Is it Still Possible to Extend TCP?

Michio Honda, Keio University
Yoshifumi Nishida, Keio University
Costin Raiciu, Universitatea Politehnica Bucuresti
Adam Greenhalgh, University College London
Mark Handley, University College London
Hideyuki Tokuda, Keio University

IRTF Open Meeting

Keio University 1858 CALAMVS GLADIO FORTIOR

82nd IETF, 15. Nov. 2011, Taipei, Taiwan

UCL

Motivation

- Extending TCP addresses many issues in the Internet
  - Inability to use multi-homing and mobility
    - Multipath TCP uses multiple paths in the connection and supports mobility
  - Security and privacy
    - TcpCrypt encrypts all the TCP traffic

- All of these would be great, but

  IS IT STILL POSSIBLE TO EXTEND TCP?

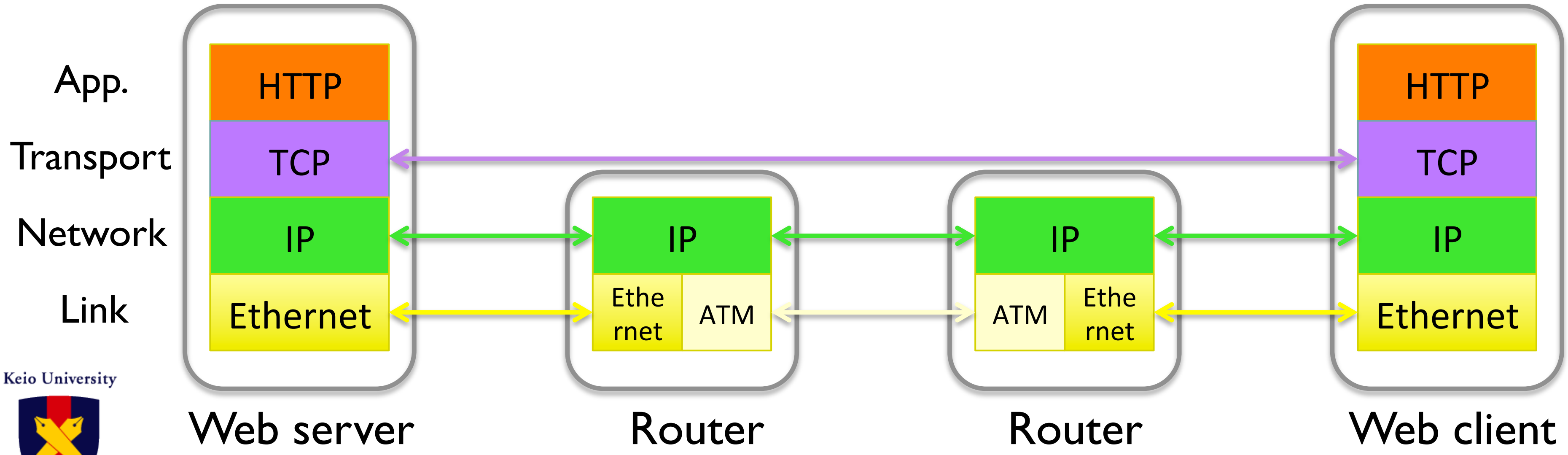
Keio University

1858

CALAMVS GLADIO FORTIOR

The Original Internet

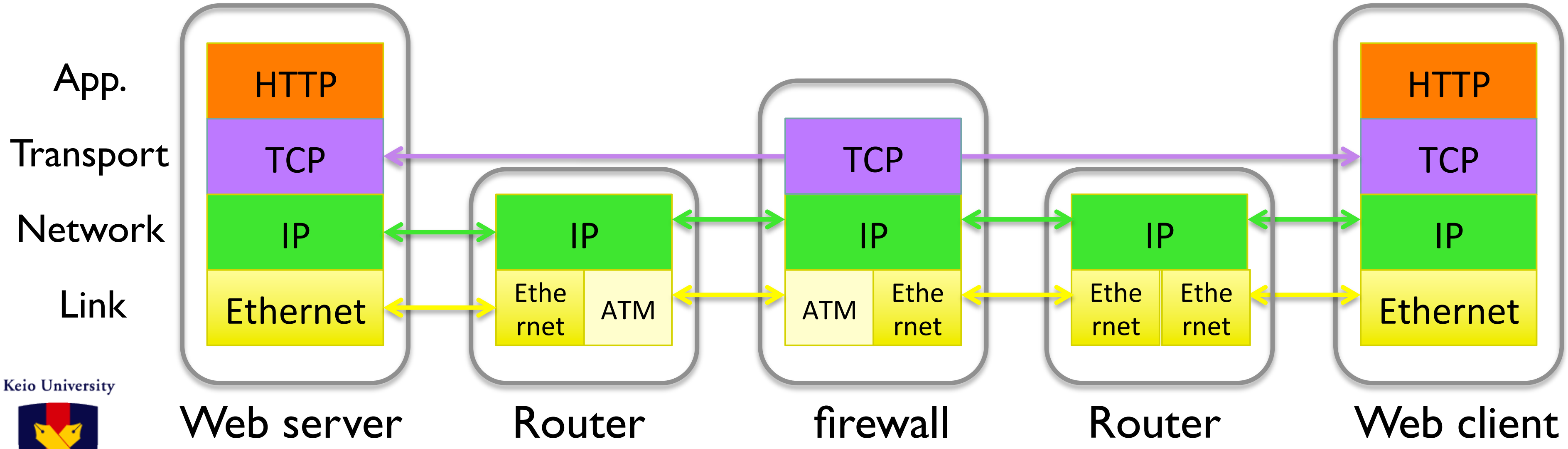
- Protocol layering
  - Routers look at IP headers to decide how to route a packet
  - TCP provides reliability via retransmission
  - Application uses OS's TCP API to do its job



Keio University
CALAMVS GLADIO FORTIOR 1858

The real Internet

- Networks look beyond the IP header
  - Middleboxes (firewalls, performance-enhancing proxies, traffic normalizer etc)
    - Optimization, security enhancement



Keio University

Extending TCP: the reality

- Extending TCP in the original Internet was easy
  - Specify what the new extension should do at end systems
- Extending TCP today is much more complicated
  - Really need to understand what middleboxes do

Problem:

We don't really know what middleboxes do in the Internet!

Keio University

1858

CALAMVS GLADIO FORTIOR

Contributions

- This paper remedies this problem
  - We run a thorough study of middlebox behaviors that are relevant for TCP extensions
  - We examined the design implications for newly proposed TCP extensions
- **IS IT STILL POSSIBLE TO EXTEND TCP?**
  - **Yes,** but we need to design very carefully

Keio University

1858

CALAMVS GLADIO FORTIOR

Measurement Methodology

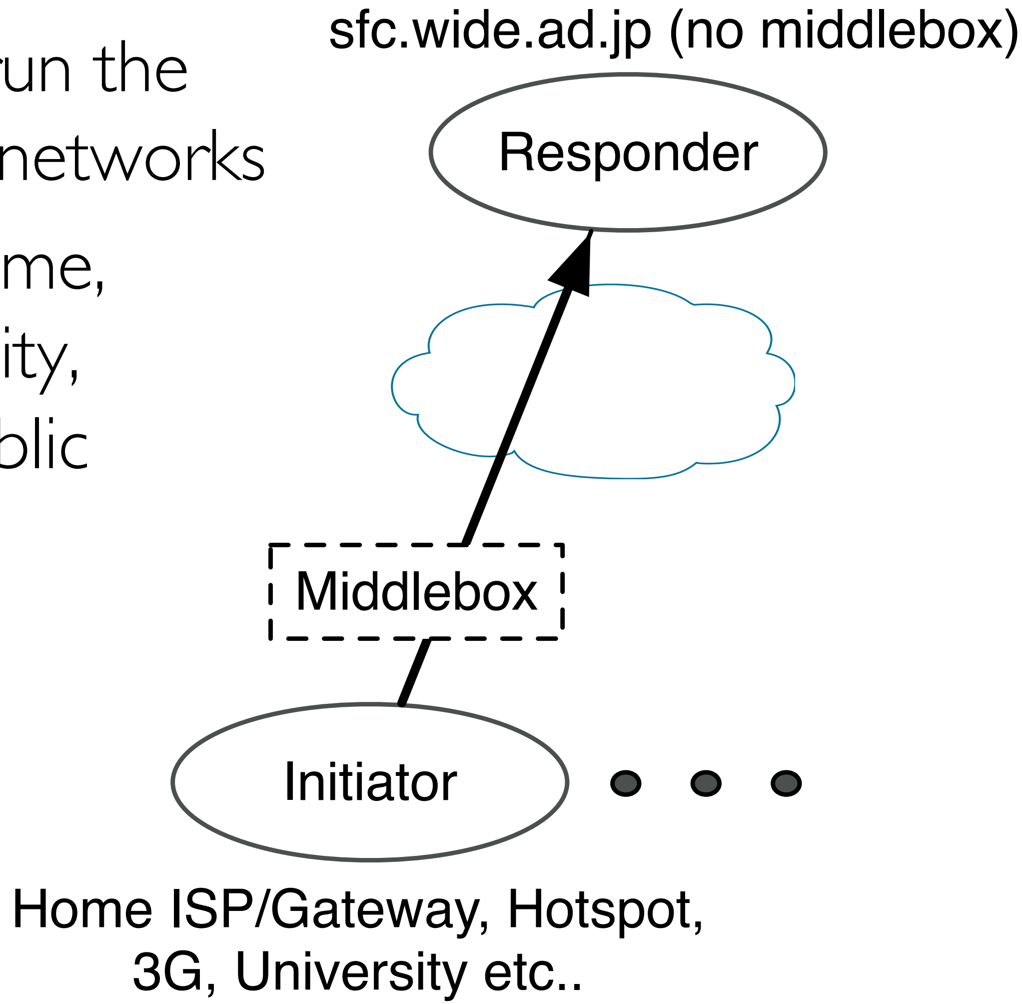
- We need to control both endpoints
  - One-sided measurements are not enough
    - Mimic TCP extensions
      - e.g., negotiating the undefined option
    - Monitor test TCP traffic at both ends
      - e.g., whether TCP options are removed
- We developed two tools that generate test TCP traffic:
  - Responder – runs at the server
  - Initiator – runs at the client

Keio University

1858 CALAMVS GLADIO FORTIOR

Experimental Setup

- The responder tool runs in the middlebox-free network
- Contributors and we run the initiator tool in access networks
- 142 paths including home, enterprise, 3G, university, hosting service and public hotspots from 24 countries



Keio University

Our Tests

- **TCP Option**
- Sequence Number Modification
- **Sequence Number Hole**
- Proxy-Acknowledgment
- Inconsistent Retransmission
- **Segment Splitting and Coalescing**
- **Intelligent NICs**

Keio University

Middlebox behaviors depend on the port number

- We performed every experiment using three server ports:
  - 80 (http)
  - 443 (https)
  - 34343 (unregistered)

Keio University

CALAMVS GLADIO FORTIOR

1858

TCP Option Tests

- Are new options removed from TCP packets?
  - Benign, but need to understand how often this happens
- Are packets with new TCP options dropped?
  - If yes, fallback to regular TCP is hard!
- If options are negotiated in the SYN exchange, will options in data segments also pass?
  - If no, we need option negotiation with data segments after that in SYN exchange

Keio University

CALAMVS GLADIO FORTIOR

1858

TCP Option Test Results

- No path dropped a packet containing the new option
  - Smooth fallback to regular TCP is possible
- 4 % of paths for port 34343, 14 % for port 80 of paths removed the SYN option
  - For port 80, ubiquitous deployment of TCP extensions might be hard
- All the paths removing options from data segments also removed the SYN option
  - SYN option exchange is reasonable to test if options in data segments pass

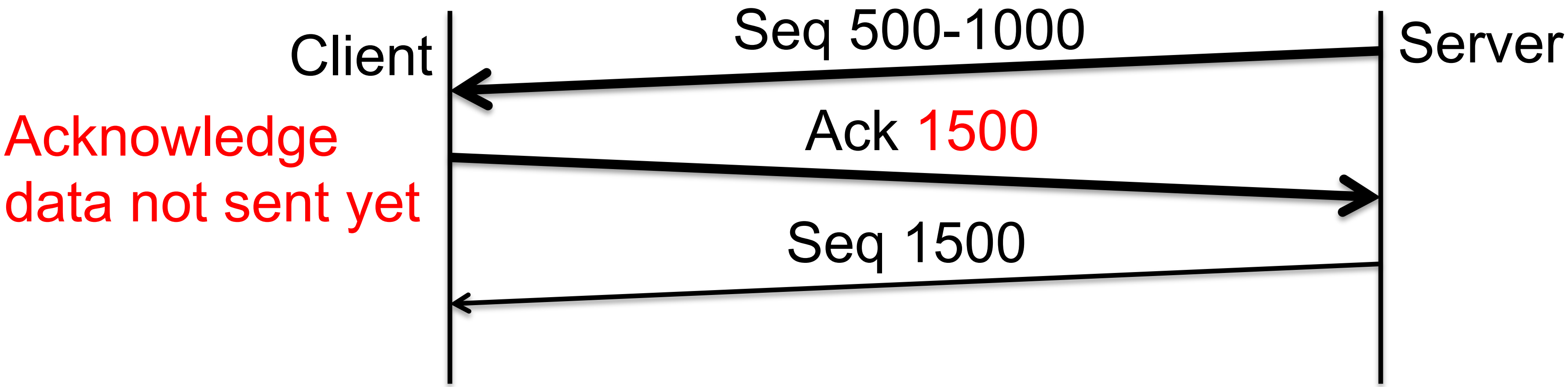
Keio University

How TCP options are removed

- For port 34343, 5 out of 6 paths just “eliminated options”
- For port 80, 16 out of 20 paths exhibited “proxy” behavior
  - The middlebox generates SYN/ACK, and options are removed as a side-effect
  - Transparently split the TCP connection
    - Split/coalesce segments, indirectly ack packets, cache segments
    - We found more such paths in 3G networks

Sequence Number Hole Test

- Can we leave holes in sequence number consistency?
  - How to skip retransmission of stale data? (e.g., for VoIP)



- Leaving sequence holes is a bad idea
  - 24% of paths (28% on port 80) do strange things

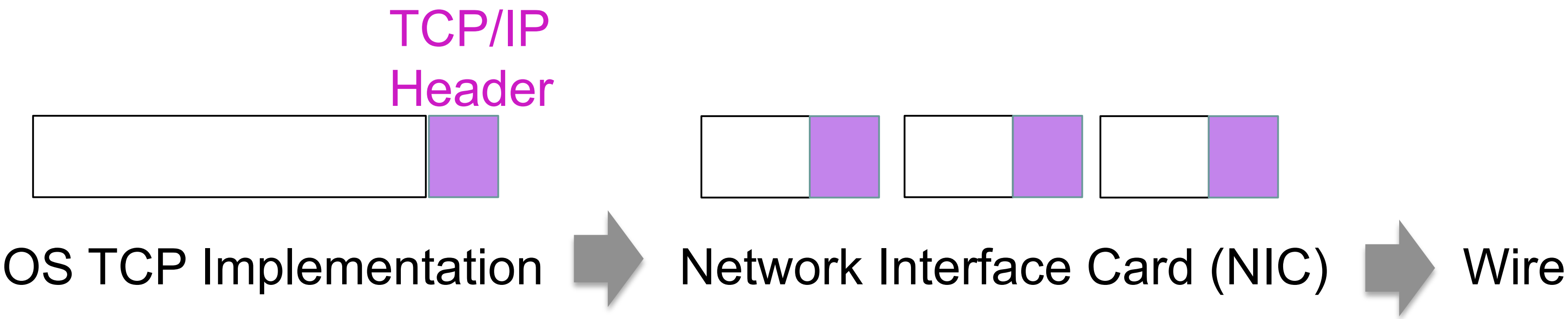
Segment Splitting/Coalescing Test

- Can we assume that segments arrive without being resegmented?
  - TCP gives a bytestream to apps
  - But TCP extensions often assume segments are not resegmented
    - TcpCrypt, MD5
- 1- 7 % of paths split/coalesce segments
  - Mostly proxies (one is not, but it does not split/coalesce when options are present)
- If options are passed in SYN exchange, we can assume packets containing options are not resegmented

Keio University
CALAMVS GLADIO FORTIOR
1858

Intelligent NICs

- TCP Segmentation Offload (TSO) is very common – "middlebox" resegmenting packets

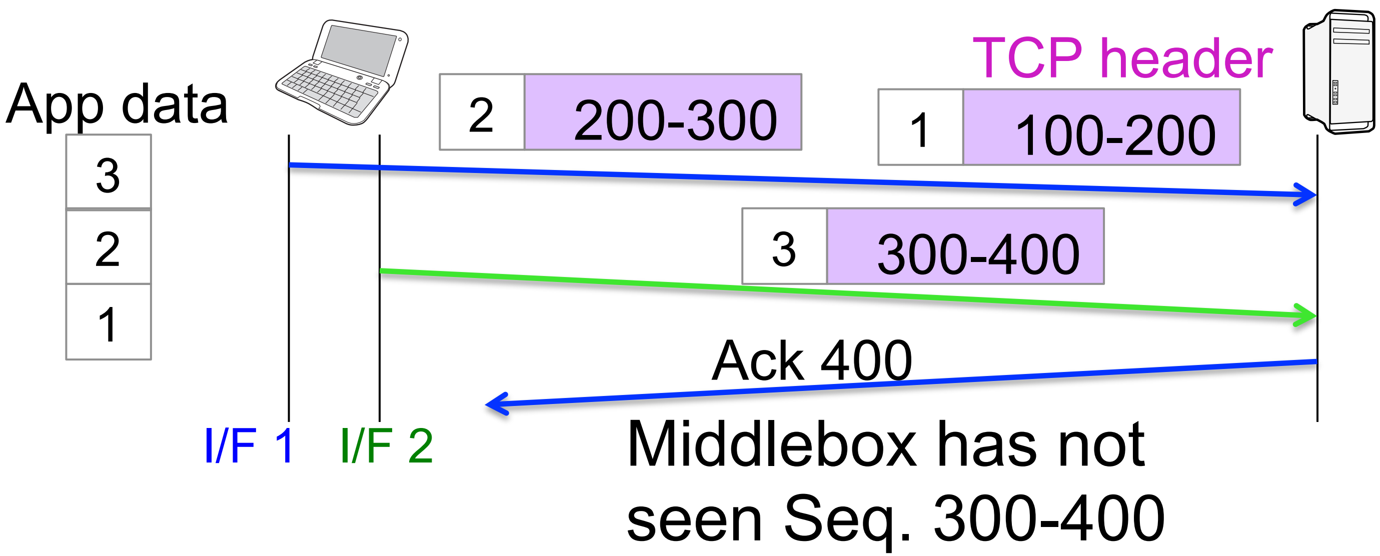


- How TCP options are treated?
- We tested 12 NICs from 4 vendors
- All the NICs copied the option to all the split segments
  - Duplicated options must be assumed

Keio University

Design Implications: Multipath TCP

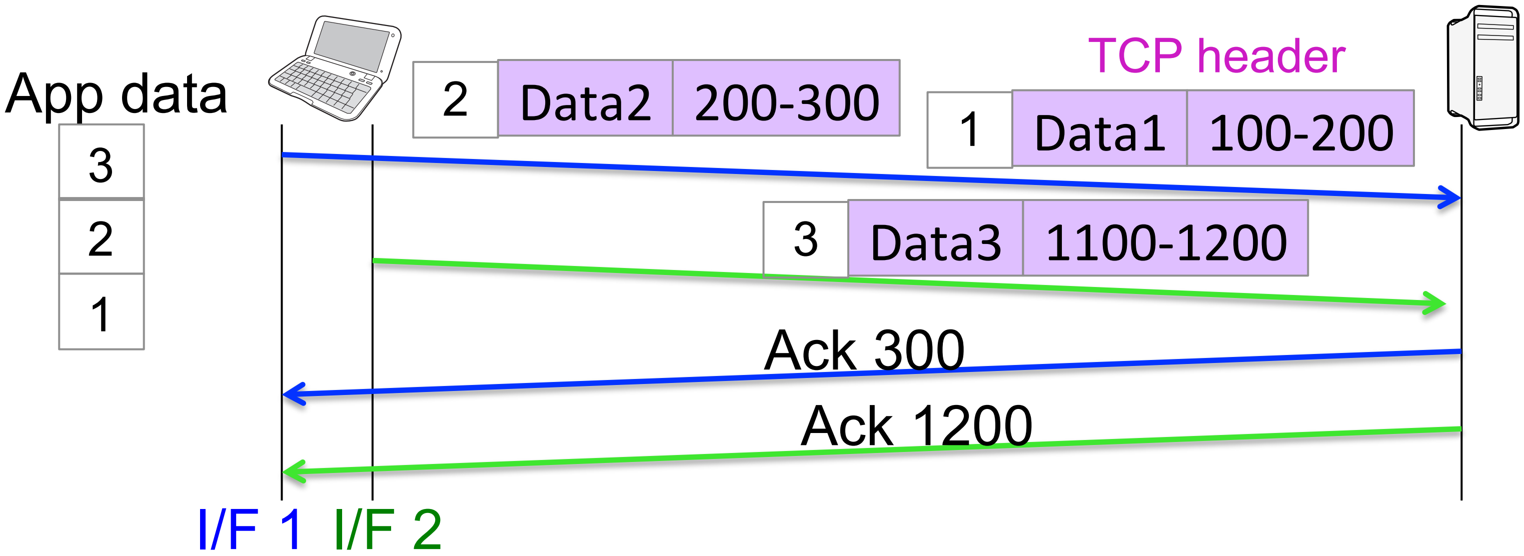
- Transmit segments across more than one path
  - Multi-homed nodes are common
- What sequence numbers should we use?
  - Option 1: reuse regular TCP sequence numbers across multiple paths



- We cannot leave a hole in sequence space on each path

Design Implications: Multipath TCP

- Need separate sequence spaces for "on-path" and "app data"
  - Regular sequence numbers are used for on-path sequence numbers
  - App data sequence number is embedded in the option



Design Implications: TcpCrypt

- TcpCrypt operation:
  - Negotiate keys
  - Encrypt payload of each segment
  - Add TCP option with MAC
- TcpCrypt is safe, but performance penalty
  - Resegmentation doesn't happen after options are negotiated
  - TCP Segmentation Offload (TSO) must be disabled

Design Implications: TCP Extended Option Space

- TCP Extended Option Space:
  - Use a part of payload as an extended option space
  - An option in the TCP header specifies "actual" data offset
- TCP Extended Option Space is unsafe
  - we cannot use different options on retransmissions

  - TSO copies the option specifying the actual data offset to all the split segments

Keio University

Design Guidelines for TCP Extensions

- Negotiate new features on the SYN exchange before use
- If options are removed, don't assume message boundaries will be preserved
- Assume segments will be split by TSO and options duplicated on these segments

Design Guidelines for TCP Extensions

- Don't assume sequence numbers arrive unmodified
  - i.e. Initial Sequence Number randomization
- Don't leave gaps in the sequence space
- Retransmitting inconsistent information is a bit risky
- Proxies are common, especially on port 80, and will strip TCP options, and resegment packets

Conclusion

- **We can still extend TCP, but extensions' design is very constrained**
- Future work
  - New tests, wider coverage
- Please join the measurement!
  - http://www.sfc.wide.ad.jp/~micchie/middlebox/cfc.html
  - Download self-contained python script, run one command, and post a result (15 min. experiment)!
  - You will see the summary of results after posting log files!

Keio University

CALAMVS GLADIO FORTIOR 1858