악의적인 호스트 검출을 위한 개선된 타임 체킹 기법*

염정돈, 강동현, 장현수, 엄영익
성균관대학교 정보통신공학부
e-mail:{youmjd, kkangsu, jhs4071, yieom}@ece.skku.ac.kr

Improved Time Checking Scheme for Detecting Malicious Host

Jeong Don Yeom, Dong Hyun Kang, Hyunsu Jang, Young Ik Eom
School of Information and Communications Engineering, Sungkyunkwan University

요 약

이동 에이전트는 분산 컴퓨팅 환경에서 유용하게 쓰이는 컴퓨터 통신 분야의 새로운 응용기술이지만, 에이전트가 이동함으로써 발생하는 여러 가지 보안상의 문제점들을 가지고 있다. 특히, 악의적인 호스트로부터 이동 에이전트를 보호하는 방법의 중요성이 점점 높아지고 있으며, 타임 체킹 기술을 이용한 보호 기법 등이 기존에 제안되었다. 그러나 기존에 제안된 타임 체킹 기술은 데이터를 누적함으로써 네트워크 부하를 초래할 수 있다. 또한, 동일한 도메인 내에서는 동일한 세션키를 사용하여 에이전트를 암호화함으로써 타임 체킹 기법의 중요한 데이터인 에이전트 도착 및 출발 시간이 조작될 가능성이 있다. 본 논문에서는 기존에 제안된 타임 체킹 기술을 기반으로 에이전트의 누적 데이터 크기를 제한하여 네트워크 부하를 줄일 수 있는 기법과 RSA 암호화 기법을 사용하여 이미 기록된 에이전트의 도착 및 출발 시간 조작을 방지할 수 있는 기법을 제안한다.

1. 서론

컴퓨터 기술 및 통신 기술의 발달은 사용자를 대신해 업무를 수행할 수 있는 에이전트를 탄생시켰다. 에이전트는 자율성(autonomy), 사회성(social ability), 반응성(reactivity), 이동성(mobility), 능동성(pro- activeness) 등의 여러 특징을 가지고 있으며, 이는 사용자의 시간과 비용을 감소시킨다. 특히, 이동성은 컴퓨터 네트워크 상에서 에이전트가 자발적으로 호스트를 이동하며 사용자 대신 작업을 수행하는 특징을 말한다[1].

그러나 이동 에이전트가 호스트를 이동함으로써 발생하는 악의적인 에이전트에 의한 호스트 공격, 악의적인 호스트에 의한 에이전트 공격으로부터 반드시 보호되어야만 한다. 특히, 악의적인 에이전트에 의한 호스트 공격은 호스트의 접근 제어 시스템과 인증 등의 여러 가지 방법이 제안되어 보완되었지만 악의적인 호스트에 의한 에이전트 공격은 아직 해결해야 할 문제로 남아 있다[2]. 최근 이에 대한 예방 방법으로 타임 체킹 기법이 제안되었다. 이 기법은 에이전트가 이주할 때마다 에이전트의 도착 시간과 출발 시간을 기록하고 기록된 시간을 3개의 타임 체킹 부등식에 대입한다. 만일 부등식이 성립하지 않으면 해당 호스트를 악의적인 호스트로 판명하여 이동 에이전트를 보호하는 기법이다.

그러나 이 기법은 이주한 호스트의 결과 데이터와 호스트의 도착 및 출발 시간 정보를 최종 호스트가 SNMS(Subnet Management Station)로 전송하기 위하여 누적해야만 한다. 따라서 이주할 호스트의 수가 많을수록 누적 데이터가 점점 증가하게 되어 네트워크에 큰 부하를 줄 수 있는 단점이 있다. 또한, 도메인 내에서 각 호스트가 같은 세션키를 공유하여 암호화 및 복호화하기 때문에 이전 호스트에 의해 기록된 에이전트의 도착 및 출발 시간을 악의적인 호스트가 세션키를 이용하여 쉽게 복호화할 수 있다. 즉, 기존에 기록된 시간을 쉽게 조작할 수 있으므로 악의적인 호스트임에도 불구하고 정상 호스트로 확인될 가능성이 있으며, 반대로 정상 호스트임에도 불구하고 악의적인 호스트로 확인될 수 있는 잘못된 검증 결과를 초래할 수 있다.

본 논문에서는 누적되는 데이터의 양을 최소화하기 위해 일정 간격으로 SNMS에 데이터를 미리 전송하고 RSA 암호화 기법을 이용하여 각 호스트에서 기록된 시간을 다중 암호화하여 악의적인 호스트에 의한 불법 수정을 방지하는 개선된 타임 체킹 기법을 제안한다.

본 논문의 2장에서는 타임 체킹 기법에 대해서 설명한다. 3장에서는 악의적인 호스트 검출을 위한 개선된 타임 체킹 기법을 설명한다. 마지막으로 4장에서는 결론 및 향후 연구 계획을 설명한다.

* 본 연구는 21세기 프론티어 연구개발사업의 일환으로 추진되고 있는 지식경제부의 유비쿼터스컴퓨팅및네트워크원천기반기술개발사업의 08B3-B1-10M 과제로 지원된 것임.

2. 관련연구

2.1 타임 체킹 기법

타임 체킹 기법은 이동 에이전트가 다른 호스트로 이주시 출발 및 도착 시간 정보를 기록한 후 기록된 정보와 세 가지 검증 부등식을 이용하여 악의적인 호스트를 검출하는 보호 기법이다. 현재 에이전트가 작업 중인 도메인을 SNMD(Subnet Management Domain)라고 하며, 초기 에이전트를 생성하는 홈 서버를 SNMS 라고 한다. SNMD에서 악의적인 호스트를 검출하기 위해 SNMS는 먼저 아래의 작업을 수행해야 한다[3].

- 서브넷에서 모든 호스트에 의해 참조되는 레퍼런스 타임을 생성한다.
- 네트워크 관리 시스템의 보안 레벨에 따라 오차 시간(time tolerance) ξ를 결정한다.
- 커뮤니케이션 키로 암호화한 세션키 k를 도메인의 모든 노드로 전송한다.
- 에이전트를 생성하고, 데이터 또는 흐름 제어를 난독화(obfuscation)한 후 에이전트를 세션키로 암호화하여 첫 번째 호스트로 전송한다.

에이전트가 호스트로 이주하면 각 호스트는 아래의 작업을 수행한다.

- 에이전트의 도착시간 T_{i-arr}을 기록한다.
- 에이전트 코드를 복호화 한 후 실행한다.
- 에이전트의 출발시간 $T_{i-leave}$을 기록한다.
- 에이전트 코드와 결과 데이터, T_{i-arr}, $T_{i-leave}$를 암호화 한다.
- 다음 호스트로 에이전트를 전송하고 마지막 호스트가 모든 결과와 코드를 SNMS로 보낼 때까지 위의 과정을 반복하여 수행한다.

마지막으로 에이전트가 SNMS로 이주하면 아래의 작업을 수행한다.

- 도착시간 T_{0-arr}를 기록한다.
- 결과와 각 호스트에서의 에이전트 도착, 출발시간을 복호화 한다.
- 타임 체킹을 수행 한다.

다음은 타임 체킹시 악의적인 호스트를 검출하기 위해 필요한 세 가지 타임 체킹 부등식이다.

$$T_{i-leave} - T_{i-arr} + T_{i-send} \leq T_{i-exe} + \xi \quad (1)$$

$$T_{i+1-arr} \geq T_{i-leave} + T_{i-send} \quad (2)$$

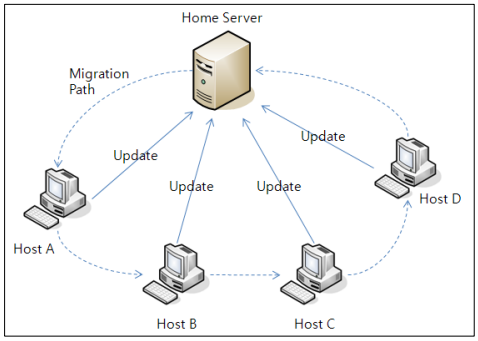
$$|T_{i+1-arr} - T_{i-leave} - T_{i-send} - T_{i-tran-i+1}| \leq \xi \quad (3)$$

(1)번 식은 실제 실행 시간이 예상 오차 시간(tolerance time)을 초과하면 해당 호스트는 믿을 수 없는 호스트임을 나타낸다. 만약 부등식이 성립하지 않으면 해당 호스트는 믿을 수 없다. (2)번 식에서는 호스트의 출발 시간이 다음 호스트의 도착시간보다 클 수 없다는 것을 나타내며, 모든 믿을 수 있는 호스트는 부등식을 만족해야 한다. 만일 부등식에 성립하지 않으면 두 개의 호스트는 모두 믿을 수 없다. (3)번 부등식은 전송지연과 관련된 부등식이다. 만일 부등식에 성립하지 않으면 두 호스트는 믿을 수 없다.

위와 같이 타임 체킹 기법은 세 부등식의 결과에 따라 악의적인 호스트를 판단한다. SNMS는 악의적인 호스트의 결과를 폐기하거나 에이전트를 다시 보내고 모든 악의적인 호스트를 제외시킨다. 오차 ξ의 크기는 직접적으로 보안성을 결정하며, 오차시간 ξ를 결정하는 것은 부등식의 정확성에 영향을 주는 중요한 부분이다.

2.2 에이전트 위치 추적 기법

에이전트는 사용자를 대신하여 여러 호스트를 이주하면서 작업을 수행하기 때문에 특정 시점에 에이전트가 어느 호스트에서 작업을 수행하고 있는지 알아내는 것은 어렵다. 그래서 에이전트 위치 추적 기법을 통해 에이전트의 위치를 파악해야 한다[6,7].



(그림 1) 홈 프록시 기법

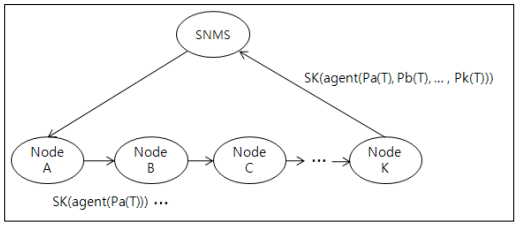
홈 프록시 기법은 에이전트 위치 추적 기법중의 하나로 이동 에이전트의 위치 정보를 이동 에이전트가 생성된 홈 호스트 또는 특정 서버에 저장하는 방식이다. 전체적인 동작 과정은 그림 1과 같다.

- 이동 에이전트는 이주시마다 자신의 위치 정보를 홈 호스트의 위치 정보 저장소에 저장한다.
- 홈 서버는 이 정보를 이용하여 이동 에이전트를 찾고 메시지를 전달한다.

3. 악의적인 호스트 검출을 위한 개선된 타임 체킹 기법

3.1 RSA 암호화 기반의 시간 조작 방지 기법

에이전트와 각 노드의 도착시간 T_{i-arr}, 출발시간 $T_{i-leave}$을 세션키(SK)로만 암호화 할 경우 이전 노드에서 기록된 시간을 다음 노드에서 조작할 수 있다. 시간이 변경되면 타임 체킹 결과에 영향을 미칠 수 있으므로 이전에 기록된 시간을 다른 노드에서 조작할 수 없도록 개선된 기법이다. 그림 2에서와 같이 RSA암호화 기법을 적용하여 각 노드에서 자신의 비밀키(Px)로 시간을 암호화한 후에 다시 세션키(SK)로 암호화하여 다음 노드로 이주하도록 한다.



(그림 2) 시간 암호화 과정

SNMS는 물론 각 노드에서도 공개키로 복호화하여 원래의 데이터를 얻을 수 있지만 각 노드는 이전 노드의 비밀키를 모르기 때문에 이전 데이터를 조작할 수 없다.

- SNMS에서 에이전트를 생성하여 세션키(SK)로 암호화 후 노드A로 전송한다. 단, 세션키는 사전에 분배된다.
- 노드A는 복호화 후 에이전트를 실행하고 도착시간 T_{i-arr}과 출발시간 $T_{i-leave}$을 기록한다.
- 기록된 시간을 노드A의 비밀키(Pa)로 암호화한 후 다시 에이전트를 세션키(SK)로 다중 암호화하여 다음 노드로 전송한다.

위의 과정을 마지막 노드가 SNMS로 전송할 때 까지 반복한다.

3.2 네트워크 부하 감소 기법

일정 간격으로 데이터를 SNMS로 미리 전송하는 방법은 일정 수의 홉 단위로 처리하는 방법과 데이터 크기를 제한하여 초과 시 SNMS로 전송하도록 하는 두 가지 방법이 있다. 이 중 홉 단위로 처리하는 방법은 각 홉에서의 데이터 크기가 매우 클 경우 무조건 일정 수의 홉 단위로 처리하므로 누적 시 데이터가 매우 커져 네트워크에 부하를 줄 수 있는 문제점이 있다. 따라서 후자인 데이터 크기를 제한하여 홉 수를 결정할 수 있도록 하는 방법을 사용하였다. 먼저 데이터 크기를 제한하여야 하기 때문에 각 호스트의 결과 데이터 크기를 누적할 변수 S와 제한 크기가 저장될 변수 L을 에이전트에 포함시킨다. 또한 에이전트는 호스트를 이주할 때마다 자신의 위치를 SNMS의 위치 저장소에 저장하여 SNMS에서 에이전트의 현재 위치를 알 수 있도록 한다.

먼저, 초기값을 설정하는 과정은 다음과 같다.

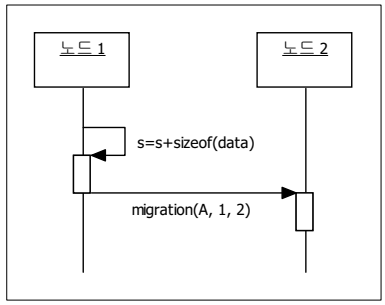
- SNMS로부터 에이전트가 이주해 오면 해당 호스트에서 에이전트를 실행한다.
- 실행 후 결과 데이터의 크기를 얻는다.
- 얻은 크기의 3배 값을 L변수에 저장한다.
  즉, 약 3홉의 단위로 처리한다. (공모 공격의 최소 홉 수)
- 누적할 변수 S를 결과 데이터 크기로 초기화 한다.

초기값 설정이 끝난 후 각 노드는 에이전트를 수신시 아래의 작업을 수행한다.

- 에이전트를 복호화 한 후 실행한다.
- 결과 데이터의 크기를 얻는다.
- 제한값 L에서 누적 데이터 크기 S를 뺀 값(L-S)과 결과 데이터의 크기를 비교한다.

그림 3과 같이 만약 노드1의 결과 데이터 크기가 L-S의 값보다 작을 경우 아래의 과정을 수행한다.

- 노드1의 데이터를 누적 가능하므로 실제 데이터를 누적하고 변수 S에는 데이터 크기를 누적하여 다음 노드에서 빈 공간을 계산 할 수 있도록 한다.
- 다음 노드로 이주 후 초기값 설정 이후의 과정을 데이터 크기가 제한 크기보다 더 커질 때 까지 반복한다.



(그림 3) 데이터 크기를 누적 후 에이전트 이주 과정

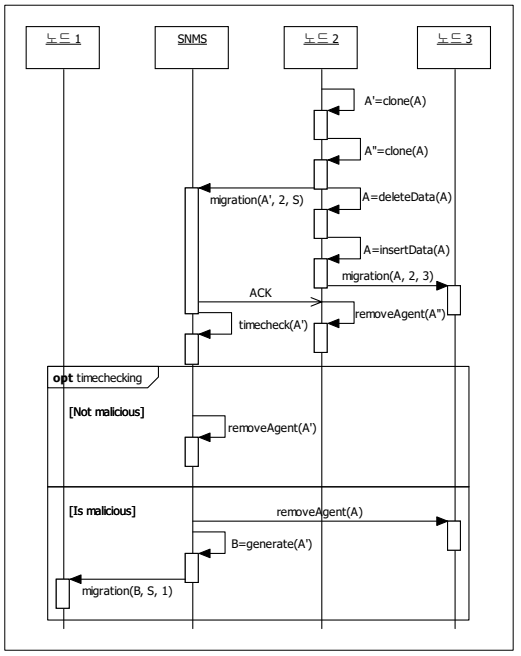
만약 결과 데이터의 크기가 제한 크기(L-S)보다 클 경우에는 그림 4와 같이 다음의 과정을 수행한다.

- 에이전트의 실행 후 결과 데이터의 크기가 L-S 즉, 누적 가능한 빈 공간보다 클 경우 데이터를 누적할 수 없으므로 이전 데이터까지만 가진 에이전트 A의 복제본 A′, A′′ 을 생성한다.
- 에이전트 A는 데이터를 삭제 후 다음 호스트로 이주시 필요한 해당 호스트의 데이터를 넣어 다음 호스트로 이주하여 작업을 계속 수행한다. 이때, 초기값 설정 과정을 수행하여 S와 L의 초기값을 다시 설정함으로써 L의 값이 매우 작게 설정되었을 경우 한 노드의 결과 데이터도 누적할 수 없게 되어 무한 루프에 빠지는 것을 방지한다.
- 동시에 A′은 SNMS로 이주시켜 타임 체킹을 하도록 하며, A′′은 A′이 SNMS로 정상적으로 이주하지 못했을 경우 재전송을 위한 복제 에이전트로 복제한 호스트에서 머무르게 한다.
- SNMS는 A′이 이주해 오면 출발 호스트로 ACK를 보내 정상적으로 A′이 정상적으로 이주했다는 것을 알리며, ACK를 받은 호스트는 복제 에이전트인 A′′을 소멸시킨다. 만약 ACK를 받지 못하면 일정 시간 이후에 A′′의 복제본을 재전송 한다.

에이전트가 SNMS로 정상적으로 이주하면 타임 체킹 결과에 따라 아래와 같이 작업을 수행한다.

- 정상적인 호스트로 확인되는 경우
  - 타임 체킹을 위해 SNMS로 이주했던 에이전트 A′을 소멸시킨다.
  - 에이전트 A는 계속 작업을 수행한다.

- 악의적인 호스트로 확인되는 경우
  - SNMS는 에이전트 위치 추적 기법인 홈 프록시 기법을 통해 에이전트 A가 수행중인 호스트를 찾아 에이전트 A를 소멸하도록 메시지를 보낸다.
  - 에이전트 A′과 같은 작업을 하는 에이전트 B를 생성한다.
  - 에이전트 B는 가장 최근에 검증받은 정상 호스트로 이주하여 초기값 설정 과정부터 반복한다.



(그림 4) 타임 체킹 전 후 과정

4. 결론

본 논문에서는 악의적인 호스트로부터 이동 에이전트를 보호하는 방법 중 하나인 타임 체킹 기법의 네트워크 부하와 보안상의 문제점을 지적하고 각각의 해결 방안을 제안하였다.

제안한 첫 번째 방법은 각 노드의 비밀키로 시간을 암호화하고 에이전트를 세션키로 다중 암호화해서 이주하도록 하여 보안상의 문제점을 해결하였다.

두 번째는 누적 데이터 크기를 제한하여 결과 데이터가 제한 크기를 초과할 시에는 데이터를 계속 누적하지 않고 미리 SNMS로 전송하도록 하여 네트워크 부하를 감소시켰다.

제안 기법을 통해 타임 체킹 기법의 가장 중요한 데이터인 시간이 조작되는 것을 막을 수 있어 타임 체킹의 신뢰성 및 정확성을 향상 시킬 수 있다. 또 에이전트가 모든 노드를 거치고 돌아왔을 때야 비로소 악의적인 호스트를 검출할 수 있었던 것을 중간 중간에 미리 악의적인 호스트를 검출할 수 있도록 하였다.

그러나 이주하는 에이전트가 계속 자신의 위치를 홈 서버로 전송해야 하는 오버헤드가 존재한다.

향후연구로는 위의 오버헤드를 줄일 수 있는 방법과 공모 공격에도 강한 타임 체킹 기법에 대한 연구가 계속적으로 이루어 져야 할 것이다.

참고문헌

[1] N. Borselius, "Mobile Agent Security," Electronics & Communication Engineering Journal, Vol. 14, No. 5, pp. 211-218, 2002.

[2] A. Wagner, "Implementing Mobile Agent Security in an Untrusted Computing Environment," Telecommunications, Proceedings of the 8th International Conference, Vol. 2, pp. 591-594, 2005.

[3] W. Jiehong, G. Xiaochun, T. Cuihua, Y. Hang, and C. Guiran, "The Study for Protecting Mobile Agents Based on Time Checking Technology," IEEE International Conference on Robotics and Biomimetics, pp. 2013-2017, 2007.

[4] T. Sander and C. Tschudin, "Towards Mobile Cryptography," IEEE Symposium on Security and Privacy, pp. 215-224, 1998.

[5] R. Rivest, A. Shamir, and L. Adleman, "A Method of Obtaining Digital Signature and Public Key Cryptosystem," Communications of the ACM, Vol. 21, No. 2, pp. 120-126, 1978.

[6] D. Deugo, "Mobile Agent Messaging Models," 5th International Symposium on Autonomous Decentralized Systems, pp. 278-286, 2001.

[7] J. Baumann, "A Comparison of Mechanisms Locating Mobile Agents," Fakultät Informatik Fakultätsbericht, 1999.