}_{(pt_1,pt_2)\to nt_1t_2}(\tilde{S}_p,\tilde{S}_q)\\&=\mathrm{CRT}_{(pt_1,pt_2)\to nt_1t_2}(S_p,S_q)+\delta_2+(\delta_1-\delta_2)t_2q((t_2q)^{-1} \bmod pt_1) \bmod nt_1t_2\\&=S+\delta_2+\tilde{\delta}'q \bmod nt_1t_2\end{aligned}$$

当 $\delta_2$ 存在时，即 $\delta_2\neq 0$，得到$\widetilde{Si}_g=S^{\delta_2+\delta_t}+\Delta'+q\Delta \bmod n$，只有当攻击者能够控制 $\Delta'$时才有可能采用本文的攻击算法获取 RSA 密钥。

### 4.3 防御建议

先前抵御 RSA-CRT 故障攻击的防御算法大多可以分为 3 个部分：首先是 $S_p$ 和 $S_q$ 两个幂运算。第二是计算签名 $Sig$

的组合运算。最后是检验故障的步骤，又分为两种情况，一种是采用条件检验是否有故障发生，如 Shamir 的防御算法，假如发生则不输出签名；另一种是产生随机数，假如有故障产生则用随机数替代签名，否则输出正确的签名。攻击者若能够在前面的操作步骤注入故障，且跳过检验步骤或检验步骤出错，则仍然可以实施故障攻击。新的攻击算法也证明了原有的防御算法并不能有效地抵御故障分析，主要原因是原有的防御措施都是在计算最终签名结果之后才进行错误检验，最终都可以通过注入故障和计算 $f()$ 得到如下的形式：

$f(\tilde{S})=\delta+q\Delta \bmod n$ 或 $f(\tilde{S})=\delta+p\Delta \bmod n$

再通过其它的组合计算 $F()$ 获得 $\gcd(F(f(\tilde{S})),n)=p$ 或 $q$，从而因式分解 $n$。

本文提出一种可行的防御措施建议：核心算法是让最终签名结果的计算在故障检验步骤之后进行，具体为：首先在组合步骤之前产生随机数，计算出一个中间结果值 $Sig^*$，然后执行故障检验步骤，假如有故障则不输出结果，否则再通过计算消除随机数，并输出正确的签名 $Sig$。

**结束语** 本文提出了针对 RSA-CRT 的故障攻击算法，介绍了基于模幂运算故障的攻击算法以及对应的防御算法，其中以 BOS 算法为分析对象，建立了新的基于错误检验故障的 RSA 故障攻击模型。与 Wagner 的攻击算法不同，本文给出了永久故障与暂时故障相结合的故障攻击算法，并进行了理论推导和仿真实验，通过实验对比了本文攻击算法与原有攻击算法的复杂度和故障利用率，实验结果验证了本文攻击算法的可行性，且比 Wagner 的攻击算法具有更高的执行效率，执行攻击所需的故障签名数更少。为了更全面地讨论攻击算法，应针对其它可能出现的故障情况作进一步分析，最后总结故障攻击的核心思想，分析了原有防御算法的不足，提出了防御建议。

目前本文仅从理论和仿真实验的角度针对基于错误检验故障的 RSA 故障攻击进行研究，研究了故障可能出现的新情况，给出了相应的攻击算法。但实际过程中故障出现的情况是多变的，针对不同情况的故障还需要做不同的分析，另外关于如何在适当的位置和时机注入故障等方面还有待进一步研究，为抵御新的防御算法设计一种有效的防御算法也是将来值得研究和关注的。

## 参 考 文 献


[1] Kocher P. Timing attack on Implementations of Diffie-Hellman, RSA, DSS, and Other systems[C]// Proceedings of Advances in Cryptology-CRYPTO'96. Springer-verlag, 1996: 104-113

[2] Boneh D, DeMillo R, Lipton R. On the Importance of Checking Cryptographic Protocols for Faults[C]// Fumy W, ed. Advances in Cryptology—Eurocrypt ' 97, LNCS1233. Springer-Verlag, 1997: 37-51

[3] Shamir A. How to Check Modular Exponentiation[C]// EUROCRYPT ' 97. Springer-Verlag, 1997

[4] Blömer J, Otto M, Seifert J-P. A New RSA-CRT Algorithm Secure Against Bellcore Attacks[C]// Sushil Jajodia, Vijayalakshmi Atluri, Trent Jaeger, eds. Proceedings of the 10th ACM Conference on Computer and Communications Security, CCS 2003. Washington, DC, USA, October 2003: 311-320

[5] Lenstra A K. Memo on RSA Signature Generation in the Presence of Faults[EB/OL]. http://cm. bell-labs. com/who/akl/ September 1996

[6] Wagner D. Cryptanalysis of a Provably Secure RSA-CRT Algorithm[C]// Vijayalakshmi Atluri, Birgit Pfitzmann, and Patrick Drew McDaniel, eds. Proceedings of the 11th ACM Conference on Computer and Communications Security, CCS 2004. Washington, DC, USA, October 2004: 92-97

[7] Schmidt J-M. Differential Fault Analysis [R]. A report from IAIK Lab in Austria. June 2008. http://www. a-sit. at/pdfs/ DFA-Report. pdf


(上接第 64 页)


[9] Symantec Inc, W32. HLLW. Fizzer@mm[OL]. http://security-response. symantec. com/avcenter/venc/data/w32. hllw. fizzer @mm. html

[10] KaZaA[OL]. http://www. kazaa. com

[11] Symantec Inc. , W32. Gnuman. Worm[OL]. http://securityresponse. symantec. com/avcenter/venc/data/w32. gnuman. worm. html

[12] Zhou Li-dong. A First Look at Peer-to-Peer Worms Threats and Defenses[C] // Miguel Castro, Robbert van Renesse, IPTPS 2005. New York, 2005: 24-35, http://research. microsoft. com/ users/lidongz/p2pworm. pdf

[13] 郑辉. Internet 蠕虫研究[D]. 天津：南开大学信息技术科学学院，2003

[14] HP Inc，基于病毒遏制技术(Virus-Throttling)的连接速度过滤[OL]. http://www. hp. com. cn/network _ pmg/pdfs/virus _ throttling_tech_brief. pdf

[15] 郑辉，吕冠一. Google Hacking 与智能蠕虫防治[J]. 信息安全与通信保密，2005，8：70-74

[16] 郑辉. DNS 抑制蠕虫传播[C]// XCON'2003. 2003

[17] Whyte D, Kranakis D, van Oorschot P C. DNS-based Detection of Scanning Worms in an Enterprise Network[C] // The 12th Annual Network and Distributed System Security Symposium (DNSS'05). San Diego, 2005

[18] Provos N A. virtual honeypot framework[R]. 03-1. Center of Information Technology Integration, University of Michigan, 2003

[19] Barber B. Cheese Worm: Pros and Cons of a ''Friendly "Worm [C]// SANS Conferences 2001. SANS Institute, 2001

[20] Symantec Inc, W32. Welchia. Worm. http://www. symantec. com/avcenter/venc/data/w32. welchia. worm. html

[21] Kern M. Codegreen beta release. 2001[OL]. http://online. securityfocus. com/archive/82/211462

[22] 文伟平，卿斯汉，蒋建春，等. 网络蠕虫研究与进展[J]. 软件学报，2004，15(8)：1208-1218

[23] Castaneda F, Sezer E C, Xu Jun. WORM VS WORM: Preliminary Studay of an Active Counter-Attack Mechanism[C]// Proc. of the ACM CSS Workshop on Rapid Malcode (WORM2004). Washington, 2004

[24] Mituzuas D. FreeBSD Scalper Worm[OL]. http://www. dammit. lt/apache-worm/

[25] F secure Inc. Global slapper worm information center[OL]. http://www. f-secure. com/slapper/

[26] Tian Bu, Towsley D. On distinguishing between Internet power law topology generators[C] // Proc. of the IEEE INFOCOM 2002. New York, IEEE, 2002: 638-647