Networking in the Ethos Operating System

Jon A. Solworth

Dept. of Computer Science and
Center for RITES
University of Illinois at Chicago

Dan Bernstein, Tanja Lange, Mike Petullo, Xu Zhang, Wenyuan Fei, Pat Gavin, Andrei Wartekin, Yaohua Li, Janosch Rux

Part I

The current state of software

Snowden revelations

- To observers of security and privacy, none of the individual capabilities disclosed by Snowden is surprising.
- We knew how software was failing under attack
- What was shocking was the breadth of activity
- And who it was aimed at
- *We have met the enemy and he is us* –Pogo
- Pogo is right is my take away from the Snowden revelations

The current state of software (prolog)

When software meets the attacker

- it fails (almost always)
- if it doesn't fail, just attack at a different layer
- attackers have to work to make it fail
- but there is plenty of motivation to do so
- for example, US spends $60 Billion a year on intelligence
- a significant amount of it is spent on serveilance

What goes wrong?

Lots of things

- Trust: relying on those who are not reliable
- Weak security services (cryptography, authentication, ...)
- Fragile semantics (buffer overflow, integer overflow, input, ..)
- Complexity
  - to program
  - to use
  - to administer
  - to secure

Trust

- This is the one issue that users cannot avoid
- Who are your adversaries?
- Who are your friends?
- Never rely on someone else when you can do it yourself
- Ex. of trust decisions
  - What Tor nodes should you use?
  - What authentication services should you use?
  - What software should you use?
  - What hardware should you use?

Security Services

- Password authentication appropriate only on local machines
- Authorization to limit what users/programs can do
- Encryption for isolation
- Problems
  - Trust (software, hardware, data)
  - Key escrow (Denial of Service)
  - Key distribution

Fragile semantics

- Programming languages:
  input verification, buffer overflow, integer overflow
- Operating systems: race conditions, isolation failures, aliasing
- Services: isolation, authorization, authentication, encryption
- Network protocols: parsing, XSS, Injection, CSRF

These issues are designed into our software.

Complexity

- Complexity favors the attacker
- The attacker has to find one execution path to compromise
- The defender has to prevent all paths from being compromised

Today's software is unfixable

Robust software—able to withstand attacks:

- must be designed for security
- must have low complexity

Its time to start over

> *Insanity: doing the same thing over and over again and expecting different results.*
>
> Albert Einstein

Lieutenant: *I think we can handle one little girl. I sent two units, they're bringing her down now.*

Agent Smith: *No lieutenant, your men are already dead.*

The Matrix

Part II

Ethos

Ethos

- Ethos' primary purpose is to make it easy to build *robust* applications
- Ethos is a clean-slate design
- It is incompatible (with the mistakes of the past)
- It tries to avoid doing things that haven't worked in the past

It's an old habit. I spent my life trying not to be careless.

Don Corleone
The Godfather (by Mario Puzzo)

How does an OS affect application security?

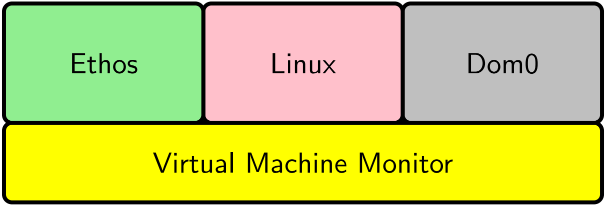
- Its part of the TCB, so its failure can destroy security
- But its impact is much more than just that
- The semantics exported by the OS determines how applications can fail
- The easiest way to see this is with a Programming Language
- A type-safe programming language cannot have buffer overflow
- Thus the system layers can have a profound impact on the types of security holes possible.
- We like to say that "Security is Semantics".

Complexity

Ethos avoids complexity to the extreme

- Because even the extreme may not enough
- One way of doing things (find the best and use that)
- Unification (make similar things look the same)
- Higher level semantics (because they fail more gracefully)
- Mindful of the pitfalls which result in security holes
- Use virtual machines for flexibility
- Modularity and information hiding
- Use declarations rather than code (because of decidability)
- Reduce cognitive load (e.g., use file system to provide privileges)

Virtual Machine Impact



- Ethos coded to one virtual machine (largely hardware independent)
- Ethos can use other OS facilities (eg. Qubes graphics)
- Your favorite OS applications can still be used
- VMs can simplify permissions and many other things

Unification examples

- Make networking very efficient so that only one networking protocol needed.
- Maximize commonality between Ethos-native and the Linux port of MinimaLT.
- The file system provides the name space for networking.
- Naming can be used to define permissions, etc.

Designed for the Internet

- Public keys are user IDs
  - Each user can have as many as they want (pseudonyms)
  - Self generated
  - Guaranteed unique (if your PRNG is not broken)
- User are added on the fly
  - With fine-grain enough authorization, this is not a problem
- Domain names
  - World-wide guaranteed unique names
  - Names which are easy to remember
- Mobile: connections are not named by their IP address/Port

Part III

Ethos Networking

Networking properties

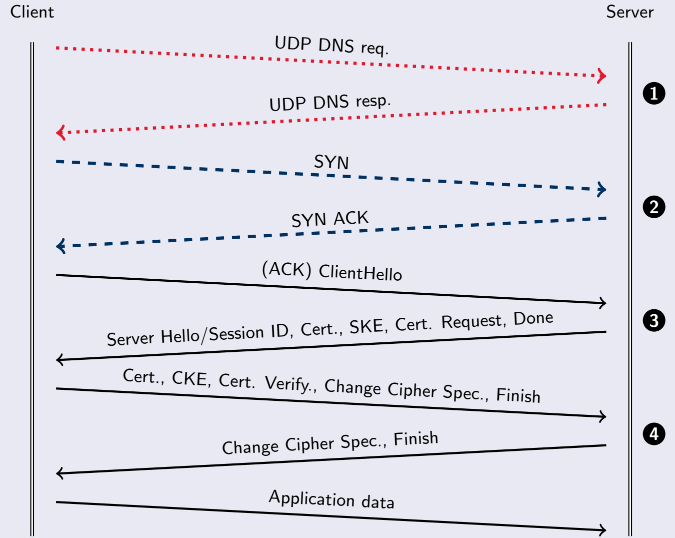
- Data on the first packet (low latency)
- All networking encrypted for confidentiality and integrity
- Ephemeral public keys used for perfect forward security
- Public key authentication of users and servers
- Tunneled to hinder traffic analysis
- Puzzles for denial-of-service protections
- Prevention of amplification attacks
- Mobile (shut down you notebook, get on a plane, open and continue connections)
- Prevent linkability of across tunnels

MinimaLT: Ethos network protocol

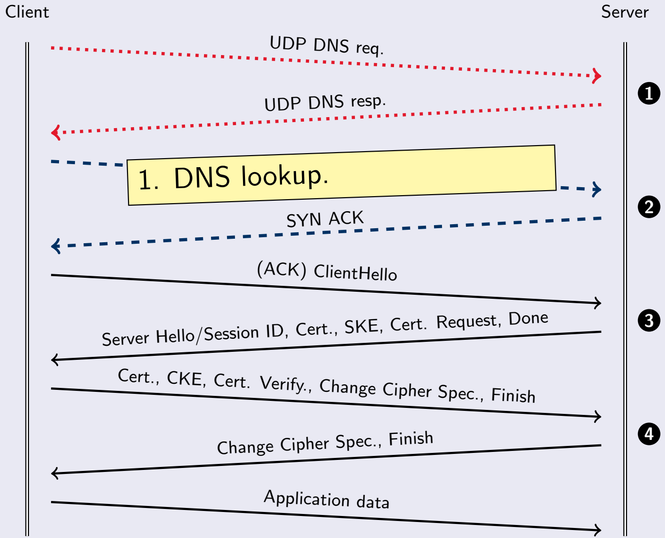
MinimaLT stands for Minimal Latency Tunneling

- ECC DH
- NaCL
- integrated with authentication servers
- implemented on Ethos and Linux

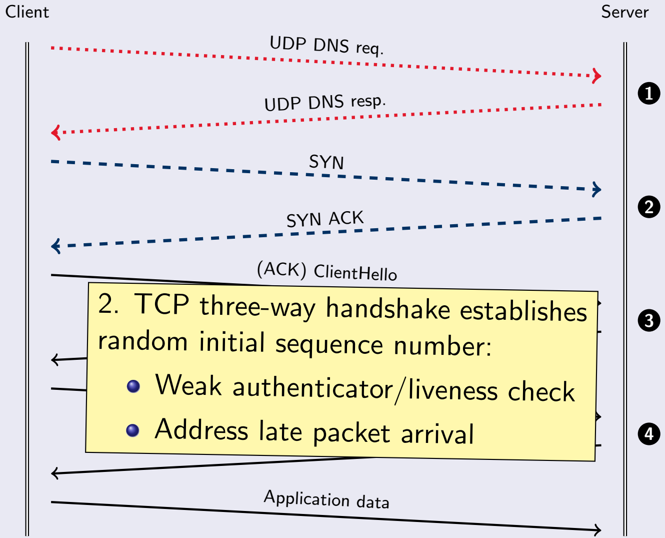
TLS: 4 round trips



TLS: 4 round trips

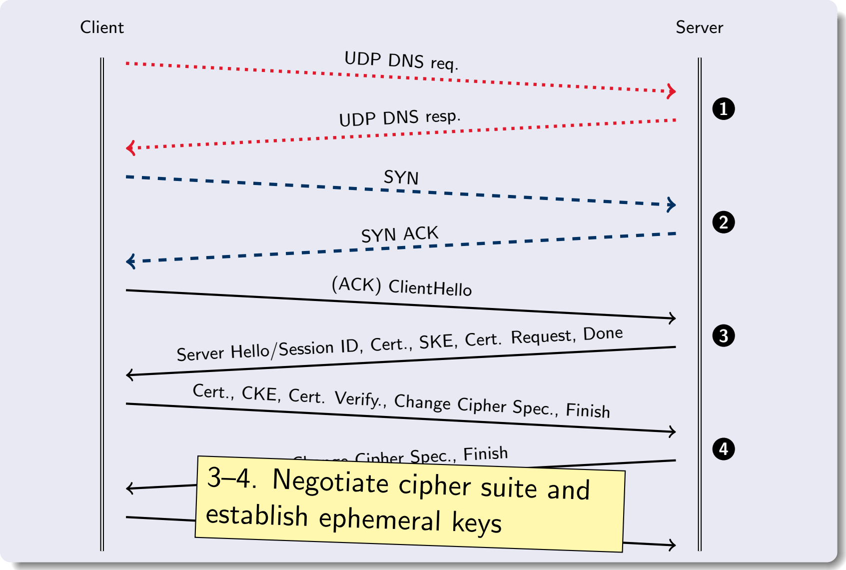


TLS: 4 round trips



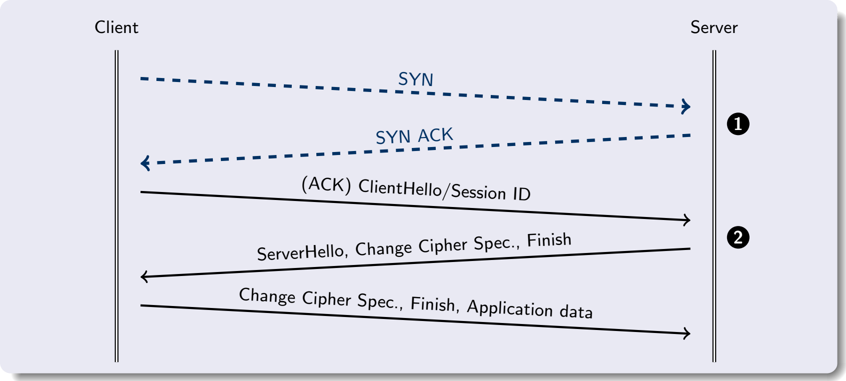
2. TCP three-way handshake establishes random initial sequence number:
- Weak authenticator/liveness check
- Address late packet arrival

TLS: 4 round trips



3–4. Negotiate cipher suite and establish ephemeral keys

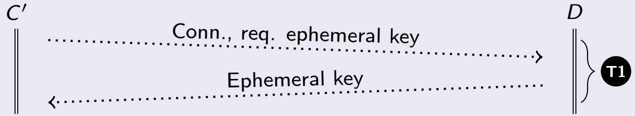
TLS (abbreviated): 2 round trips



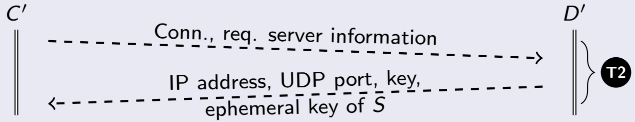
Only possible on a *reconnect*

MinimaLT round trips

Obtaining *D*'s ephemeral key (only at boot time):

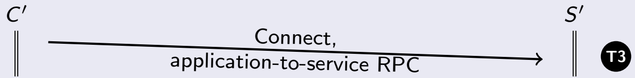


DNS-like lookup (once per host):



MinimaLT Connection Establishment

Connection establishment:



- TCP's SYN/ACK handshake unnecessary in cryptographic protocol
- One round trip for directory lookup (same as any internet protocol)
- Data goes on first packet to server in all cases
- if tunnel already established, create new connections w/i tunnel

Packet information

On the unencrypted part of the packet is

- IP and UDP for routing
- Client ephemeral public key (if tunnel initiation packet)
- Tunnel ID
- Time-based Nonce
- Encrypted payload with
  - Sequence and Acknowledge fields encrypted
  - Control fields (other than for routing) not exposed example TCP/IP RST.
  - Integrity protected

Key Rollover

- Rekeying occurs periodically (every minute)
- Key is a hash of the previous key
- Key rollover is identified by a changed tunnel ID
- Changed tunnel ID looks like a tunnel initiation packet
- Provide PFS even over long running connections

Mobility

- Tunnel is identified by a tunnel ID
- Tunnel ID is the hash of the client's ephemeral public key
- When changing IP address, simultaneously do a tunnel rekey
- Looks like a new tunnel
- Inhibits location tracking of clients

Programming

client:

```
fd = ipc("/service/messaging", "example.com")
```

server:

```
iFd = advertise("/service/messaging")
fd, user = import(iFd)
```

- Crypto is transparent
- Returns UserID to server
- Authorized by UserId
- Simpler than POSIX network APIs
- Application programmer can't screw up encryption, authentication, ...

Other issue

- Uses Sayl, a scalable authentication infrastructure
  - Scales to the Internet
  - Enables user and host authentication
  - Very efficient
- Denial of Service protections built into MinimaLT
- All information is typed, no need to
  - serialize and
  - parse
- All control messages sent by RPC
  - Start a new connection anonymously
  - Start a new connection with a pseudonym
  - Close a tunnel
  - Next tunnel ID

Part IV

Conclusion

Conclusion

- Today's widely used systems have failed under attack
- They are brittle, break disastrously
- They are unfixable, and will have to be replaced
- Need systems which are much stronger
- Ethos is designed to drive down complexity and remove pitfalls while providing strong security services
- We focused on Ethos networking here
  - Faster than unencrypted TCP/IP
  - More secure than TLS
  - Very simple