

網路秘密通訊機制

彭垂業 許志行 楊武
國立交通大學 資訊科學系

中文摘要

現今的網路環境最大的缺失就是安全性不足，因此近年來成為熱門的研究方向。而大部分改善網路安全性的問題，是使用 Secret Key 對 Packet 內部 data 的部份做加密。此方法可以讓第三者無法得知 packet 的內容，但是卻可經由 IP Header 得知某兩端正在通訊，沒有隱密性。因此希望網路環境不但有安全性也要有隱密性。我們使用 FreeBSD 開放原始碼的特性，將現有的 TCP/IP 做一些修改，並不需要修改現有的應用程式，來達成網路的隱密性。所謂的隱密性是利用混淆的方式，透過其他的電腦來轉送 packet，讓第三者很難去判斷 packet 真正的目的地。最後的實作結果，此系統可以正常運作，效能也有不錯的表現。

關鍵詞：網路、秘密通訊

1. 介紹

1.1 研究動機

近年來網際網路最大的問題就是安全性太薄弱，對於現今的電子交易，會在網際網路上傳送重要且隱私的個人資料，因此網際網路的安全性是越來越重要。而絕大多數都是使用密碼學來解決安全性的不足，並且都針對於 Data 的部份作保護，對於兩端通訊者的身份卻還是暴露在網際網路上。

因此希望在現有的網際網路架構上，能提供雙方通訊間的隱密性。所謂的隱密性，是不要讓第三者輕易的就能得知通訊者身分。以現有網際網路架構上，只能用混淆的方式來增加困難度。

1.2 研究目標

利用 FreeBSD 開放原始碼的特

性，將原本的 TCP/IP 做修改來達成所要的目的。對於混淆的方式，能透過其他特定的電腦，藉由在它們之間來傳遞封包，達成混淆的效果，而在這些特定電腦的接收與傳送封包使用 FreeBSD 所提供的函式庫來達成。在隱藏的部份，透過對稱式加密法的方式來達成，使用 OpenSSL 所提供相關的加解密函式庫。

秘密通訊機制要有以下特性：

1. 能保有原來的 TCP/IP 的連線方式，與修改過的 TCP/IP 的連線方式，兩者並存，且能互相切換。
2. 能保有 TCP 的所有機制。
3. END TO END 的架構。
4. Sender 自己決定路徑，並且能自動更新路徑。
5. Application 不須要更改，即可使用秘密通訊。

1.4 論文架構

本論文一共分為四節，第一節為緒論，對整篇論文做一個簡單的介紹。第二節則說明系統的架構、設計的原理與系統如何實作。第三節則是實作的結果，並且對系統效能與正常模式做比較。第四節為本篇論文的總結，並且對未來的發展提出一些意見。

2 系統設計

2.1 系統架構

系統由三個部分組合而成別是 Host、Onion Server、Onion Router。Onion Server 負責管理 Onion Router 的狀態。Host 向 Onion Server 取得 Onion Router 的資訊，並且自行決定路徑。Onion Router 負責轉送封包，由轉送的過程來混淆真正的目的端。如圖 1。

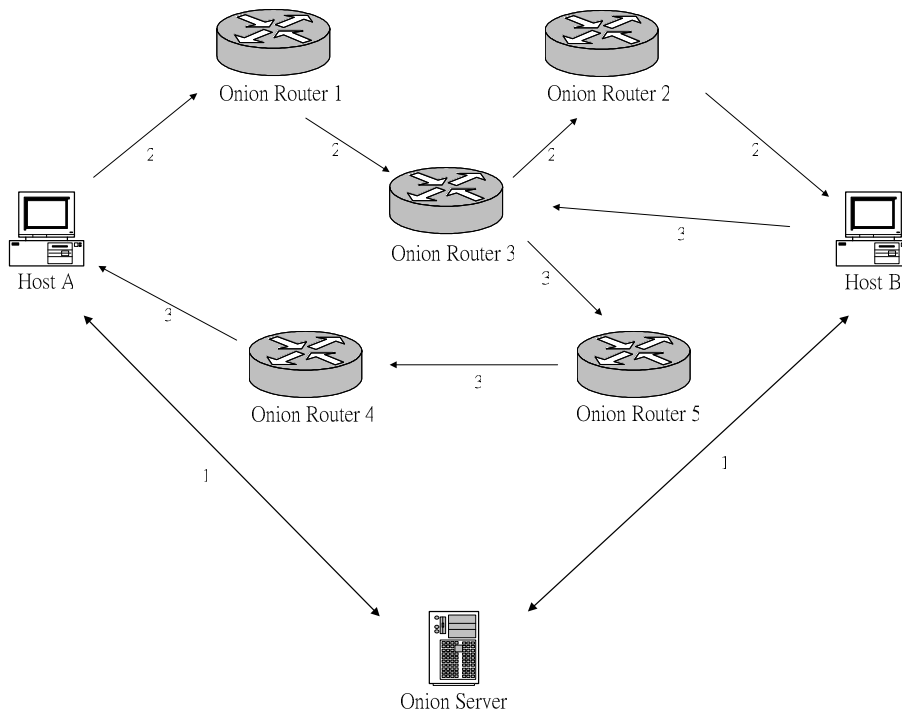


圖 1 秘密通訊架構圖

首先 Onion Router 必須先向 Onion Server 註冊，讓 Onion Server 得到所有 Onion Router 的資訊。接著 Host A 與 Host B 分別向 Onion Server 取得 Onion Router 的資訊，Onion Server 回應部份 Onion Router 的資訊給 Host A 與 Host B，分別是 Onion Router 1、2、3 與 Onion Router 3、4、5，如圖 1 的編號 1。

當 Host A 取得 Onion Router 的資訊，便可以決定封包的路徑，如圖 1 的編號 2，Host A → Onion Router 1 → Onion Router 3 → Onion Router 2 → Host B。Host B 決定的封包路徑為 Host B → Onion Router 3 → Onion Router 5 → Onion Router 4 → Host A，如圖 1 的編號 3。

2.2 封包格式

原始的 IP datagram 仍然會保留，但必須加密起來，外面再加入一層 IP Header，Aaaaa

其中 protocol 欄位為 111 表示為秘密通訊的特定封包，而目的地 IP Address 為第一個經過的 Onion Router。外層的 IP Header 與原始的 IP Header 之間放入第二個至最後一個經過的 Onion Router 的 IP Address 與另一端 Host 的 IP Address。

從 Host 送出的封包格式，會將最內部的原始 IP 封包使用雙方 Host 的 secret key 加密，再用 Onion Router 的 secret key 將它之後的路徑資訊與原始 IP 封包一層一層地加密，見圖 2。

當 Onion Router 收到封包後，會將一層解密開，接著將封包底部亂數隨機 Padding，如圖 3.5。透過 Onion Router 對封包的處理，會使處理前與處理後的封包，payload 的部份會不一樣，封包長度也不一樣，找不出這兩個封包的相關性，而達到混淆的效果。

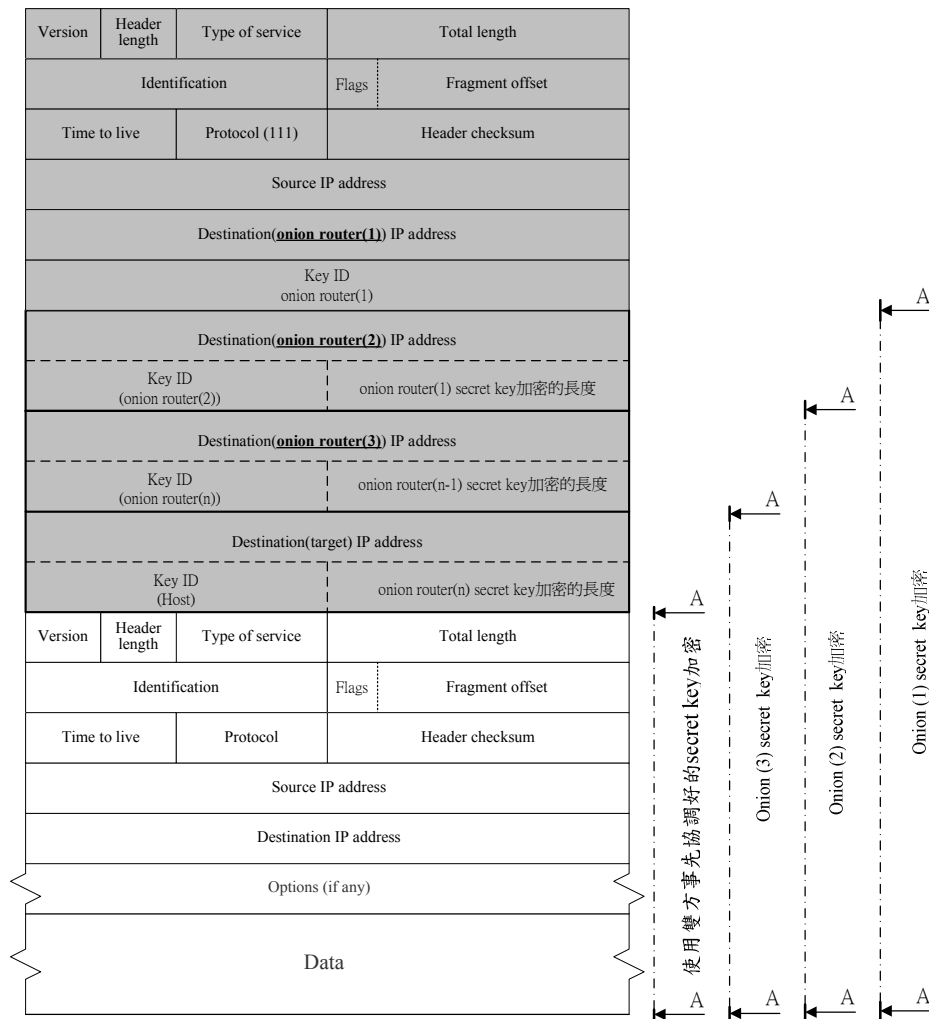


圖 2 Host 送出的秘密通訊封包

2.3 介紹 implementation

系統功能有下列三項：

- (1) 可設定正常模式或者秘密通訊模式。當設定為 enable 時為秘密通訊模式，此時所有的封包便會以 3.2 所敘述的封包格式送出。設定為 disable 時，封包便會恢復成原始的格式。
- (2) 可設定經過 Onion Router 的數目。根據需求可設定路徑長度，Onion Router 的數目越多隱密性越高，但是效率越差。Onion Router 的數目越少隱密性越低，但是效率越好。
- (3) 動態調整 segment 大小。當 Onion Router 的數目越多，IP 封包的長度就會越長，有可能會超過 MTU

的長度。因此根據 Onion Router 的數目，動態地設定 segment 的大小，可讓 IP 封包的長度不會超過 MTU。秘密通訊使用特定的封包格式，必須修改 TCP/IP 原始碼，因此使用開放原始碼的作業系統 FreeBSD，版本為 5.1。Host 端修改原始碼的部份包括 ip_output.c, ip_input.c, 及 tcp_output.c。Onion Router 使用兩個 thread，一個 thread 使用 Raw Socket 的方式來收送秘密通訊使用特定的封包；另一個 thread 用來與 Onion Server 通訊。Onion Server 使用三種 thread，一種 thread 用來與 Onion Router 通訊，對於每一個 Onion Router 都會產生一個 thread 來與 Onion Router 通訊；另一種 thread 用來與 Host 通訊；最後一種 thread 用來檢查 Onion Router 是否 timeout。

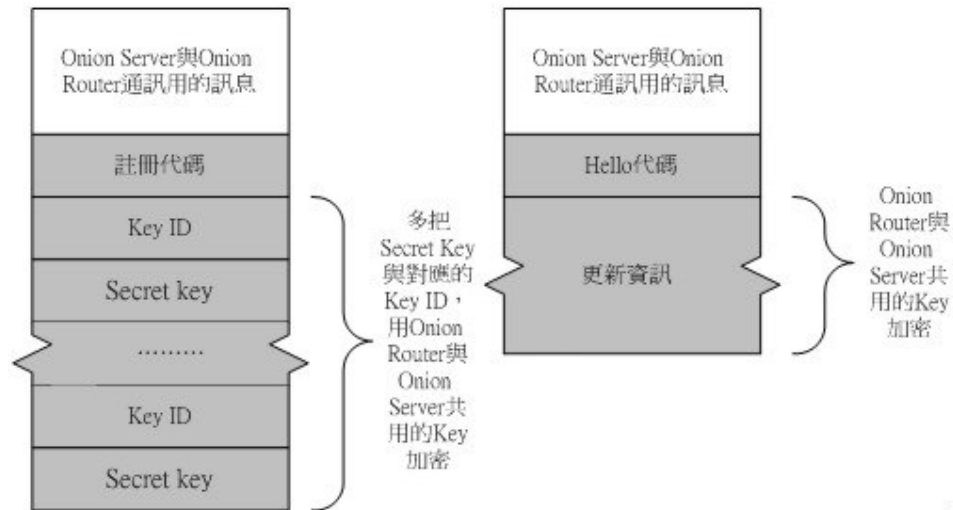


圖 3 Onion Router 與 Onion Server 的通訊

2.4 系統運作

2.4.1 Onion Server

有三個主要部份，分別為與 Onion Router 通訊、與 Host 通訊以及檢查 Onion Router 是否 timeout。會使用多個 thread 來完成這三個部份，其中與 Onion Router 通訊的部份，對於每一個 Onion Router 都會產生相對應的 thread 來服務。

(1) Onion Server 與 Onion Router 通訊：

所有的 Onion Router 都必須向 Onion Server 註冊，因此 Onion Server 知道所有 Onion Router 的資訊(包含 Onion Router 的 IP address、secret key 和對應的 key ID)，如圖 3。每一個 Onion Router 的 Key ID 都會有 2^{16} 個，分別對應 65536 把不同的 secret key，其目的是要讓每個 Host 對於同一個 Onion Router 都拿到不同的 secret key，因此每一個 Onion Router 可以服務 65536 個 Host。並且每一把 secret key 都有時間性，當 Host 的封包一段時間沒有經過某台 Onion Router 時，該把 secret key 便會改變。Onion Router 的資訊必須使用與 Onion Server 共有的 Secret key 加密。當完成註冊的 Onion Router 每隔一段時間必須發送 hello message 給 Onion

Server，如果 Onion Server 一段時間沒有收到 hello message，表示 Onion Router

已不存在，Onion Server 能夠立即更新。hello message 包含 secret key 的更新，如圖 3。

(2) Onion Server 與 Host 通訊：

秘密通訊是由多群 Onion Server-Onion Routers 所組成，每一群 Onion Server-Onion Routers 會有一台 Onion Server 管理多個 Onion Routers。

當 Host 要作秘密通訊時，可以有三種選擇，第一種必須要向多個 Onion Servers 詢問有哪些 Onion Routers 可以使用，如圖 4，目的是要分散風險，避免某一台 Onion Server 被入侵而得知那台 Onion Server 所管轄的所有 Onion Router 的資訊，因此 Host 在選擇路徑時，不可以全部都選同一 Onion Server-Onion Routers 群內的 Onion Router。每台 Onion Server 回報現在已存在的一部分 Onion Routers 的資訊(包含 Onion Router 的 IP address、secret key 和對應的 key ID)給 Host。每一個 Host 所得到的 Onion Router 資訊可能都不一樣，以避免每一個 Host 都使用相同的 Onion Router 與相同的 secret key。此 Onion Router 資訊必須使用 Onion Server 與 Host 共有的 secret key 來加密。當 Host 所擁有的 Onion Router 有變動時，Onion Server 會主動告知 Hos

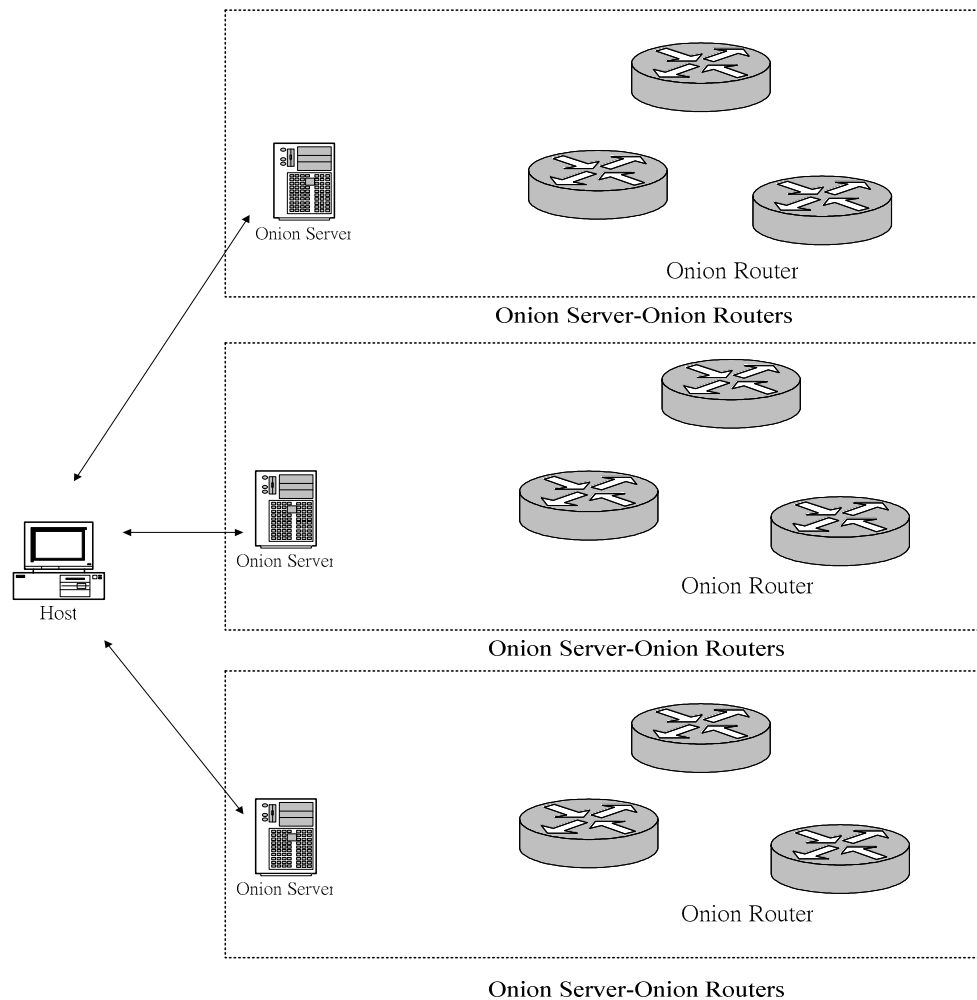


圖 4 向多台 Onion Server 取得 Onion Router 的資訊

最新的 Onion Router 資訊，因此兩端 Host 在通訊時，並不會因為 Onion Router 的變動而造成斷線。

第二種必須要向一個 Onion Servers 提出詢問，且必須知道除了目的端之外的其他 Host，可以是一台或多台，如圖 5。當 Host A 要與 Host C 通訊時，Host A 選擇 Host B 當作中繼站。封包抵達中繼站 Host B 時，Host B 再選擇它所在群組的 Onion Router，封包經過 Host B 群組的 Onion Router 之後會抵達目的端 Host C，因此可以透過中繼站來跨越不同群組。

第三種只要向一個 Onion Servers 提出詢問，傳送的封包只會在單一群組的 Onion Router 之間行走，之後直接抵達目的端。

可藉由 Onion Server-Onion Routers 群數目的增加，讓秘密通訊的 Host 一直增加。因此可讓無限多個 Host 加入秘密通訊系統。

(3) 檢查 Onion Router 是否 timeout：

當 Onion Server 發現某一台 Onion Router 已經 timeout，便會通知擁有此 Onion Router 的 Host。可讓 Host 選擇路徑時，不會選到此台有問題的 Onion Router。

2.4.2 Hosts

◆ 秘密通訊模式的啟動

利用 Raw Socket 的特性，可將 message 傳送到 ip_output，來進行秘密通訊模式的開啟與關閉。啟動秘密通訊模式包括下列三項工作：

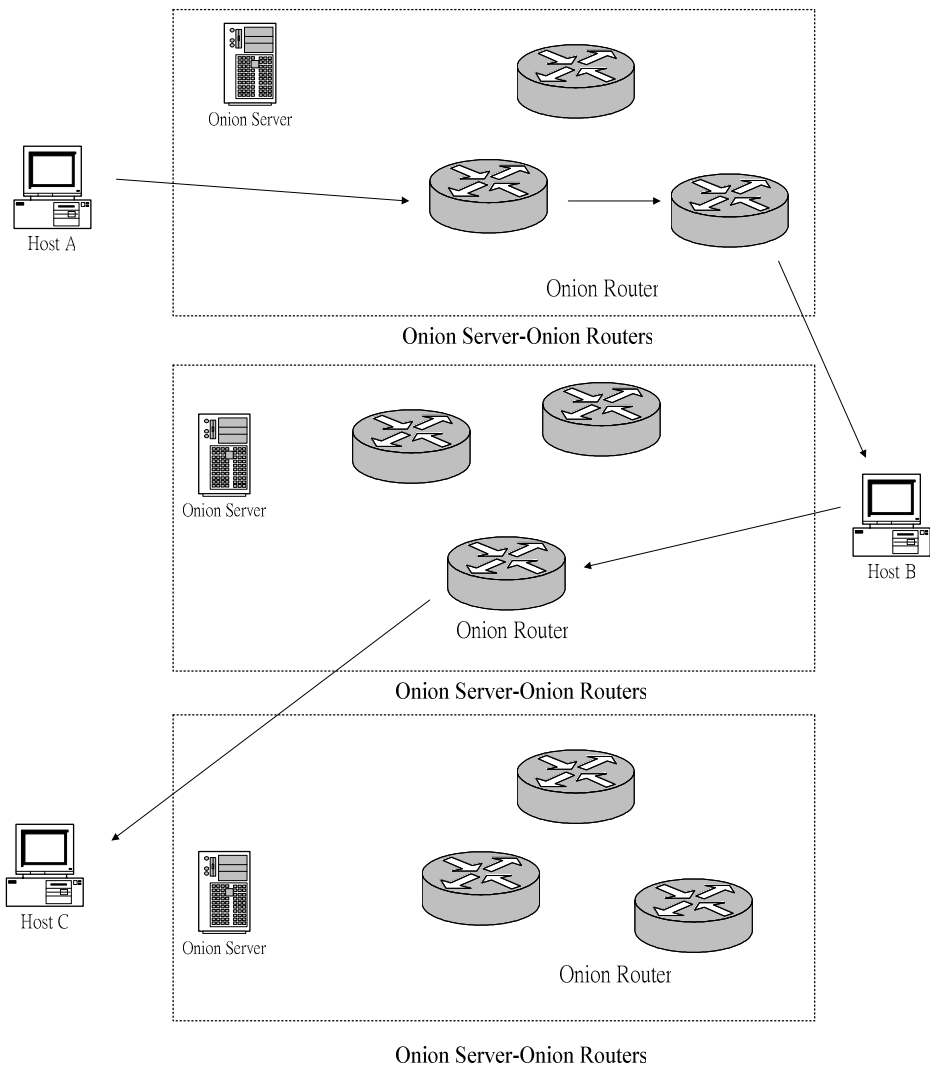


圖 5 Host 使用中繼站的秘密通訊

- (1) 向 Onion server 取得 Onion Router 的資訊 (包括 Onion Router 的 IP address、secret key 和對應的 key ID)。
- (2) 獲得 Onion Routers 資訊後，若是目前無存在任何 Onion Router，則無法開啟秘密通訊模式；若是有一台以上的 Onion Router 存在，則秘密通訊模式的啟動程式會傳送開啟秘密通訊模式的 message 以及 Onion Router 的資訊到 ip_output，如圖 6。
- (3) 若有兩台以上 Onion Router，則可讓使用者設定封包傳送的路徑長度。

◆ 秘密通訊模式的運作

秘密通訊模式的運作是透過修改 tcp_output、ip_output、ip_input 的方式達成。

- (1) 由 transport layer 傳下來的封包先將決定好的路徑資訊填入，再加入外層 IP Header，最後進行加密的動作。首先使用雙方事先協調好的 key 來對內部原始封包加密，接著依序使用每台 Onion Router 的 secret key 對之後的 payload 加密，從 Onion Router (n)、Onion Router (n-1)、...、一直到 Onion Router (2)，如圖 2。
- (2) 一旦 Onion Router 有所變動時，便會收到新的 Onion Router 資訊，因此會自動改變路徑。

- (3) 當封包經過所有的路徑之後，最後會到達目的端 Host。
- (4) 目的端 Host 在 ip_input 收到此格式的封包後，先去掉 IP header 與 padding，再解密原始 IP 封包內容，即可得到原始資訊。

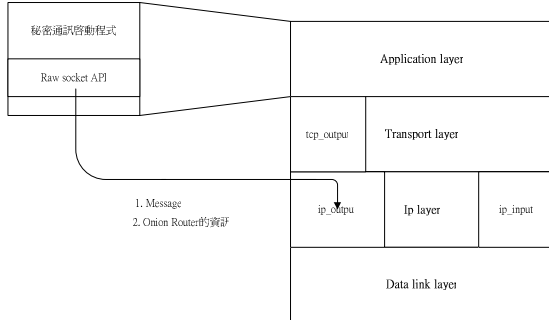


圖 6 啟動程式的系統內部運作圖

2.4.3 Onion Router

有兩個主要部份，分別為 Register 及 Packet forward。使用兩個 thread 來完成這兩個部份。

(1) Register :

當 Onion Router 啟動時，首先產生 65536 把 Secret Key 與對應的 Key ID，Key ID 是用來識別每一把 Secret Key。接著必須向 Onion Server 註冊，註冊的資訊包含 Onion Router 的 IP address、secret key 和對應的 key ID，此資訊必須用與 Onion Server 共有的 Secret key 加密。之後每隔一段時間發送 hello message 給 Onion Server 表示其 Onion Router 還存在。

(2) Packet forward :

利用 Raw Socket 可接收有特定 protocol 值的封包之特性，可將 IP layer 收到秘密通訊模式的封包傳送到 Application layer。因此便可以直接處理這類型的封包，之後再透過 Raw Socket，將封包直接傳送到 IP layer，便可完成 forward 的動作。

當收到 packet 時，先從 Key ID 欄位得知使用加密的 secret key，將放在外層 IP Header 後面的資訊(已用此 Onion Router 的 Key ID 之 secret key 加密)解密。解密第一塊 block(8-bytes)後，可取

得 1)下一站的位址；2)下一站的 Key ID；3)用此 secret key 加密的長度。由第 3)點可以將剩下的部分解密，再將外層的 IP Header 去除，重新產生一個新的外層 IP Header，Destination address 欄位填入 1)下一站的位址；Source address 欄位填入自己的 IP address。由第 3)點可以間接得知收到封包的 padding 長度，必須先去除此 padding，再產生隨機長度的 padding，其目的要讓收到的封包與送出的封包長度沒有相關性，具有混淆的效果。整個封包長度確定後，將 Total length 欄位填入，再將 IP Header 的其他欄位填入。最後 Key ID 欄位填入 2)下一站的 Key ID，便立刻送出。例如圖 7，當 Onion Router R1 收到 Packet 1 時，由外層 IP Header 之後的 Key ID 欄位為 23，表示必須使用 ID 為 23 的 secret key 解密。首先解開第一塊 block，獲得 1)下一站的位址，R1；2)下一站的 Key ID，50；3)用此 secret key 加密的長度，1478。由外層 IP Header 的 Total length 欄位得知 Packet 1 的長度為 1500 bytes 以及 3)加密的長度為 1478，及可判斷封包後面的 padding 長度；外層 IP Header 的長度為 20 bytes 加上加密的長度 1478 bytes 與 Key ID 的長度 2 bytes，共為 1500 bytes，表示 Packet 1 沒有 Padding。由 3)加密的長度的資訊，再將剩餘的部分解密後，分別將 1)和 2)的資訊填入新的外層 IP Header 相對應的欄位。最後決定 Padding 的長度為 7 bytes，Packet 2 的 Total length 為 $1478+2-8+7+20=1499$ bytes，Onion Router R1 就完成 Packet 2，將它傳送出去。

每個 Onion Router 只能夠解開下一站的 IP address，並無法得知其他的 IP address，也無法得知會經過多少個 Onion Router。

3. 效能分析

關於實際效能的部份，使用三個不同大小的檔案，分別為 5139KB、10.69MB、

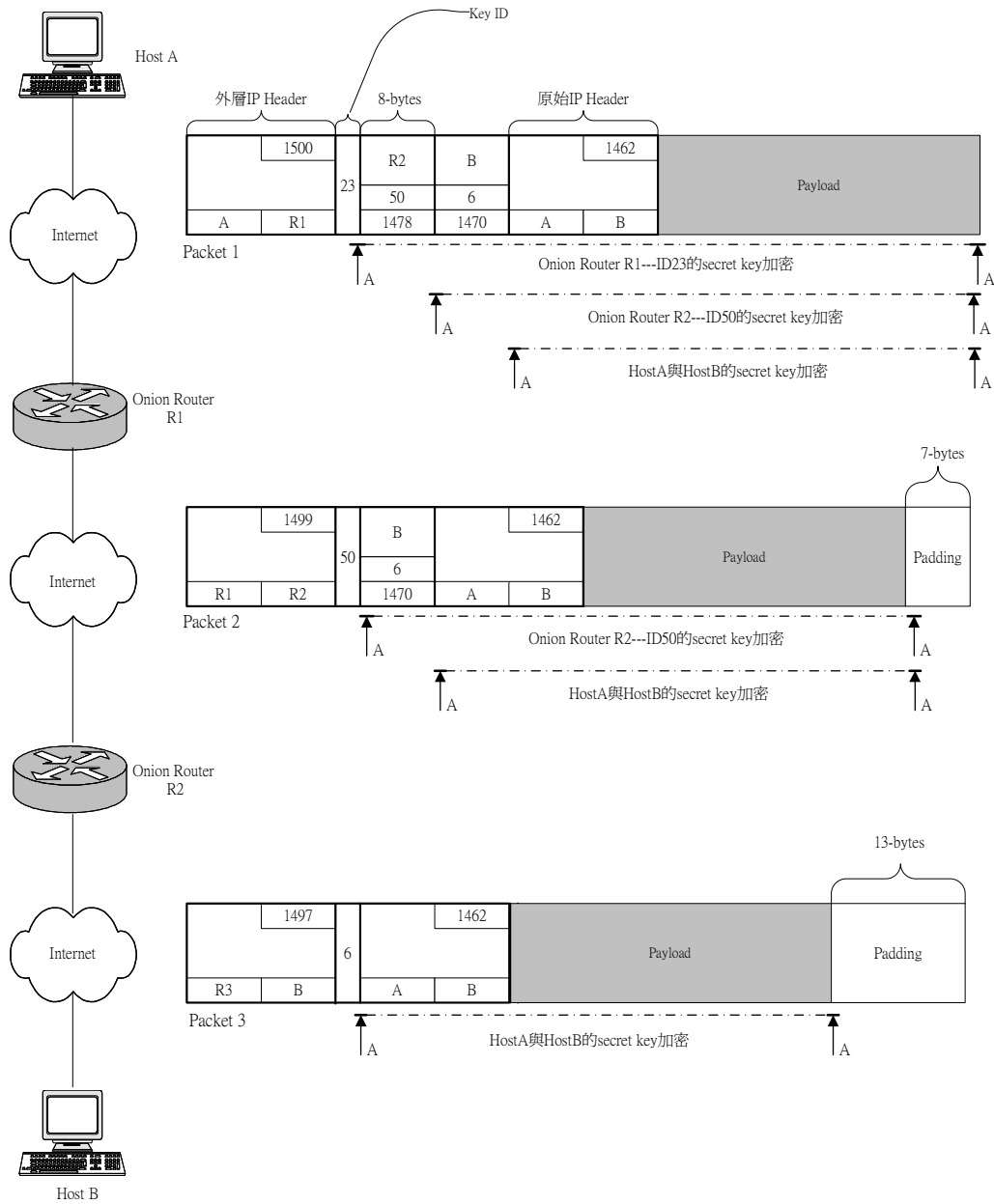


圖 7 秘密通訊的封包運作過程圖

173MB，透過上傳與下載這些檔案來分析不同情況的效能。我們將分成五種情況，第一種為正常連線，未啟動秘密通訊的一般正常連線；第二種為啟動秘密通訊，經過一台 Onion Router R1 但未加密；接下來的三種為啟動秘密通訊，有加密，分別經過的 Onion Router 為 R1 一台、R2 一台、R1 和 R2 兩台，見表 8。

上傳：Host A 將檔案傳往 Host B。

下載：Host B 將檔案傳往 Host A。

由表 8 的正常連線欄位可得知，Host A 傳送檔案至 Host B 的極限為 4.5MB/s 左

右，Host B 傳送檔案至 Host A 的極限為 10MB/s 左右。由無加密(R1)與加密(R1)兩個欄位，可得知加密對整體的效能影響最大，因此整個系統負擔最重為 Sender，由下載那一欄位便可看出，Host B 的 CPU 為 333MHz 導致效能會變的不理想。由加密(R1)與加密(R2)兩個欄位，可得知 Onion Router 的快慢的效能影響不大。檔案的大小則對效能毫無影響。

3.1 安全性分析

當一個封包從 Sender 端送出，經過一

Size:5139KB	正常連線	無加密(R1)	加密(R1)	加密(R2)	加密(R1、R2)
上傳	4.72MB/s	3.73MB/s	1.65MB/s	1.81MB/s	1.43MB/s
下載	10.12MB/s	4.16MB/s	1.22MB/s	1.21MB/s	879.52KB/s

Size:10.69MB	正常連線	無加密(R1)	加密(R1)	加密(R2)	加密(R1、R2)
上傳	4.45MB/s	3.70MB/s	1.58MB/s	1.81MB/s	1.55MB/s
下載	10.13MB/s	4.11MB/s	1.20MB/s	1.21MB/s	853.47KB/s

Size:173MB	正常連線	無加密(R1)	加密(R1)	加密(R2)	加密(R1、R2)
上傳	4.47MB/s	3.69MB/s	1.59MB/s	1.77MB/s	1.48MB/s
下載	9.20MB/s	4.10MB/s	1.21MB/s	1.21MB/s	881.25KB/s

表 8 效能分析

台 Onion Router(r1)，往第二台 Onion Router(r2)傳送，最後 Onion Router(r2)將封包傳送到 Receiver 端。一個封包在 Sender 端透過三層加密，再經由兩台 Onion Router 的一層解密與 Padding，最後抵達 Receiver 端。可讓單一封包從 Sender 端到 Receiver 端，被混淆成三個獨立的封包，這三個封包長度不一樣，內容也都不一樣。

原始的 IP Header 的 Source IP Address 與 Destination IP Address，記錄 Sender 與 Receiver 的身分，由於經過多層的加密以及不容易察覺原始的 IP Header 位於封包的相對位址，因此很難得知通訊兩方的身分。

4 結論

本篇論文中，我們設計一個可在網路

上進行秘密通訊的系統來降低被第三者竊聽與 traffic analysis 的風險，在盡可能增加安全性的同時也能有好的效能表現。使用者可根據自己的需求決定路徑長度，路徑越長安全性越高，但是效能會越差，反之亦然。

在現今網路上沒有百分之百安全的系統能夠防止竊聽與 traffic analysis，只能增加被竊聽與 traffic analysis 的困難度，使安全性提高。我們利用密碼學與 Onion Router 的轉送封包來增加混淆程度，提高竊聽與 traffic analysis 的困難度。我們修改作業系統的核心，但還保有原始的功能，透過啟動程式可選擇要使用原始的連線方式或者是具有隱密性與安全性的秘密通訊連線，而不是一個只供特殊應用的單一系統。

參考文獻

- [1] David Goldschlag, Michael Reed, Paul Syverson, "Anonymous connections and onion routing", in Proc. IEEE Journal on Selected Areas in Communications, on Volume 16, Issue 4, May 1998
- [2] David Goldschlag, Michael Reed, Paul Syverson, "Anonymous connections and onion routing", in Proc. 1997 IEEE Symp. on Security and Privacy, Oakland, CA, May 1997
- [3] David Goldschlag; Michael Reed, Paul Syverson, "Onion routing for anonymous and private Internet connections", Association for Computing Machinery. Communications of the ACM; Feb 1999
- [4] David Goldschlag, Michael Reed, Paul Syverson, "Onion routing access configurations", DARPA Information Survivability Conference and Exposition, 2000
- [5] David Goldschlag, Michael Reed, Paul Syverson, "Proxies for anonymous routing", in Proc. 12th An. Computer Security Applications Conf., San Diego, CA, 1996
- [6] Paul Syverson, "Onion routing for resistance to traffic analysis", in proceedings of the DARPA Information Survivability Conference and Exposition on Volume 2, April 2003
- [7] Andrew Oram, Steve Talbott, "Managing Projects with make, 2/e", O'Reilly, 1991
- [8] Behrouz A. Forouzan, Tyler Gregory Hicks, "TCP/IP Protocol Suite, 2/e", McGraw-Hill, 2002
- [9] Gary R. Wright, W. Richard Stevens, "TCP/IP Illustrated, Volume 2: The Implementation", Addison Wesley, 1995
- [10] Michael Lucas, Jordan Hubbard, "Absolute BSD: The Ultimate Guide to FreeBSD", No Starch Press, 2002
- [11] W. Richard Stevens, "UNIX Network Programming, Volume 1, 2/e", Prentice Hall, 1997
- [12] W. Richard Stevens, "TCP/IP Illustrated, Volume 1: The Protocols", Addison Wesley, 1995
- [13] 林慶德, "UNIX 網路程式設計--網路應用程式設計介面 Sockets 與 XTI", 培生, 2002
- [14] 施勢帆, 林毓能, 吳國華, "FreeBSD 實務手冊", 旗標, 2004
- [15] 蔣大偉, "make 專案開發工具", O'Reilly, 2000
- [16] "Ethereal", <http://www.ethereal.com>
- [17] "OpenSSL", <http://www.openssl.org>
- [18] "Winpcap", <http://winpcap.polito.it/>