KOMUNIKASI DATA

"MODEL REFERENSI OSI LAYER"

Matheus S. Rumetna, S.Kom

(*email* : matheus.rumetna@gmail.com)

matheusrumetnas.com

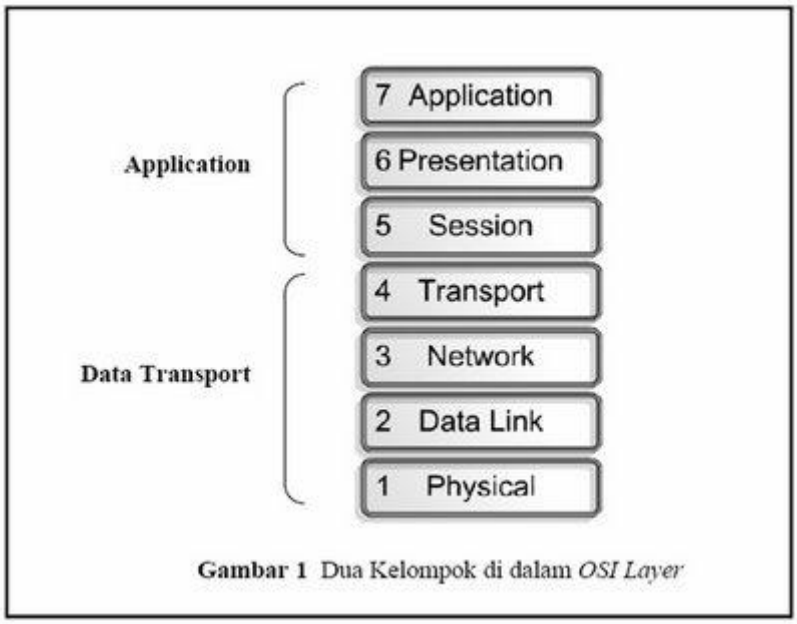
BAB I
PENDAHULUAN

1.1 Pengantar

Model Referensi OSI merupakan salah satu arsitektur jaringan komputer yang dibuat oleh ISO (*International for standarization Organization*) untuk memecahkan masalah kompatibilitas *device* antar *vendor*, dengan menyediakan standarisasi yang dapat digunakan oleh para *vendor* dalam membuat *device*. Model referensi OSI mengidentifikasikan semua proses yang dibutuhkan untuk melakukan komunikasi dan membaginya ke dalam kelompok secara logika yang disebut layer. OSI menjelaskan bagaimana data dan informasi dari sebuah aplikasi pada sebuah komputer melewati media jaringan berkomunikasi ke aplikasi yang berada di komputer lain. OSI juga merupakan sebuah framework dalam pembuatan dan mengimpementasikan standar jaringan. OSI terdiri dari tujuh layer, yang secara umum terbagi dalam dua kelompok, yakni *Upper layer* (*Application Layer*) dan *lower layer* (*data transport layer*). Layer yang tergolong dalam uper layer mendefinisikan bagaimana aplikasi pada sebuah *host* akan berkomunikasi dengan user dan host lainnya. Sedangkan *lower layer* mendefinisikan bagaimana data terkirim dari satu *host* ke *host* lainnya. Model referensi OSI terdiri dari tujuh layer, antara lain :

1. *Application Layer*
2. *Presentation Layer*
3. *Session Layer*
4. *Transport Layer*
5. *Network Layer*
6. *Data Link Layer*
7. *Physical Layer*

matheusrumetna@wordpress.com



Gambar 1 Dua Kelompok di dalam *OSI Layer*

matheusrumetna@wordpress.com

1. ***Aplication Layer***

   Aplication layer berfungsi sebagai *interface* antara *user* dan komputer. *Layer* ini bertanggung jawab untuk mengidentifikasi ketersediaan dari *partner* komunikasi, menentukan ketersediaan *resources* dan melakukan proses sinkronisasi komunikasi. *Application layer* menentukan identitas dan ketersediaan dari *partner* komunikasi untuk sebuah aplikasi dengan data yang dikirim Beberapa contoh aplikasi yang bekerja di *application layer* antara lain :

   a. Telnet (*Telecommunication Network*)

      Telnet merupakan program yang menyediakan kemampuan bagi *user* untuk dapat mengakses *resource* sebuah mesin (telnet *server*) dari mesin lain (telnet *client*) secara *remote*, seolah olah *user* berada dekat dengan mesin dimana *resource* tersimpan.

   b. FTP (*File Transfer Protocol*)

      FTP merupakan sebuah program yang berfungsi mengirimkan *file* dari suatu *host* ke *host* lain melalui jaringan.

c. DNS (*Domain Name System*)

Mekanisme pemetaan antara FQDN (*Fully Qualified Domain Names*) dengan alamat IP. FQDM merupakan sebuah hierarki yang secara logika menempatkan sistem berbasis pada domain pengenal.

d. SMTP (*Simple Mail Transfer Protocol*)

SMTP merupakan sebuah protokol (program yang dieksekusi oleh program lain) yang berfungsi untuk mengatur pengiriman e-mail.

e. SNMP (*Simple Network Manajemen Protocol*)

SNMP merupakan salah satu jenis protokol yang memberikan kemampuan untuk mengawasi dan mengatur peralatan-peralatan dalam jaringan komputer.

2. *Presentation Layer*

Presentation Layer berfungsi untuk :

a. Menyediakan sistem penyajian data ke *aplication layer*.

b. Menyediakan sistem pembentuk kode (format *coding*), misalnya format ASCII yang digunakan komputer IBM *compatible* dan format EBDIC digunakan oleh mesin IBM.

c. Menyediakan proses konversi antar format *coding* yang berbeda.

d. Menyediakan layanan *translation*. *Presentation layer* menjamin data yang dikirimkan dari *application layer* suatu sistem dapat dibaca oleh layer aplikasi di sistem yang lain.

e. Menyediakan sarana untuk melakukan *compression, decompression, encriprion* dan *decryption*. Beberapa contoh aplikasi yang bekerja di presentation layer antara lain:

(1) PICT, TIFF, JPEG, merupakan format data untuk aplikasi berupa gambar (*image*).

(2) MIDI, MPEG dan quicktime, merupakan format data untuk aplikasi *sound* & *movie*.

(3) EBDIC dan ASCII, merupakan format data untuk informasi dalam bentuk teks.

matheusrumetna@wordpress.com

3. Session Layer

Berfungsi dan bertanggung jawab:

a. Mengkoordinasi jalannya komunikasi antar sistem.

b. Melakukan proses pembentukan, pengelolaan dan pemutusan *session* antar sistem aplikasi.

c. Mengendalikan dialog antar *device* atau nodes. Berikut ini adalah beberapa contoh protokol yang bekerja di *session layer* :

(1) *Remote Procedure Call* (RPC). Merupakan protokol yang menyediakan mekanisme *client/server* pada sistem operasi windows NT.

(2) *Structure Query Language* (SQL) , dibangun oleh IBM untuk menyediakan kemudahan bagi *user* dalam mendefinisikan kebutuhan-kebutuhan informasi yang terdapat di sistem lokal atau remote sitem.

(3) *Network File System* (NFS), dibangun oleh Sun Microsistem dan digunakan oleh *workstation* TCP/IP dan Unix agar dapat mengakses remote *resource*.

(4) X Windows, merupakan protokol yang menyediakan mekanisme *client/server* pada sistem operasi Unix

(5) Apple Talk Session Protokol (ASP), merupakan protokol yang menyediakan mekanisme *client/server* pada mesin-mesin Apple.

4. *Transport Layer*

Transport Layer bertanggung jawab dalam proses :

a. Pengemasan data *upper layer* ke dalam bentuk *segment*.

b. Pengiriman *segment* antar *host*.

c. Penetapan hubungan secara logika antar *host* pengirim dan penerima dengan membentuk virtual circuit.

d. Secara opsional, menjamin proses pengiriman data yang dapat diandalkan.

matheusrumetna@wordpress.com

Proses pengiriman pada *transport layer* ini dapat dilakukan dengan 2 mekanisme :

(1) *Connection Oriented*

Proses pengiriman yang menggunakan *Connection oriented* dapat diilustrasikan pemberian pesan kepada seseorang yang dipisahkan oleh jarak yang jauh. Pemberikan pesan tersebut dilakukan melalui telepon. Proses pemberian pesan akan dilakukan jika lawan bicara adalah orang yang dituju sehingga dapat dipastikan bahwa pesan diterima oleh orang yang dimaksudkan.

Berdasarkan ilustrasi tersbut dapat kita simpulkan bahwa data yang dikirimkan dengan menggunakan mekaisme *connection oriented* dapat diandalkan. TCP (*Transmission Control Protocol*) merupakan jenis protokol yang mampu mengirimkan data yang *reliable*.

(2) *Connection Less*

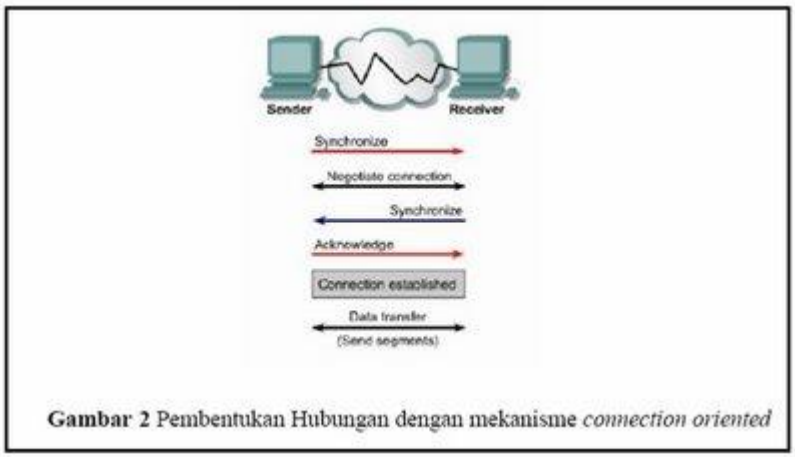
Mekanisme *connectionless* diilustrasikan dengan proses pemberikan pesan yang dilukukan melalui surat. Pengiriman surat mengkin sampai ke tempat tujuan tetapi penerima di tempat tujuan belum tentu orang yang dimaksudkan sehingga pesan belum tentu sampai ke orang yang dimaksud. Dari ilustrasi tersebut dapat kita simpulkan bahwa data yang dikirmkan dengan menggunakan mekanisme *Connectionless* kurang dapat diandalkan.

UDP (*User Datagram Protocol*) mengirimkan data unreliable Pengiriman data dengan menggunakan TCP tidak berarti selalu tanpa kesalahan.Kesalahan dapat terjadi tetapi kesalahan tersebut dapat dideteksi dan dapat dilakukan proses pengiriman ulang atas segment yang salah. Proses pembentukan hubungan *connection-oriented* dilakukan melalui beberapa langkah yakni :

1. Pengiriman segment *syschronization* untuk menetapkan *connection agreement*.

matheusrumetna@wordpress.com

2. Segment kedua dan ketiga adalah *acknowledge* yang meminta dan menetapkan parameter-parameter antar *host.*
3. Segment terakhir merupakan sebuah *acknowledgement*, *segment* ini memberitahu *host* tujuan bahwa *connection agrement* telah diterima dan hubungan telah ditetapkan, sehingga dan sudah mulai dikirimkan.



Gambar 2 Pembentukan Hubungan dengan mekanisme *connection oriented*

Connection oriented memiliki karakteristik sebagai berikut :

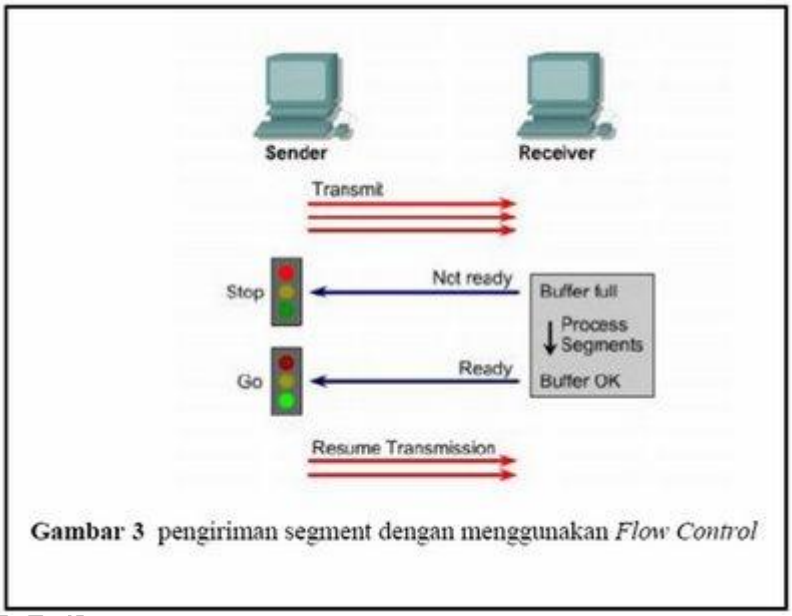
1. Setelah menerima *segment* dari pengirim, *station* penerima akan mengirimkan *segment acknowledge back* ke *station* pengirim.
2. *Station* pengirim akan mengulang pengiriman *segment* ketika menerima *acknowledge* dari penerima.
3. *Segment-segment* akan disusun kembali oleh penerima ke dalam susunan yang tepat.
4. Dapat mengelola alilran data sehingga tidak terjadi *congestion*, *overload* dan kehilangan data.

Ketika menerima data dari komputer lain, sebuah komputer akan menyimpan dalam sebuah memori yang disebut *buffer*. Teknik *buffering* merupakan salah satu teknik untuk mengatasi *congestion*. Teknik *buffering* terbatas untuk penerimaan data dalam jumlah tertentu karena kapasitas *buffer* sangat terbatas. Untuk menangani

matheusrumetna@...om

keterbatasan ukuran *buffer*, *layer transport* menyediakan mekanisme *flow control*.

Flow control mencegah *host* pengirim melakukan pengiriman data yang menyebabkan terjadinya *overlow* dan kehilangan data pada sisi *host* penerima. Pencegahan dilakukan dengan mengiriman sinyal *not ready* pada pengirim ketika kapasitas *buffer* sudah penuh pada sisi penerima, sehingga *host* pengirim menghentikan sementara proses pengiriman data sampai menerima sinyal *go*. Proses di atas diilustrasikan pada gambar dibawah ini

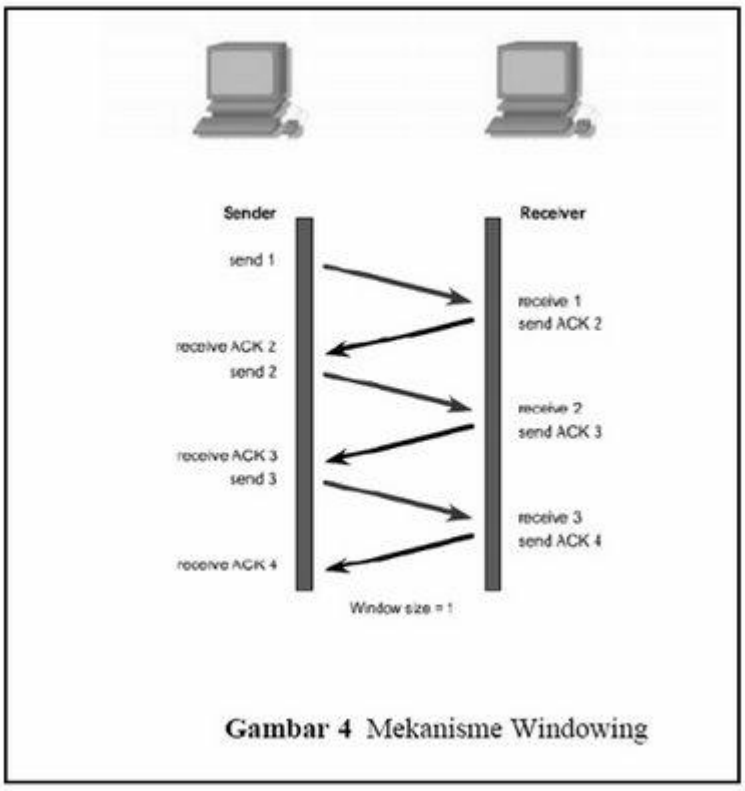


Gambar 3 pengiriman segment dengan menggunakan *Flow Control*

Pengiriman data akan berjalan lambat jika *host* pengirim selalu menunggu *acknowledgment* setelah mengirimkan tiap *segment*-nya. Banyak waktu terbuang karena *host* pengirim hanya dapat melakukan pengiriman *segment* berikutnya setelah selesai menerima *acknowledgment* dari *host* penerima. Masalah banyaknya waktu yang terbuang dapat diatasi dengan mekanisme *windowing*. Sejumlah *segment* yang diperbolehkan untuk dikirimkan tanpa menunggu *acknowledgment* disebut *window*.

matheusr...ess.com

Windowing mengontrol berapa banyak informasi yang dikirimkan dari satu *host* ke *host* lainnya. Gambar dibawah ini menampilkan proses pengiriman dengan ukuran *window* watu dan untuk meningkatkan *performance* ukuran *window* diubah menjadi tiga.

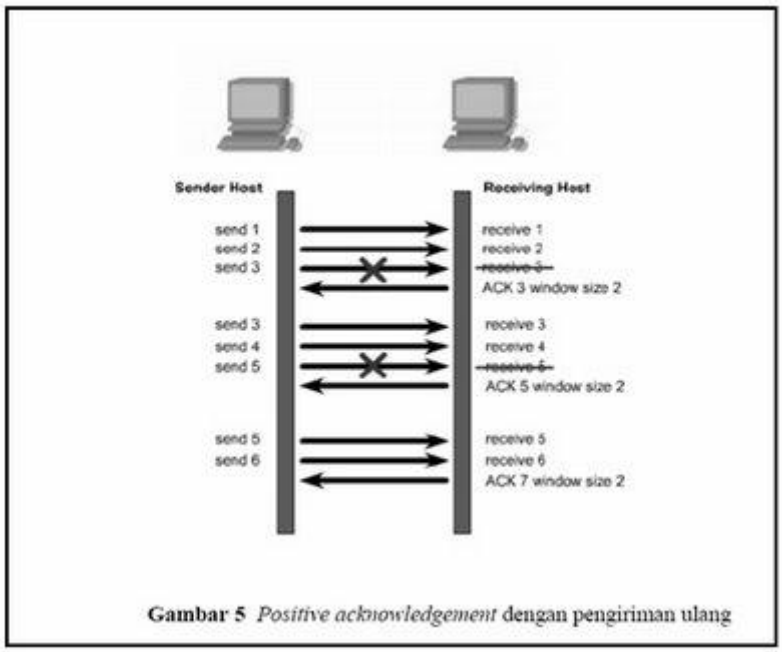


Gambar 4 Mekanisme Windowing

Dengan memperbesar ukuran *window* menjadi tiga, maka *acknowledgment* hanya akan dikirimkan oleh penerima ketika telah menjadi tiga *segment*. Sesuai dengan ukuran *window*. *Host* pengirim akan mencatat setiap segment yang dikirim dan menunggu *acknowledgement* dari *host* penerima sebelum mengirimkan *segment* berikutnya. Jika dalam jangka watu tertentu tidak menerima *acknowledgement* maka *host* pengirim akan melakukan pengiriman ulang.

matheusrum.com

Dalam dibawah ini diperlihatkan bahwa sebuah *host* mengirimkan *segment* 1, 2, 3. *Host* penerima memberitahu *host* pengirim bahwa *segment-segment* tersebut telah diterima dan meminta *segment* ke 4. karena menerima *acknowledgment* 4 maka *host* pengirim akan mengirimkan *segment* ke 4, 5 dan 6. *Segment* 5 mengalami masalah dalam proses pengirimannya dan mengakibatkan *host* penerima memberitahu kejadian tersebut pada *host* pengirim dan meminta pengiriman ulang terhadap segment 5. Ketika *host* penerima telah menerima *segment* ke 5, *acknowledgment* yang diberikan kepada *host* pengirim adalah *acknowledge* untuk meminta *segment* 7.



Gambar 5 *Positive acknowledgement* dengan pengiriman ulang

Beberapa protokol yang bekerja di layer ini adalah sebagai berikut :

1. ATP (*Appletalk Transaction Protokol*) dan NBP (*Name Binding Protocol*), merupakan protokol-protokol di jaringan apple yang bertugas membentuk hubungan antar *host*.
2. NetBios/NetBEUI, menetapkan dan mengelola komunikasi antar komputer sedangkan NetBEUI menyediakan layanan transport data untuk melakukan komunikasi.

matheus

3. SPX (*Sequenced Packet Exchange*) dan NWLink *protocol connection oriented* pada jaringan *Netware* yang digunakan untuk menjamin pengiriman data.
4. TCP (*Transmission Control Protocol*)
   Bagian dari protokol TCP/IP yang bertanggung-jawab untuk mengirimkan data.

5. *Network Layer*

Network Layer bertanggung jawab untuk:

a. Melakukan mekanisme *routing* melalui *internetwork*, *router* merupakan *device* yang berfungsi membawa trafik antar *host* yang terletak dalam *network* yang berbeda.
b. Mengelola sistem pengalamatan logika terhadap jaringan komputer.
c. Data berupa *segment* yang diterima dari *trasport layer* akan dikemas ke dalam bentuk *packet*. Ketika *packet* diterima oleh *interface* sebuah *router*, maka alamat tujuan akan diperiksa jika alamat tujuan tidak ditemukan maka packet tersebut akan dibuang. Tetapi jika alamat tujuan ditemukan dalam *routing table* (sebuah tabel yang terdapat di dalam *router* berisi informasi tentang alamat *network* yang dapat dijangkau oleh *router*) maka *packet* akan dikeluarkan melalui *outbound interface* menuju ke alamat tujuan.

Pada *network layer* terdapat dua jenis *packet* yaitu :

1.) *Packet Data*, digunakan untuk membawa data milik *user* dikirimkan melalui jaringan dan protokol yang digunakan untuk mengelola *packet* data disebut *Routed Protocol*. Contoh protokol yang tergolong ke dalam *routed protocol* antara liain IP dan IPX.

2.) *Route Update Packet*, digunakan untuk meng-*update* informasi yang terdapat dalam *routing table* milik *router* yang terhubung dengan *router* lainnya. Protokol yang mengelola *routing table* disebut dengan

matheusrumetna@wordpress.com

routing protocol. Contoh protokol yang tergolong dalam routing protokol antara lain RIP, IGRP, OSPF dan sebagainya.

Beberapa contoh protokol yang bekerja di network layer adalah sebagai berikut :

a. DDP (*Delivery Datagram Protocol*), merupakan protokol *transport* yang biasa digunakan oleh jaringan komputer apple.
b. IP (*Internet Protocol*), bagian dari Protokol TCP/IP yang menyediakan informasi *routing* dan sistem pengalamatan logika.
c. IPX (*Internet Packet Exchange*) dan NWLink merupakan protokol yang disediakan oleh sistem operasi *netware* yang dibuat oleh novell, digunakan untuk *routing* paket.
d. NETBEUI dibangun oleh IBM dan Microsoft, menyedikan layanan *transport* untuk NetBIOS.

6. *Data Link Layer*

Packet yang diperolah dari *network layer* dibungkus (dienkapulasi oleh *data link layer* ke dalam sebuah *frame*. *Data Link layer* bertugas menjamin pesan yang dikirimkan ke media yang tepat dan menterjemahkan pesan dari *network layer* ke dalam bentuk bit di *physical layer* untuk dikirimkan ke *host* lain. *Data link layer* akan membentuk *packet* ke dalam bentuk *frame* dan menambahkan sebuah *header* yang berisi alamat *hardware* (*physical/hardware addressing*).

Data Link terbagi dalam dua sublayer :

a. *Logical Link Control* (LLC) 802.2, bertanggung jawab mengidentifikasikan protokol *network layer* dan kemudian melakukan enkapsulasi protokol-protokol tersebut. Isi LLC akan menentukan langkah selanjutnya yang harus dilakukan ketika menerima *frame* dari *host* lain (LLC bertindak sebagai *service access point*). Sebagai contoh, ketika *host* menerima *frame*, LLC akan mengerti bahwa *packet* ditujukan untuk protokol IP di *Network Layer*.

matheusrumetna@wordpress.com

b. *Media Access Control* (MAC) 802.3, mendefinisikan bagaimana *packet* ditempatkan pada sebuah media dalam *sublayer* ini sistem pengalamatan *hardware* didefinisikan.

7. *Physical Layer*

Tanggung jawab dari *layer* ini adalah melakukan pengiriman dan penerimaan bit. *Physical layer* secara langsung menghubungkan media komunikasi yang berbeda-beda. *Physical layer* menetapkan kebutuhan-kebutuhannya secara *electrical*, *mechanical* prosedural untuk mengaktifkan, memelihara dan memutuskan jalur antar sistem secara fisik.

matheusrumetna@wordpress.com