Towards a Provably Secure DoS-Resilient Key Exchange Protocol with Perfect Forward Secrecy

Lakshmi Kuppusamy, Jothi Rangasamy, Douglas Stebila, Colin Boyd, and Juan Gonzalez Nieto

Information Security Institute, Queensland University of Technology,
GPO Box 2434, Brisbane, Queensland 4001, Australia
{l.kuppusamy,j.rangasamy,stebila,c.boyd,j.gonzaleznieto}@qut.edu.au

Abstract. Just Fast Keying (JFK) is a simple, efficient and secure key exchange protocol proposed by Aiello et al. (ACM TISSEC, 2004). JFK is well known for its novel design features, notably its resistance to denial-of-service (DoS) attacks. Using Meadows' cost-based framework, we identify a new DoS vulnerability in JFK. The JFK protocol is claimed secure in the Canetti-Krawczyk model under the Decisional Diffie-Hellman (DDH) assumption. We show that security of the JFK protocol, when reusing ephemeral Diffie-Hellman keys, appears to require the Gap Diffie-Hellman (GDH) assumption in the random oracle model. We propose a new variant of JFK that avoids the identified DoS vulnerability and provides perfect forward secrecy even under the DDH assumption, achieving the full security promised by the JFK protocol.

Keywords: Denial of service, Meadows' cost-based framework, Just Fast Keying, client puzzles, key agreement, perfect forward secrecy

1 Introduction

Denial-of-service (DoS) attacks that are mounted to exhaust the processing, memory, or network resources of target systems have become a common occurrence. These attacks can easily disable servers, so it is important for the server to detect and filter bogus connection requests as early as possible. Cryptographic techniques such as authentication can assist a server in detecting bogus connections, but the computationally expensive operations involved in authentication may open a new DoS vulnerability. Hence care must be taken to design a protocol that implements defense strategies to efficiently tackle DoS attackers and to protect itself from exhausting the resources.

A number of DoS countermeasures such as client puzzles [3, 11, 20], stateless connections [2] and gradual authentication [14, 19] are available for building DoS-resilient protocols. Resistance to DoS attacks is the main design goal for network protocols such as JFK [1], CA-RSA [8], IKE [10], IKEv2 [12], MIKE [13] and HIP [15]. Recently, Stebila et al. [23] described a model for assessing DoS-resistance in protocols by following the approach of Stebila and Ustaoglu [22]. They also gave a generic construction for DoS-resistant protocols.

Just Fast Keying (JFK) is a simple, efficient and secure key exchange protocol proposed by Aiello et al. [1]. JFK is well known for its novel design features, including its resistance to DoS attacks. The designers of JFK provided a careful analysis of its security properties but stopped short of providing a fully formal security proof. They pointed out that the basic structure of JFK, what they called the "cryptographic core", already has a proof of its key exchange security properties [6], and that the additional mechanisms of JFK beyond this core will not degrade key exchange security.

The JFK designers also discuss formal approaches to other protocol properties, namely privacy and forward secrecy. However, they do not provide any formal discussion of the DoS-resilience of JFK, one of its main properties. The main technique for achieving DoS-resilience in JFK is the reuse of ephemeral public keys, but this comes at the obvious expense of perfect forward secrecy.

Smith et al. [21] performed a detailed analysis of JFK's DoS resistance using Meadows' cost-based modelling framework [14] and identified two DoS attacks against the protocol that are possible when an attacker is willing to reveal the source IP address. The first attack was found with the direct application of Meadows' cost-based framework and the second attack was possible considering the presence of co-ordinated DoS attackers.

Contributions. The main goal of our work is to improve the DoS resistance of JFK, although our approach can be applied to any Diffie-Hellman (DH)-based protocols. Our contributions can be summarised as follows:

- We show that the security of JFK protocol with ephemeral DH reuse appears to require the GDH assumption in the random oracle model.
- We also analyse the JFK protocol in detail with the help of Meadows' cost-based framework and identify a new DoS vulnerability in JFK which is possible in the presence of co-ordinated DoS attackers.
- We propose a variant of JFK that efficiently generates fresh ephemeral keys from precomputed values using a technique of Boyko et al. [5] up to 3.4 times faster than in JFK. The proposed JFK variant not only avoids the identified DoS attack but also achieves both the security level promised by the original JFK protocol and perfect forward secrecy. We give a detailed yet simple security analysis of the resulting BPV-JFK protocol in the Canetti-Krawczyk model under the DDH assumption in the standard model.
- We implement the generic technique of Stebila et al. [23] in JFK and BPV-JFK to achieve strong DoS resilience in the Stebila et al. model.

A comparison of the security properties of JFK and the protocols proposed in this paper appears in Table 1.

2 Just Fast Keying Protocol and its DoS Vulnerabilities

Aiello et al. [1] presented two variants of the JFK protocol namely JFKi and JFKr, both with same level of DoS resistance; we focus on the JFKi variant, which appears in Figure 1.

Protocol	Cost-based vulnerability	Security assumptions	Perfect Forward Secrecy	DoS-resilience
JFK [1]	Yes	GDH, ROM	Only with no reuse	No
DoS-JFK (§ 2.5)	No	GDH, ROM	Only with no reuse	Yes
BPV-JFK (§ 3)	No	DDH	Yes	No
DoS-BPV-JFK (§ 3.6)	No	DDH	Yes	Yes

Table 1. Comparison of properties of JFK-based protocols

1. $\mathbf{I} \rightarrow \mathbf{R}$: $N_{\mathbf{I}}^{'}$, g^x, $\mathsf{ID}'_{\mathbf{R}}$
2. $\mathbf{R} \rightarrow \mathbf{I}$: $N'_{\mathbf{I}}$, $N_{\mathbf{R}}$, g^y, $\mathsf{grpinfo}_{\mathbf{R}}$, $\mathsf{ID}_{\mathbf{R}}$, $\mathsf{S}_{k_{\mathbf{R}}}[g^y, \mathsf{grpinfo}_{\mathbf{R}}]$, token
3. $\mathbf{I} \rightarrow \mathbf{R}$: $N_{\mathbf{I}}$, $N_{\mathbf{R}}$, g^x, g^y, $\{\mathsf{ID}_{\mathbf{I}}, \mathsf{sa}, \mathsf{S}_{k_{\mathbf{I}}}[N'_{\mathbf{I}}, N_{\mathbf{R}}, g^x, g^y, \mathsf{ID}_{\mathbf{R}}, \mathsf{sa}]\}_{K_a}^{K_e}$, token
4. $\mathbf{R} \rightarrow \mathbf{I}$: $\{\mathsf{S}_{k_{\mathbf{R}}}[N'_{\mathbf{I}}, N_{\mathbf{R}}, g^x, g^y, \mathsf{ID}_{\mathbf{I}}, \mathsf{sa}, \mathsf{sa}']\}_{K_a}^{K_e}$

$\mathsf{token} = H_{\mathsf{HKR}}(g^y, N_{\mathbf{R}}, N'_{\mathbf{I}}, IP_{\mathbf{I}})$, $K_e = H_{g^{xy}}(N'_{\mathbf{I}}, N_{\mathbf{R}}, 1)$
$K_a = H_{g^{xy}}(N'_{\mathbf{I}}, N_{\mathbf{R}}, 2)$, $K_{xy} = H_{g^{xy}}(N'_{\mathbf{I}}, N_{\mathbf{R}}, 0)$

$N_{\mathbf{I}}, N_{\mathbf{R}}$	Random nonces chosen by $\mathbf{I}$ and $\mathbf{R}$respectively.
$H(\cdot)$	Unkeyed hash function.
$N'_{\mathbf{I}}$	Initiator's initial nonce, computed as $H(N_{\mathbf{I}})$.
g	Generator of a multiplicative group of order q.
g^x, g^y	Public key of the initiator and the responder.
HKR	Transient, hash key known only to the responder.
$H_K(\cdot)$	Keyed hash function (secure MAC) using key K.
$IP_{\mathbf{I}}$	Initiator's network address.
$\mathsf{ID}_{\mathbf{I}}, \mathsf{ID}_{\mathbf{R}}$	Information to identify $\mathbf{I}$ and $\mathbf{R}$ public keys.
$\mathsf{ID}'_{\mathbf{R}}$	Initiator indicates to the responder which authentication information (e.g. certificates) should be used.
$\mathsf{sa}, \mathsf{sa}'$	Information used to establish a security association.
$\mathsf{grpinfo}_{\mathbf{R}}$	Groups, algorithms and hash functions supported by the responder,.
$\mathsf{S}_{k_{\mathbf{I}}}[\cdot]$	Digital signature computed by $\mathbf{I}$ using long-term secret key $k_{\mathbf{I}}$.
$\mathsf{S}_{k_{\mathbf{R}}}[\cdot]$	Digital signature computed by $\mathbf{R}$ using long-term secret key $k_{\mathbf{R}}$.
$\{\cdot\}_{K_a}^{K_e}$	Encryption using key K_e and generating a message authentication code (MAC) over the resultant cipher text using key K_a.
k_{xy}	Session key computed using shared secret g^{xy}.

Fig. 1. JFKi protocol and its message components

The main technique for improving efficiency in the JFK protocols is for the participants to reuse their Diffie-Hellman (DH) exponentials (g^x and g^y) across multiple sessions with different parties. In the case of the responder, this technique drastically reduces the computational load by avoiding the exponentiation cost due to the generation of new g^y for each session and by allowing the signature $\mathsf{S}_{k_{\mathbf{R}}}[g^y, \mathsf{grpinfo}_{\mathbf{R}}]$ to be kept constant as long as the DH exponential is reused. Unfortunately, this technique eliminates forward secrecy since y becomes part of the long-term key. Interestingly, JFK with ephemeral DH reuse achieves adaptive forward secrecy (AFS), the term defined by Aiello et al. [1] as below:

Definition 1 (AFS). *If an adversary knows the DH exponent during the time period t_{i+1} and if all the session keys generated during previous time periods $t_0, t_1, \ldots t_i$ remain secure, then the protocol is said to achieve AFS.*

2.1 Smith et al.'s Analysis of JFK in Meadows Framework

One of the design features of JFK is to delay the computational commitments at the responder until the initiator reveals the IP address. Smith et al. [21] argued that the protocols should resist DoS attacks given the ready availability of IP addresses and in the presence of the adversary willing to reveal the source IP address. A direct application of Meadows' cost-based framework to the JFK protocol captured the first attack. The second attack was identified when assessing DoS resistance of the protocol in the presence of coordinated attackers.

1. $\mathbf{I} \rightarrow \mathbf{R}$: $\mathsf{compnonce1}(N_\mathbf{I}), N'_\mathbf{I} = \mathsf{hash1}(N_\mathbf{I})$, $\mathsf{genexp1}(g^x)$ ||
   $N'_\mathbf{I}, g^x$ || $\mathsf{vergroup}(g^x), \mathsf{accept1}$
2. $\mathbf{R} \rightarrow \mathbf{I}$: $\mathsf{compnonce2}(N_\mathbf{R})$, $\mathsf{token} = \mathsf{genmac1}_{\mathsf{HKR}}(g^y, N_\mathbf{R}, N'_\mathbf{I}, IP_\mathbf{I})$ ||
   $N'_\mathbf{I}$, $N_\mathbf{R}$, g^y, $\mathsf{grpinfo}_\mathbf{R}$, $\mathsf{ID}_\mathbf{R}$, $sig1 = \mathsf{S}_{k_\mathbf{R}}[g^y, \mathsf{grpinfo}_\mathbf{R}]$, token ||
   versig1, accept2
3. $\mathbf{I} \rightarrow \mathbf{R}$: $\mathsf{genDH1}(g^{xy})$, $K = \mathsf{compKeys1}(N'_\mathbf{I}, N_\mathbf{R}, g^{xy})$,
   $s = \mathsf{gensig2}(N'_\mathbf{I}, N_\mathbf{R}, g^x, g^y, \mathsf{ID}_\mathbf{R}, \mathsf{sa})$, $C_1 = \mathsf{encrypt1}_K(\mathsf{ID}_\mathbf{I}, s, \mathsf{sa})$,
   $C = \mathsf{genmac2}_K(C_1)$ || $N_\mathbf{I}$,$N_\mathbf{R}$, g^x, g^y, token, C_1, C ||
   $N'_\mathbf{I} = \mathsf{hash}_2(N_\mathbf{I})$, $\mathsf{ver1}(\mathsf{token} = \mathsf{genmac3}_{\mathsf{HKR}}(g^y, N_\mathbf{R}, N'_\mathbf{I}, IP_\mathbf{I}))$,
   $\mathsf{genDH2}(g^{xy})$, $K = \mathsf{compKeys2}(N'_\mathbf{I}, N_\mathbf{R}, g^{xy})$,
   $\mathsf{ver2}(C = \mathsf{genmac4}_K(C_1))$, $\mathsf{decrypt1}_K(C_1)$, $\mathsf{versig2}(s), \mathsf{accept3}$
4. $\mathbf{R} \rightarrow \mathbf{I}$: $W = \mathsf{gensig3}(N'_\mathbf{I}, N_\mathbf{R}, g^x, g^y, \mathsf{ID}_\mathbf{I}, \mathsf{sa}, \mathsf{sa}')$, $D_1 = \mathsf{encrypt2}_K(W, \mathsf{sa}')$,
   $D = \mathsf{genmac5}_K(D_1)$ || D_1, D || $\mathsf{ver}(D = \mathsf{genmac6}_K(D_1))$,
   $\mathsf{decrypt2}_K(D_1)$, $\mathsf{versig3}(W)$, accept4

compnonce :	selecting a random bit string. $\delta(\mathsf{compnonce}) = \textit{cheap}$.
hash :	computing a hash value. $\delta(\mathsf{hash}) = \textit{cheap}$.
genmac :	generating a MAC that is based on a keyed hash function. $\delta(\mathsf{generatemac}) = \textit{medium}$.
gensig :	generating a digital signature. $\delta(\mathsf{gensig}) = \textit{expensive}$.
genexp :	computing a DH ephemeral key. $\delta(\mathsf{genexp}) = \textit{expensive}$.
genDH :	generating a shared secret. $\delta(\mathsf{genDH}) = \textit{expensive}$.
versig :	verifying a digital signature. $\delta(\mathsf{versig}) = \textit{expensive}$.
vergroup :	verifying that a given DH value belongs to an acceptable group. $\delta(\mathsf{vergroup}) = \textit{medium}$.
compKeys :	computing the keys for encryption and authentication is based on a keyed hash function. $\delta(\mathsf{compKeys}) = \textit{medium}$.
encrypt :	encrypt the message using symmetric key. $\delta(\mathsf{decrypt}) = \textit{medium}$.
decrypt :	decrypt the message using symmetric key. $\delta(\mathsf{decrypt}) = \textit{medium}$.

Fig. 2. Annotated Alice-and-Bob Specification of JFKi

Message Processing Costs in JFK. Meadows used the Alice-and-Bob specification style (for more details refer to Appendix A) to show how messages are processed during the protocol execution. In Figure 2, the annotated Alice-and-Bob specification of JFK protocol is provided [21]. The cost-based framework assigns cost to every action performed by the communicating principals. The cost for performing a particular action is defined to be an element in the set $\mathcal{S} = \{cheap, medium, expensive\}$ [21]. The cost function δ (defined in [21]) associated with performing the following actions also appears in Figure 2.

The public-key validation (vergroup in Figure 2) should not be expensive to avoid DoS-attack. Since the adversary is capable of spoofing messages and signatures for which verifications result in failures we have $\delta(\mathsf{spoof}) = cheap$.

2.2 A New DoS Vulnerability in JFK

The main motivation behind the reuse technique in JFK is to protect the responder from a certain type of DoS attack. However, the re-using technique introduces an attack we describe in this section. Smith et al. identified a second DoS attack on JFK under the assumption that both the initiator and the responder are reusing their ephemeral DH exponentials. However a similar type of DoS attack is possible in the case where the initiator does not reuse the ephemeral DH exponential while the responder reuses its ephemeral DH exponential.

To be precise, the initiator generates g^x for the first session and then computes $g^{2x}, g^{3x}, \ldots$ for further sessions using the previously computed DH values. In this way, the ephemeral DH values of the attackers looks different. On the other hand, the responder computes g^y once and reuses it for all sessions it participates during a time period. In this attack, the goal of the attackers is to trigger the responder to perform the expensive signature verification versig2.

The responder performs computationally expensive operations like genDH2 and versig2 only in step 3 of the protocol (Figure 2). To make the responder engage in signature verification versig2, an attacker must have performed medium to expensive operations such as genMac2, encrypt1, genexp1, genDH1 and gensig2.

Though a DoS attacker is capable of sending a spoof signature spoofgensig2 instead of gensig2, it must perform expensive computations such as genexp and genDH1 to convince the responder to perform up to versig2 computation. If an attacker can somehow decrease the cost of computing genexp and genDH1, then it can cause the responder to perform many expensive versig2 computations. Considering the presence of coordinated initiators, the cost for performing genexp and genDH1 computations can be amortised across all attackers resulting in many cheaply generated communications.

2.3 Cost Calculations

We follow Meadows' framework modified by Smith et al. [21] to demonstrate the impact of sharing Diffie-Hellman values by a group of coordinated initiators and analyse DoS susceptibility. For further details of the notations used in this section, refer to Appendix A.

Assume that there are n coordinated initiators/attackers who generate g^x and multiply g^x $n-1$ times to compute the remaining $n-1$ Diffie-Hellman values say $g^{2x}, g^{3x}, \ldots g^{nx}$. The coordinated initiators engage in protocol runs with the responder that is currently re-using its Diffie-Hellman value g^y for reducing its computational burden. The initiators compute shared secrets $g^{xy}, \ldots g^{nxy}$ in a similar way as above. As a result, the cost for computing genexp in message 1 and genDH in message 3 are amortised across all attackers. For larger values of n, the attackers cost function for sharing genexp denoted by $\phi(\mathsf{shareexp})$ is:

$$\begin{aligned}\phi(\mathsf{shareexp}) &= \frac{1}{n} \times \phi(g^x) + (n-1) \times \phi(\mathsf{modmul}) \\ &= \frac{1}{n} \times \mathit{expensive} + (n-1) \times \mathit{cheap} = \mathit{cheap}\end{aligned}$$

as computing modular multiplication $\phi(\mathsf{modmul})$ is *cheap*. The cost for sharing genDH is $\phi(\mathsf{shareDH}) = \mathit{cheap}$.

Hence the attack cost function Θ for the attacker in constructing n valid-looking third messages to convince the responder to perform up to n decrypt1 operations and proceed with n versig2 operations is dominated by events of medium costs. Hence the attacker's total cost for performing a single decrypt1 operation in order to trigger the responder to proceed with versig2 is:

$$\begin{aligned}\Theta(\mathsf{decrypt1}) = &\phi(\mathsf{shareDH}) + \delta(\mathsf{compKeys1_K}) + \phi(\mathsf{spoofsig}) \\ &+ \delta(\mathsf{encrypt1}) + \delta(\mathsf{genmac2}) = \mathit{medium} \qquad (1)\end{aligned}$$

On the other hand, to detect that the n signatures are spoofed, the responder does the following computation n times including hash2, genmac3, genDH2, decrypt1, and versig2. Hence a single message processing cost $\delta'(\mathsf{versig2})$ for the responder is dominated by events of two expensive costs:

$$\begin{aligned}\delta'(\mathsf{versig2}) = &\delta(\mathsf{hash2}) + \delta(\mathsf{genmac3}) + \delta(\mathsf{genmac4}) + \delta(\mathsf{genDH2}) \\ &+ \delta(\mathsf{compKeys2}) + \delta(\mathsf{decrypt1}) + \delta(\mathsf{versig2}) \\ &= \mathit{expensive} \qquad (2)\end{aligned}$$

From Equations (1) and (2), we see that $(\delta'(\mathsf{versig2}), \Theta(\mathsf{decrypt1}))$ which is equal to $(\mathit{expensive}, \mathit{medium})$ does not belong to the tolerence relation $\mathcal{T}$ defined in [14] and in Appendix A; hence the protocol is vulnerable to a DoS attack.

2.4 Basic Security of the JFK Protocol

Aiello et al. [1] compared the basic JFK protocol with the ISO 9798-3 protocol. Since the ISO 9798-3 protocol is SK-secure in the UM model under the DDH assumption, they argued that the JFK protocol inherits the same security. The main design feature of the JFK protocol is that the responder can reuse its ephemeral DH key when it is under attack. If the security of the basic JFK (ISO 9798-3) protocol is analysed with the assumption that the responder reuses its ephemeral DH key, then its SK security may not be reduced to DDH assumption.

Moreover, Aiello et al. argued that JFK protocol with DH reuse achieves AFS security and identity protection under the combined H/DDH assumption (where H is a pseudo-random function) along with secure signature, encryption and MAC. Unfortunately, their assumptions are true only when they prove the security in a weakened model which considers only the presence of passive adversaries. In this case, JFK achieved AFS security only against passive adversaries. To prove JFK security against active adversaries, their assumptions are not enough as discussed above. We will see in section 3.4 that JFK can be shown secure using GDH assumption.

2.5 Resisting Other Type of DoS Attacks in JFK

Note that in JFK protocol, upon receiving third message, the responder first computes the shared secret g^{xy}, next verifies the MAC, and finally verifies the signature on the decrypted message. Therefore, as seen in the described DoS attack on JFK, if the responder is reusing its ephemeral DH key g^y, then attackers can compute valid shared secrets of the form g^{mxy} without performing exponentiations. With these efficiently generated keys they can trigger the responder perform two expensive operations, namely genDH2 and versig2 by sending fake signature with valid MACs. This attack could be thwarted if a responder can set a fresh ephemeral DH key in an efficient manner for each session which led the attacker to perform an exponentiation to compute the shared secret g^{xy} for each session in order to succeed in the attack.

Otherwise to avoid the exponentiation cost, the attacker can take another approach: do not compute the shared secrets and send bogus MACs in the third message. Here the attacker's goal is not to cause the responder perform up to versig2, but up to the expensive genDH2 operation. Unfortunately, JFK will be vulnerable to this DoS attack since the protocol requires the responder to compute the shared secret g^{xy} first, even to verify the MAC on the received message. Therefore in the JFK protocol the attacker, without computing the shared secret, can cause the responder to proceed up to the genDH2 operation. However it is easy to avoid such an attack by incorporating client puzzles in the protocol provided that the cost for computing the puzzle solution should be equal to or higher than the cost for performing genDH2 and versig2 operations. By verifying the puzzle solutions in an efficient manner the responder ensures that the cost for the initiator/attacker will be much higher than that of the responder at any stage of the protocol.

3 BPV-JFK

Although adding certain type of client puzzles with JFK (resulting in a protocol we call DoS-JFK) can thwart the above mentioned DoS attacks, we propose a new protocol, which we call BPV-JFK, for the following reasons. First, JFK with ephemeral key reuse can be proven under the GDH but not under the standard DDH assumption as claimed by the JFK designers; in this case, the

security analysis will have to be in the random oracle model. Second, JFK with ephemeral key reuse does not achieve perfect forward secrecy, but gives AFS as per Definition 1. With BPV-JFK, we not only achieve the full security promised by the original JFK protocol but achieve perfect forward secrecy. Interestingly, by computing a fresh ephemeral DH value for each session in an efficient manner and by adding client puzzles to BPV-JFK, we avoid the above mentioned attacks.

3.1 BPV Generator

We use a technique due to Boyko et al. [5] that enables the responder to compute g^y with fewer modular multiplications (compared to a full exponentiation).

Definition 2 (BPV Generator). *Let p be a DSA modulus such that the prime q divides $p-1$. Select a random element g of order q in the multiplicative group $\mathbb{Z}_p^*$ For integer parameters $N \geq \ell \geq 1$, the BPV generator generates pairs of the form (i, g^i) as follows:*

BPV-Pre: Pre-processing *Generate N random integers $x_1, x_2, \ldots x_N \in \mathbb{Z}_q$. Compute $X_i = g^{x_i} \mod n$ for each i and store the pair (x_i, X_i) in a table.*

BPV-Gen: Pair generation *Whenever a pair (y, g^y) is needed, generate a random set $S \subseteq_R \{1, \ldots, N\}$ such that $|S| = \ell$. Compute $y = \sum_{j \in S} x_j \mod q$. If $y = 0$, stop and generate S again. Otherwise compute $g^y = \prod_{j \in S} g^{x_j} \mod n$ and return (y, g^y).*

In [16], Nguyen et al. presented the extended BPV generator (EBPV) which is exactly the BPV generator for certain parameter choices. Using the following theorem, they established the security of some discrete logarithm based signature schemes that use EBPV under adaptive chosen message attack . The theorem shows that for a fixed q, with overwhelming probability on the choice of x_i's, the distribution of the output y of the BPV generator is statistically close to the uniform distribution on $\mathbb{Z}_q$. Nguyen and Stern presented a similar result in [17]. In particular, a polynomial time adversary cannot distinguish the two distributions for appropriate choices of N and ℓ. Possible N and ℓ values are listed in Section 3.3.

Theorem 1. *Let q be a prime, and let $N \geq \ell \geq 1$. Then,*

$$\frac{1}{q^N} \sum_{\boldsymbol{x} \in \mathbb{Z}_q^N} \sum_{y \in \mathbb{Z}_q} \left| \Pr_{S \subseteq [1,N]:|S|=\ell} \left(\sum_{j \in S} x_j \equiv y \mod q \right) - \frac{1}{q} \right| \leq \sqrt{q / \binom{N}{\ell}} \qquad (3)$$

Moreover, there exists a $c > 0$, such that the following holds with probability at least $1 - 2^{-cN}$. If $x_1, \ldots, x_N$ are chosen independently and uniformly from $[0, q-1]$ and if $y = \sum_{j \in S} x_j \mod q$ is computed from a random set $S \subseteq [1, N]$ of ℓ elements, then the statistical distance between the distribution of y and the uniform distribution is bounded by $\sqrt{q / \binom{N}{\ell}}$.

Intuition on Theorem 1. Note that this result holds regardless of whether the pre-computed x_i's are known to a distinguisher or not. In other words, even if a distinguisher knew from the x_i's which elements of $\mathbb{Z}_q$ were more (or less) likely to be generated by the BPV generator, these elements are still generated only with negligibly more (or less) frequency compared to uniform. For example, the theorem implies that for appropriate choices of the N and ℓ values, the BPV generator outputs almost all the elements of $\mathbb{Z}_q$ and the proportion of elements not output by the BPV generator is very small.

To see this, fix random $x_1, \ldots, x_N$. Let z be the proportion of elements of $\mathbb{Z}_q$ which are output by the BPV generator for these x_i's. That is, for qz of the elements of $\mathbb{Z}_q$, there exists a subset of size ℓ of $\{x_1, \ldots, x_N\}$ that sum to that element, whereas for $q(1-z)$ of the elements no such subset exists. Suppose (for simplicity) that each one of the qz elements of $\mathbb{Z}_q$ that is output is output with equal frequency (namely, $1/qz$). Then consider inequality (3) and substitute:

$$\begin{aligned}
\sqrt{q/\binom{N}{\ell}} &\geq \frac{1}{q^N} \sum_{\boldsymbol{x} \in \mathbb{Z}_q^N} \sum_{y \in \mathbb{Z}_q} \left| \Pr\left(\sum_{j \in S : |S| = \ell} x_j \equiv y \mod q \right) - \frac{1}{q} \right| \\
&= \frac{1}{q^N} \sum_{\boldsymbol{x} \in \mathbb{Z}_q^N} \left(qz \left| \frac{1}{qz} - \frac{1}{q} \right| + q(1-z) \left| 0 - \frac{1}{q} \right| \right) = 2(1-z) \\
\Rightarrow z &\geq 1 - \frac{1}{2} \sqrt{q/\binom{N}{\ell}}
\end{aligned}$$

For appropriate choices of N and ℓ, $\sqrt{q/\binom{N}{\ell}}$ can be made negligibly small, so z can be made very close to 1. In other words, the proportion of elements of $\mathbb{Z}_q$ that are output by the BPV generator is very close to 1. Hence the theorem implies that the BPV generator outputs almost all the elements of $\mathbb{Z}_q$ with almost same probability.

3.2 The BPV-JFK Protocol and its Security Analysis

We now replace the reuse technique in the JFK protocol with the BPV generator to generate new ephemeral DH value g^y for each session efficiently. We denote the resulting protocol, depicted in Figure 3, as BPV-JFK. The rest of the security features, such as inclusion of identity protection, addition of encryption to protocol and anti-replay cache, follow directly from the JFK design [1].

Aiello et al. [1] analysed JFKi protocol in two phases. In the first phase, Aiello et al. analysed the basic cryptographic core of the JFKi protocol and in the second phase, they analysed the additional security features implemented in the protocol. Since our BPV-JFK protocol is similar to the JFKi protocol, we follow the approach of Aiello et al. and separate the analysis of the basic cryptographic core of the BPV-JFK protocol from the analysis of the complete BPV-JFK protocol which has additional security features. As the security features are already well analysed by Aiello et al., we do not repeat their security analysis

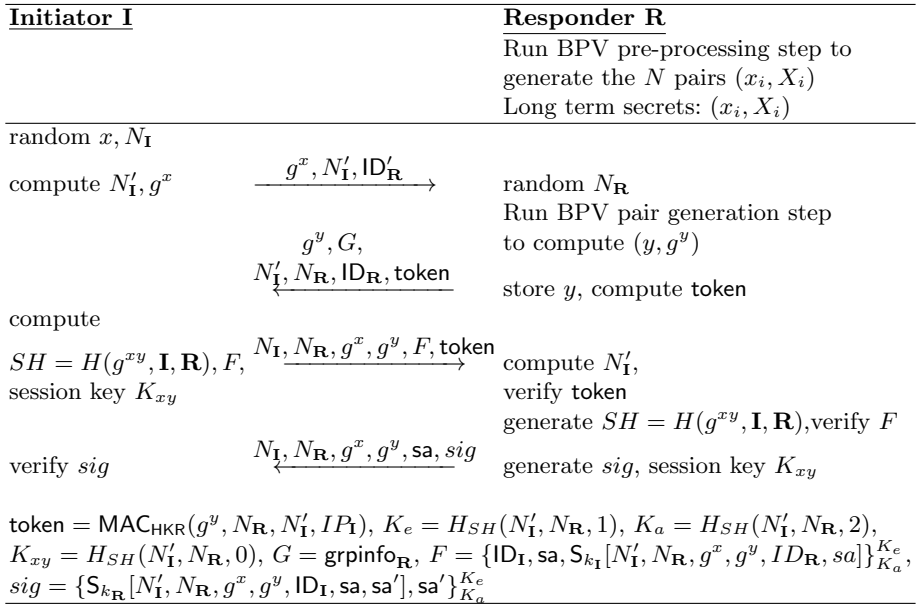

Fig. 3. BPV-JFK protocol

and analyse only the basic BPV-JFK protocol. In this section, we first analyse the security of the basic BPV-JFK protocol in Canetti-Krawzyck model [6] and show that the basic BPV-JFK is SK-secure in the unauthenticated link model (UM). Refer to [6] for a detailed analysis of the CK01 model.

To resist other type of DoS attacks described in Section 2.5, we follow the approach of Stebila et al. [23] to turn BPV-JFK into a DoS-resistant protocol. We call the resultant protocol DoS-BPV-JFK and show in Section 3.6 that DoS-BPV-JFK in Figure 5 satisfies the definition of Stebila et al..

Basic BPV-JFK protocol and its Security Analysis. The basic BPV-JFK protocol is depicted in Figure 4. The responder computes y and g^y using the ℓ random pairs (x_i, X_i) as in the BPV pair generation step. We prove in the following theorems that the basic BPV-JFK is SK-secure in the UM with PFS if the DDH assumption holds.

Definition 3 (Decisional Diffie-Hellman (DDH) assumption). *Let k be a security parameter. Let g be an element of order q in $\mathbb{Z}_p^*$ for primes p and q such that $q|p-1$ Then the probability distributions of $\mathcal{D}_0 = \{\langle p, g, g^x, g^y, g^{xy}\rangle : x, y \leftarrow_R \mathbb{Z}_q\}$ and $\mathcal{D}_1 = \{\langle p, g, g^x, g^y, g^z\rangle : x, y, z \leftarrow_R \mathbb{Z}_q\}$ are computationally indistinguishable.*

Remark 1. Note that the responder requires only the value y to compute the shared secret g^{xy} in future. Hence it stores only y for each session but not the ℓ

1. $\mathbf{I} \rightarrow \mathbf{R}$: $\mathsf{N}_{\mathbf{I}}$, g^x, $\mathsf{ID}_{\mathbf{I}}$
2. $\mathbf{R} \rightarrow \mathbf{I}$: $\mathsf{N}_{\mathbf{I}}$, $\mathsf{N}_{\mathbf{R}}$, g^y, $\mathsf{S}_{k_{\mathbf{R}}}[\mathsf{N}_{\mathbf{I}}, \mathsf{N}_{\mathbf{R}}, g^x, g^y, \mathsf{ID}_{\mathbf{R}}]$
3. $\mathbf{I} \rightarrow \mathbf{R}$: $\mathsf{N}_{\mathbf{I}}$, $\mathsf{N}_{\mathbf{R}}$, $\mathsf{S}_{k_{\mathbf{I}}}[\mathsf{N}_{\mathbf{I}}, \mathsf{N}_{\mathbf{R}}, g^x, g^y, \mathsf{ID}_{\mathbf{I}}]$

shared session secret : $\sigma = g^{xy}$

Fig. 4. Basic BPV-JFK protocol

pairs (x_i, X_i) as they are already stored in the long-term table. Hence the session state reveal or session corruption query outputs the value y and session key (if it is computed) that are stored in the responder memory.

Theorem 2 (SK-security of BPV-JFK in AM). *If G is a group where the* DDH *assumption holds, then the basic BPV-JFK protocol (without signatures) is SK-secure in the authenticated links model (AM).*

Proof. During the protocol execution, if both the uncorrupted parties P_i and P_j complete the protocol, then they compute the same shared secret g^{xy}. Hence the first requirement of SK-security definition, namely correctness (refer to [6]) is satisfied.

Let $\mathcal{A}$ be an adversary against basic BPV-JFK protocol. Using $\mathcal{A}$, we can construct an adversary $\mathcal{B}$ that can distinguish between $\mathcal{D}_0$ and $\mathcal{D}_1$. The distinguishuer $\mathcal{B}$ is given a DDH challenge tuple (p, g, g^u, g^v, g^w) chosen from $\mathcal{D}_0$ or $\mathcal{D}_1$ with probability 1/2 as input.

Assume that $\mathcal{B}$ creates an AKE experiment that involves n honest parties $P_i, i = 1, \ldots, n$ and the adversary $\mathcal{A}$. Each party can be activated to participate in at most s AKE sessions. Among all n parties, $\mathcal{B}$ randomly selects one party, call it P_j. Assume that P_j plays the role of the responder in all its sessions. For all parties activated as responders, $\mathcal{B}$ runs the BPV pre-processing step to compute the corresponding long-term secret pairs of $(x_i, X_i), i = 1 \ldots N$ as described in the protocol. Also $\mathcal{B}$ selects a random session denoted by s^* among the sessions where P_j plays the role of the responder. Let P_i be the initiator to the session s^* which could be either at P_i or P_j. Except for session s^*, $\mathcal{B}$ executes all session establishments by following the protocol specification.

When $\mathcal{A}$ corrupts a party other than P_i and P_j, $\mathcal{B}$ submits the secrets possessed by the party. If $\mathcal{A}$ issues a corrupt query to either of the parties P_i and P_j then $\mathcal{B}$ declares failure.

Assume that a session $s' \neq s^*$ is activated between two uncorrupted parties P_k (initiator) and P_j. In this case $\mathcal{B}$ can compute the shared session key (since the ephemeral DH public keys of P_k and P_j are set by $\mathcal{B}$) of s'. If s' is corrupted by $\mathcal{A}$, then the adversary $\mathcal{B}$ submits the session key to $\mathcal{A}$. Suppose that a session $s'' \neq s^*$ is activated between the adversary controlled initiator P_m (initiator) and the responder P_j. Now $\mathcal{B}$ can compute the session key g^{xy} even if it does not know x.

For the test session s^*, $\mathcal{B}$ assigns the DDH challenge values g^u and g^v in the ephemeral public keys of parties P_i and P_j respectively. $\mathcal{B}$ declares failure and

stops the experiment if $\mathcal{A}$ tries to corrupt the party P_i or P_j and the session s^*. For the test query issued by $\mathcal{A}$, $\mathcal{B}$ submits g^w to the adversary. $\mathcal{A}$ continues with the experiment even after the test query, but it is not allowed to expose the test session. At the end of the experiment $\mathcal{A}$ stops and returns a bit b' as its guess.

Analysis. Provided that parties the P_j or P_j or the session s^* are not corrupted, the AKE experiment simulated by $\mathcal{B}$ is perfect except with negligible probability. With probability $1/n$, $\mathcal{B}$ selects P_j as the responder of the session s^*. The event that $\mathcal{A}$ selects s^* as test session happens with probability at least $1/s$. For the test session, the ephemeral public keys of the parties are not generated using BPV pair generation step. By Theorem 1, the statistical distance between the computed y and ephemeral secret of the assigned challenge v is bounded by $\sqrt{q/\binom{N}{\ell}}$.

The adversary $\mathcal{A}$ was provided with g^w as answer to the test query. In this case, if the challenge tuple belongs to the distribution $\mathcal{D}_0$, then the actual session key was provided to $\mathcal{A}$. Otherwise, if the tuple belongs to the distribution $\mathcal{D}_1$, then the response was a random key. Whenever $\mathcal{A}$ wins the experiment with probability $1/2 + \epsilon$ for non-negligible ϵ, the adversary $\mathcal{B}$ also distinguishes between the two distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ with probability $1/2 + \epsilon/ns + \sqrt{q/\binom{N}{\ell}}$ since the adversary chooses s^* as test session and P_j as the responder with probability $1/ns$. □

Theorem 3 (SK-security of BPV-JFK in UM). *The basic BPV-JFK protocol is SK-secure in the unauthenticated links model (UM).*

Proof. We showed in Theorem 2 that the basic BPV-JFK protocol without signatures is SK-secure in the AM. Then the proof follows by Theorem 6 in [6] which states that any protocol which is SK-secure in the AM can be transformed in to a SK-secure protocol in the UM by including authenticators such as digital signatures. □

Perfect Forward Secrecy (PFS). Except for the generation of g^y the basic BPV-JFK protocol in Figure 3 is the same as the ISO 9798-3 protocol (the basic JFK protocol without g^y reuse). In the basic BPV-JFK protocol, the BPV generator is used to compute g^y whereas in the ISO 9798-3 protocol a random y is used to compute g^y.

The ISO 9798-3 protocol is SK-secure [6] in the UM model with PFS if the signature scheme is secure, the pseudo-random function H is secure and the DDH assumption holds. We now provide a proof sketch using game hopping technique that the basic BPV-JFK is SK-secure in the UM model with PFS.

Let E_i be the event that the adversary wins in game G_i. Assume that the long-term secret N pairs (x_i, X_i) of the responder are revealed. Assume that G_0 is the game in which the BPV generator is used to compute g^y in the BPV-JFK

protocol. Now we define game G_1 which is same as game G_0 except that a random y is used to compute g^y. An adversary that can distinguish G_0 from G_1 is effectively a distinguisher for the BPV generator in Theorem 1. Note that, whereas the proof of security in Theorem 2 did not have the x_i revealed to the adversary, they are revealed to the adversary in the PFS argument. Nonetheless, Theorem 1 still applies, and the games are statistically indistinguishable: $|\Pr(E_0) - \Pr(E_1)| \leq \sqrt{q/\binom{N}{\ell}}$, which is negligible for appropriate choices of N and ℓ.

Now, in G_1, the modified BPV-JFK protocol is same as the ISO 9798-3 protocol with the pre-computed N pairs which are independent of the different g^y values used to compute the session keys. If the adversary wins game G_1, then the challenger breaks the PFS security of the ISO protocol. The ISO protocol was proven to have PFS [6], so $\Pr(E_1)$ is negligible, and hence $\Pr(E_0) \leq \mathrm{negl}(k) + \sqrt{q/\binom{N}{\ell}}$.

Remark 2. In the BPV-JFK protocol, the responder has to store the value y for each session. This small amount of storage for each session is not a resource constraint since a very small fraction (less than one one-millionth, say) of the responder's memory will be used to store such values when it executes thousands of key exchange sessions per second. Since the BPV-JFK protocol focus on minimizing the computational based expenditure to the responder, the amount of storage needed for each session is considered to be minimal.

3.3 Efficiency Comparison and Parameter Sizes

Table 2 compares the general efficiency of BPV-JFK with the JFK protocol, both when JFK reuses the ephemeral public keys and when it does not.

Protocol	Efficiency technique	Server operations in message 1	Perfect Forward Secrecy
JFK with no reuse	None	1 mod. exp.	Yes
JFK with reuse	Reuse of g^y	None	No
BPV-JFK	BPV generator	$\ell - 1$ mod. add., $\ell - 1$ mod. mul.	Yes

Table 2. Efficiency comparison of JFK and BPV-JFK

The specific efficiency of the BPV-JFK protocol depends on the number of elements ℓ in the random set S the responder choose to compute (y, g^y) in the BPV pair generation step. The responder may prefer to reduce the number of online modular multiplications required for generating g^y. Hence it might be appropriate for the responder to choose a bigger value of N (polynomial in $\log q$) to make ℓ smaller. Values for N and ℓ for a 160-bit q and their corresponding BPV-Pre and BPV-Gen running time are presented in Table 3. We ran the experiment

on a single core of a 3.06 GHz Intel Core i3 with 4GB RAM, compiled using `gcc -O2` with architecture x86_64. The big integer arithmetic from OpenSSL 0.9.8r is used to implement the software.

			Runtime	
N	ℓ	$\sqrt{q/\binom{N}{\ell}}$	BPV-Pre (s)	BPV-Gen (ms)
$2^{11} = 2048$	48	2^{-82}	0.939	0.226
$2^{12} = 4096$	40	2^{-80}	1.892	0.196
$2^{13} = 8192$	35	2^{-81}	3.758	0.168
$2^{14} = 16384$	31	2^{-81}	7.527	0.156
$2^{16} = 65536$	26	2^{-83}	30.148	0.134

Table 3. Parameter sizes, security bound for Theorem 1, and runtime for BPV generator with 1024-bit modulus and 160-bit exponent. For comparison, a single 160-bit modular exponentiation takes 0.461 ms.

Our experimental results verify that computing g^y using the BPV-Pre and BPV-Gen algorithms is faster than performing a 160-bit modular exponentiation to compute g^y from a random y (which requires 0.461 ms). The advantage factor of BPV generation over modular exponentiation based on the parameter values listed in Table 3 is between 2 and 3.4. For example, if we choose $N = 2^{14}$ and $\ell = 31$, then g^y can be computed 3 times faster using the BPV generator.

3.4 On the Security of JFK

If the security of the basic JFK (ISO 9798-3) protocol is analysed with the assumption that the responder reuses its ephemeral DH key, then the security might not be reduced to DDH assumption for the following reason: consider the scenario as in session s'' of Theorem 2, where $\mathcal{A}$ controls the initiator P_m and activates the session s'' with responder P_j. Since the responder reuses the ephemeral DH key g^y for several sessions, the adversary $\mathcal{B}$ assigns one of the DDH challenge g^v in the place of g^y. If a session key reveal query is issued by $\mathcal{A}$, then $\mathcal{B}$ cannot compute the session key as it does not know both x and y.

The adversary $\mathcal{B}$ simulates the environment perfectly only if it can respond to this query consistently. Here the DDH assumption itself is not enough and therefore the security proof for the basic JFK with reuse technique may require the GDH assumption and the session key is set as $H(g^{xy}, \mathbf{I}, \mathbf{R})$. If H is a random oracle and the adversary has access to DDH oracle, then the session key reveal queries issued for session s'' could be answered correctly. Hence if H is a random oracle and $\mathbb{G}$ is a group where GDH assumption holds, then the basic JFK protocol (with g^y reuse) is SK-secure in the UM model.

3.5 Cost Calculations for BPV-JFK.

Now, we calculate the cost functions for BPV-JFK protocol and show that the protocol is resistant to the DoS-attack described in Section 2.2. From Equa-

tions (1) and (2) it is clear that the attacker cost function for mounting a coordinated DoS-attack on JFK and the responder cost function to process the fake message does not satisfy the tolerance relation $\mathcal{T}$. In this section, we calculate the cost functions for both the attacker and the responder and show that our BPV-JFK protocol satisfies the tolerance relation and hence resistant to the attack.

Here the n coordinated initiators/attackers can compute the ephemeral DH values as $g^x, g^{2x}, g^{3x}, \ldots g^{nx}$ and can engage in protocol runs with the responder that uses the BPV generator to generate a new g^y for each session. Since g^y is unique for each session, the n shared secrets are independent of each other. That is for n different $g^{y_i}, i = 1, \ldots, n$, the attacker has to perform n exponentiations to compute the n shared secrets $g^{xy_1}, \ldots, g^{nxy_n}$. As a result, the cost for computing genexp in message 1 could be amortised across all attackers but not for the computation of genDH in message 3.

For large values of n, $\phi(\mathsf{shareexp})$, the attacker's cost for sharing genexp, is *cheap*. Hence the attack cost function Θ for the attacker in constructing n valid-looking third messages is dominated by events of expensive cost since the attacker must compute new shared secret for each session. Hence the attacker's total cost is $\Theta(\mathsf{decrypt1}) = \textit{expensive}$. However, to process a single message $\delta'(\mathsf{versig2})$, the operations performed by the responder are dominated by events of two expensive costs. Hence, $\delta'(\mathsf{versig2}) = \textit{expensive}$. Therefore, we see that $(\delta'(\mathsf{versig2}), \Theta(\mathsf{decrypt1})) = (\textit{expensive}, \textit{expensive})$ which belongs to $\mathcal{T}$ and hence the protocol is resistant to the DoS attack.

3.6 Analysing the BPV-JFK Protocol in the Stebila et al. model

Stebila et al. [23] gave a generic technique to transform any protocol into a DoS resistant protocol. Their technique uses strongly difficult interactive client puzzles as a DoS countermeasure and message authentication codes (MAC) for integrity of stateless connections. The server in the protocol must not perform any expensive operation until it verifies the MAC and the puzzle solution. Refer to [23] for a detailed description of the model.

In this section we follow the approach of Stebila et al. to turn BPV-JFK into a DoS-resistant protocol. We combine a strongly difficult interactive client puzzle with the BPV-JFK protocol and show that the resultant protocol, denoted DoS-BPV-JFK in Figure 5, satisfies the definition of Stebila et al.

DoS-BPV-JFK Protocol Specification. Let us assume Puz —consisting of Setup, GenPuz, $\mathsf{FindSoln}$, and $\mathsf{VerSoln}$ algorithms — is a strongly difficult client puzzle [23]. Let DoS-BPV-JFK be the protocol consisting of the following algorithms:

- $\mathsf{GlobalSetup}(1^l)$: Set $\rho\mathsf{Space} \leftarrow \{0,1\}^\ell$ and $\mathsf{NonceSpace} \leftarrow \{0,1\}^\ell$.
- $\mathsf{ServerSetup}(\hat{S} \in \mathsf{Servers})$: Set $\mathsf{HKR} \leftarrow_r \{0,1\}^\ell$ and $\rho_{\hat{S}} \leftarrow \mathsf{HKR}$
- $\mathsf{CActionj}_{\mathsf{DoS-BPV-JFK}}(\cdots), \mathsf{SActionj}_{\mathsf{DoS-BPV-JFK}}(\cdots)$: As specified by the protocol in Figure 5.

Client $\hat{C}$		**Server** $\hat{S}$
		long-term secrets: $\rho_{\hat{S}} = \mathsf{HKR}$,
		$((x_i, X_i))_{i=1}^{N} \leftarrow \mathsf{BPVPre}(g, n, N, q)$
$\underline{\mathsf{CAction1}_{\mathsf{DoS-BPV-JFK}}}$:		
1. $N_{\mathbf{I}} \leftarrow_r \mathsf{NonceSpace}$		
2. compute $N'_{\mathbf{I}}$		
3. $\underline{\mathsf{CAction1}_{\mathsf{BPV-JFK}}}$		
4. compute g^x	$\xrightarrow{g^x, N'_{\mathbf{I}}, \mathsf{ID}'_{\mathbf{R}}}$	$\underline{\mathsf{SAction1}_{\mathsf{DoS-BPV-JFK}}}$:
5.		$N_{\mathbf{R}} \leftarrow_r \mathsf{NonceSpace}$
6.		$str \leftarrow (\hat{C}, \hat{S}, N'_{\mathbf{I}}, N_{\mathbf{R}})$
7.		$puz \leftarrow GenPuz(Q, str)$
8.		$\underline{\mathsf{SAction1}_{\mathsf{BPV-JFK}}}$
9.		$(y, g^y) \leftarrow \mathsf{BPVGen}(g, n, \ell, q, (x_i, X_i))_{i=1}^{N})$
10.	$g^y, puz,$	$\lambda = \mathsf{MAC}_{\mathsf{HKR}}(g^y, N_{\mathbf{R}}, N'_{\mathbf{I}}, IP_{\mathbf{I}}, str, puz)$
11. $\underline{\mathsf{CAction2}_{\mathsf{DoS-BPV-JFK}}}$:	$\xleftarrow{N'_{\mathbf{I}}, N_{\mathbf{R}}, \mathsf{ID}_{\mathbf{R}}, G, \lambda}$	store y
12. soln$\leftarrow$ FindSoln(str, puz)		
13. $\underline{\mathsf{CAction2}_{\mathsf{BPV-JFK}}}$:		
14. compute		
15. $SH = H(g^{xy}, \mathbf{I}, \mathbf{R}), F$	$N_{\mathbf{I}}, N_{\mathbf{R}}, g^x, g^y, F,$	
	$puz, soln, str, \lambda$	
	$\xrightarrow{}$	$\underline{\mathsf{SAction2}_{\mathsf{DoS-BPV-JFK}}}$:
16.		compute $N'_{\mathbf{I}}$, reject if
17.		$\lambda \neq \mathsf{MAC}_{\mathsf{HKR}}(g^y, N_{\mathbf{R}}, N'_{\mathbf{I}}, IP_{\mathbf{I}}, str, puz)$
18.		reject if $\neg$ VerSoln(str, puz, soln)
19.		$\tau \leftarrow (N_{\mathbf{I}}, N_{\mathbf{R}}, puz, soln)$
20.		verify no existing presession $[\hat{C}, \hat{S}, \tau]$
21.		accept and store presession $[\hat{C}, \hat{S}, \tau]$
22.		$\underline{\mathsf{SAction2}_{\mathsf{BPV-JFK}}}$
23.		generate $SH = H(g^{xy}, \mathbf{I}, \mathbf{R})$,verify F
24. verify sig	$N_{\mathbf{I}}, N_{\mathbf{R}}, g^x, g^y, sa, sig$ $\xleftarrow{}$	generate sig

Fig. 5. DoS-BPV-JFK protocol

Theorem 4. *Let Puz be an $\epsilon_{\ell,Q,n}(t)$-strongly difficult interactive puzzle and that MAC is a family of secure message authentication codes. Then DoS-BPV-JFK is an $\epsilon'_{\ell,Q,n}(t)$-denial-of-service-resistant protocol, for $\epsilon'_{\ell,Q,n}(t) = \epsilon_{\ell,Q,n}(t + cq) + negl(\ell)$, where c is a constant and q is the number of Send queries issued, assuming $t \in poly(\ell)$.*

Proof. Since *Puz* is an $\epsilon_{\ell,Q,n}(t)$-strongly difficult interactive puzzle and MAC is a secure message authentication code, the proof for condition 1 of the DoS definition of Stebila et al. in [23] directly follows from Theorem 3 in [23]. It remains to show that $\mathsf{SAction1}_{\mathsf{BPV-JFK}}$ does not involve any expensive operation. It is clear from Figure 5 that $\mathsf{SAction1}_{\mathsf{BPV-JFK}}$ involves only a small number of modular additions and modular multiplications. □

4 Conclusion and Future Work

Denial of service attacks are a growing concern to open networks. They may arise in a number of ways; in this work we focused on resource exhaustion DoS attacks (on network protocols) which cause a server to perform many expensive operations. We identified such an attack on the JFK protocol. Though there are many other security features offered by the JFK protocol, it achieves only adaptive forward secrecy. To achieve perfect forward secrecy and to resist the identified attack, we propose to use a technique introduced by Boyko et al. in place of ephemeral key reuse and show that the new protocol is SK-secure in CK01 model under the DDH assumption. The resultant protocol can easily be shown to be DoS-resistant after incorporating client puzzles and secure MACs. While there are many interesting features offered by the resultant protocol, it would be interesting to see if the proposed techniques can be embedded in any other DH based key exchange protocols.

Acknowledgements. The authors are grateful to Dr. Berkant Ustaoglu for his critical comments and to anonymous referees for helpful comments. This work is supported by Australia-India Strategic Research Fund project TA020002.

References

1. W. Aiello, S. M. Bellovin, M. Blaze, R. Canetti, J. Ioannidis, A. D. Keromytis, and O. Reingold. Just Fast Keying: Key agreement in a hostile Internet. *ACM Transactions on Information and System Security*, 7(2):1–30, May 2004.
2. T. Aura and P. Nikander. Stateless connections. In Y. Han, T. Okamoto, and S. Qing, eds., *ICICS 1997*, volume 1334 of *LNCS*, pages 87–97. Springer, 1997.
3. T. Aura, P. Nikander, and J. Leiwo. DOS-resistant authentication with client puzzles. In B. Christianson, B. Crispo, J. A. Malcolm, and M. Roe, eds., *Security Protocols: 8th International Workshop*, volume 2133 of *LNCS*, pages 170–177. Springer, 2000.
4. M. Bellare and P. Rogaway. Entity authentication and key distribution. In D. R. Stinson, ed., *CRYPTO '93*, volume 773 of *LNCS*, pages 232–249. Springer, 1993.
5. V. Boyko, M. Peinado, and R. Venkatesan. Speeding up discrete log and factoring based schemes via precomputations. In K. Nyberg, ed., *EUROCRYPT '98*, volume 1403 of *LNCS*, pages 221–235. Springer, 1998.
6. R. Canetti and H. Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In B. Pfitzmann, ed., *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 453–474. Springer, 2001.
7. R. Canetti and H. Krawczyk. Security Analysis of IKE's Signature-Based Key-Exchange Protocol. In M. Yung, ed., *CRYPTO 2002*, volume 2442 of *LNCS*, pages 143–161. Springer, 2002.
8. C. Castelluccia, E. Mykletun, and G. Tsudik. Improving secure server performance by re-balancing SSL/TLS handshakes. In F. Lin, D. Lee, B. Lin, S. Shieh, and S. Jajodia, eds., *ASIACCS 2006*, pages 26–34. ACM, 2006.
9. L. Gong and P. Syverson, Fail-Stop Protocols: An Approach to Designing Secure Protocols. In R.K. Iyer, M. Morganti, V. Glogor and W.K. Fuchs eds., *Proc.*

Dependable Computing for Critical Applications, pages 44–55. IEEE Computer Society, 1998.
10. D. Harkins, D. Carrel, et al. The Internet Key Exchange (IKE), 1998. URL: www.ietf.org/rfc/rfc2409.
11. A. Juels and J. Brainard. Client puzzles: A cryptographic countermeasure against connection depletion attacks. In *NDSS 1999*, pages 151–165. Internet Society, 1999.
12. C. Kaufman. Internet Key Exchange (IKEv2) protocol, December 2005. RFC 4306.
13. K. Matsuura and H. Imai. Modification of Internet Key Exchange resistant against Denial-of-Service. In *Pre-Proceeding of Internet Workshop (IWS) 2000*, pages 167–174, Feb 2000.
14. C. Meadows. A formal framework and evaluation method for network denial of service. In *CSFW 1999*, IEEE, 1999.
15. R. Moskowitz, P. Nikander, P. Jokela, and T. R. Henderson. Host Identity Protocol, April 2008. RFC 5201.
16. P. Nguyen, I. Shparlinski, and J. Stern. Distribution of modular sums and the security of the server aided exponentiation. In *Proc. Workshop on Cryptography and Computational Number Theory (CCNT'99)*, pages 257–268. Birkhäuser, 2001.
17. P. Q. Nguyen and J. Stern. The hardness of the hidden subset sum problem and its cryptographic implications. In M. J. Wiener, ed., *CRYPTO '99*, volume 1666 of *LNCS*, pages 31–46. Springer, 1999.
18. T. Okamoto and K. Tanaka and S. Uchiyama. Quantum Public-Key Cryptosystems. In M. Bellare, ed., *CRYPTO '2000*, volume of *LNCS*, pages 147-165. Springer, 2000.
19. J. Rangasamy, D. Stebila, C. Boyd and J. González Nieto. An integrated approach to cryptographic mitigation of denial-of-service attacks. In R. Sandhu and D.S. Wong eds., *ASIACCS 2011*, pages 114–123. ACM 2011.
20. R. L. Rivest, A. Shamir, and D. A. Wagner. Time-lock puzzles and timed-release crypto. Technical Report TR-684, MIT Laboratory for Computer Science, March 1996.
21. J. Smith, J. González Nieto, and C. Boyd. Modelling denial of service attacks on JFK with Meadows's cost-based framework. In R. Buyya, T. Ma, R. Safavi-Naini, C. Steketee, and W. Susilo, ads., *AISW-NetSec 2006*, volume 54 of *CRPIT*, pages 125–134. Australian Computer Society, 2006.
22. D. Stebila and B. Ustaoglu. Towards Denial-of-Service-Resilient Key Agreement Protocols. In C. Boyd and J.G. Nieto eds., *ACISP 2009*, volume 5594 of *LNCS*, pages 389–406. Springer, 2009.
23. D. Stebila, L. Kuppusamy, J. Rangasamy, C. Boyd, and J. González Nieto. Stronger difficulty notions for client puzzles and denial-of-service-resistant protocols. In A. Kiayias, ed., *CT-RSA 2011*, volume 6558 of *LNCS*, pages 284–301. Springer, 2011.

A Meadows' Cost-Based Framework [14]

Meadows [14] developed a cost-based framework for analysing vulnerabilities of DoS attacks in protocols. The model assigns cost to every action performed by the communicating principals (initiator and responder) to compare the amount of resources expended on both initiator and responder sides. In Meadows framework, a protocol is resistant to DoS attacks if the cost for an attacker to successfully interrupt the protocol at any stage exceeds the given threshold.

A.1 Notation

Meadows used the Alice-and-Bob specification style to specify the protocols for the framework to calculate the cost associated with every action that includes generating, verifying and accepting message. The following definitions appears in [14] and [21].

Definition 4. *An* Alice-and-Bob specification *is a sequence of statements of the form* $A \rightarrow B : M$ *where* M *represents the message is sent from* A *to* B.

The method of annotating an Alice-and-Bob specification to include message processing steps needed at both the communicating parties is described below.

Definition 5. *An* annotated Alice-and-Bob specification *is a sequence of statements of the form*
$A \rightarrow B : T_1, ..., T_k||M||O_1, ..., O_n$. *The operations performed by* A *to produce message* M *is represented by the ordered sequence* T_i. *Similarly, the ordered sequence* O_j *represents the operations performed by* B *to process and validate message* M.

Normally the events correspond to a sequence of generating, sending, receiving, validating and accepting a message. The different types of events are: (1) *normal*, (2) *verification*, and (3) *accept*.

A.2 Cost Sets and Cost Functions

The following definitions are used to compute the costs for participating in a protocol. Usually this is done by calculating the costs of individual events and summing them for each of the steps in a protocol.

Definition 6. *A* cost set $\mathcal{C}$ *is a monoid with the monoid operation* $+$ *and partial order* $<$ *such that* $x \leq x + y$ *and* $y \leq x + y$ *for all* x, y *in* $\mathcal{C}$

Definition 7. *Assume that the events are defined in an Alice-and-Bob specification. A function* δ *that maps events to a cost set* $\mathcal{C}$ *and maps events to 0 on accept events is called an* event cost function.

Definition 8. *Let* $\mathcal{P}$ *be an annotated Alice-and-Bob specification protocol. The* message processing cost function $\delta^{'}$ *associated with the event cost function* δ *on verification events following receipt of a message is as follows: If a line* $A \rightarrow B : T_1, ..., T_k||M||V_1, ..., V_n$ *appears in* $\mathcal{P}$ *an annotated Alice-and-Bob specification of a protocol, then for each verification event* $V_j : \delta^{'}(Vj) = \delta(V_1) + ... + \delta(V_j)$.

We now define the costs to a responder for engaging in the protocol.

Definition 9. *Let* $\mathcal{P}$ *be an annotated Alice-and-Bob specification protocol. The protocol engagement cost function* Δ *associated with* δ *defined on accept events is as follows: If the line* $A \rightarrow B : T_1, ..., T_k||M||V_1, ..., V_n$ *appears in* $\mathcal{P}$, *and* V_n *is an accept event, then the protocol engagement cost* $\Delta(V_n)$ *is defined as the sum of the costs of all events occurring at* B *up to the accept event* V_n *plus cost of the event* T_i.

A DoS attacker has the capability to send bogus messages or to engage in bogus protocol runs by spoofing the IP address and hence the attacker is not restricted to the same events as a legitimate protocol participant. Therefore, it is necessary to define an independent set of cost functions to define the cost for actions performed by the attacker.

Definition 10. *Let us denote the attacker cost set by $\mathcal{G}$ and the set of actions performed by the attacker by $\mathcal{I}$. Let ϕ be the function that maps the attacker's actions to their cost set $\mathcal{G}$. Then the cost function denoted by Φ is defined on a sequence of actions performed by the attacker as, $\Phi(i_1, ..., i_n) = \phi(i_1) + ... + \phi(i_n)$ for $i_k \in \mathcal{I}$*

Definition 11. *Let Θ be the attacker cost function that maps events in $\mathcal{P}$ to a cost set $\mathcal{C}$. The protocol $\mathcal{P}$ is said to be* fail stop *if for any event $\mathsf{E} \in \mathcal{P}$, if the attacker interferes with a message arriving before E, then the events arriving after E will not occur unless the cost expended by the attacker is $\Theta(\mathsf{E})$*

Definition 12. *Let us denote the responder and attacker cost sets by $\mathcal{C}$ and $\mathcal{G}$ respectively. A tolerance relation $\mathcal{T}$ is the subset of $\mathcal{C} \times \mathcal{G}$ consisting of all pairs (c, g) such that if the cost expended by the attacker is greater than g, then the protocol designer will tolerate the cost expended by the responder up to c. A tuple $(\mathsf{c}_0, \mathsf{g}_0)$ is said to be within $\mathcal{T}$ if there exists $(\mathsf{c}, \mathsf{g}) \in \mathcal{T}$ with $\mathsf{c}_0 \leq \mathsf{c}$ and $\mathsf{c}_0 \leq \mathsf{c}$.*

A.3 Evaluating DoS Resistance

The procedure for assessing DoS resistance in a protocol using this framework is as follows [14]:

- determine the attacker's capabilities and the attack cost function Θ for each step of the protocol execution,
- decide on the tolerance relation $\mathcal{T}$, and
- verify that $(\delta^{'}(\mathsf{E}_2), \Theta(\mathsf{E}_1)) \in \mathcal{T}$ for a verification event E_2 immediately followed by the event E_1, and that $(\Delta(\mathsf{E}), \Theta(\mathsf{E})) \in \mathcal{T}$ if E is an accept event.

To analyse JFK using Meadows' cost based framework, we follow the definition of cost sets and tolerance relation proposed by Smith et al. [21]: the cost sets for both the responder and the attacker, denoted respectively by $\mathcal{C}$ and $\mathcal{G}$, consist of elements *cheap*, *medium* and *expensive* such that $0 < cheap < medium < expensive$. Adding two elements in the cost set results in assigning the maximum of two: $x + y = max(x, y)$. The tolerance relation $\mathcal{T}$ is defined as ([14], [21]):

$$\mathcal{T} = \begin{pmatrix} (cheap, cheap); & (cheap, medium); \\ (medium, cheap); & (cheap, expensive) \\ (medium, medium); & (medium, expensive); \\ (expensive, expensive); & \end{pmatrix}$$

A protocol may become vulnerable to a cheaply mounted DoS attack if the pair $(expensive, cheap)$ and $(expensive, medium)$ ever occurs in the analysis.