SCIS2000 The 2000 Symposium on Cryptography and Information Security
Okinawa, Japan, January 26-28, 2000

Evaluation of DoS Resistance in Relation to Performance Dynamics

Kanta Matsuura *

Abstract— In key-agreement protocols, we have a trade-off between security and availability; if authenticated for reasons of security, the protocol would be more vulnerable to a Denial-of-Service (DoS) attack composed of quite a large number of bogus requests which cost an honest responder a lot. DoS-protection strategies such as *network ingress filtering* and *falling-together (FT) mechanism* can make the protocol better but we should estimate the availability or server-blocking probability to see how well the trade-off is solved. This paper gives an estimation in a situation where the responder's computational performance depends on the number of on-going processes. The results show that optimized implementation of the FT mechanism can improve the DoS resistance more significantly than other heuristics such as memory increase, connection timeout, and packet discarding.

Keywords: Denial-of-Service attack; key agreement; ingress filter; falling-together mechanism; server-blocking probability; connection timeout.

1 Introduction

In order to enjoy secure communication over an open network, how to establish secret session keys is a fundamental problem. And unfortunately, key-agreement protocols have a trade-off between security and availability; if authenticated for reasons of security (*e.g.* against man-in-the-middle attack), the protocol would be more vulnerable to a Denial-of-Service (DoS) attack composed of quite a large number of bogus requests which cost an honest responder a lot. For instance, Internet Key Exchange (IKE) [1] is an authenticated version of Diffie-Hellman (DH) key-agreement protocol [2]. Public-key based authentication mechanisms in IKE can be abused by DoS attackers since an honest responder must perform heavy public-key operations before he/she becomes sure about the identity of the initiator [3].

A protection strategy against such a CPU exhaustion was proposed in [4] and called *falling-together (FT)* mechanism. FT mechanism can improve IKE and the improvement was evaluated in terms of server-blocking probability in a simplified model where the performance or precomputation speed is kept constant [5], [6]. This paper provides a more advanced evaluation which considers (1) the dependence of the precomputation performance on the number of on-going processes and (2) heuristic optimization in implementation. Specifically, Section 2 reviews FT mechanism and a basic evaluation model. The paper will then go on to an advanced version of the model in Section 3. After showing the evaluation results in Section 4, Section 5 gives concluding remarks.

* Institute of Industrial Science, the University of Tokyo, Roppongi 7-22-1, Minato-ku, Tokyo 106-8558, JAPAN.

2 Basic Model

2.1 Falling-Together Mechanism

Let us assume that the initiator I and the responder R can use the same public-key infrastructure where public and secret keys of an entity X are denoted by PK_X and SK_X, respectively. Then three-pass key-agreement protocols can be typically described as in Fig. 1, where we can find nine steps involved. FT mechanism can be described as the following strategies:

1. Do not use heavy computation such as modular exponentiation in Step ((4)).
2. Carry a random fresh material by Message ((5)) implicitly in a way that the reconstruction of the material requires heavy computation in Step ((6)) by the initiator with PK_R and/or SK_I. This computational load must be comparable to or higher than that of Step ((8)).
3. Use the random material for authentication.
4. Equip Message ((7)) with an acknowledgment material derived from the reconstructed random material.
5. Verify the acknowledgment material by inexpensive computation at the beginning of Step ((8)).

Once conformed, this mechanism discourages DoS attackers by "falling-together" nightmare; if the attacker and the responder or target have the same level of computational power, the attacker must exhaust his/her own resource in order to exhaust that of the target since a bogus acknowledgment material is detected by the light verification procedure before expensive computation in Step ((8)).

Step	**(Initiator)**		**(Responder)**
((1))	Precomputation (as a regular or periodical background job, *etc.*).		Precomputation (as a regular or periodical background job, *etc.*).
((2))	Computation by using PK_R and/or SK_I.		
((3))	Request Message	$\Longrightarrow$	
((4))			Computation by using PK_I and/or SK_R. (Session-key computation may be included.)
((5))		$\Longleftarrow$	Reply Message
((6))	Computation by using PK_R and/or SK_I. (Session-key computation may be included.)		
((7))	Acknowledgment Message	$\Longrightarrow$	
((8))			Computation by using PK_I and/or SK_R.
((9))	Key establishment.		Key establishment.

Figure 1: Typical structure of three-pass authenticated key-agreement protocols. PK_I and PK_R are the public keys of the initiator and the responder, respectively. SK_I and SK_R are the corresponding secret keys of the initiator and the responder, respectively.

2.2 Simple Evaluation Model

When the responder has sufficient memory such that precomputed values are never exhausted, FT mechanism alone well defeats DoS attacks. When insufficient, specific DoS-resistance is not trivial but can be evaluated in terms of server-blocking probability; the server or responder could not accept a new request when all the precomputed materials are used.

In [5] and [6], we used a statistically equilibrated model illustrated in Fig. 2 where *network ingress filter* [7] limits the amount of bogus packets per attack and attackers send the same number (M) of bogus requests per attack.

As typical parameter assignment, we can use

- $C = 1000$ as responder's *memory capacity* for precomputed materials,
- $R = 0.001$ as *rate of valid request*, and
- $M = 256$ as the number of bogus requests per attack. This corresponds to a local router which controls IP addresses of class C.

Precomputation power U was kept constant as a time unit. We consider the security level equivalent to 1024-bit RSA composite. So if the performance level is in the order of several hundreds Kbps for public-key primitives, $C = 1000$ corresponds to a memory of a few hundreds KB. In this case, $R = 0.001$ causes a few valid requests per minute.

We can interpret the state-transition diagram into a set of equations as follows:

$$\frac{dP_C}{dt} = UP_{C-1} - (D+R)P_C \tag{1}$$

$$\frac{dP_i}{dt} = UP_{i-1} - (D+R+U)P_i + RP_{i+1} \tag{2}$$

$$(C - M < i < C)$$

$$\frac{dP_i}{dt} = UP_{i-1} - (D+R+U)P_i + RP_{i+1} + DP_{i+M} \tag{3}$$

$$(0 < i \leq C - M)$$

$$\frac{dP_0}{dt} = -UP_0 + RP_1 + D\sum_{j=1}^{M} P_j \tag{4}$$

$$\sum_{j=0}^{C} P_j = 1 \tag{5}$$

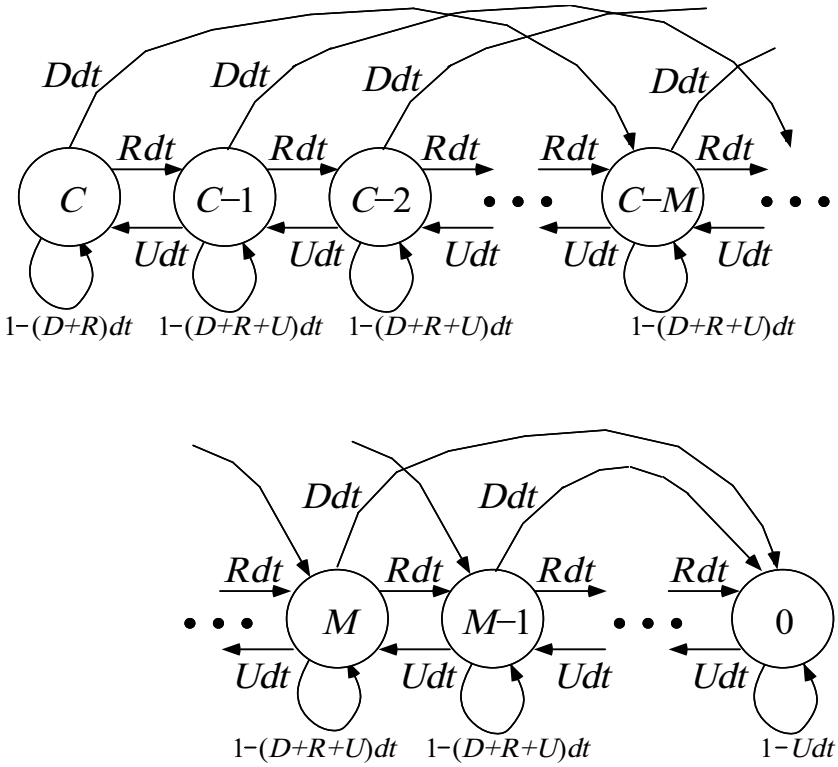


Figure 2: State-transition diagram of a responder (server) in a basic model where the precomputation power is kept constant. Each state is represented by the number of available sets of precomputed materials.

where P_i is the probability that the server is in state i and D represents the *rate of bogus request*. When statistically equilibrated, we can set each derivative $\frac{dP_i}{dt}$ to be zero. Thus $P_0, P_1, \cdots, P_C$ can be computed by solving the resultant following linear equations.

$$UP_{C-1} - (D+R)P_C = 0 \tag{6}$$

$$UP_{i-1} - (D+R+U)P_i + RP_{i+1} = 0 \quad (C-M < i < C) \tag{7}$$

$$UP_{i-1} - (D+R+U)P_i + RP_{i+1} + DP_{i+M} = 0 \quad (0 < i \leq C-M) \tag{8}$$

$$\sum_{j=0}^{C} P_j = 1 \tag{9}$$

where P_i is the probability that the responder is in state i. P_0 is called *server-blocking probability* since precomputed materials are exhausted at state 0.

Figure 3 shows blocking probability P_0 as a function of rate of bogus request D. The blocking probability P_0 is lower than 10% even if the rate of attack D is 0.1. This suggests the responder is usually alive even when attacked by several hundred sets of bogus requests per minute.

3 Advanced Model

The basic model in the previous section assumes that the precomputation power U is kept constant. This is due to CPU protection by FT mechanism. However, actual implementation may be not optimized and the precomputation power may be reduced as the state approaches to the exhaustion. In order to investigate the influence of this reduction, this paper assumes U is linearly reduced from U_{max} to U_{min} in the studied region of D. We will refer to U_{min}/U_{max} as *performance-reduction rate*, in the following.

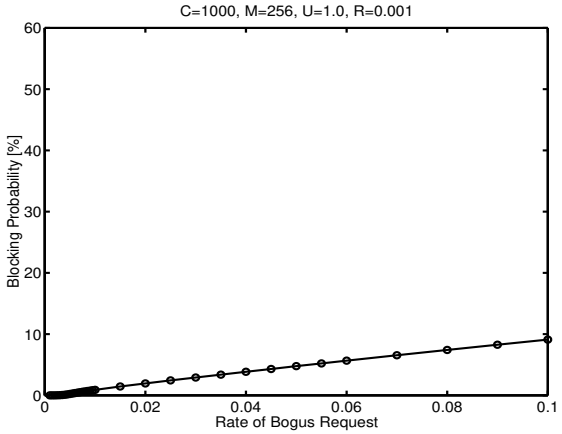
Another issue ignored by the basic model is the effect of heuristic optimization in implementation: for instance, memory increase, connection timeout, and packet discarding. The effect of memory increase can be easily studied by using larger memory capacity. This section interprets the effect of the other two optimizations by replacing M with a smaller value M'.

In summary, our advanced model (Fig. 4) can be written by the following linear equations:

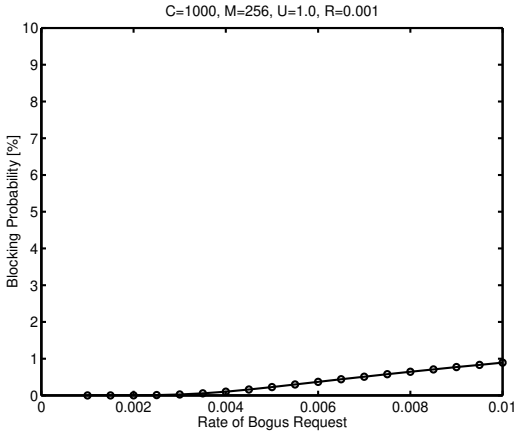
$$U_{C-1}P_{C-1} - (D+R)P_C = 0 \tag{10}$$

$$U_{i-1}P_{i-1} - (D+R+U_i)P_i + RP_{i+1} = 0 \quad (C-M' < i < C) \tag{11}$$

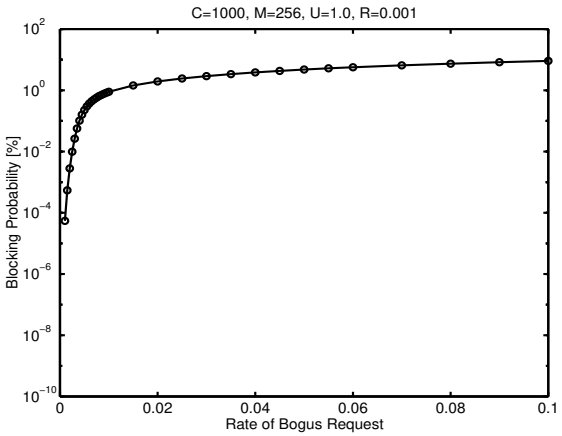
$$U_{i-1}P_{i-1} - (D+R+U_i)P_i + RP_{i+1} + DP_{i+M'} = 0 \quad (0 < i \leq C-M') \tag{12}$$



(a) linear plot



(b) around the threshold



(c) semi-logarithmic plot

Figure 3: Blocking probability as a function of *rate of bogus request* in a basic model with memory capacity $C = 1000$.

$$\sum_{j=0}^{C} P_j = 1 \tag{13}$$

where $M' \leq M$ and the precomputation power at state i ($i = 0, 1, \cdots, C$) is determined by

$$U_i \equiv U_{min} + \frac{U_{max} - U_{min}}{C}\, i. \tag{14}$$

4 Advanced Evaluation

4.1 Results

We computed the server-blocking probability P_0 for different performance-reduction rates such as 0.1, 0.5, and 0.9. Figs. 5–7 show the evaluation results; (a) linear plot when the rate of bogus request is up to a hundred times as large as the rate of normal request, (b) linear plot around a threshold rate of bogus request where the server-blocking probability makes a rapid increase into the order of several percents, and (c) semi-logarithmic plot in the same region as in (a). The parameters are summarized in Table 1.

Table 1: Parameters in the advanced evaluation. M' is the equivalent number of each cluster of bogus requests when implementation is optimized by the use of connection timeout and/or packet discarding. C is memory capacity of the server.

Figure Number	M'	C
5	256	1000
6	160	1000
7	256	10000

4.2 Discussion

In comparison with Fig. 3, Figs. 5–7 tell us that the performance reduction worse than 10% yields significantly bad effect on the server-blocking probability. This suggests that the importance of the implementation of FT mechanism. Our concern is then the effect of the other countermeasures considered in these evaluations.

First, comparing Fig. 6 with Fig. 5, we can see that the optimization in implementation improves the threshold rate of bogus request where the blocking probability starts rapid increase in linear plot. By contrast, comparing Fig. 7 with Fig. 5, we can see that the memory increase improves the blocking probability when the rate of bogus request is smaller than the threshold rate.

Finally, the most important result is that the significant difference in DoS resistance is given by performance-reduction rate U_{max}/U_{min}. Not by the optimization in implementation (M'), nor by the memory capacity (C). If we can keep the performance-reduction rate under 10%, we can keep almost the same DoS resistance. If we could not keep it, the blocking probability gets significantly worse regardless of the heuristic optimization nor memory increase.

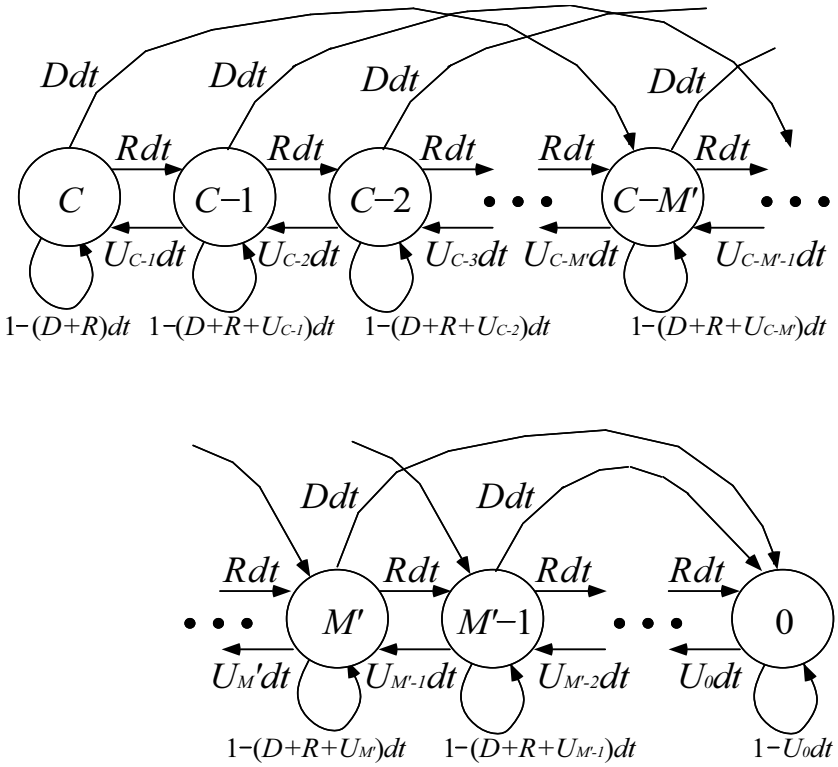


Figure 4: State-transition diagram of a responder (server) in an advanced model where the precomputation power depends on the server state. The effect of connection timeout and packet discarding is interpreted by the equivalent number of requests M'.

5 Concluding Remarks

This paper gives an advanced evaluation of DoS resistance in three-pass key agreement protocols. The evaluation model considers linear dependence of the precomputation power on the number of on-going processes. The results show that optimized implementation of the FT mechanism can improve the DoS resistance more significantly than other heuristics such as memory increase, connection timeout, and packet discarding. Specifically, if the FT mechanism can keep the performance-reduction rate under 10%, we can keep almost the same acceptable DoS resistance: at most 10% blocking probability even when attacked by several ten thousands of bogus requests per minute.

Acknowledgments

This work was performed in part of Research for the Future Program (RFTF) supported by Japan Society for the Promotion of Science (JSPS) under contact no. JSPS-RETF 96P00604.

This work is partly supported by the 16th Research Promotion Fund (No. 28) from Casio Science Promotion Foundation.

References

[1] D. Harkins and D. Carrel. "The Internet Key Exchange (IKE)". rfc2409, November 1998.

[2] W. Diffie and M. Hellman. "New directions in cryptography". *IEEE Trans. Information Theory*, Vol. IT-22, No. 6, pp. 644–654, 1976.

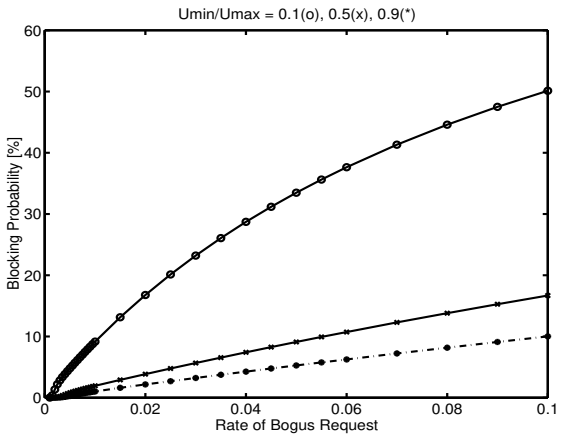
[3] K. Matsuura and H. Imai. "A revised signature mode for the Internet Key Exchange". *Internet Draft*, draft-matsuura-sign-mode-01.txt, Sep. 1999.

[4] K. Matsuura and H. Imai. "Protection of authenticated key-agreement protocol against a Denial-of-Service attack". *Proceedings of 1998 International Symposium on Information Theory and Its Applications (ISITA'98)*, pp. 466–470, October 1998.

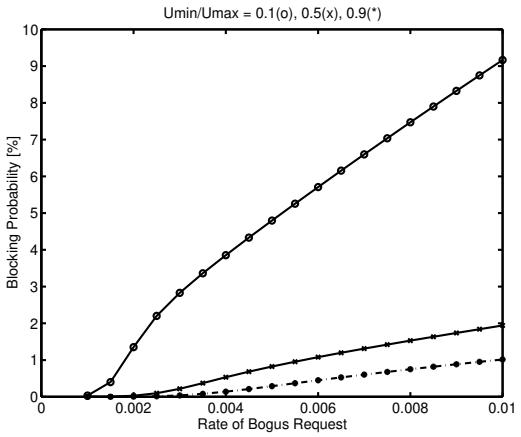
[5] K. Matsuura and H. Imai. "Blocking Probability in Key-Agreement Protocol with Ingress Filtering". Proc. of 1999 Symposium on Cryptography and Information Security (SCIS'99), Vol. II, pp. 893–898, January 1999. (in Japanese)

[6] K. Matsuura and H. Imai. "Resolution of ISAKMP/Oakley Key-Agreement Protocol Resistant against Denial-of-Service Attack". In *Pre-Proc. of Internet Workshop'99 (IWS'99)*, pp. 17–24, February 1999.

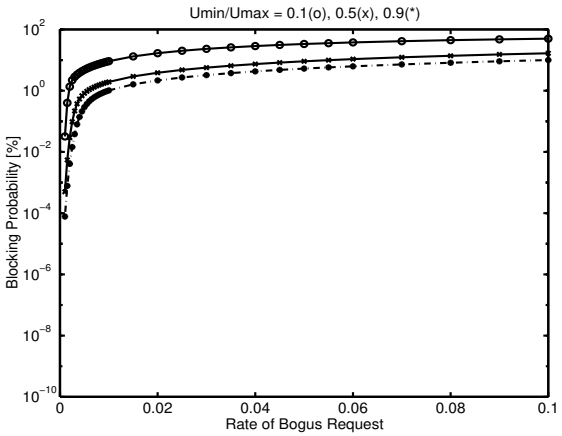
[7] P. Ferguson and D. Senie. "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing". rfc2267, January 1998.



(a) linear plot

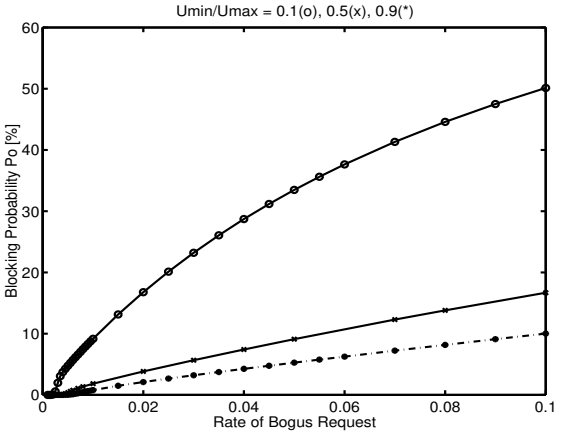


(b) around the threshold

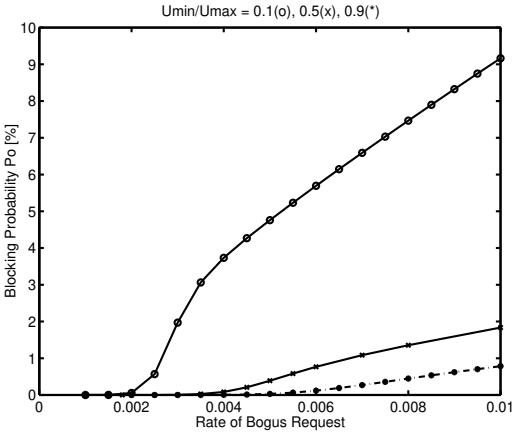


(c) semi-logarithmic plot

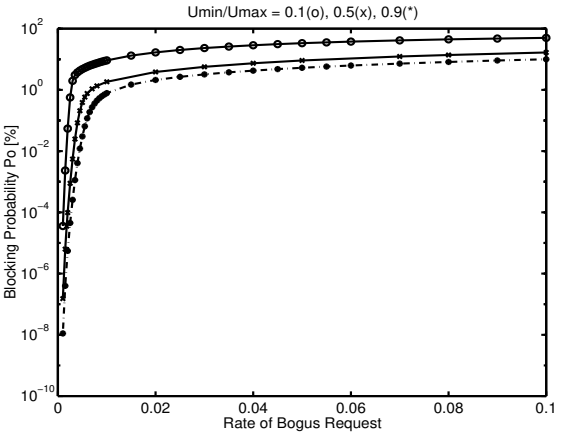
Figure 5: Blocking probability as a function of *rate of bogus request* when $M' = M = 256$ and memory capacity $C = 1000$.



(a) linear plot

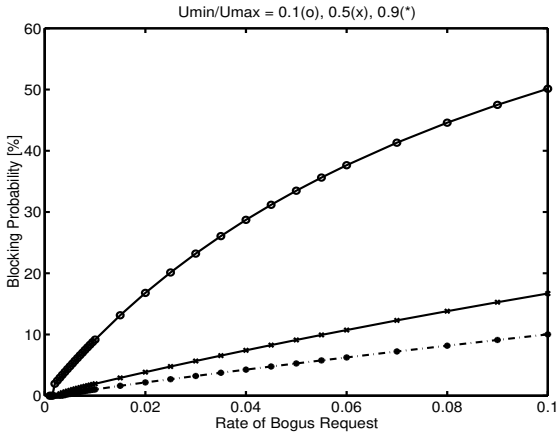


(b) around the threshold

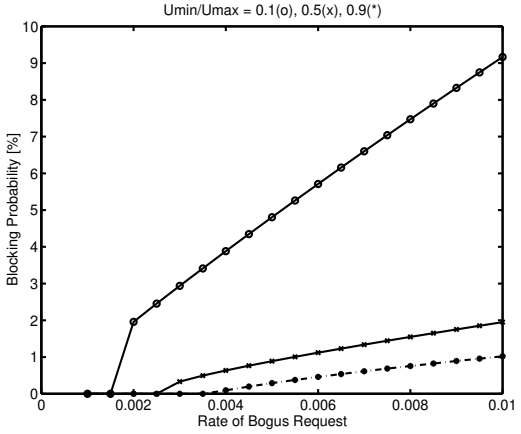


(c) semi-logarithmic plot

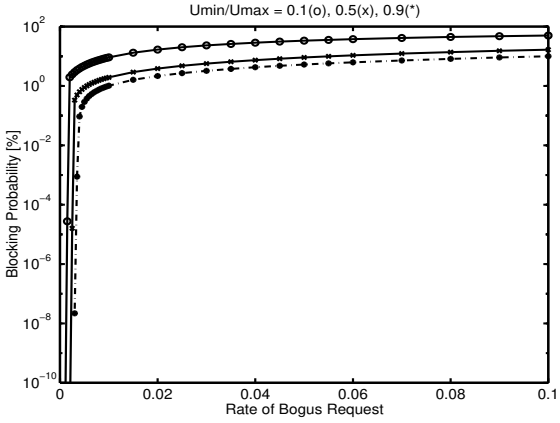
Figure 6: Blocking probability as a function of *rate of bogus request* when the implementation is optimized to give $M' = 160 < M = 256$ but memory capacity is not increased ($C = 1000$).



(a) linear plot



(b) around the threshold



(c) semi-logarithmic plot

Figure 7: Blocking probability as a function of *rate of bogus request* when $M' = M = 256$ but memory capacity $C = 10000$.