Attack Mitigation in IoT Devices

Kamal Lakhara[1], Sunil Shinde[1]
[1] *Department of Computer Science and Engineering,*
Indian Institute of Technology, Guwahati, India 781039
{lkamal, s.dashrath}@iitg.ac.in

Abstract - Internet of Things devices have access to new areas of data and are capable of controlling physical devices. Security is the biggest concern in adopting Internet of Things technology. In order to ensure security in the Internet of Things (IoT), it is imperative to have a comprehensive analysis of the exploits and attacks on the 6LoWPAN adaptation layer. Fragmentation is an important part for communication in IoT devices because Large packet transmission is not always feasible and easy. But fragmenting a packet into smaller fragments invites many security issues. The most notable issues with IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) are the Buffer Reservation Attack and Fragment Duplication Attack. In this term project, we have provided some energy efficient solutions to prevent such kinds of Buffer Reservation Attack and Fragment Duplication Attack.

Keywords - Internet of Things, Fragmentation, Buffer Reservation, Duplication, SHA-2

I. Introduction

IoT devices have taken over the world. We can find IoT devices everywhere. From wearables(fitness tracking devices) to Smart TVs to Security Systems to Home Automation, we are surrounded by IoT devices. The idea that small processing devices should be capable of engaging in IoT and becoming a part of the Internet of Things lead to the idea of 6LoWPAN. It is IPv6 for Low Power Wireless Personal Area Networks. Before diving into the Prevention Algorithms, let us try to understand what are these attacks and how an attacker can use these attacks.

1.1 Buffer Reservation Attack

As the name suggests the attacker tries to reserve the space/buffer of the compromised node. The attacker sends a fake fragment to the compromised node. The compromised node now reserves the buffer for the upcoming fragments thinking that the fragments are coming from the node it expects to come from. Now when the original sender tries to send the fragments and the

compromised node cannot accept it as it has already reserved the buffer for the attacker node. Buffer Reservation Attack causes Denial Of Service for the legitimate/compromised nodes. For example, consider that Sender wants to send Fragments to the compromised node or victim. But before it can send any fragment to it the attacker finds the channel free and sends a fake fragment to the compromised node and the victim node reserves the buffer for the attacker. Now when the original sender sends the 3 fragments to the compromised node, the fragments are dropped as there is no buffer available.

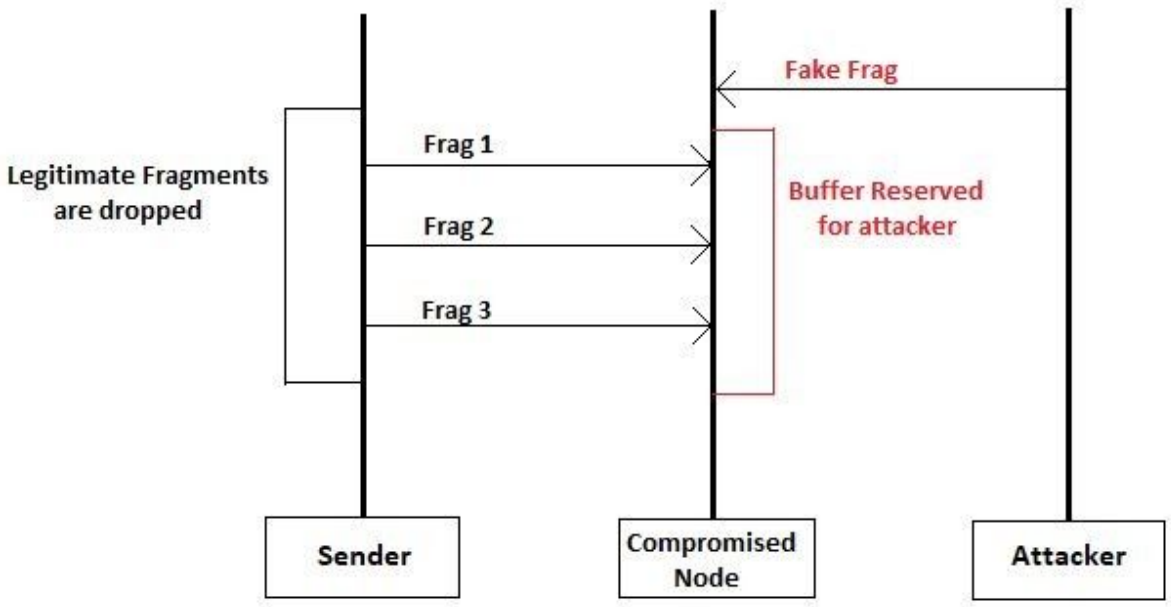

Fig 1.1 Buffer Reservation Attack

1.2 Fragment Duplication Attack

As the name suggests, the attacker tries to send a duplicate fragment to the compromised node and forces the compromised node to stop the communication by dropping all the received fragments. The attacker takes advantage of the fact that the compromised node cannot verify at the 6LoWPAN layer if the fragment received has come from the same source as the previous fragments. To execute such an attack, the attacker first sniffs and observes all the fragments that are sent to the compromised node and then sends a Duplicate Fragment. A receiver node cannot

the packet with two same fragments and ultimately has to drop the fragments and ask for retransmission of the packet. For example, consider that Sender wants to send 3 Fragments of any packet say P1. Sender sends 2 Fragments to the receiving node. During this time, the attacker node observes the channel and sends the Duplicate Fragment 3 before the Sender sends it. Now when the receiver or victim node receives the Fragment 3 from the sender, it realizes that the channel has been compromised and cannot decide which fragment to use for the reassembly purpose at the 6LoWPAN layer. Compromised/Receiver node has to drop all these fragments.Fragment Duplication Attack can also cause DoS(Denial of Service) at the receiver node.

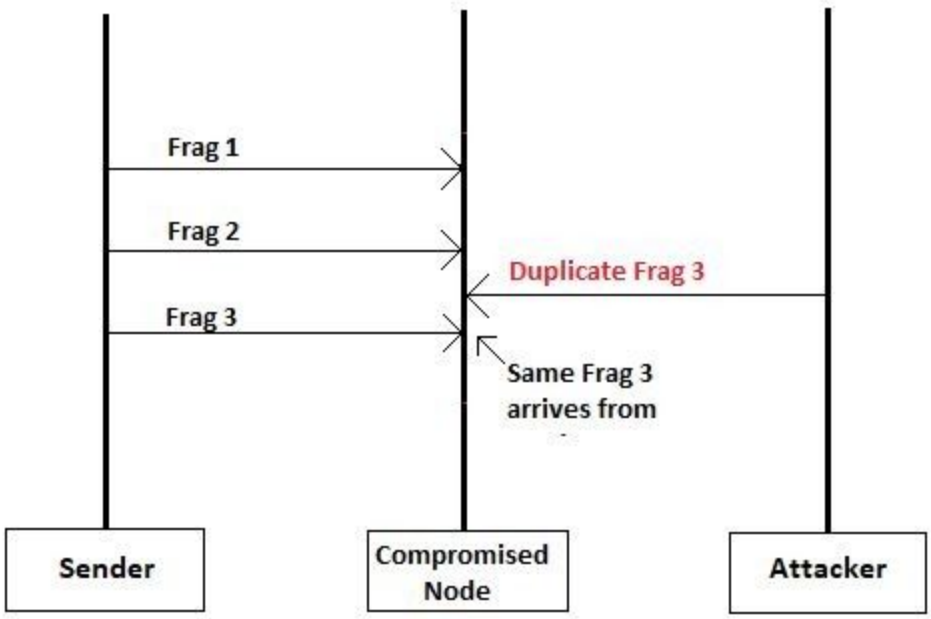

Fig 1.2 Fragment Duplication Attack

II. Related Work

2.1 Buffer Reservation Attack

Fragmenting a packet can be an issue in IP as well as 6LowPAN based communication. Different and notable work have been done on prevention of Buffer Reservation Attack. Authors in Paper [1] used a split buffer method in which a fragment sized buffer was allocated for every new buffer arriving. After the buffer gets full, we can drop any fragment based on its characteristics.

Attackers cannot send a single fragment and reserve the buffer if this approach is followed. But this approach can be computationally expensive. Paper [2] have implemented BRAIN (Buffer Reservation Attack prevention using Legitimacy Score in 6LoWPAN Network). The paper implements a Buffer Allocation Algorithm that keeps track of certain parameters while allocating the buffer. This approach has given good performance against Buffer Reservation Attack. An increased packet drop ratio was observed in the result of this paper which clearly shows that the authors were able to detect and stop such Buffer Reservation Attacks. Paper [3] discusses Reputation Based Buffer Allocation method to prevent Buffer Overflow. In this they have provided an approach to protect devices with less Buffer capacity from overflowing. Our proposed approach follows the same pattern as Paper[2] but with a different and modified algorithm. We have tried to use the Congestion Control approach used at Transport Layer in one of our algorithms.

2.2 Fragment Duplication Attack

Notable works have been done previously for prevention of this attack. Paper [4] discusses one such prevention mechanism where content chaining method is used. In content chaining method, cryptography is used which verifies if the fragments received are from the same packet or different one. In this method, when a fragment is received it is processed and a token that will be used to verify the upcoming node is stored with the receiver node. When the second fragment is received, a hash is computed of the received fragment. If it matches with the token stored, it is declared as a valid fragment. If it doesn't match with the token the fragment is declared fake and dropped immediately. Hence, receiving nodes are able to detect fake fragments using token value.This method requires the received node to store and update the token value everytime a new fragment is received. Our proposed solution uses SHA-2 to prevent the Duplication of Fragments. The idea of using this SHA-2 algorithm striked our mind because one of our faculties at Indian Institute of Technology, Dr. Amit Awekar used to compute the MD5 hash of code we used to submit during our Programming Language Lab course last summer. He did this to ensure nobody could change the code once it is submitted. We cannot use MD5 as we are dealing with online data transmission and MD5 hash is not secure anymore.

III. Proposed Work

3.1 Buffer Reservation Attack

3.1.1 Algorithm-1

Algorithm 1 given below uses the concept of Additive Increase/Multiplicative decrease that is used while Congestion Control at Transport Layer. Algorithm revolves around a Score that keeps updating with every Fragment received. This Score value is used to allocate Buffer to every incoming fragment. If the fragment received is the 1st Fragment of any Packet then this value is initialized with some minimum value based on the first fragment received. After this, on every successful reception of fragment this value keeps doubling till its value becomes ALB/2 where ALB is Available Buffer. After this it increments linearly. In case if the Reassembly Timeout occurs and there is a failure caused by a node, then the score directly becomes the minimum value that is used at the start and the count of failure increments by 1. At any point if the count of failure exceeds the maximum failure allowed then that node is blacklisted and transmission stops. In this Algorithm available buffer size plays an important role in determining attacker nodes.

Notation	Meaning
COUNT	Keeps count of number of Failures
MAF	Maximum Failures allowed
ALB	Available Buffer
RPL	Remaining Packet Length
minVal	Stores Minimum Score Value
LRP	Last Received Packet

Table given above contains the meaning of the abbreviated words used in the algorithm.

Algorithm 1: Buffer Allocation Algorithm

```
COUNT = 0;
ALB = BUFFER.LENGTH ;
MAF = 6;
procedure BufferAllocation(frag)
 if FragmentOffset = 0 then
    if LastReceivedPacket = NULL then
       Score = minVal;
    else
       if LastReceivedPacket = SUCCESS then
          if Score > ALB/2 then
             Score = Score+1;
          else
             Score = min(2*Score,ALB)
          end
       else
          if LastReceivedPacket = TimeOutReAssembly(Failure) then
             Score = minVal ;
             COUNT = COUNT + 1;
             if COUNT > MAF then
                SuspendTranmission (Node);
             end
             return False
          else
       end
    end
    Allocation = min(Score , PacketLength) ;
 else
    if ALB > FragmentSize then
       Allocation = min(Score,RPL - FragmentOffset * SizeOfFragment);
    end
 end
 BufferAllocated = AllocateBuffer(Allocation);
 if BufferAllocated = True then
    ALB = ALB - Allocation;
    return True
 else
    return False
 end
 end procedure
```

For the analysis of this algorithm we can simulate a Buffer Reservation Attack on Contiki Cooja OS with appropriate simulation parameters and check the performance of this Algorithm. For

some unfortunate reasons we could not simulate the attack. We expected that the performance of this algorithm would at least come on par with the Paper[2] algorithm.

3.1.2 Algorithm-2

In this Algorithm, ReAssemblyTimeout timer plays an important role. This time we have used it as a main factor in our Algorithm. Algorithm revolves around a Score that keeps updating with every Fragment received. This Score value is used to allocate Buffer to every incoming fragment. This time we have created a function called getScore() that updates the value based upon the nature of LastReceivedPacket. If there is no LastReceivedPacket i.e 1st packet then the Score is initialized with the minimum value between ALB/10 and Average Packet Length. If the LRP was a success then this score updates based upon its current value. If it is greater than ALB/2 it is assigned with the minimum value of (Score+ALB/10) and (Score+1). And if it is not then Score is incremented by one tenth of ALB. Now if the LRP was a failure or a Timeout occurred, then the score is not directly assigned with minimum value but is decremented by one tenth of current score. This is the situation where this approach is different with existing approaches. If there are three consecutive TimeOut without any successful reception of fragments in between, this approach simply blocks the transmission with that node and blacklists the node. Also if there are more failures collectively i.e there can be failures which are not consecutive and if such failure counts exceed a certain limit, the transmission with that node is blocked and the node is blacklisted. This Score value is used to allocate Buffer to every incoming fragment. Given below is the Score Updation Algorithm followed by getScore function used in the algorithm. Score Updation Algorithm uses the getScore function to update the score value. getScore function updates the value of the score on the basis of received fragment and the status of the last received fragment.

Algorithm 1: Score Updation Algorithm

```
static flag = 0;
procedure ScoreUpdation(frag)
 if FragmentOffset = 0 then
    if LRP = NULL then
        flag = 0;
        Score = getScore(Score,NULL);
        return True
    else
        if LRP = SUCCESS then
            flag = 0;
            Score = getScore(Score,SUCCESS);
            return True
        else
            if LRP = TOA(Failure) then
                if flag==2 then
                    Blacklist the node;
                    return False
                end
                Score = getScore(Score,TOA) ;
                flag++;
                return True
            end
        end
    end
    Allocation = min(Score , PacketLength) ;
else
    if ALB > FragmentSize then
        Allocation = min(Score,RPL - FragmentOffset * SizeOfFragment);
    end
end
BufferAllocated = AllocateBuffer(Allocation);
if BufferAllocated = True then
    ALB = ALB - Allocation;
    return True
else
    return False
end
end procedure
```

Algorithm 1: getScore Function

```
procedure getScore(Score, Status)
  static minimumvalue = min(AVB/10 , AveragePacketLength);
static count = 0;
static MAF = 6;
if Status == NULL then
    Score = minimumValue;
else
    if Status == Success then
        if Score > ALB/2 then
            Score = min(Score+ALB/10,Score+1);
        else
            Score = Score + ALB/10;
        end
    else
        count = count+1;
        if count >= MAF then
            Blacklist the Node;
            return False
        end
        Score = Score - Score/10;
    end
end
end procedure
```

For the analysis of this algorithm, we can simulate a Buffer Reservation Attack on Contiki Cooja OS with appropriate simulation parameters and check the performance of this Algorithm. We expected that the performance of this algorithm would at least come on par with the Paper[2] algorithm.

3.2 Fragment Duplication Attack

3.2.1 Approach-1

We could have used the MD5 hash function or even SHA-1 instead of SHA-2 but they both are less secure than SHA-2. MD-5 hash is prone to well known attacks like birthday paradox attacks. Also SHA-2 is more secure than MD5 for obvious reasons. In this approach, we have used the concept of SHA-2 hash function to prevent this attack. SHA-2 algorithm is a hash function which

takes in input and produces a 160 bit long hash value known as message digest corresponding to that input In this approach, when the sender sends the fragment to the receiving client, it also sends the SHA-2 message digest of the payload along with it. Now, when the attacker node tries to send a duplicate node with the same header but different payload, the receiver node again computes the message digest using the payload of the duplicate fragment. If the value computed doesn't match with the original SHA-2 value that was sent along with the fragment then the receiver node ensures that this is a duplicate node and drops it immediately.

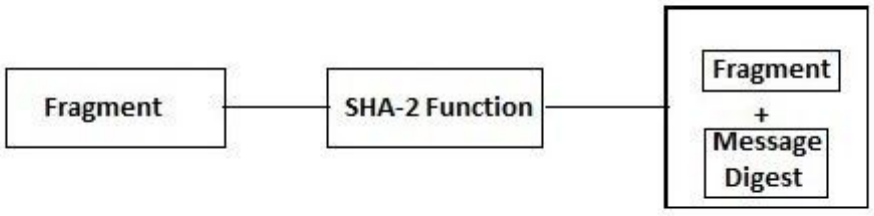

Fig 3.1 Sender Side

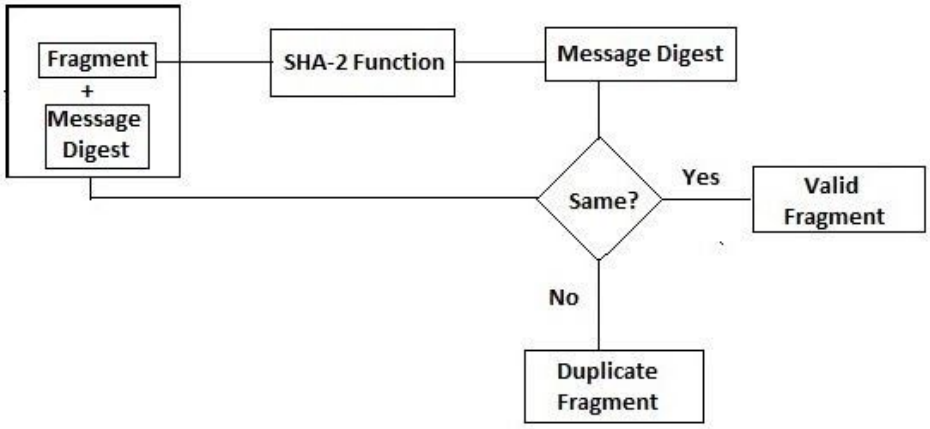

Fig 3.2 Receiver Side

Consider an example. Alice is sending a packet to Bob. While sending every fragment, Alice computes the SHA-2 message digest of the message and sends the message digest along with the payload. Bob also computes the message digest of the payload and matches with the message digest sent by Alice. If it matches Bob is ensured it is a valid fragment. Now consider Alice has sent all the fragments of the packet. Meanwhile, an attacker, say Charlie who wants to interrupt the communication, was observing all the fragments sent by Alice and created his own duplicate Fragment Frag 3' with the same header but loaded some random junk as the payload. Bob is now confused and cannot decide which is the legimitate Frag 3. Bob computes the message digest of the Frag 3's payload and matches with the message digest received with this fragment. Both the SHA-2 message digest values turned out to be different. Bob is now ensured that Frag 3' is a duplicate node and discards it. Bob can now reassemble all the fragments.

3.2.2 Approach-2

In this approach, a protocol can be agreed using a secured channel between the entities before the communication starts. In this protocol, both the entities can agree upon a series of predefined keys/numbers/nonce that are known only to sender and receiver. During transmission, the sender sends a nonce with each fragment in an order along with the payload. Receiver, when receives all the fragments and is reassembling the fragments can verify if any fragment is compromised or a duplicate one with the nonce present in it. Order of these keys/numbers/nonce plays an important role in this. If they are not in order or if any nonce is different than the one agreed, the receiver can ensure it is a duplicate or compromised fragment and can discard it.

In the Figure[3.3], we can clearly see that Bob was expecting Nonce as 10 for Fragment number 5. But the attacker had no idea about the predefined nonce series and he sent the Duplicate Fragment 5' with a random number. Bob is ensured that Frag 5' is a duplicate fragment and discards it while reassembling.

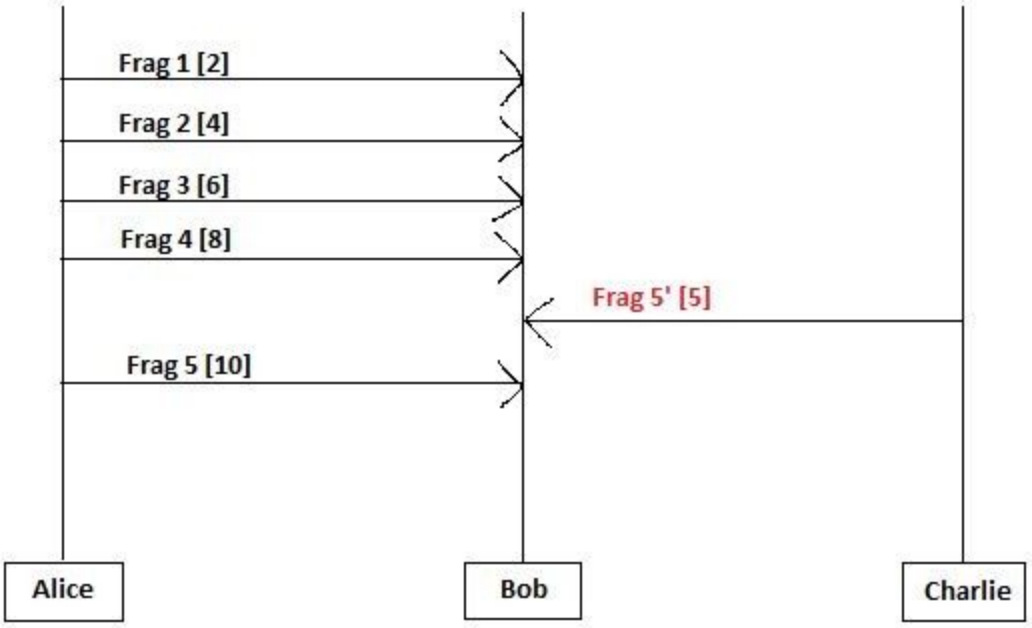

Fig 3.3 Nonce Strategy

IV. Analysis

We believe our Buffer Reservation Prevention Algorithm 1 that uses the concept of AIMD and Algorithm 2 that uses Reassembly Timeout as a major tool would be at par with previous techniques or algorithms used to prevent this attack on a performance basis.The content-chaining scheme discussed in Paper [1] reduces the fragment duplication attack by offering efficient per-fragment sender authentication. We have used the SHA-2 hash function as discussed before to mitigate this attack. We are hopeful that this will reduce such attacks at the 6LoWPAN layer in IoT devices. The 2nd approach discussed above for fragment duplication uses an approach of using some series of predefined key/number/nonce. We are hopeful that these two approaches could mitigate the problems of Fragment Duplication.

V. Conclusion

During this research work, we came across various attacks that are possible on network devices, IoT devices to be specific. We learned how the attacks can be implemented and in this report we have discussed how these attacks can be mitigated or even prevented. With the discussions made above, we learned that vulnerabilities can be explored easily on IoT devices. Security against such attacks or any other attacks should be an important factor while developing any new protocols for IoT devices or networking devices in general. Future work includes finding new attacks that are possible on IoT devices and the strategies to mitigate them. Researchers could work on and improve the approaches discussed here.

Acknowledgment

This research work was guided and supported by Prof. Sukumar Nandi, IIT Guwahati and Pradeepkumar Bhale, Research Scholar at IIT Guwahati. We would like to thank them for providing this wonderful opportunity to work in this research area. Special thanks to Pradeepkumar Bhale for providing continuous guidance during this research work.

References

1. Hummen, R., Hiller, J., Wirtz, H., Henze, M., Shafagh, H., Wehrle, K.: 6LoWPAN fragmentation attacks and mitigation mechanisms. In: Proceedings of the sixth ACM conference on Security and privacy in wireless and mobile networks, ACM(2013) 55-66

2. Bhale, Pradeepkumar & Prakash, Satya & Biswas, Santosh & Nandi, Sukumar. (2019). BRAIN: Buffer Reservation Attack PreventIoN Using Legitimacy Score in 6LoWPAN Network. 10.1007/978-3-030-37484-6_12.

3. Mahmud Hossain, Yasser Karim, and Ragib Hasan. 2018. SecuPAN: A Security Scheme to Mitigate Fragmentation-Based Network Attacks in 6LoWPAN. In Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy (CODASPY '18). Association for Computing Machinery, New York, NY, USA, 307–318.

4. I. Butun, P. Österberg and H. Song, "Security of the Internet of Things: Vulnerabilities, Attacks, and Countermeasures," in IEEE Communications Surveys & Tutorials, vol. 22, no. 1, pp. 616-644, First Quarter 2020, doi: 10.1109/COMST.2019.2953364.

5. A. Becher, Z. Benenson, and M. Dornseif. Tampering with motes: real-world physical attacks on wireless sensor networks. In Proc. of SPC, 2006.