creative commons

COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

공학석사학위논문

소셜네트워크와 유전알고리즘을 이용한 빠른 패치 전파 기법

Fast Patch Distribution Strategy Using Social Network and Genetic Algorithm

2014 년 2 월

서울대학교 대학원

전기 · 컴퓨터 공학부

강 동 화

공학석사학위논문

소셜네트워크와 유전알고리즘을 이용한 빠른 패치 전파 기법

Fast Patch Distribution Strategy Using Social Network and Genetic Algorithm

2014 년 2 월

서울대학교 대학원

전기 · 컴퓨터 공학부

강 동 화

소셜네트워크와 유전알고리즘을 이용한 빠른 패치 전파 기법

Fast Patch Distribution Strategy Using Social Network and Genetic Algorithm

지도교수 조 유 근

이 논문을 공학석사학위논문으로 제출함

2013 년 11 월

서울대학교 대학원

전기 · 컴퓨터 공학부

강 동 화

강동화의 석사학위논문을 인준함

2013 년 12 월

위 원 장	민 상 렬	(인)
부위원장	조 유 근	(인)
위 원	김 지 홍	(인)

요약

온라인 소셜네트워크 사이트 사용자가 늘어남에 따라 온라인 소셜네트워크 사이트를 대상으로 한 공격이 증가하고 있다. 온라인 소셜네트워크 사이트 사용자들 사이에는 강한 연결성을 가지고 있어서 사용자들의 흥미를 끄는 정보가 빠르게 전파 된다. 사용자들의 흥미를 끌게 하는 정보로 위장한 악성코드가 온라인 소셜네트워크 사이트에 퍼질 경우 큰 피해를 일으킬 수 있다. 또한 새로운 유형의 악성코드는 예방하기 어려우며, 최소한의 시간을 소요해서 패치(Patch)를 만들어 배포한다 하더라도 악성코드의 전파가 시작 된 이후에 패치 전파가 시작 된다. 따라서 악성코드의 활동을 억제 시킬 수 있는 빠른 패치 전파 기법이 필요하다.

본 논문에서는 온라인 소셜네트워크를 기반으로 하면서 페이지랭크(PageRank)를 이용한 빠른 패치 전파 기법과 유전알고리즘(Genetic Algorithm)을 이용한 빠른 패치 전파 기법을 제시하고 그 성능을 비교 분석 한다. 제안 기법들에서는 온라인 소셜네트워크 사용자들 간의 교류 관계를 그래프로 표현한다. 페이지랭크를 이용한 기법에서는 먼저 그래프를 분할 후 각 부분 그래프마다 페이지랭크 계산을 통해서 패치 전파에 가장 많이 기여 할 수 있을 것으로 기대되는 사용자들을 찾는다. 유전알고리즘을 이용한 기법에서는 반복적인 유전알고리즘 연산을 통해서 패치 전파에 가장 많이 기여 할 수 있을 것으로 기대되는 사용자들을 찾는다. 선택 된 사용자들에게 1차로 패치를 전달하고 패치를 받은 사용자들은 자신과 교류가 있는 사용자들에게 패치를 전파 한다. 나머지 사용자들도 패치를 받게 되면 자신과 교류가 있는 사용자들에게 패치를 전파 한다.

제안 기법을 검증하기 위해 온라인 소셜네트워크 사용자들의 사용 패턴을 모델링하고 제안된 기법을 구현해서 시뮬레이션 하였다. 시뮬레이션 결과,

사용자들을 무작위로 선택하는 방법과 교류하는 사용자들의 수가 많은 사용자들을 선택하는 방법에 비해서 페이지랭크를 사용하는 방법과 유전알고리즘을 사용한 방법에서 더 빠르게 패치를 전파했음을 확인 할 수 있었다. 또한 유전알고리즘을 사용한 방법이 페이지랭크를 사용하는 방법보다 더 빠르게 패치를 전파했음을 확인 할 수 있었다.

주요어: 소셜네트워크, 그래프, 유전알고리즘, 패치 전파, 핵심 노드, 보안
학번: 2012-20725

목차

요약 **i**

목차 **iii**

그림 목차 **v**

표 목차 **vii**

제 1 장 서론 **1**

제 2 장 온라인 소셜네트워크 개요 **4**

2.1 온라인 소셜네트워크 구조의 특징 4

2.1.1 멱함수 법칙 4

2.1.2 짧은 네트워크 지름 5

2.2 소셜네트워크를 기반으로 전파 된 악성코드 6

2.2.1 Samy Worm 6

2.2.2 Koobface Worm 6

2.2.3 Bom sabado Worm 6

2.2.4 Ramnit 7

2.2.5 haha photo message 7

제 3 장 관련 연구 **8**

3.1 소셜네트워크를 이용한 공격 기법 8

3.2 소셜네트워크를 이용한 공격 탐지 기법 10

3.3 소셜네트워크를 이용한 방어 기법 12

제 4 장 소셜네트워크를 이용한 패치 전파 기법 **16**

4.1 소셜 그래프 생성 16

4.2 핵심 노드 선택 18

4.2.1 페이지랭크를 이용한 핵심 노드 선택 18

4.2.2 유전알고리즘을 이용한 핵심 노드 선택 22

4.3 패치 전파 30

제 5 장 시뮬레이션 결과 및 분석 **31**

5.1 시뮬레이션 환경 31

5.2 소셜네트워크에서의 패치 전파 모델 31

5.3 시뮬레이션 결과 분석 32

5.3.1 시간에 따른 패치 전파율 33

5.3.2 시간에 따른 악성코드 감염률 35

5.3.3 소셜 그래프의 변화에 따른 최대 피해율 38

5.3.4 악성코드 감지 시간에 따른 최대 피해율 40

제 6 장 결론 **41**

Abstract **46**

그림 목차

그림 1.1 페이스북 사용자 증가 추세 1
그림 1.2 시간에 따른 새미 웜에 의한 피해 규모 2

그림 2.1 멱함수 법칙을 따르는 유튜브 사용자들의 차수 분포 4

그림 3.1 공격 코드가 내장 되어 있는 페이스북 어플리케이션 9
그림 3.2 'Photo of the Day' 어플리케이션의 HTTP 요청 규모 9
그림 3.3 Xu 등이 제안 한 소셜네트워크에서의 웜 감지 시스템 구조 10
그림 3.4 사용자 간 MMS 트래픽 량을 기반으로 생성 된 그래프 . . . 13
그림 3.5 그래프의 가중치를 고려해서 분할 된 그래프 14

그림 4.1 소셜네트워크를 이용한 패치 전파 기법 17
그림 4.2 $n = 11$인 소셜 그래프 18
그림 4.3 Zhu등이 정의한 'Connectivity' 19
그림 4.4 페이지랭크 계산 예 23
그림 4.5 유전알고리즘의 전체 구조 24
그림 4.6 핵심 노드에 따른 품질 계산 결과 26
그림 4.7 일점 교차 연산 27
그림 4.8 다점 교차 연산 (교차점이 2개인 경우) 28
그림 4.9 변이 연산 29
그림 4.10 수선 29

그림 5.1 정보를 다른 사용자에게 전달하는데 소요되는 시간 분포 . 32
그림 5.2 페이스북에서의 패치 전파 34
그림 5.3 트위터에서의 패치 전파 34

그림 5.4 구글 플러스에서의 패치 전파 35
그림 5.5 페이스북에서의 악성코드 감염률 36
그림 5.6 트위터에서의 악성코드 감염률 36
그림 5.7 구글플러스에서의 악성코드 감염률 37
그림 5.8 소셜 그래프 변화에 따른 최대 감염률 39
그림 5.9 DT(Detection Threshold)에 따른 최대 감염률 40

표 목차

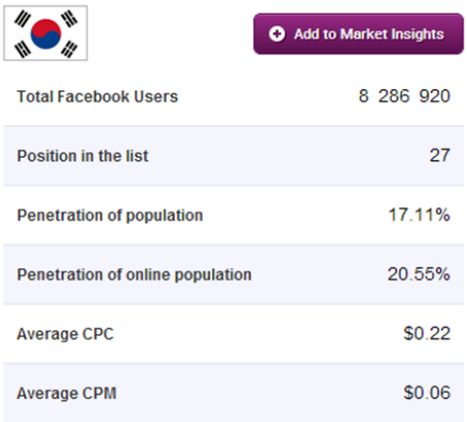
표 2.1 웹 페이지, 플리커, 라이브저널을 표현 한 그래프의 지름과 반지름 5

표 5.1 시뮬레이션에 사용 된 하드웨어와 운영체제 31

표 5.2 시뮬레이션에 사용 된 소셜네트워크 그래프 데이터 33

제 1 장 서론

온라인 소셜네트워크 서비스 사용자가 증가하고 있다. 그림 1.1은 온라인 소셜네트워크 서비스 중 하나인 페이스북(Facebook)의 사용자 증가 추세를 보여주고 있다 [5]. 사용자가 많은 온라인 소셜네트워크 사이트에서 흥미로운 정보가 노출 될 경우 짧은 시간에 많은 사용자들에게 전파 된다. 국내에서는 페이스북, 트위터(Twitter) 사용자가 많으며, 정보나 일상 공유를 목적으로 사용되고 있다 [1, 5, 7]. 기업이나 공공기관에서는 정보 전파 속도가 빠르다는 점을 이용해서 홍보를 목적으로 많이 사용하고 있다. 온라인 소셜네트워크 서비스들은 대부분 사용자가 이용하기 쉬운 웹 페이지(Web Page)를 기반으로 제공하는 경우가 대부분이다. 정보를 제공하는 사용자들은 공유하고자 하는 정보를 텍스트(Text)뿐만 아니라 웹 페이지에서 사용되는 자바스크립트(Javascript), HTML(Hypertext Markup Language) 등을 활용하여 표현 할 수도 있다.



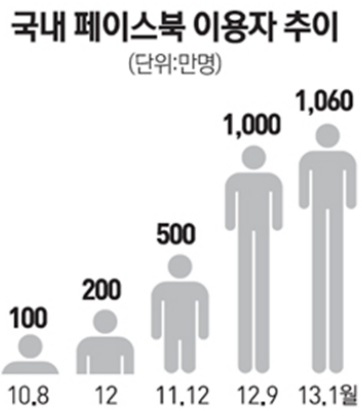


그림 1.1 페이스북 사용자 증가 추세

하지만 OWASP(Open Web Application Security Project) Top 10에 따르면 웹 페이지는 여전히 SQL Injection, XSS(Cross-Site Scripting) 등에 취약하며

사용자들은 이러한 공격들을 인지하지 못하는 경우가 대부분이다 [4]. 온라인 소셜네트워크 사이트들도 웹 페이지를 기반으로 서비스를 제공하는 경우가 대부분이다. 따라서 온라인 소셜네트워크 사이트들은 웹 페이지 취약점들을 악용한 다양한 공격에 노출되어 있으며 빠른 전파력을 가지고 있기 때문에 그 피해가 짧은 시간에 크게 증가한다. 그림 1.2는 마이스페이스(Myspace)에서 전파 되었던 새미 웜(Samy Worm)에 감염 된 사용자 수를 시간에 따라서 보여주고 있다 [10]. 공격을 막기 위한 패치를 짧은 시간 안에 만들어 전파시키더라도 공격이 시작 된 시점 이후에 패치 전파가 시작 될 수밖에 없다. 따라서 이미 시작 된 공격에 대한 피해를 최소화시키기 위해서는 패치 전파 속도를 공격 전파 속도보다 더 빠르게 하는 기법이 필요하다.

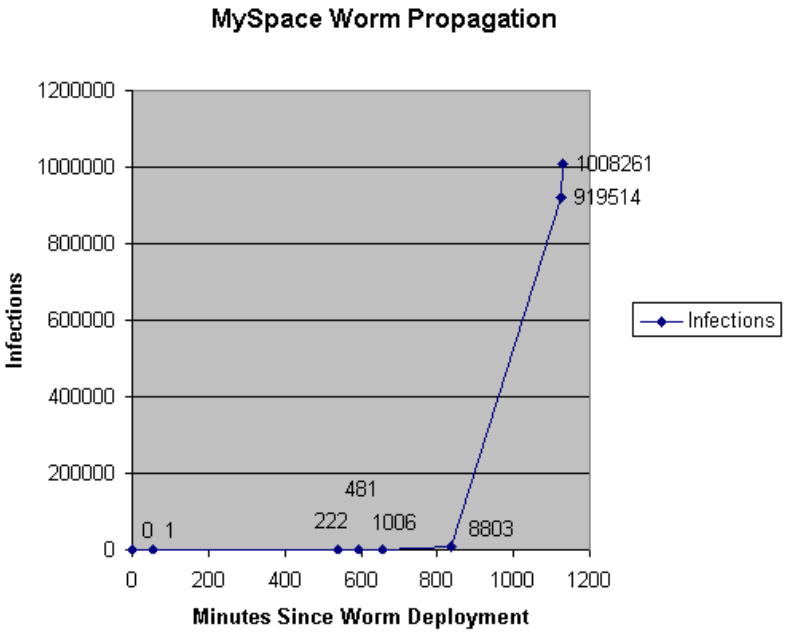


그림 1.2 시간에 따른 새미 웜에 의한 피해 규모

본 논문에서는 소셜네트워크 사이트 사용자들의 교류 관계를 기반으로 하고 페이지랭크를 이용한 빠른 패치 전파 기법과 유전알고리즘을 이용한 빠른 패치 전파 기법을 제안한다. 제안 기법에서는 다음과 같은 사항들을 가정한다.

- 공격자는 악성코드를 전파하고 방어자는 패치를 전파한다.
- 악성코드와 패치는 공격자 또는 방어자가 선택 한 사용자들에게 1차로 부여되며 이후에는 사용자들 사이의 교류관계로 인해서 전파 된다.
- 악성코드와 패치는 감염 된 사용자가 사용하는 하드웨어, 운영체제, 웹 브라우저와 무관하게 전파 가능하다.
- 악성코드와 패치는 그 자체가 직접 전달되는 경우뿐만 아니라 다운로드 및 실행을 유도하는 URL(Uniform Resource Locator)이 전달되는 경우에도 전파 된 것으로 판단한다.
- 공격자는 소셜네트워크 사이트 사용자들의 교류 관계를 파악 할 수 없고, 방어자는 사용자들의 교류 관계를 파악하고 분석 할 수 있다.

제안 기법에서는 온라인 소셜네트워크 사용자들의 교류관계를 파악해서 그래프로 표현 하고, 페이지랭크와 유전알고리즘을 사용해서 패치 전파에 가장 많이 기여 할 것으로 보이는 사용자들을 찾는다. 페이지랭크를 이용한 기법에서는 사용자들 사이의 연결 관계를 고려해서 분할과정을 거친 후 페이지랭크 계산을 하고, 유전알고리즘을 이용한 기법에서는 대규모 네트워크에서 일부 사용자를 찾는 규모가 큰 문제를 다루기 때문에 최적에 가까운 해를 찾을 수 있다. 선택 된 사용자는 최초로 패치를 받게 되며, 자신과 교류하고 있는 사용자에게 패치를 전파한다. 또한 나머지 노드들도 자신에게 패치가 오면 자신과 교류하는 사용자들에게 패치를 전파한다.

본 논문의 구성은 다음과 같다. 2장에서는 소셜네트워크의 개요에 대해서 알아본다. 3장에서는 관련 연구에 대해서 알아본다. 4장에서는 제안 된 소셜네트워크를 이용한 빠른 패치 전파 기법에 대해서 설명한다. 5장에서는 시뮬레이션을 통해서 제안 된 기법의 성능을 측정하고 그 결과를 분석한다. 마지막으로 6장에서 결론을 맺는다.

제 2 장 온라인 소셜네트워크 개요

본 장에서는 온라인 소셜네트워크 사용자들의 연결 관계를 나타낸 그래프의 특성에 대한 기존 연구에 대해서 알아본다. 또한 기존에 전파 되었던 온라인 소셜네트워크 악성코드에 대해서 알아본다.

2.1 온라인 소셜네트워크 구조의 특징

2.1.1 멱함수 법칙

멱함수 법칙(Power Law)은 네트워크 과학에 적용되는 법칙이다. 식 2.1은 변수 x에 대한 멱함수 법칙을 보여준다. 온라인 소셜네트워크에서는 사용자들을 노드로 표현하고 사용자들의 교류 관계를 간선으로 표현한 그래프를 생성 할 수 있다. Mislove 등은 온라인 소셜네트워크 서비스 중 하나인 유튜브(YouTube) 사용자들의 관계를 표현한 그래프에서 각 노드들의 차수의 분포가 멱함수 법칙을 따른다는 것을 보였다 [15]. 그림 2.1은 유튜브 사용자들의 관계를 표현한 그래프 내 노드들의 차수 분포를 보여주고 있다.

$$f(x) = ax^k \tag{2.1}$$

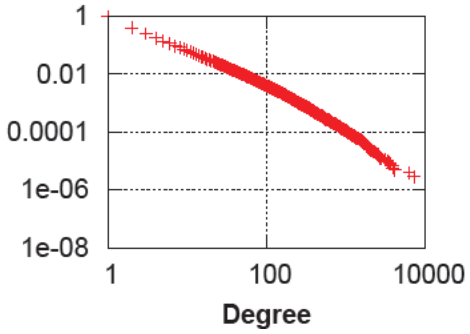


그림 2.1 멱함수 법칙을 따르는 유튜브 사용자들의 차수 분포

통계학적 분석으로 멱함수 법칙과의 유사도를 파악하기 위해서 멱함수 분포와 유튜브 사용자들을 표현한 차수 분포에 대해서 콜모고로프 스미르노프 검증(Kolmogorov-Smirnov Test)을 수행하였다. 수행 결과 두 분포의 거리(Distance)는 0.0094가 나왔고 이 값은 유튜브 네트워크가 멱함수 법칙을 잘 따르고 있음을 보여준다. 온라인 소셜네트워크 구조가 멱함수 법칙을 따른다는 것은 소수의 노드들이 매우 높은 차수를 가지고 있음을 보여준다.

2.1.2 짧은 네트워크 지름

Mislove 등은 온라인 소셜네트워크 서비스인 플리커(Flickr), 라이브저널(LiveJournal) 사용자들의 관계를 표현한 그래프를 생성하였다. 또한 비교 분석을 위해서 웹 페이지에 대해서도 같은 과정을 수행 하였다. 플리커와 라이브저널을 표현한 그래프에서는 각 노드는 사용자들을 나타내며 각 간선은 사용자들의 교류 관계를 나타낸다. 웹 페이지를 표현한 그래프에서 각 노드는 각 웹 페이지를 나타내며 각 간선은 웹 페이지들 사이의 하이퍼링크를 나타낸다. 생성한 각 그래프의 반지름(Radius)과 지름(Diameter)을 측정하였다. 표 2.1은 각 네트워크의 반지름과 지름을 보여준다.

Network	Users/Pages	Radius	Diameter
웹 페이지	8,600,000	475	905
플리커(Flickr)	1,846,198	13	27
라이브저널(LiveJournal)	5,284,457	12	20

표 2.1 웹 페이지, 플리커, 라이브저널을 표현 한 그래프의 지름과 반지름

웹 페이지에 비해서 온라인 소셜네트워크 서비스인 플리커와 라이브저널을 표현하는 그래프의 지름과 반지름이 상당히 작음을 알 수 있다. 지름과 반지름이 작으면 임의의 두 노드 사이의 거리가 짧다는 것을 의미한다. 이는 그래프 상에서 정보가 빠르게 전파 될 수 있음을 보여준다.

2.2 소셜네트워크를 기반으로 전파 된 악성코드

2.2.1 Samy Worm

새미 웜은 온라인 소셜네트워크를 기반으로 최초 악성코드이며 2005년에 전파 되었다. 미국에서 제공 되었던 온라인 소셜네트워크 서비스인 마이스페이스(Myspace) 사용자들 사이에서 전파 되었다 [10]. 당시 마이스페이스에서는 스크립트를 허용하지 않았다. 하지만 웹 페이지의 스타일을 지정하는 CSS(Cascading Style Sheets)를 위해서 <DIV> 태그는 허용 되어 있었다. 공격자는 이 취약점을 악용해서 악성코드를 작성 하였다. 사용자가 악성코드가 들어 있는 페이지에 방문하기만 하면 자동으로 악성코드에 감염 되어 자신의 의지와는 상관없이 자동으로 자신의 친구 목록에 있는 사용자들에게 악성코드를 전파 시켰다. 공격 시작 후 20시간 만에 피해자 수는 100만 명이 넘어갔다. 그림 1.2는 시간에 따른 감염자 수를 보여준다.

2.2.2 Koobface Worm

2009년에는 Koobface Worm이 페이스북으로 유포되었다 [18]. 페이스북 사용자에게 특정 URL이 전달된다. 수신 된 URL을 따라가면 유튜브 사이트로 위장한 웹 페이지에 연결 된다. 연결 된 웹 페이지에는 가짜 동영상 플레이어를 보여준다. 또한 동영상 플레이어에 필요한 프로그램처럼 보여주면서 악성코드 실행 파일을 다운로드하고 실행하도록 유도한다. 감염 된 사용자는 다른 사용자들에게 악성코드를 유포하게 되며 공격자의 제어를 받게 된다.

2.2.3 Bom sabado Worm

Bom sabado worm은 2010년에 온라인 소셜네트워크 서비스 중 하나인 오르컷(Orkut)을 통해서 전파 되었다 [3]. 2.2.1절에서 소개 한 새미 웜과 동일하게 스크립트를 이용해서 작성 되었다. 사용자가 페이지에 접속을 하자마자 자바 스크립트가 자동으로 실행된다. 실행 된 스크립트는 사용자를 공격자의 커뮤니티에 자동으로 가입시키도록 한다. 그리고 자동으로 사용자의 친구목

록을 찾아서 ‘Bom Sabado!’라는 메시지를 보내도록 한다. 이 메시지 내부에도 사용자를 감염 시킨 스크립트의 내용과 동일한 스크립트를 포함하도록 해서 악성코드를 전파시켰다.

2.2.4 Ramnit

2011년에 페이스북 사용자 계정 정보를 탈취하는 악성코드가 전파 되었다 [11]. Koobface와 유사하게 악성 URL을 통해서 감염 된다. 탈취한 계정을 이용해서 해당 계정에 등록되어 있는 친구들에게 동일한 악성 URL을 전송하는 방식으로 악성코드를 전파 시켰다. 2011년 9월부터 11월 사이에만 약 80만대가 감염 되었다.

2.2.5 haha photo message

2012년에 페이스북에서 ‘haha photo’라는 제목으로 메시지에 악성 URL을 포함시켜서 보내는 사례가 있었다. URL을 따라가면 ‘picture14.jpg’과 ‘www.facebook.exe’문자열이 포함 된 실행파일을 다운 받게 된다. 이 파일을 실행하게 되면 사용자의 친구들에게 자동으로 ‘haha photo’라는 제목으로 동일한 메시지를 전파한다.

제 3 장 관련 연구

본 장에서는 소셜네트워크를 이용한 공격, 공격 탐지, 방어 기법에 대해서 알아본다.

3.1 소셜네트워크를 이용한 공격 기법

Athanasopoulos 등은 온라인 소셜네트워크 사이트를 악용해서 소셜네트워크를 봇넷(Botnet)으로 만드는 기법을 제안 했다 [8]. 온라인 소셜네트워크는 이용하는 사용자 수가 최근에 급증하고 있다. 또한 사용자들끼리는 서로가 제공하는 정보를 공유하려는 경향이 있다 [9]. 이러한 점을 이용해서 공격자는 흥미를 끄는 정보로 사용자들이 많이 접근하도록 유도하였다.

온라인 소셜네트워크 사이트 중에서 페이스북에는 자체에서 제공하는 어플리케이션 개발 플랫폼이 있다 [2]. 사용자들은 페이스북 내에서 제공하는 개발 플랫폼에서 직접 어플리케이션을 개발하여서 다른 페이스북 사용자들이 사용하도록 할 수 있다. Athanasopoulos 등은 실험을 위해서 'Photo of the Day'라는 페이스북 어플리케이션을 제작하여 사용자들에게 배포 하였다. 'Photo of the Day' 어플리케이션에는 공격 코드가 내장 되어 있다.

그림 3.1은 어플리케이션 코드의 일부를 보여준다. 코드의 3번째 줄에 'victim-host'에 이미지 파일을 요청하는 코드가 있다. 소셜네트워크 상에 많은 사용자들이 'Photo of the Day' 어플리케이션을 사용하게 될 경우 해당 코드가 실행되어 'victim-host'에 많은 양의 HTTP 요청을 보내게 되어 공격이 이루어진다. 그림 3.2는 시간에 따른 HTTP 요청 규모를 보여주고 있다. 공격이 시작 후 1주일인 1월 29일에 가장 많은 HTTP 요청이 발생했음을 알 수 있다.

```
<iframe name="1" style="border: 0px none #ffffff;
width: 0px; height: 0px;"
src="http://victim-host/image1.jpg?
fb_sig_in_iframe=1&amp;
fb_sig_time=1202207816.5644&amp;
fb_sig_added=1&amp;
fb_sig_user=724370938&amp;
fb_sig_profile_update_time=1199641675&amp;
fb_sig_session_key=520dabc760f374248b&amp;
fb_sig_expires=0&amp;
fb_sig_api_key=488b6da516f28bab8a5ecc558b484cd1&amp;
fb_sig=a45628e9ad73c1212aab31eed9db500a">
</iframe><br/>
```

그림 3.1 공격 코드가 내장 되어 있는 페이스북 어플리케이션

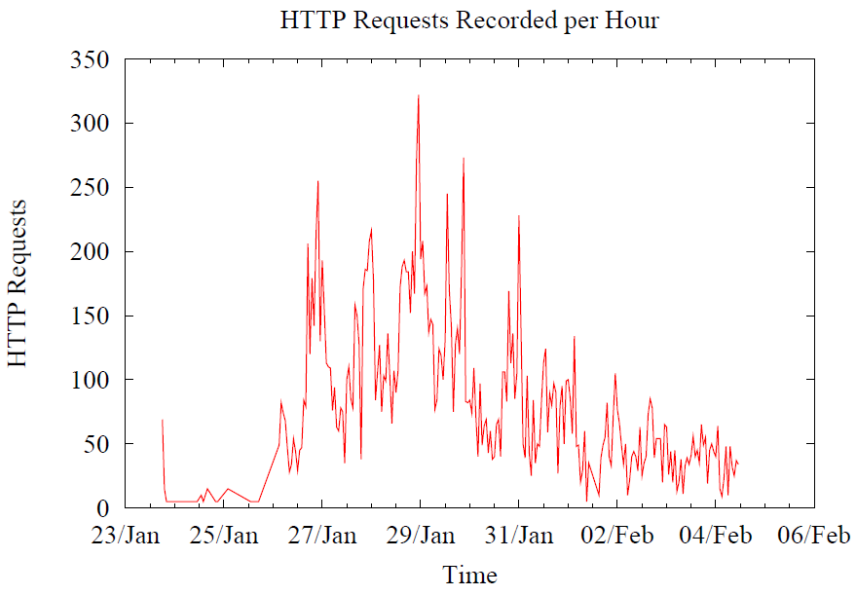


그림 3.2 'Photo of the Day' 어플리케이션의 HTTP 요청 규모

3.2 소셜네트워크를 이용한 공격 탐지 기법

Xu 등은 그래프 구조를 이용해서 소셜네트워크에서 전파되는 웜(Worm)을 빠르게 탐지 할 수 있는 기법을 제안 했다 [19]. 감지 시스템은 총 4개의 모듈로 되어 있다. 그림 3.3은 4개의 모듈을 포함 한 전체 웜 감지 시스템 구조를 보여준다.

그림 3.3 Xu 등이 제안 한 소셜네트워크에서의 웜 감지 시스템 구조

설정 모듈

설정 모듈에서는 온라인 소셜네트워크 사이트에서 통신 모듈을 통해 소셜 그래프를 얻는다. 소셜 그래프의 노드는 사용자를 나타내고 간선은 사용자들의 친구 관계를 나타낸다. 소셜 그래프를 분석해서 웜 탐지의 기반이 되는 증거를 수집 할 노드들을 선택한다. 노드를 선택하는 방법은 알고리즘 1을 따른다. 알고리즘 1을 따라서 선택 된 각 노드들끼리는 각자 노드의 반경 2-hops 안에 있는 노드들이 거의 겹치지 않는다. 선택 된 노드들은 각자 2개의 서로 다른 노드(사용자 계정)들이 친구 목록에 추가 된다. 추가되는 2개의 노드가 나타내는 사용자 계정들은 실제로 사용자가 소셜네트워크 사이트를 사용하기 위해서 만든 계정이 아니며 증거 수집을 목적으로 하는 계정이다. 추가 된 노드를 'Decoy Friend'라고 부른다.

Algorithm 1 Maximum Coverage Algorithm

Input: Graph $G = (V, E)$

Output: Monitored user set C

```
C ← φ
while |C| < K and V ≠ φ do
    maxcover ← 0
    for ∀v ∈ V do
        cover_v ←2-hops coverage of v
        if maxcover < cover_v then
            maxcover ← cover_v
        end if
    end for
    C ← C ⋃ {v}
    V ← V − {v}
    for ∀u ∈ V and (v, u) ∈ E do
        for (u, w) ∈ E do
            V ← V − {w}
        end for
        V ← V − {u}
    end for
    Update degree of each v ∈ V
end while
```

증거 수집 모듈

증거 수집 모듈에서는 온라인 소셜네트워크 사이트에서 전파되고 있는 메시지들 중에서 웜으로 의심 되는 메시지들을 수집한다. 감염된 사용자의 계정은 자신의 친구들에게 웜을 유발하는 메시지를 전파시키려고 한다. 'Decoy Friend'는 이러한 메시지들을 수집해서 웜 탐지 모듈에서 메시지의 웜 유발 가능성 여부를 판단한다.

웜 탐지 모듈

웜의 여부를 판단하기 위해서 2개의 'Decoy Friend'들이 받은 메시지의 유사도를 측정하였다. 수신 된 메시지가 웜이라고 판단되면 통신 채널을 통해서 온라인 소셜네트워크 관리자에게 웜 발생을 알려준다.

통신 모듈

통신 모듈에서는 온라인 소셜네트워크 사이트와 모듈들 사이에서 필요한 모든 통신을 지원한다. 각 모듈들은 통신 모듈을 통해서 필요한 정보들을 주고 받는다.

3.3 소셜네트워크를 이용한 방어 기법

모바일 악성코드는 주소록, 최근 통화내역, 최근 문자 수신 내역 및 발신 내역을 사용해서 MMS(Multimedia Messaging Service)로 악성코드를 전파 시킨다. Zhu 등은 그래프 구조를 이용해서 MMS를 기반으로 한 소셜네트워크에서 전파되는 악성코드를 억제하는 기법을 제안 하였다 [20]. 제안 기법은 총 4 단계로 이루어져 있으며 각 단계는 다음과 같다.

- 그래프 생성
- 그래프 분할

- 핵심 노드(Key Node) 선택
- 패치 전파

그래프 생성

MMS 통신이 가능한 사용자들 사이의 트래픽의 총합을 기반으로 해서 그래프를 생성한다. 각 사용자들이 노드가 되며 사용자들 사이의 MMS 트래픽의 총합의 상대적인 값이 간선의 가중치가 된다. 그림 3.4는 사용자들의 MMS 트래픽 량을 기반으로 생성 된 그래프의 간단한 예를 보여주고 있다.

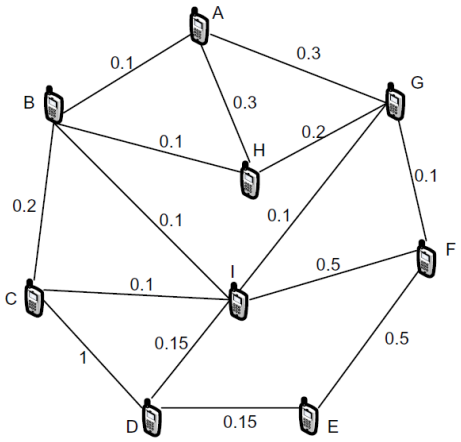


그림 3.4 사용자 간 MMS 트래픽 량을 기반으로 생성 된 그래프

그래프 분할

생성된 그래프를 노드들과 간선들의 가중치를 고려해서 분할한다. 그래프를 분할 할 때 가능하면 가중치가 높은 간선에 연결 된 노드들끼리는 같은 부분 그래프에 속하도록 한다. 그림 3.5는 그림 3.4에 있는 그래프를 분할 한 결과를 보여주고 있다. 상대적으로 가중치가 낮은 간선에 연결 된 노드들은 다른 부분 그래프에 속해있는 것을 확인 할 수 있다.

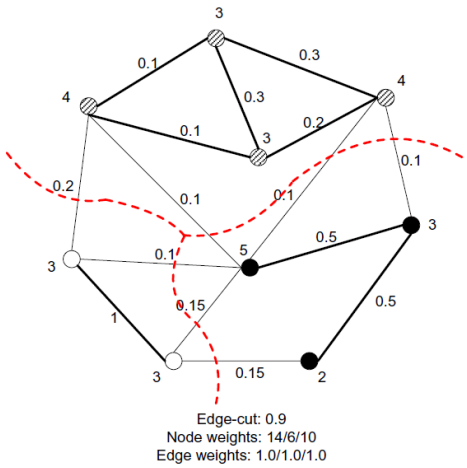


그림 3.5 그래프의 가중치를 고려해서 분할 된 그래프

핵심 노드 선택

그래프 분할이 끝나면 최초로 패치를 전달 할 핵심 노드들을 선택한다. 알고리즘 2는 핵심 노드들을 선택하는 알고리즘을 나타내고 있다. 이 때 'Cut Edge'는 부분 그래프 사이를 연결하는 간선을 가리킨다. 'Cut Edge'들은 상대적으로 가중치가 낮은 간선들이다.

Algorithm 2 Minimum Vertex Separator Algorithm

Input: E_C: the set of cut edges;

1: $V_S \leftarrow \phi$
2: **while** $E_C \neq \phi$ **do**
3: Select $v \in V'$ which is shared by the most number of cut edges in E_C
4: Add v to V_S
5: Remove from E_C any cut edge whose end point is v
6: **end while**

Output: V_S: the set of vertex separators;

패치 전파

선택 된 핵심 노드들에게 최초로 패치를 부여하고 패치를 받은 노드들은 인접한 노드들에게 패치를 전달한다. 패치를 전달 받은 나머지 노드들도 자신하고 인접한 노드들에게 패치를 전달해 준다.

장점과 단점

Zhu 등이 제안한 기법은 MMS 교류 량이 많은 사용자들끼리 같은 부분 그래프에 속하도록 해서 같은 부분 그래프 내에 있는 노드들 사이에는 패치 전파가 빠르다. 하지만 알고리즘 2에 따라서 핵심 노드들을 선택하게 되면 'Cut Edge'들에게 연결 되어 있는 노드들이 선택 된다. 'Cut Edge'들은 가중치가 상대적으로 낮기 때문에 'Cut Edge'들에게 연결 되어 있는 노드들은 주변 노드와 MMS로 통신 할 기회가 낮다는 단점이 있다.

제 4 장 소셜네트워크를 이용한 패치 전파 기법

본 장에서는 소셜네트워크를 이용해서 패치 전파하는 기법을 제안한다. 그림 4.1은 소셜네트워크를 이용한 패치 전파 하는 기법의 전체 과정을 보여준다. 소셜네트워크를 이용한 패치 전파는 크게 세 단계로 볼 수 있다. 첫 번째 단계에서는 온라인 소셜네트워크 서비스 사용자들의 교류 관계를 그래프로 표현한다. 3.2절에서 Xu 등이 제안한 기법과 동일하게 본 논문에서는 이 그래프를 소셜 그래프라고 정의한다. 두 번째 단계에서는 생성 된 소셜 그래프에서 가장 패치 전파를 빠르게 할 것으로 기대 되는 노드들을 찾아낸다. 찾아낸 각각의 노드들을 핵심 노드(Key Node)라고 정의한다. 핵심 노드를 선택하는 제안 된 두 가지 기법은 각각 페이지랭크를 이용한 기법과 유전알고리즘을 이용한 기법이 있다. 마지막 단계에서는 선택 된 핵심 노드들은 최초로 패치를 받아서 주변 노드에게 전파시킨다.

4.1 소셜 그래프 생성

온라인 소셜네트워크 서비스에서 사용자들의 교류 관계를 그래프로 표현한다. 각 노드는 온라인 소셜네트워크 서비스 사용자 계정을 나타낸다. 각 간선은 두 사용자 계정 사이의 교류 여부를 표시하며 가중치는 부여하지 않는다. 두 사용자 계정 사이로 정보 교류를 하는 경우에는 두 계정을 표현하는 노드 사이에 간선을 생성하고 그렇지 않은 경우에는 간선을 생성하지 않는다. 두 사용자 계정 사이의 교류 여부를 결정하는 방법은 다음과 같다. 계정 A와 계정 B가 서로 친구 관계를 가지고 있는 경우에는 두 계정 사이로 정보 교류가 있다고 간주한다. 그렇지 않은 경우에는 정보 교류가 없다고 간주한다. 위와 같은 과정을 모든 노드들의 쌍에 대해서 조사하면 소셜 그래프가 생성된다.

생성 된 그래프 $G = (V, E)$에서 V에 존재하는 모든 노드에게 서로 다른

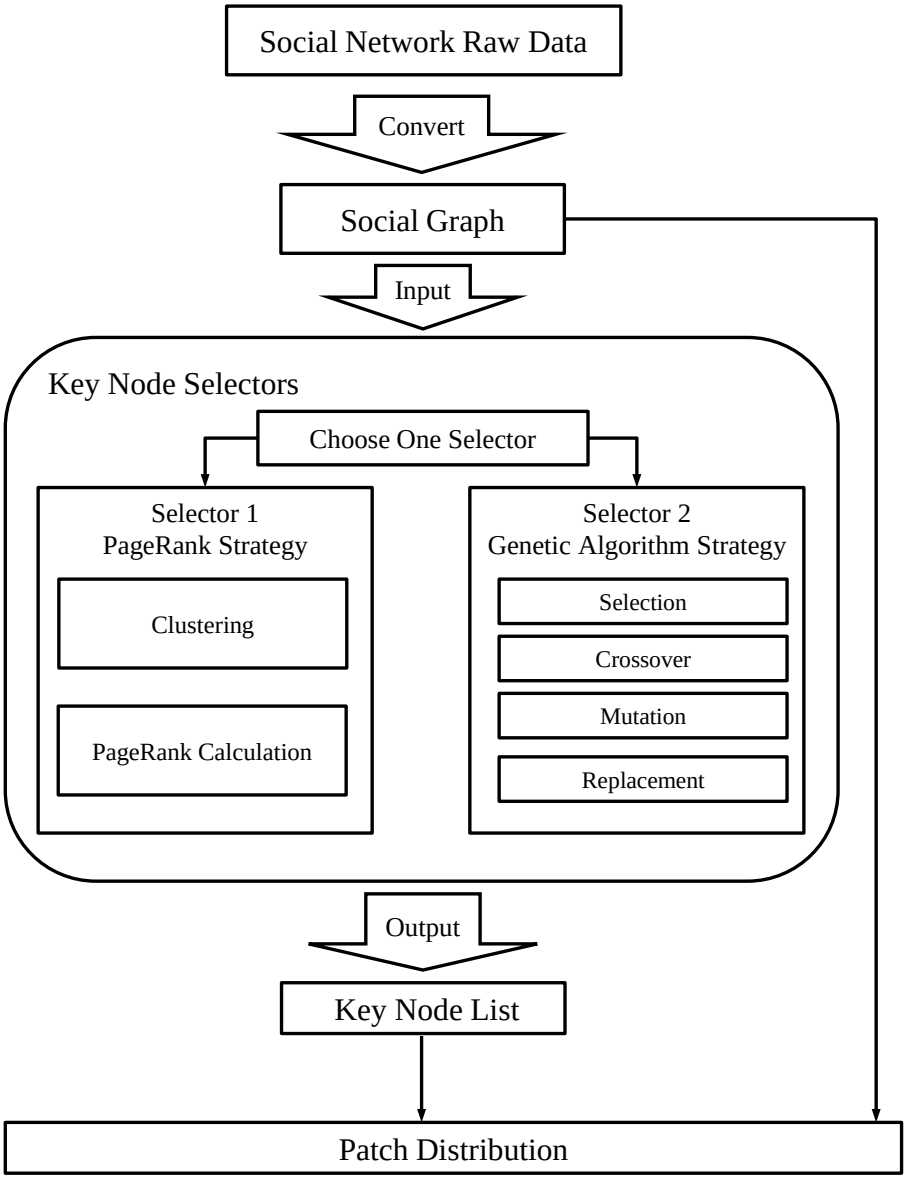


그림 4.1 소셜네트워크를 이용한 패치 전파 기법

식별 번호를 부여한다. 식별 번호는 정수로 표현하며 노드의 개수 $|V| = n$ 일 때, 노드 번호 i의 범위는 $0 \leq i \leq n-1$이 된다. 그림 4.2는 $n = 11$인 소셜 그래프의 예를 보여준다.

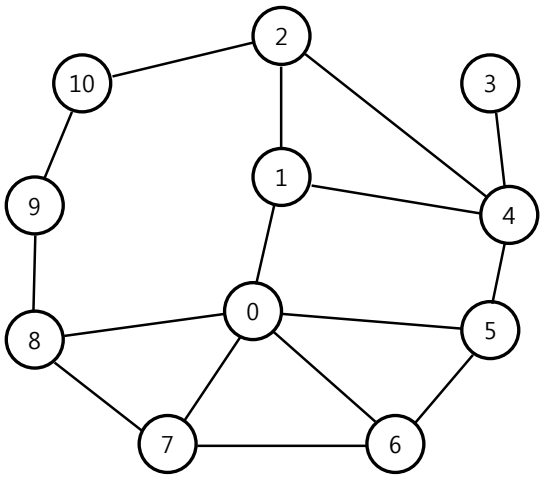


그림 4.2 $n = 11$인 소셜 그래프

4.2 핵심 노드 선택

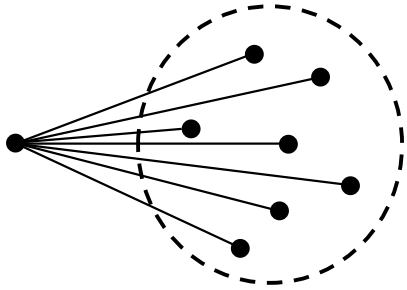
4.2.1 페이지랭크를 이용한 핵심 노드 선택

페이지랭크를 이용한 핵심 노드 선택하는 방법은 두 과정으로 나눌 수 있다. 첫 번째 과정으로 그래프 분할 과정이 진행 되고 이어서 페이지랭크 계산 과정이 진행 된다.

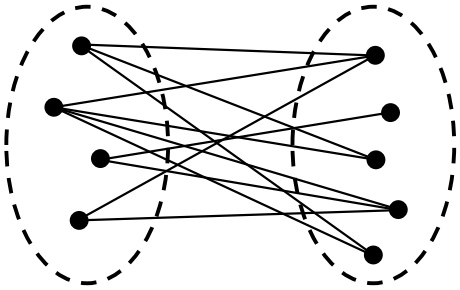
그래프 분할

그래프 분할 과정은 Zhu등이 제안한 패치 전파 기법에 있는 그래프 분할 방법을 일부 수정해서 적용하였다. 그래프 분할 과정을 설명하기 위해서, Zhu의 패치 전파 기법에서 정의 된 'Connectivity'에 대해서 다룬다 [20]. 그림 4.3은 'Connectivity'를 표현하고 있다.

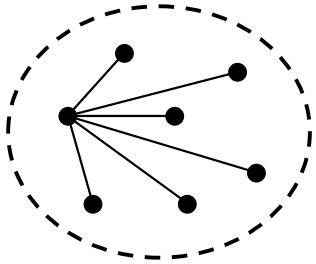
(a) 두 노드 사이의 ‘Connectivity’



(b) 하나의 노드와 하나의 부분 그래프 사이의 ‘Connectivity’



(c) 두 개의 부분 그래프 사이의 ‘Connectivity’



(d) 하나의 부분 그래프의 ‘Connectivity’

그림 4.3 Zhu등이 정의한 ‘Connectivity’

- 두 노드 사이의 'Connectivity': 노드 a와 노드 b 사이에 간선이 있는 경우, 'Connectivity' $C(a,b)$는 그 간선의 가중치가 된다.

- 하나의 노드와 하나의 부분 그래프 사이의 'Connectivity': 부분 그래프 S와 노드 a가 있고 $a \notin S$인 경우, 'Connectivity' $C(S,a)$는 $\sum_{b \in S} C(a,b)$ 이다.

- 두 개의 부분 그래프 사이의 'Connectivity': 두 부분 그래프 S_1, S_2에 대해서, 'Connectivity' $C(S_1,S_2)$는 $\sum_{a \in S_1} C(a,S_2)$가 된다.

- 하나의 부분 그래프의 'Connectivity': 부분 그래프 S와 노드 a가 있고 $a \in S$라고 하자. 그리고 부분 그래프 $S_{_a}$는 부분 그래프 S에서 노드 a를 제외시킨 부분 그래프라고 하자. 이 때 'Connectivity' $C(S)$는 $\frac{\sum_{a \in S} C(a,S_{_a})}{|S|}$ 이다.

'Connectivity'에 기반을 두어서 그래프를 분할하기 위해, 모든 간선에 대해서 0이 아닌 임의의 가중치를 부여한다. 그래프 분할은 '확장 단계', '축소 단계', '복구 단계'로 총 3단계로 이루어져 있다.

'확장 단계'에서는 임의의 간선을 선택해서 그 간선과 인접한 두 노드로 부분 그래프 S를 만든다. 그 다음, 부분 그래프 S와 인접한 노드들 중에서 부분 그래프 S와 가장 'Connectivity'가 높은 노드 x를 찾는다. 노드 x를 부분 그래프 S에 포함 시켜서 부분 그래프 S'를 만든다. $C(S') \geq C(S)$인 경우에는 S'는 갱신된 S가 되고 '확장 단계'를 다시 반복하게 된다. $C(S') < C(S)$인 경우나 인접한 노드가 없어서 S'를 만들 수 없는 경우에는 확장을 중단한다. 확장을 중단한 이후에는 남아 있는 간선들 중 임의로 하나를 선택해서 '확장 단계'를 다시 시작한다. 모든 노드가 부분 그래프에 포함 될 때 까지 '확장 단계'를 반복한다. 그래프 분할이 완성 되면 부분 그래프의 총 개수를 센다. 부분 그래프 개수를 셀 때는 하나의 노드로만 이루어진 부분 그래프는 부분 그래프 개수에 포함시키지 않는다. 부분 그래프 개수가 미리 지정 해 둔 값 m 보다 클 경우에는 '축소 단계'로 넘어가고 그렇지 않은 경우에는 '복구 단계'로 넘어간다.

'축소 단계'에서는 하나의 부분 그래프를 하나의 노드로 축소시키는 과정이다. 그래프 $G_i = (V_i, E_i)$는 i번의 '축소 단계'를 거친 직후의 그래프라고 했을 때, '축소 단계'는 그래프 $G_i = (V_i, E_i)$로 부터 축소 된 그래프 $G_{i+1} = (V_{i+1}, E_{i+1})$를 만들어 내는 과정이다. 그래프 $G_{i+1} = (V_{i+1}, E_{i+1})$에서 각 노드는 그래프 $G_i = (V_i, E_i)$에서의 각 부분 그래프를 나타낸다. 그래프 $G_{i+1} = (V_{i+1}, E_{i+1})$에 있는 노드 a와 b가 각각 그래프 $G_i = (V_i, E_i)$에 있는 부분 그래프 S_a와 S_b를 나타낸다고 하자. $C(S_a, S_b) > 0$인 경우에는 노드 a와 b사이에 가중치가 $C(S_a, S_b)$인 간선을 만든다. 그래프 $G_{i+1} = (V_{i+1}, E_{i+1})$를 모두 생성한 뒤에는 다시 '확장 단계'로 넘어간다.

'복구 단계'에서는 축소 된 노드들을 다시 원래 그래프로 복구하는 과정이다. n번의 '축소 단계'를 거친 후의 그래프를 $G_n = (V_n, E_n)$ 라고 하자. 그래프 $G_n = (V_n, E_n)$에 있는 각 노드를 원래 그래프 $G_0 = (V_0, E_0)$을 기반으로 해서 대체 시킨다. '복구 단계'를 거친 후에는 그래프 G'_n을 얻게 된다.

페이지랭크 계산

Larry의 페이지랭크 계산 방법을 사용해서 핵심 노드를 선택한다. 페이지랭크를 계산하기 전에, 부분 그래프 사이를 연결하는 모든 간선들을 제거한다. 그리고 각 부분 그래프 마다 페이지랭크를 계산한다. 페이지랭크 계산 방법은 다음과 같다.

$N_i (1 \leq i \leq n)$를 n개의 각 노드라고 하고, $N_{i,k}$를 k번의 계산 후의 노드 N_i의 점수라고 하자. 각 노드의 초기 점수 $N_{i,0} (1 \leq i \leq n)$는 동일하게 0보다 큰 임의의 값을 부여한다. D_x를 노드 N_x의 차수라고 하고 $P_{x,y}$를 노드 N_x로부터 N_y에게 전달하게 되는 점수의 비중이라고 하자. 식 4.1은 $P_{x,y}$를 정의 한다.

$$P_{x,y} = \begin{cases} if\ N_x\ is\ connected\ with\ N_y & : \ 1/D_x \\ otherwise & : \ 0 \end{cases} \tag{4.1}$$

그리고 $N_{j,k-1} (1 \leq i \leq n)$값들로 부터 $N_{i,k} (1 \leq i \leq n)$값들을 계산 하는 방법은

식 4.2와 같다.

$$N_{i,k} = \sum_{j=1}^{n} N_{j,k-1} P_{j,i} \tag{4.2}$$

식 4.2를 각 노드들의 점수가 수렴 할 때 까지 반복해서 계산한다. 수렴 조건에 사용 되는 값 e에 대해서 수렴 조건은 식 4.3과 같다.

$$E = \sum_{i=1}^{n} |N_{i,k-1} - N_{i,k}| < e \tag{4.3}$$

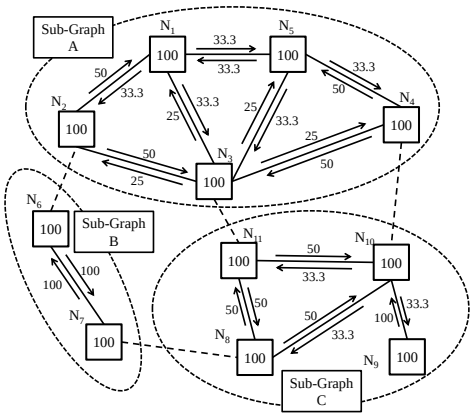
페이지랭크 계산이 모두 끝나면 각 노드마다 수렴 된 점수를 알 수 있다. 각 부분 그래프에서 수렴 된 점수가 가장 높은 하나의 노드가 핵심 노드로 선택 된다. 하나의 부분 그래프 내에서 두 개 이상의 노드가 가장 높으면서 같은 점수를 갖고 있는 경우에는 두 노드 중에서 임의의 하나의 노드를 핵심 노드로 선택한다. 그림 4.4는 페이지랭크 계산의 한 예를 보여주고 있다. 3개의 부분 그래프로 분할되어 있으며, 핵심 노드로 선택 된 노드는 N_3, N_6, N_{10}이다.

4.2.2 유전알고리즘을 이용한 핵심 노드 선택

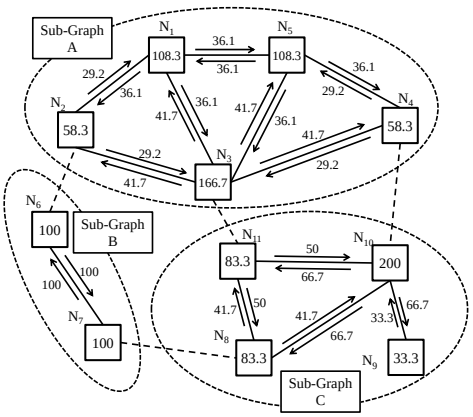
본 절에서는 유전알고리즘을 이용해서 핵심 노드를 선택하는 기법을 설명한다. 선택 할 핵심 노드들의 개수를 정해두고, 임의의 핵심 노드들로 이루어진 염색체들을 여러 개 생성한다. 생성한 염색체들을 대상으로 유전알고리즘을 적용해서 가장 품질이 좋은 염색체를 찾는다. 염색체의 품질 계산은 노드들 사이의 거리를 기반으로 계산된다.

유전알고리즘

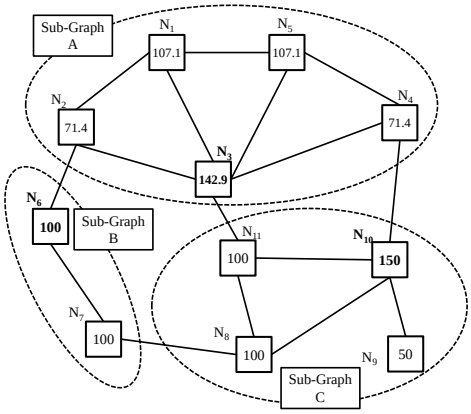
유전알고리즘(Genetic Algorithm)은 존 홀랜드(John Holland)에 의해서 1975년에 개발 된 최적화 기법이다 [12]. 생물의 진화를 모방하였으며, 선택(Selection), 교차(Crossover), 변이(Mutation), 대치(Replacement) 연산을 사용한다 [14]. 주어진 문제가 TSP(Traveling Salesman Problem) 문제처럼 계산 불가능한 수준으로 복잡한 경우 유전알고리즘을 사용하면 최적에 가까운 해를 얻을 수 있다. 그림 4.5는 유전알고리즘의 전체 구조를 보여주고 있다.



(a) 초기 상태



(b) 1회 계산 후 상태



(c) 마지막 상태

그림 4.4 페이지랭크 계산 예

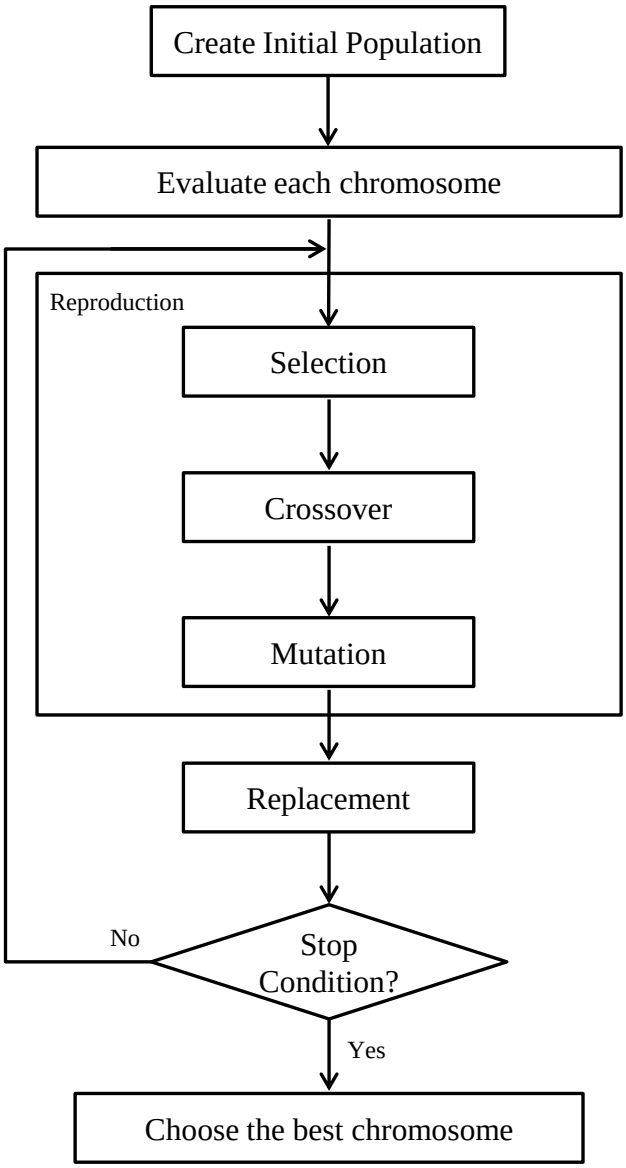


그림 4.5 유전알고리즘의 전체 구조

염색체 표현형태

찾아내려고 하는 노드들의 개수가 k일 때 각각의 염색체는 k개의 노드들의 집합이 된다. 각각의 염색체는 일차원으로 표현한다. 예를 들어서 염색체 내 노드들의 개수 $k = 5$이고 그래프 내 노드들의 개수 $n = 100$ 일 때 임의의 염색체 p는 식 4.4와 같이 표현 될 수 있다.

$$p = \{93, 2, 15, 62, 23\} \tag{4.4}$$

염색체 p에는 5개의 노드가 있으며, 각 노드의 식별 번호는 93, 2, 15, 62, 23 이다. 또한 모든 식별 번호는 100을 넘지 않는다.

품질 계산

소셜 그래프 $G = (V, E)$가 주어지고, 임의의 염색체 p에 k개의 노드가 있고, 소셜 그래프 $G = (V, E)$ 내부 모든 노드에게 $|V| = n$ 미만인 서로 다른 식별 번호 부여 되었다고 하자. 소셜 그래프 $G = (V, E)$ 내부 각 노드 v_i $(0 \leq i \leq n-1)$에 대해서 임의의 염색체 p의 품질 C_p은 식 4.5와 같이 정의 된다.

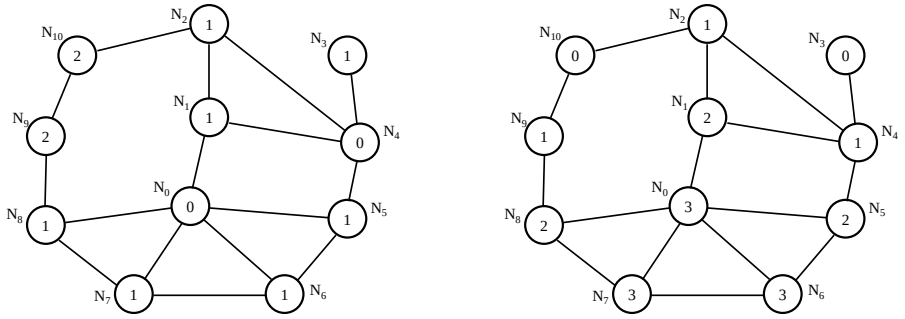
$$C_p = \sum_{i=0}^{n-1} \min_{u \in p} d(u, v_i) \tag{4.5}$$

이 때 $d(u, v_i)$는 노드 u와 v_i 사이의 최단 거리를 나타낸다. 그림 4.6은 같은 그래프에 대해서 핵심 노드에 따른 대한 품질 계산 결과를 보여주고 있다.

그림 4.6(a)의 경우는 C_p 값이 $1 \times 5 + 2 \times 2 = 9$가 된다. 그림 4.6(b)의 경우는 C_p 값이 $1 \times 3 + 2 \times 3 + 3 \times 3 = 18$이 된다.

문제 정의

소셜 그래프 $G = (V, E)$와 k개의 노드를 보유하고 있는 임의의 염색체 p에 대해서 품질 C_p를 구할 수 있다. C_p의 값이 작을수록 염색체 p의 품질이 더 좋다는 것을 의미한다. 본 논문에서 제시하는 문제는 주어진 소셜 그래프 $G = (V, E)$에서 품질 C_p값을 최소가 되는 염색체 p를 구하는 것이다.



(a) 핵심 노드가 N_0, N_4인 경우 (b) 핵심 노드가 N_3, N_{10}인 경우

그림 4.6 핵심 노드에 따른 품질 계산 결과

유전알고리즘 연산자

유전알고리즘에서는 선택, 교차, 변이, 대치 연산을 사용한다. 각 연산에는 여러 가지 연산자가 존재한다. 그 중에서 본 논문에서 제시하는 문제에 적합한 연산자를 선택한다.

선택 연산

선택 연산은 룰렛 휠 선택(Roulette Wheel Selection)을 사용한다. 룰렛 휠 선택은 선택 연산에서 가장 대표적인 방법이다. 각 해의 품질을 평가한 다음 가장 좋은 해의 적합도가 가장 나쁜 해의 적합도보다 k 배가 되도록 조절한다. 본 논문에서 제시하는 문제에서는 염색체의 품질 C_p값이 작을수록 좋은 염색체가 된다. 해 집단 내에서 C_w가 최악의 해, C_b가 최상의 해일 때, 해 i의 적합도 f_i는 식 4.6과 같이 계산 한다.

$$f_i = (C_w - C_i) + (C_w - C_b)/(k-1), k > 1 \tag{4.6}$$

일반적으로 k값의 범위는 $3 \leq k \leq 4$ 이다. k값을 높이면 선택압(Selection Pressure) 이 높아지며 다양성을 떨어뜨리는 결과가 발생하게 된다. 알고리즘 3은 룰렛 휠 선택을 기술하고 있다.

Algorithm 3 룰렛 휠 선택

```
point ← random(0, SumOfFitness)
sum ← 0
for i := 0 → n − 1 do
    sum ← sum + f_i
    if point < sum then
        return i
    end if
end for
```

교차 연산

교차 연산은 일점 교차(One-Point Crossover)또는 다점 교차(Multi-Point Crossover)를 사용한다. 일점 교차는 일차원으로 이루어진 염색체에서 수행되는 대표적인 교차 연산이다. 길이가 같은 두 염색체를 임의의 교차점에서 분할하고 서로 다른 염색체로부터 분할 된 염색체를 재조합한다. 그림 4.7은 일점 교차의 예를 보여준다. 길이가 n인 일차원 염색체의 경우 일점 교차로 분할하는 방법은 $n-1$가지가 된다.

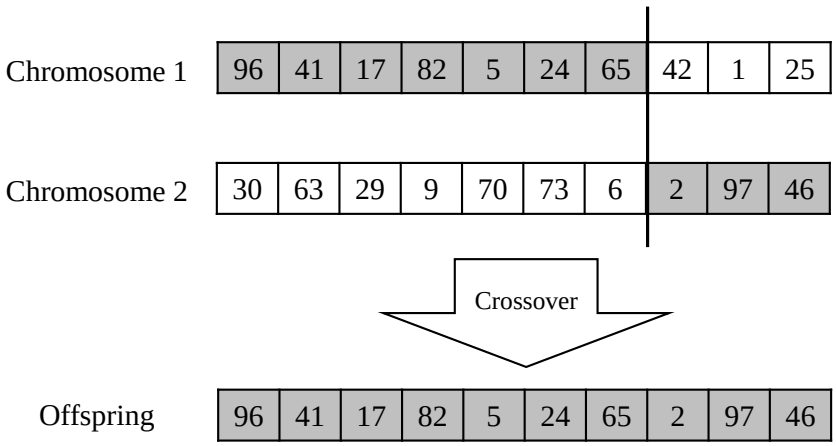


그림 4.7 일점 교차 연산

다점 교차는 일점 교차와 유사하나 염색체 분할점이 여러 개인 교차 방법이다. 그림 4.8은 교차점이 2개인 다점 교차의 예를 보여준다. 길이가 n인 일차원 염색체이면서 교차점이 k개인 경우 염색체 분할 방법은 총 $\binom{n-1}{k}$가지가 된다.

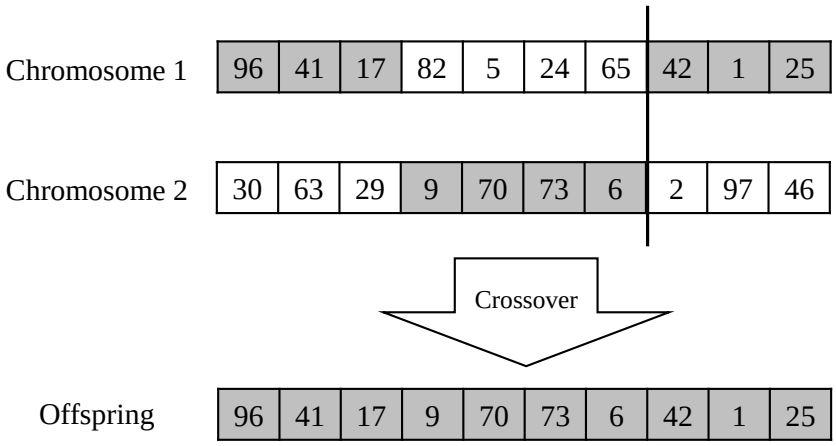


그림 4.8 다점 교차 연산 (교차점이 2개인 경우)

변이 연산

변이 연산은 부모 해에 없는 속성을 주입해서 해 공간을 더 넓게 탐색하려는 목적을 가지고 있다. 난수를 생성하여 일정 임계값 미만의 수가 나올 경우에 해당 유전자를 임의로 변형 시킨다. 그림 4.9는 변이 연산의 예를 보여준다. 진화 초반에는 변이 연산의 결과로 더 좋은 해가 나올 수도 있다. 반면에 진화 후반에는 이미 좋은 해들로 구성 되어 있기 때문에 더 좋은 해가 나올 가능성이 낮다. 품질이 더 나쁘게 나온 해는 진화를 거치면서 사라지게 되며, 품질이 좋게 나온 해는 전체 해 집단의 품질 향상에 기여한다.

교차와 변이연산을 거친 해가 문제 해결에 적합하지 않은 경우가 있다. 본 논문에서 제시하는 문제에서는 해 안에 있는 노드 번호들이 중복되어서는 안된다. 하지만 교차와 변이 연산을 거칠 경우 중복되는 경우가 발생 할 수도 있다. 그림 4.10은 교차연산으로 인해 노드 번호가 중복되는 경우와, 그것을 수선하

는 과정을 나타낸다. 두 염색체를 교차 연산한 결과 '24'가 중복 된다. 이렇게 될 경우 핵심 노드의 개수가 하나 줄어들게 된다. 따라서 두 개의 '24' 중에서 하나를 임의의 숫자로 변경한다. 변경할 때 기존에 있는 번호와 중복되지 않는 번호를 선택한다.

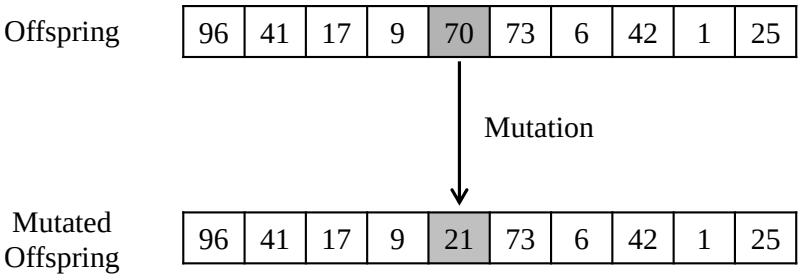


그림 4.9 변이 연산

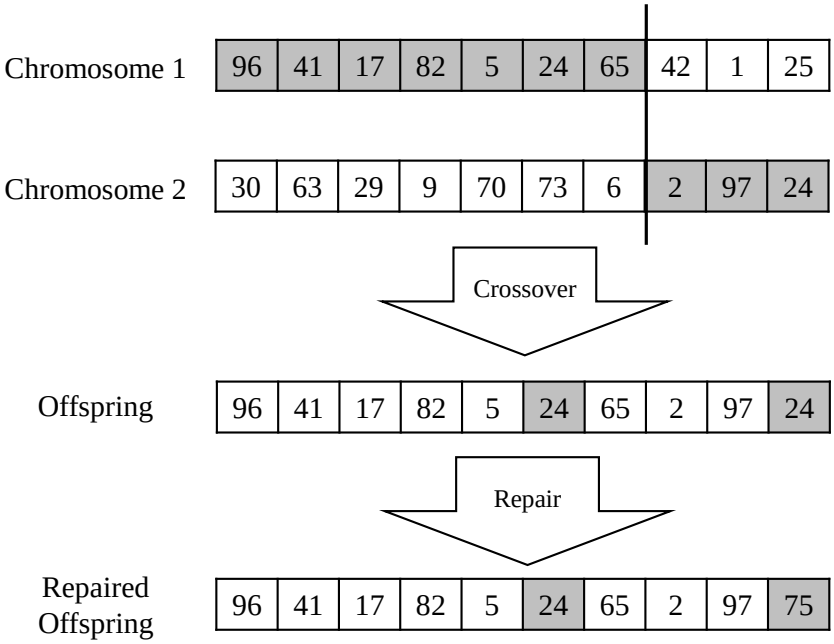


그림 4.10 수선

대치 연산

본 논문에서는 안정 상태 유전알고리즘을 수행한다. 안정 상태 유전알고리즘의 경우에는 하나 또는 소수의 해를 해 집단 내에 있는 해들과 대치하는 방식이다. 사용 가능한 대치 연산자는 다음과 같다.

첫 번째로는 해 집단 내에서 가장 품질이 낮은 해를 대치하는 것이다. 이 방법은 강한 수렴성을 가지며 품질이 좋은 해는 항상 살아남게 된다. 대신 수렴성이 강하기 때문에 해 공간을 넓게 탐색하지 못할 수도 있다는 단점이 있다.

두 번째로는 두 부모 해 중에서 나쁜 해와 대치하는 방법이 있다. 이 방법은 자신과 가장 닮은 두 부모 해 중에서 대치하기 때문에 해 집단의 다양성을 오래 유지 할 수 있다는 장점이 있다. 또한 두 부모해 보다 품질이 나쁜 경우에는 대치를 포기 할 수도 있다. 이 방법은 다양성을 더욱 오래 유지 할 수 있지만 수렴성이 떨어지기 때문에 충분한 시간이 필요하다.

4.3 패치 전파

패치 전파 단계에서는 1차로 모든 핵심 노드들에게 패치를 전달한다. 패치를 받은 노드들은 자신과 인접한 모든 노드들에게 패치를 전달한다. 나머지 노드들도 패치를 받게 되면 자신과 인접한 모든 노드들에게 패치를 전달한다. 다른 노드를 통해서라도 최소 하나의 핵심 노드와 연결 되어 있는 노드들은 패치를 전달 받게 된다.

제 5 장 시뮬레이션 결과 및 분석

본 장에서는 제안 된 페이지랭크를 이용한 패치 전파 기법과 유전알고리즘을 이용한 패치 전파 기법을 시뮬레이션 한 결과를 살펴본다. 제안 된 두 기법들을 성능 관점에서 비교 분석한다.

5.1 시뮬레이션 환경

본 논문의 시뮬레이션은 PC에서 진행 되었다. 표 5.1은 시뮬레이션에 사용된 하드웨어와 운영체제 환경을 나타낸다.

프로세서	Inter(R) Core(TM) i5-2500 CPU @ 3.30GHz
메모리	Samsung DDR3 8GB PC3-12800
운영체제	Ubuntu 12.10
커널버전	3.5.0-43

표 5.1 시뮬레이션에 사용 된 하드웨어와 운영체제

시뮬레이션 코드는 그래프 분석을 위해 C++, STL(Standard Template Library), Boost Graph Library(BGL)을 사용해서 작성 되었다 [17].

5.2 소셜네트워크에서의 패치 전파 모델

본 논문에서는 신뢰성 있는 결과를 도출하기 위해서 사용자들의 온라인 소셜네트워크 사용 패턴을 조사하였다. 온라인 소셜네트워크 사용자들은 실시간으로 사이트에 접속해서 정보를 받지 않는 경우가 대부분이다. 사용자가 자신의 계정에 정보(악성코드 또는 패치)를 수신하고 난 후 사용자가 그 정보를 다른 사용자에게 전달하는데 걸리는 시간 분포에 대한 모델이 필요하다. Nazir

등은 페이스북 사용자들에 대해서 정보에 대해서 응답하는데 걸리는 시간 분포를 도출하였다 [16]. 그림 5.1은 그 분포를 나타낸다.

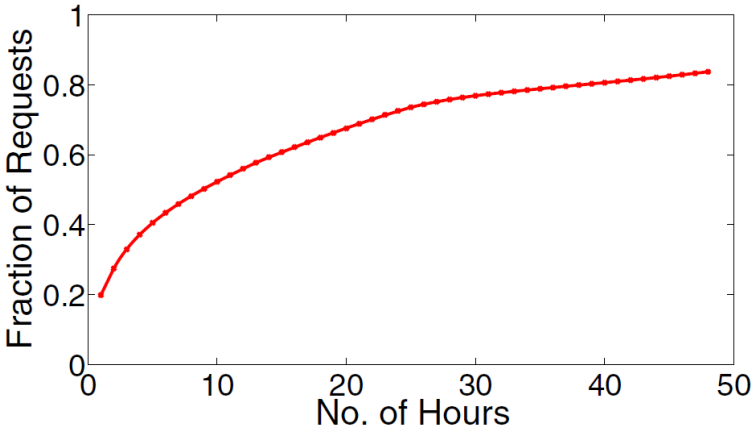


그림 5.1 정보를 다른 사용자에게 전달하는데 소요되는 시간 분포

위 분포를 모델링해서 유사한 분포를 도출한 결과 식 5.1과 유사한 분포가 나왔다.

$$y = (\ln x + 0.6732)/5.4253 \tag{5.1}$$

확률에 따른 응답 소요시간을 예측하기 위해서 식 5.1로 부터 역함수를 구했다.

$$x = e^{5.4253y - 0.6732} \tag{5.2}$$

식 5.2는 식 5.1에 대한 역함수를 나타낸다. 소셜네트워크 사용자들의 사용패턴을 적용하기 위해서 시뮬레이션에 식 5.2를 적용하였다.

5.3 시뮬레이션 결과 분석

시뮬레이션을 총 3개의 소셜네트워크 데이터 셋(Data Set)에 대해서 진행하였다. 표 5.2는 시뮬레이션에 사용 된 소셜네트워크 그래프 데이터의 규모를 나타낸다 [6].

온라인 소셜네트워크	노드(사용자) 수	간선 수
페이스북(Facebook)	4,039	88,234
트위터(Twitter)	81,306	1,768,149
구글 플러스(Google+)	107,614	13,673,453

표 5.2 시뮬레이션에 사용 된 소셜네트워크 그래프 데이터

시뮬레이션을 수행 할 때 적용한 핵심 노드 선택 방법은 제안 기법을 포함하여 다음과 같이 총 4가지이다.

- 무작위로 선택하는 방법
- 노드의 차수가 가장 높은 순서대로 선택하는 방법
- 제안 된 페이지랭크 기법을 이용한 방법
- 제안 된 유전알고리즘 기법을 이용한 방법

페이지랭크를 이용한 방법에서는 그래프 분할 과정에서 부분 그래프의 개수가 정해지기 때문에 핵심 노드의 개수를 임의로 정할 수 없다. 객관적인 비교를 위해서 나머지 3가지 방법에서 핵심 노드들을 선택 할 때 그 개수는 페이지랭크를 이용해서 나오는 핵심 노드들의 개수와 동일하게 하였다.

핵심 노드들을 무작위로 선택하는 방법의 경우에는 매번 다른 해가 나오게 된다. 유전알고리즘을 선택하는 경우에도 간혹 다른 해가 나오는 경우가 있다. 또한 사용자의 응답 소요시간도 확률이 적용되기 때문에 결과가 다르게 나온다. 따라서 시뮬레이션을 100회 수행하여 그 평균을 계산하였다.

5.3.1 시간에 따른 패치 전파율

본 절에서는 패치가 전파된 노드들의 비율을 시간에 따라서 관측하였다. 또한 전파된 노드들의 비율을 핵심 노드들을 선택하는 4가지 방법에 대해서 서로 비교하였다. 그림 5.2, 5.3, 5.4는 그 결과를 보여준다.

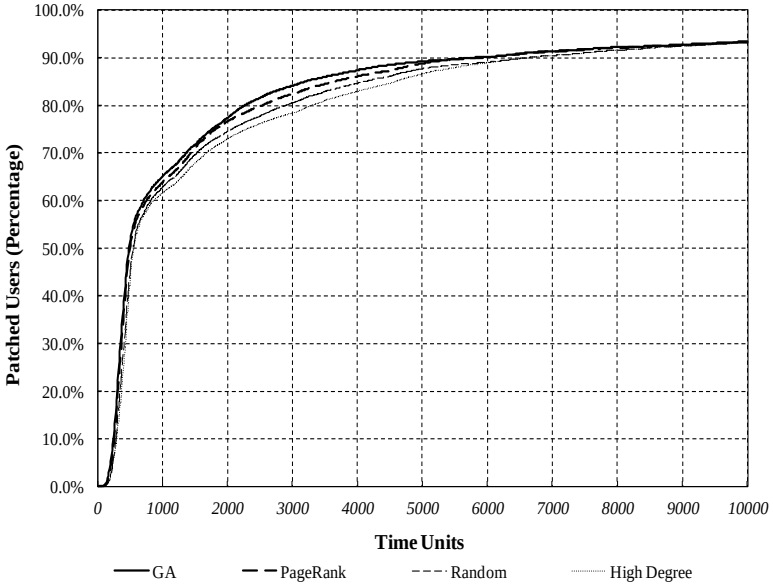


그림 5.2 페이스북에서의 패치 전파

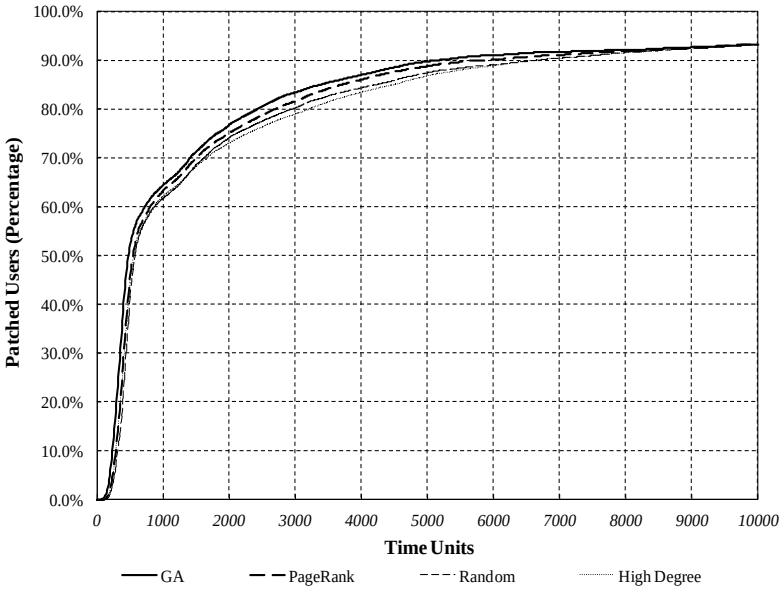


그림 5.3 트위터에서의 패치 전파

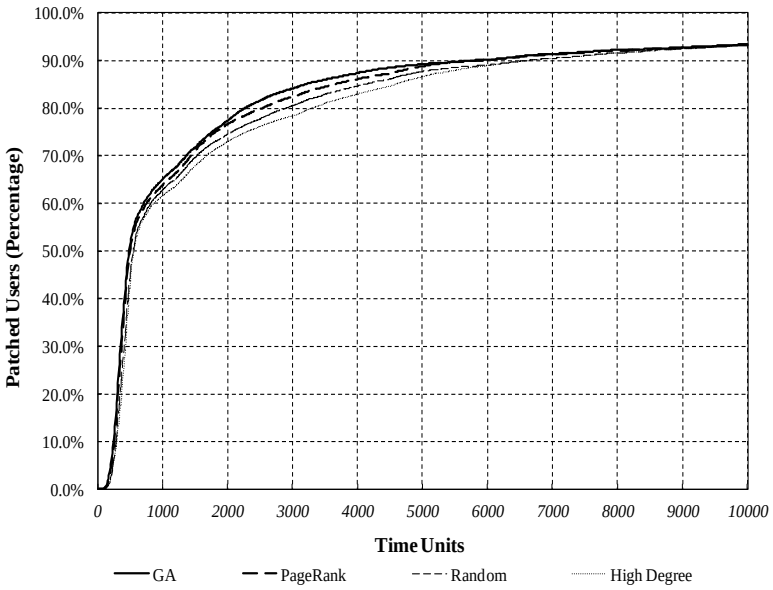


그림 5.4 구글 플러스에서의 패치 전파

패치전파 속도는 네 가지 방법 모두 초반에 빠르게 전파 되며 약 65%가 넘는 시점부터는 전파 속도가 급격하게 감소하였다. 또한 네 가지 방법 사이의 패치 전파 속도 차이가 크지 않았다. 하지만 전체 노드의 약 70%부터 약 90%가 패치를 수신 하는 동안에는 그 격차가 다소 드러났음을 확인 하였다. 유전 알고리즘이 가장 빨랐으며 그 이후로 페이지랭크, 무작위 선택, 높은 차수를 선택하는 방식 순으로 빠르게 패치가 전파 되었다.

5.3.2 시간에 따른 악성코드 감염률

본 절에서는 악성코드에 감염 된 노드들의 비율을 시간에 따라서 관측하였다. 악성코드의 종류와 패치의 종류는 각각 한 가지만 있다고 가정 하였다. 또한 최초로 악성코드에 감염 된 노드들의 개수는 전체의 0.02%로 하였으며 전체의 0.4%의 노드가 감염 되었을 때 패치 전파를 시작 하였다. 악성코드에 감염 된 노드들의 비율을 핵심 노드들을 선택하는 4가지 방법에 대해서 서로 비교하였다. 그림 5.5, 5.6, 5.7은 그 결과를 보여준다.

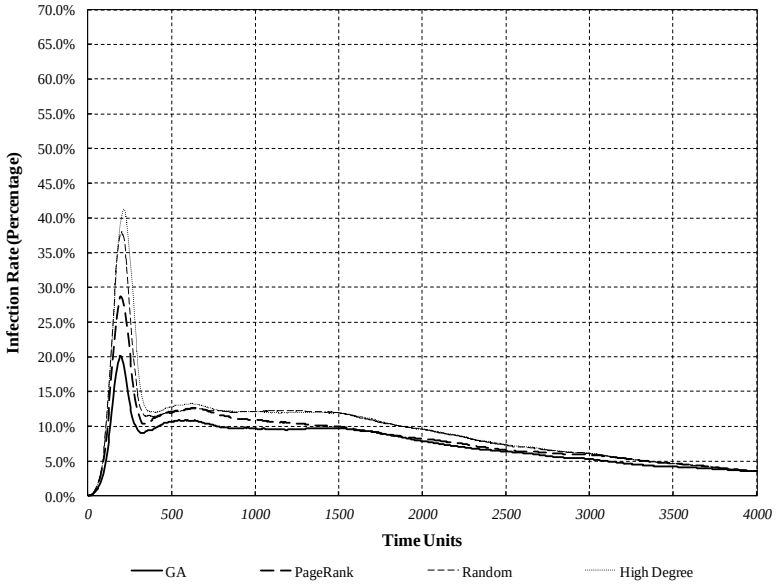


그림 5.5 페이스북에서의 악성코드 감염률

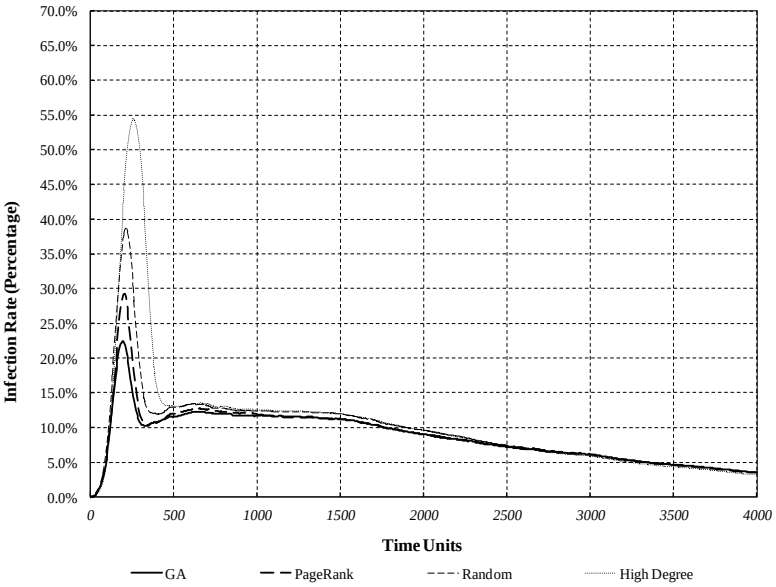


그림 5.6 트위터에서의 악성코드 감염률

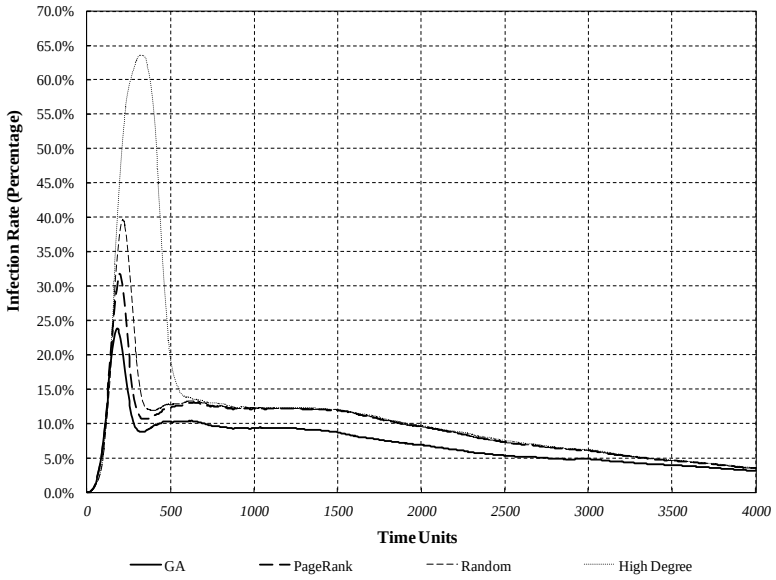


그림 5.7 구글플러스에서의 악성코드 감염률

악성코드 전파가 패치 전파보다 먼저 시작하기 때문에 초반에 악성코드 감염률이 빠르게 상승한다. 하지만 패치가 전파되는 시점부터 악성코드 감염률이 급격하게 감소했음을 확인 할 수 있다. 또한 악성코드 감염률이 1차로 대폭 감소 한 뒤에 아주 잠시 그 감염률이 증가하는 형태를 관찰 할 수 있었다. (그림 5.5의 경우 약 350 Time Unit 지점) 그 이후에는 악성코드 감염률이 일정 기간 동안 유지되다가 서서히 감소하였다. 네 가지 방법 모두 비슷한 형태를 가졌지만 최대 감염률에서는 많은 차이를 보였다. 모든 그래프에 대해서 유전알고리즘을 적용한 경우 최대 감염률이 가장 적었다. 그 다음으로는 페이지랭크를 적용한 경우가 최대 감염률이 가장 적었다. 하지만 무작위로 핵심 노드들을 선택한 경우와 차수가 높은 노드들을 고른 경우에는 최대 감염률이 높았다. 특히 높은 차수를 고른 경우에는 그래프에 중심 부분에 집중해서 핵심 노드들이 선택되었기 때문에 효과적인 패치 전파가 어려웠다. 그래프의 규모가 커질수록 네 가지 방법에 대한 최대 감염률의 차이는 더 증가 하였다.

5.3.3 소셜 그래프의 변화에 따른 최대 피해율

본 절에서는 유전알고리즘으로 핵심 노드들을 선택한 기법에 대해서 소셜 그래프의 변화에 따른 최대 피해율을 관측하였다. 신뢰성 있는 결과를 도출하기 위해서 Kumar등이 제시한 온라인 소셜네트워크의 진화 모델을 사용 하였다. Kumar등이 제시한 모델에서는 온라인 소셜네트워크 사용자를 3가지로 분류한다 [13].

- Passive users: 다른 사용자의 권유나 흥미에 의해서 온라인 네트워크에 참여하였으나 다른 사용자들과의 교류에 소극적이다.
- Inviters: 온라인 소셜네트워크에 기존에 없던 새로운 사용자들을 적극적으로 참여시킨다.
- Linkers: 이미 온라인 소셜네트워크에 속해 있는 사용자들과 적극적으로 연결한다.

Kumar등이 제시한 모델에서 표현되는 소셜 그래프 $G = (V, E)$에서도 각 노드는 온라인 소셜네트워크 사용자를 나타내며 각 간선은 사용자들 사이의 친구관계 여부를 타나낸다. 또한 'Passive user'들의 집합 P, Inviter들의 집합 I, Linker들의 집합 V에 대해서 $V = P \cup I \cup L$을 만족한다. 각 단위 시간 마다 소셜 그래프 내 하나의 새로운 노드와 ε개의 새로운 간선들이 생성 된다고 모델링하였다. 새로 생성 된 노드의 유형은 확률 $p = (P, I, L)$에 의해서 결정 된다. 새로 생성 된 각 간선 (u, v)에 대해서 두 노드 u, v를 결정한다. 노드 u의 차수를 $d(u)$라고 하고 매개 변수 β가 있을 때, 식 5.3은 두 노드를 선택하는 확률 분포를 나타낸다.

$$D^{\beta}(u) \propto \begin{cases} \beta \cdot (d(u)+1) & u \in L \\ d(u)+1 & u \in I \\ 0 & otherwise \end{cases} \tag{5.3}$$

노드 u는 $D^1(u)$로 결정한다. $u \in I$인 경우에는 노드 v는 새로 생성 된 노드가 된다. $u \in L$인 경우에는 노드 v는 $D^{\gamma}(u)$로 결정한다. 이 때, 매개변수 γ는 그

래프 내 가장 큰 연결 요소(Connected Component)에 대한 선호도를 나타내며 값이 클수록 선호도가 높다.

본 논문의 시뮬레이션에 사용 된 데이터 셋 중 구글플러스의 사용자 유형별 분포는 $P : I : L = 0.13 : 0.32 : 0.55$이다. 이 분포는 Kumar등이 분석하고 모델링한 온라인 소셜네트워크 플리커의 분포와 다소 유사하였다. 따라서 본 시뮬레이션에서는 Kumar등이 플리커를 모델링할 때 사용 된 매개 변수 γ, ε 값을 반영하여 $p = (0.13, 0.32, 0.55)$, $\gamma = 15$, $\varepsilon = 6$ 으로 설정하였다 최초로 악성코드에 감염 된 노드들의 개수는 전체의 0.02%로 하였으며 전체의 0.4%의 노드가 감염 되었을 때 패치 전파를 시작 하였다. 전체 노드 대비 추가 된 노드의 비율을 각각 0.1%, 0.2%, 0.4%, 0.8%, 1.6%로 변경 시키면서 시뮬레이션을 수행했다. 또한 패치에 사용 된 핵심 노드들은 그래프에 노드가 추가되기 전에 선택 된 핵심 노드들을 사용하였다. 그림 5.8은 그 결과를 보여준다.

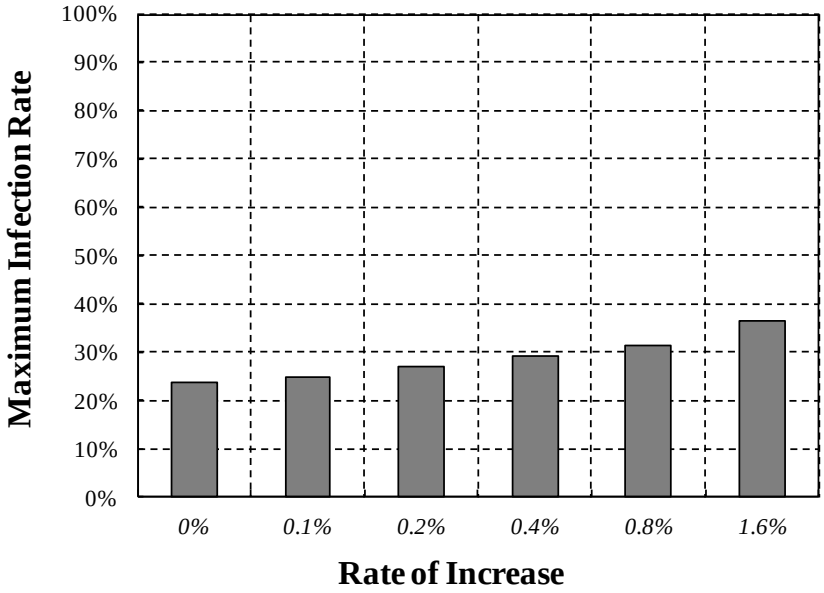


그림 5.8 소셜 그래프 변화에 따른 최대 감염률

노드가 많이 추가 될수록 최대 감염률이 증가하였지만 그 증가폭이 크지 않았다. 소셜 그래프가 변경되어도 기본적인 형태를 많이 벗어나지 않으며 그

래프 내 간선이 추가 되어 연결성이 높아지기 때문에 성능저하가 크지 않음을 확인 할 수 있었다.

5.3.4 악성코드 감지 시간에 따른 최대 피해율

본 절에서는 유전알고리즘으로 핵심 노드들을 선택한 기법에 대해서 악성코드 감지 시간에 따른 최대 감염률을 관측하였다. 최초로 악성코드에 감염 된 노드들의 개수는 전체의 0.02%로 하였다. 악성코드에 감염 된 노드가 전체 노드 중 DT%(Detection Threshold)이상이 되는 순간 패치 전파를 시작하도록 하였다. DT값을 0.1, 0.2, 0.4, 0.8, 1.6으로 변경 시키면서 시뮬레이션을 수행했다. 그림 5.9는 그 결과를 보여준다.

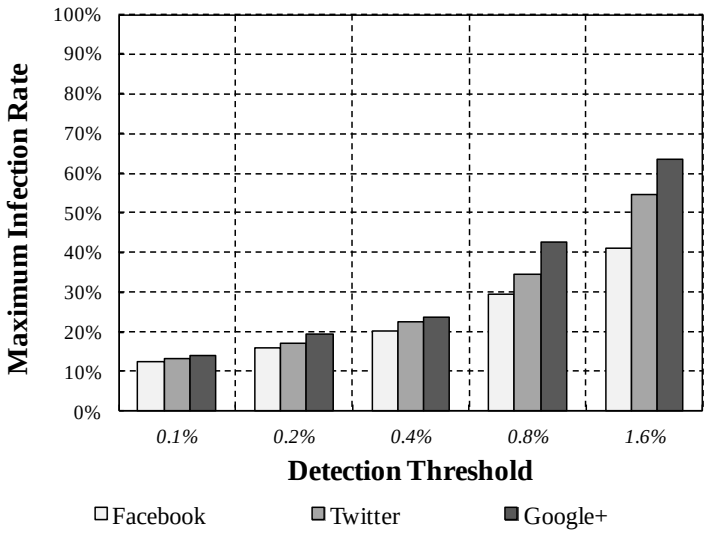


그림 5.9 DT(Detection Threshold)에 따른 최대 감염률

패치 전파를 늦게 시작 할수록 악성코드에 의한 피해가 증가 하였다. 또한 규모가 큰 그래프일수록 그 비율이 높게 증가하였음을 확인하였다.

제 6 장 결론

본 논문에서는 악성코드에 의한 피해를 최소화시키기 위해서 빠른 패치 전파 기법을 제안 하였다. 제안 된 기법은 소셜네트워크를 그래프로 만들고 핵심 노드들을 선택해서 핵심 노드들에게 1차로 패치를 준 뒤 전파시키는 기법이다. 핵심 노드들을 선택하는 방법으로 페이지랭크를 선택하는 방법과 유전알고리즘을 사용하는 방법을 제안하였다. 페이지랭크를 이용하는 방법에서는 그래프를 노드들 사이의 연결 상태를 고려해서 같은 부분 그래프 내 노드들 사이의 연결성이 좋도록 분할하였다. 분할 된 각 부분그래프마다 페이지랭크 계산을 수행하고 수렴 된 점수가 가장 높은 노드를 핵심 노드로 선택해서 패치 전파 속도를 높였다. 또한 유전알고리즘을 이용하는 방법에서는 핵심 노드들의 집합을 하나의 염색체로 정의하였다. 염색체의 품질은 염색체 내 핵심 노드들과 그래프 내 모든 노드들 사이의 거리로 정의하였다. 정의 된 염색체와 품질에 대해서 유전알고리즘의 선택, 교차, 변이, 대치 연산을 반복해서 최적에 가까운 해를 탐색 하였다.

세 가지의 소셜네트워크 데이터 셋을 대상으로 시뮬레이션 한 결과, 무작위로 핵심 노드들을 고르거나 단순히 차수가 높은 노드들을 핵심 노드들로 고르는 방법에 비해서 페이지랭크나 유전알고리즘을 사용해서 핵심 노드들을 선택하는 방법에서 더 빠르게 패치를 전파 시켰다. 그리고 유전알고리즘을 사용했을 때 페이지랭크를 사용 했을 때 보다 더 빠른 패치를 전파 시킬 수 있는 핵심 노드들을 선택하였음을 확인 하였다. 특히 악성코드에 의한 최대 피해율 관점으로 봤을 때 그 효과가 더욱 커졌음을 확인 하였으며 시간이 지남에 따라 소셜 그래프에 노드가 추가 되더라도 성능저하가 거의 발생하지 않았음을 확인하였다.

다양한 소셜네트워크 서비스가 생기고 그 사용자가 급증하고 있다. 또한 빠

른 전파력을 갖는다는 특징을 악용해서 악성코드를 전파 시키는 시점에서 본 논문에서 제안하는 기법은 여러 가지 소셜네트워크에서 발생하는 악성코드에 의한 피해를 최소화 하는 데에 큰 기여를 할 것으로 예상 된다.

참고문헌

[1] Facebook. https://www.facebook.com/.

[2] Facebook developers. https://developers.facebook.com/.

[3] Mau sabado for orkut users. http://www.symantec.com/connect/blogs/mau-sabado-orkut-users/.

[4] Owasp. https://www.owasp.org/.

[5] Social media marketing, statistics & monitoring tools. http://www.socialbakers.com/.

[6] Stanford large network dataset collection. http://snap.stanford.edu/data/.

[7] Twitter. https://twitter.com/.

[8] E. Athanasopoulos, A. Makridakis, S. Antonatos, D. Antoniades, S. Ioannidis, K. G. Anagnostakis, and E. P. Markatos. Antisocial networks: Turning a social network into a botnet. In *Information Security*, pages 146–160. Springer, 2008.

[9] N. B. Ellison et al. Social network sites: Definition, history, and scholarship. *Journal of Computer-Mediated Communication*, 13(1):210–230, 2007.

[10] M. R. Faghani and H. Saidi. Social networks' xss worms. In *Computational Science and Engineering, 2009. CSE'09. International Conference on*, volume 4, pages 1137–1141. IEEE, 2009.

[11] L. Garber. Security, privacy, and policy roundup. *IEEE Security & Privacy*, pages 15–17, 2012.

[12] J. H. Holland. *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence.* U Michigan Press, 1975.

[13] R. Kumar, J. Novak, and A. Tomkins. Structure and evolution of online social networks. In *Link Mining: Models, Algorithms, and Applications*, pages 337–357. Springer, 2010.

[14] M. Melanie. An introduction to genetic algorithms. *Cambridge, Massachusetts London, England, Fifth printing*, 3, 1999.

[15] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee. Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 29–42. ACM, 2007.

[16] A. Nazir, S. Raza, and C.-N. Chuah. Unveiling facebook: a measurement study of social network based applications. In *Proceedings of the 8th ACM SIGCOMM conference on Internet measurement*, pages 43–56. ACM, 2008.

[17] J. G. Siek, L.-Q. Lee, and A. Lumsdaine. *Boost Graph Library: User Guide and Reference Manual, The.* Pearson Education, 2001.

[18] K. Thomas and D. M. Nicol. The koobface botnet and the rise of social malware. In *Malicious and Unwanted Software (MALWARE), 2010 5th International Conference on*, pages 63–70. IEEE, 2010.

[19] W. Xu, F. Zhang, and S. Zhu. Toward worm detection in online social networks. In *Proceedings of the 26th Annual Computer Security Applications Conference*, pages 11–20. ACM, 2010.

[20] Z. Zhu, G. Cao, S. Zhu, S. Ranjan, and A. Nucci. A social network based patching scheme for worm containment in cellular networks. In *Handbook of Optimization in Complex Networks*, pages 505–533. Springer, 2012.

Abstract

Fast Patch Distribution Strategy Using Social Network and Genetic Algorithm

Donghwa Kang
Department of Computer Science and Engineering
The Graduate School
Seoul National University

As the number of online social network service users continues to increase, malicious users are increasingly able to attack against the service. Online social network users tend to connect to each other so any interesting information in the network is propagated in a short time. If a malware with any interesting information is exposed in the online social network service, it can cause serious damage. It is also difficult to prevent new types of malwares. Even though the patch against a malware is distributed in a short time, the beginning of patch distribution always comes after the malware propagates. Thus, we need a fast patch distribution strategy to stop malwares.

In this thesis, we propose two patch distribution strategies. One is based on a social network with PageRank and the other is based on a social network with genetic algorithm. Then we compare and analyze the strategies in terms of performance. Both strategies are illustrated in a graph that represents communications between online social network users. In the strategy with PageRank, the PageRank calculation is ran for each sub-graph and one user who got the

highest score in each sub-graph is selected. In the strategy with genetic algorithm, genetic algorithm operators are implemented repeatedly and a set of users is selected as the best solution. The patch is then distributed initially to all selected users. When the users receive the patch, they forward the patch to all users who communicate with them. Also, when others users receive the patch, they also forward the patch to all users who communicate with them.

In order to verify the patch distribution strategy, we implemented the strategy and performed simulations. We also modeled information propagation between social network users. Simulation results showed that our strategies allowed lower infection rate than random selection and high degree selection. Also, the genetic algorithm strategy distributed the patch faster than the PageRank strategy.

Keywords: Social Network, Graph, Genetic Algorithm, Patch Distribution, Key Nodes, Security

Student Number: 2012-20725