# Wrangling a Large Herd of Internet Clocks

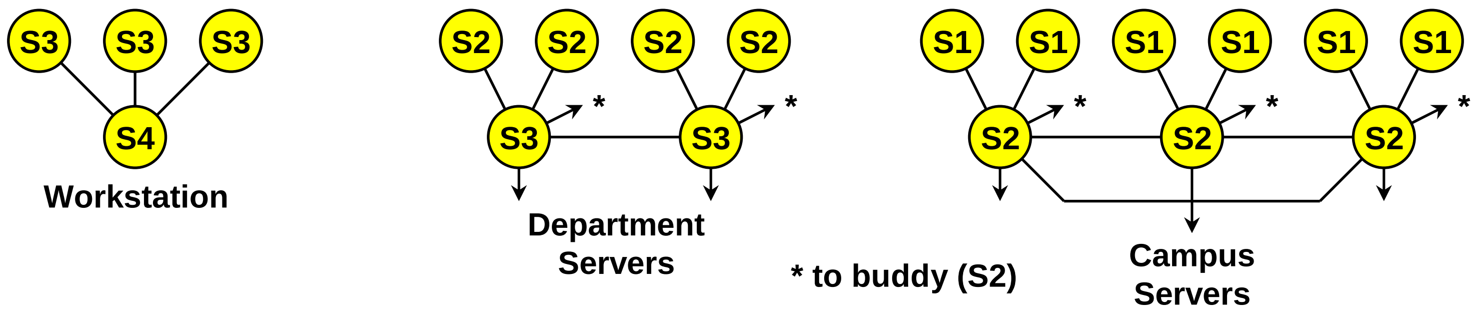
DIMACS Workshop
27 October 1997

David L. Mills
University of Delaware
mills@udel.edu

HTML, PostScript and PowerPoint versions of this presentation are available at http://www.eecis.udel.edu/~mills

Sir John Tenniel; *Alice's Adventures in Wonderland,* Lewis Carroll

# Introduction

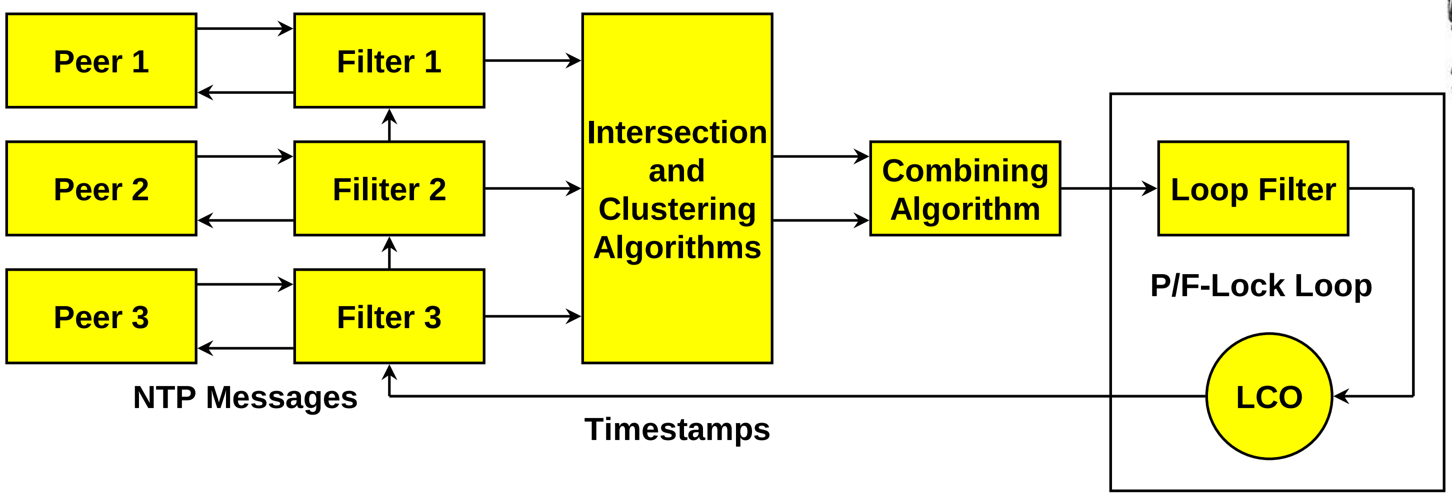



- Network Time Protocol (NTP) synchronizes clocks of hosts and routers in the Internet
- Provides submillisecond accuracy on LANs, low tens of milliseconds on WANs
- Unix NTP daemon ported to almost every workstation and server platform available today - from PCs to Crays
- Well over 100,000 NTP peers deployed in the Internet and its tributaries all over the world

# Goals

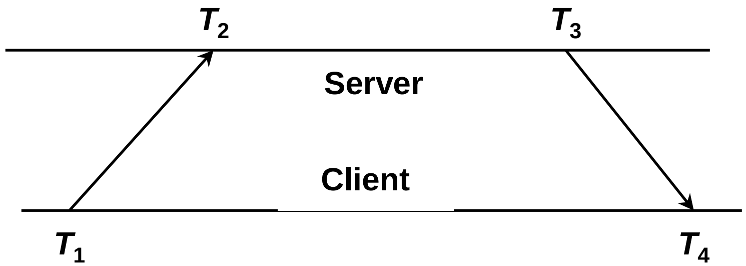
- Robustness to many and varied kinds of failure, including Byzantine disagreements, malicious attacks and implementation bugs.
  - Our approach is based on diverse network paths, redundant servers and a suite of intricately crafted mitigation algorithms.
- Autonomous server and client configuration to optimize performance under resource constraints.
  - Our approach is based on Internet multicasting and manycasting, together with engineered drop-add heuristics.
- Autonomous authentication using a combination of public-key and private-key cryptography.
  - Our approach uses automatically generated and managed keys with controlled lifetimes and engineered algorithms designed to avoid loss of accuracy due to encryption delays.

# NTP architecture



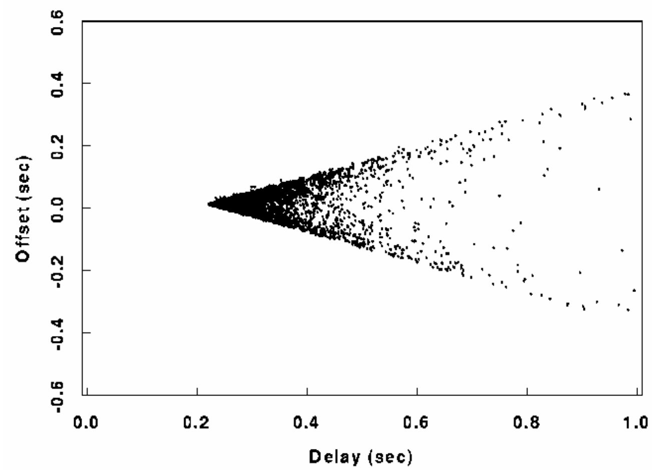

- Multiple synchronization peers for redundancy and diversity
- Clock filters select best from a window of eight clock offset samples
- Intersection and clustering algorithms pick best subset of peers and discard outlyers
- Combining algorithm computes weighted average of offsets for best accuracy
- Loop filter and local clock oscillator (LCO) implement hybrid phase/frequency-lock feedback loop to minimize jitter and wander

# Clock filter algorithm



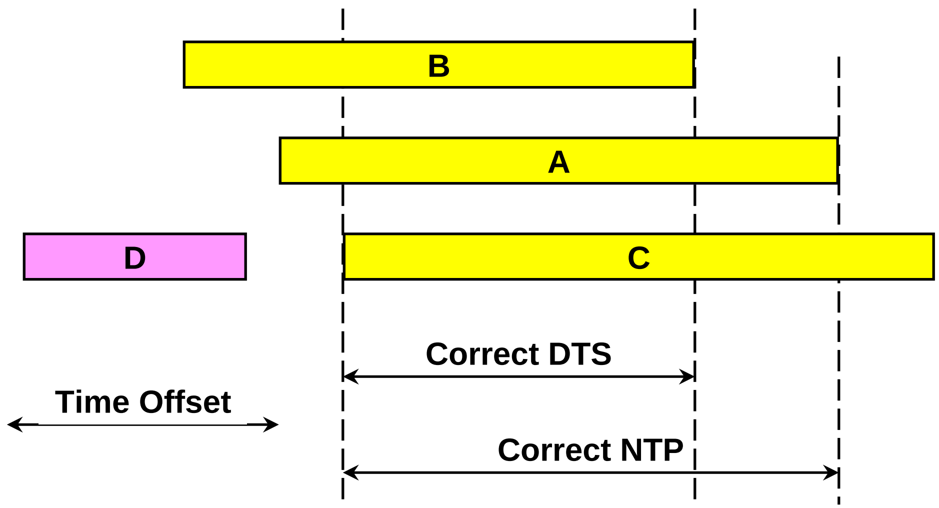

Offset $\theta = \frac{1}{2}[(T_2 - T_1) + (T_3 - T_4)]$

Delay $\delta = (T_4 - T_1) - (T_3 - T_2)$




- Most accurate clock offset $\theta$ is measured at the lowest delay $\delta$ (apex of the wedge diagram)
- Phase dispersion $\varepsilon_r$ is weighted average of offset differences over last eight samples - used as error estimator
- Frequency dispersion $\varepsilon_f$ represents clock reading and frequency tolerance errors - used in distance metric
- Synchronization distance $\lambda = \varepsilon_f + \delta/2$ - used as distance metric and maximum error bound, since correct time $\theta_0$ must be in the range $\theta - \lambda \leq \theta_0 \leq \theta + \lambda$

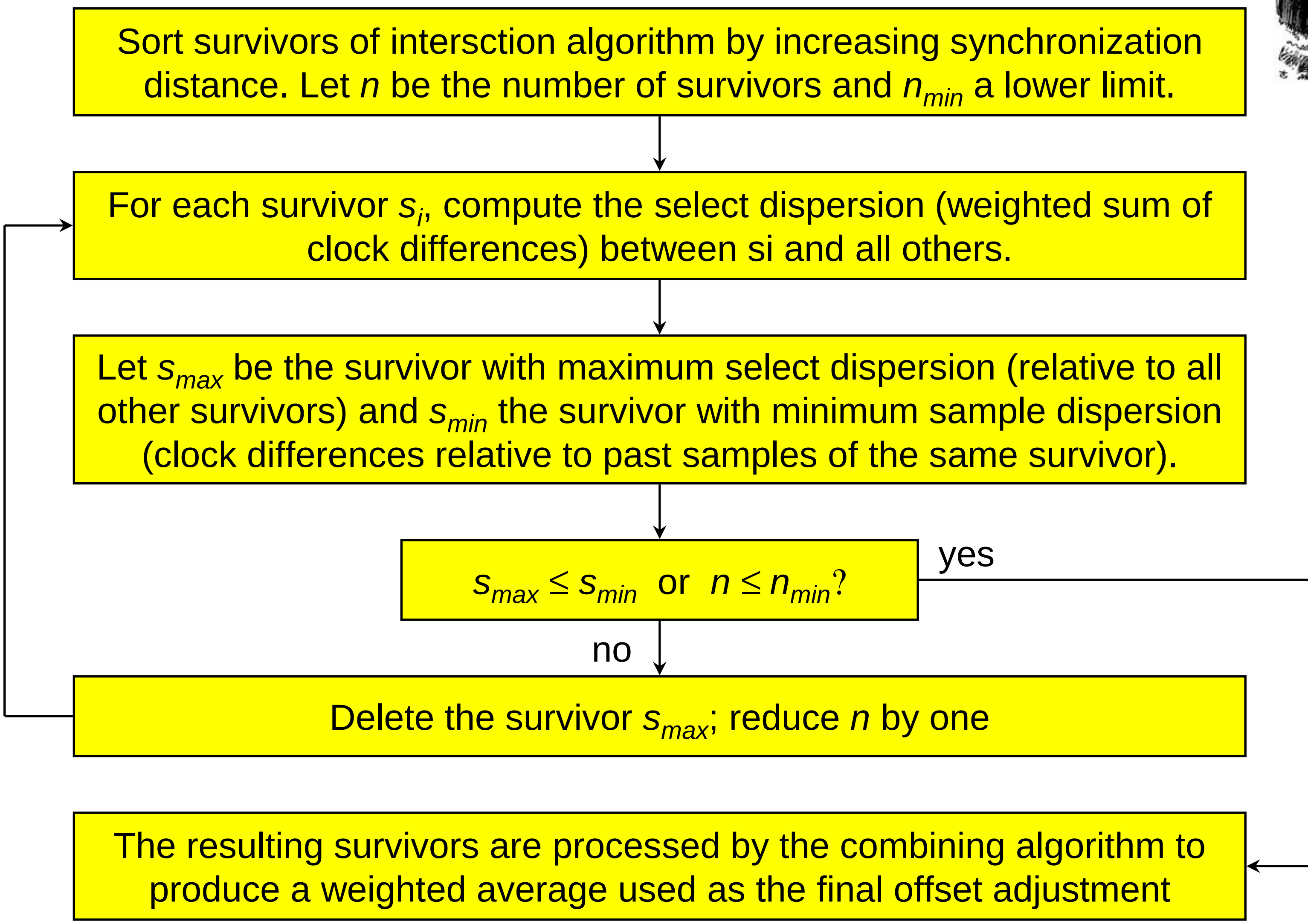
# Intersection algorithm




**m = number of clocks**
**f = number of presumed falsetickers**
**A, B, C are truechimers**
**D is falseticker**

- Initially, set falsetickers *f* and counters *c* and *d* to zero
- Scan from far left endpoint: add one to *c* for every lower endpoint, subtract one for every upper endpoint, add one to *d* for every midpoint
- If $c \geq m - f$ and $d \geq m - f$, declare success and exit procedure
- Do the same starting from the far right endpoint
- If success undeclared, increase *f* by one and try all over again
- if $f \leq m/2$, declare failure

# Clustering algorithm



# NTP autonomous configuration - approach

- Dynamic peer discovery schemes
  - Primary discovery vehicle using NTP multicast and manycast modes
  - Augmented by DNS, web and service location protocols
  - Augmented by NTP subnet search using standard monitoring facilities
- Automatic optimal configuration
  - Distance metric designed to maximize accuracy and reliability
  - Constraints due to fanout limitations and maximum distance
  - Complexity issues require intelligent heuristic
- Candidate optimization algorithms
  - Multicast mode with or without initial propagation delay calibration
  - Manycast mode with administrative and/or TTL delimited scope
  - Distributed, hierarchical, greedy add/drop heuristic

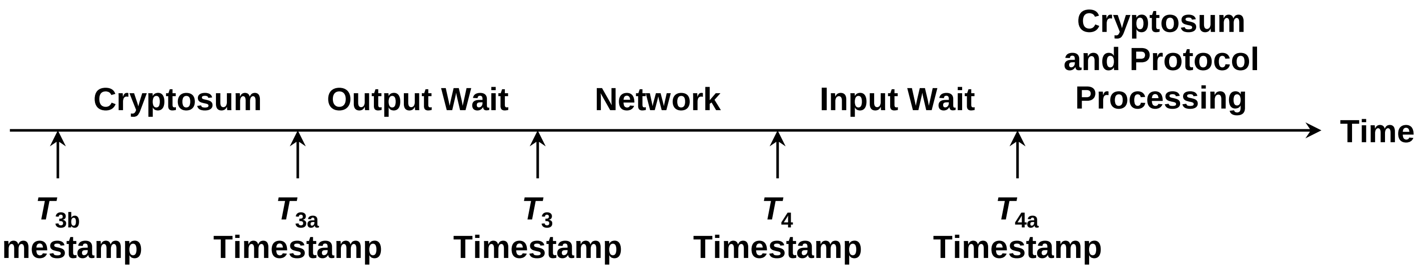
# NTP configuration scheme

- Multicast scheme (moderate accuracy)
  - Servers flood local area with periodic multicast response messages
  - Clients use client/server unicast mode on initial contact to measure propagation delay, then continue in listen-only mode
- Manycast scheme (highest accuracy)
  - Initially, clients flood local area with a multicast request message
  - Servers respond with multicast response messages
  - Clients continue with servers as if in ordinary configured unicast client/server mode
- Both schemes require effective implosion/explosion controls
  - Expanding-ring search used with TTL and administrative scope
  - Excess network traffic avoided using multicast responses and rumor diffusion
  - Excess client/server population controlled using NTP clustering algorithm and timeout garbage collection

# NTP autonomous authentication - approach

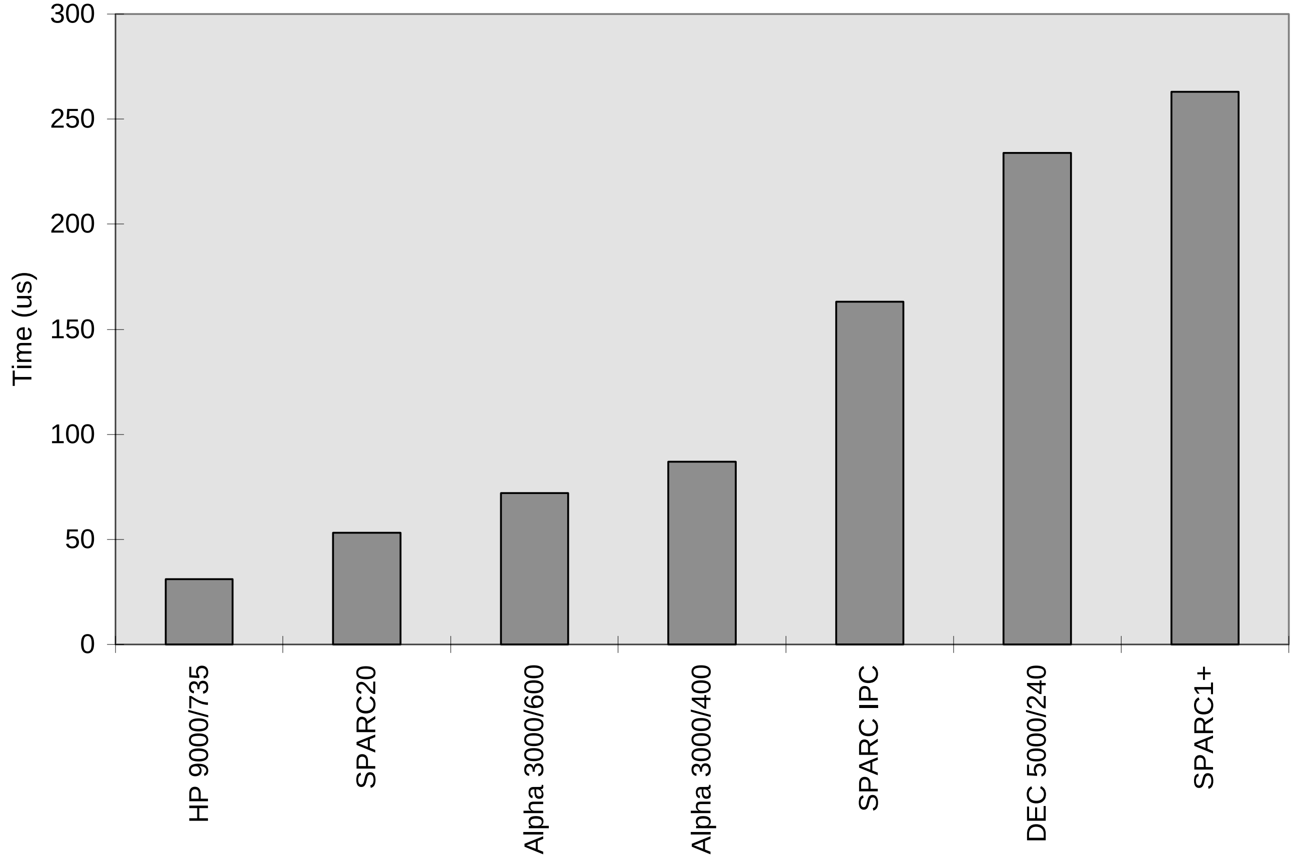
- The circular dilemma:
  - Cryptographic keys must not endure beyond enforced lifetimes
  - Enforced lifetime requires secure timekeeping
  - Secure timekeeping requires cryptographic authentication
- Authentication and synchronization protocols work independently for each peer, with each allowed to reach a tentative outcome
- When both authentication and synchronization are complete, the peer is admitted to the population used to synchronize the system clock
- Complicating this scheme are requirements that the lifetimes of all public keys, including those used to sign certificates, must be enforced as well
- However, the Achilles heel using public-key cryptography is that it is too slow for good timekeeping

# Message propagation time budget



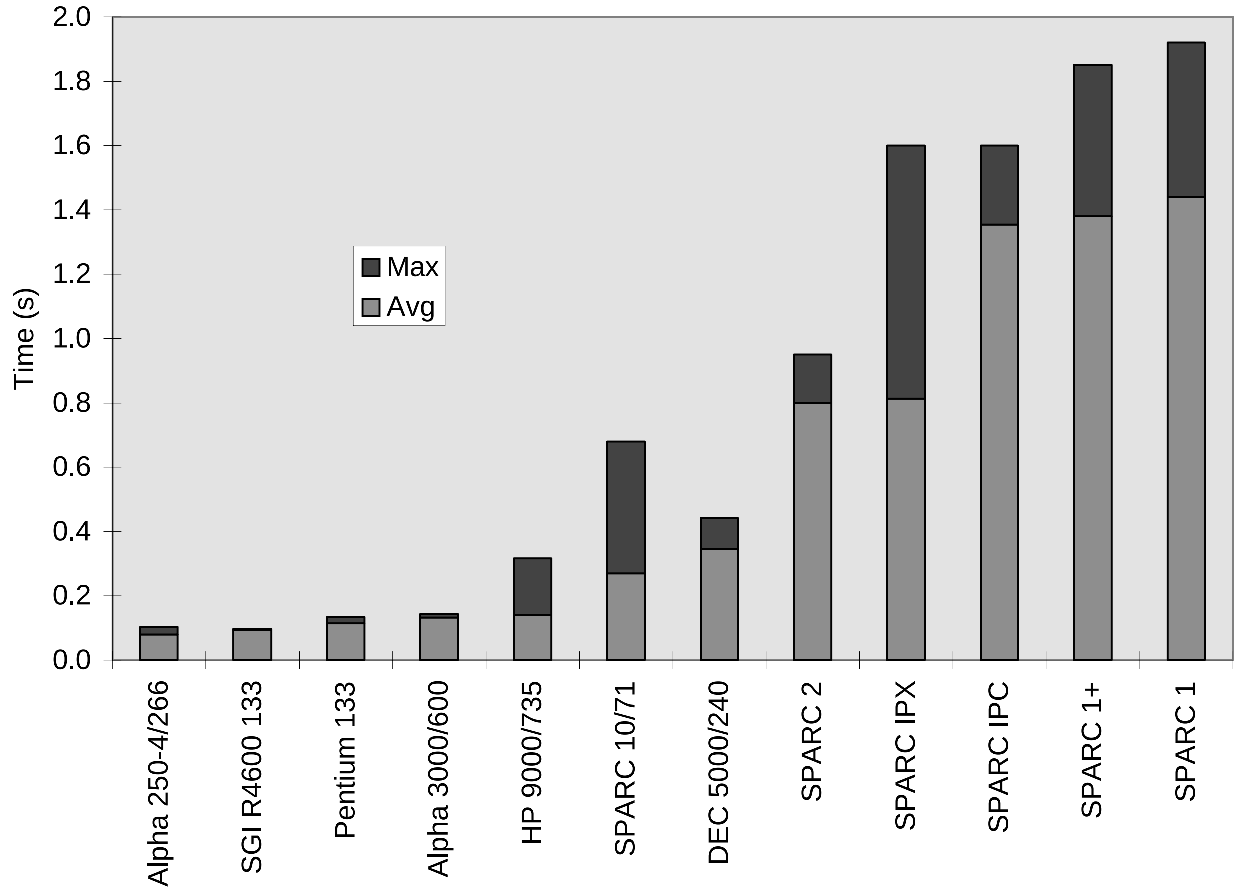

- We want $T_3$ and $T_4$ timestamps for accurate network calibration
  - If output wait is small, $T_{3a}$ is good approximation to $T_3$
  - $T_{3a}$ can't be included in message after cryptosum is calculated, but can be sent in next message; use $T_{3b}$ as best approximation to $T_3$
  - $T_4$ captured by most network drivers at interrupt time; if not, use $T_{4a}$ as best approximation to $T_4$
- Largest error is usually output cryptosum
  - Private-key algorithms (MD5, DES-CBC) running times range from 10 μs to 1 ms, depending on architecture, but can be predicted fairly well
  - Public-key algorithms (RSA) running times range up to 100 ms, depending on architecture, but are highly variable and depend on message content

## MD5 message digest



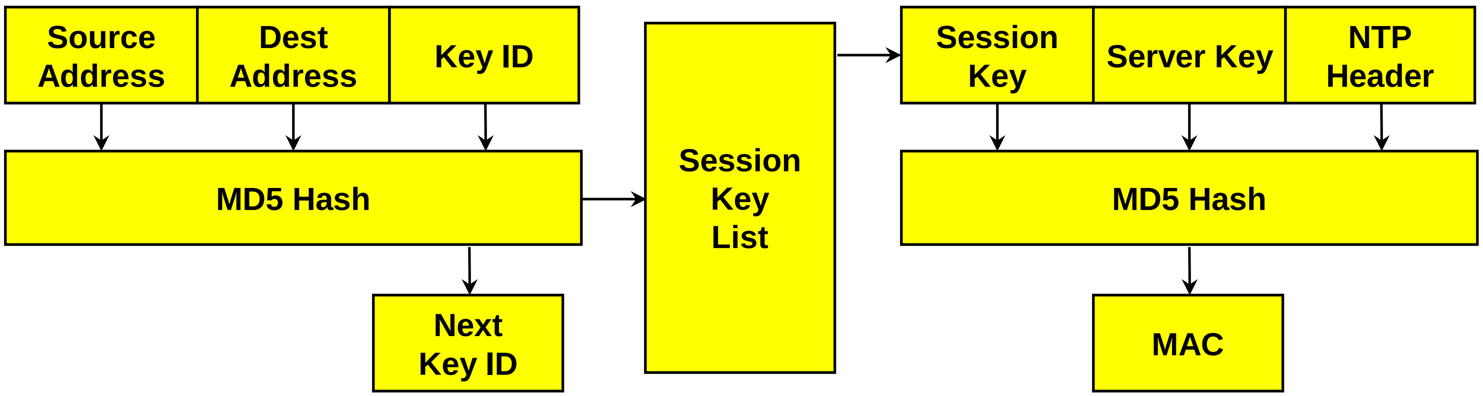

- Measured times to construct 128-bit hash of 48-octet NTP header using MD5 algorithm in RSAREF

## MD5/RSA digital signature




- Measured times (s) to construct digital signature using RSAREF
- Message authentication code constructed from 48-octet NTP header hashed with MD5, then encrypted with RSA 512-bit private key

# NTP authentication scheme




- Session keys are generated using IP addresses and key identifiers
- Initial key identifier is random; each succeeding one is hashed from the previous one
- Session key list is used in reverse order; clients verify hash of current session key matches most recent session key identifier
- At intervals, a server generates a random server key and generates a public value by encrypting it with RSA
- When the server key changes, clients obtain and decrypt the public value and verify it matches the server key

# NTP Version 4 current progress and status

- NTP Version 4 architecture and algorithms implemented and in test
  - Simple NTP (SNTP) Version 4 specification now an Internet draft
  - Improved local clock model now standard NTP feature
  - Precision time kernel modifications now in Digital Unix 4.0 and Sun Solaris 2.6
- Autonomous configuration
  - Multicast server discovery now standard NTP feature
  - Manycast server discovery implemented and in test
  - Distributed add/drop greedy heuristic designed and simulated
  - Span-limited, hierarchical multicast groups using NTP distributed mode and add/drop heuristics under study
- Cryptographic authentication
  - Autokey scheme implemented and in test
  - Public-key certificate discovery and verification scheme expected to follow IETF model

# Future plans

- Complete NTP Version 4 protocol testing and validation project
  - Deploy, test and evaluate NTP Version 4 daemon in local network
  - Deploy and test in DARPA testbeds (DARTnet and CAIRN)
  - Deploy and test at friendly sites in the US, Europe and Asia
- Prosecute standards agendae in IETF, ANSI, ITU, POSIX
  - Revise the NTP formal specification and launch on standards track
  - Participate in deployment strategies with NIST, USNO, others
- Develop scenarios for other applications such as web caching, DNS servers and other multicast services

# NTP online resources

- Internet (Draft) Standard RFC-1305 Version 3
  - Simple NTP (SNTP) Version 4 specification RFC-2030
  - Designated SAFEnet standard (Navy)
  - Under consideration in ANSI, ITU, POSIX
- NTP web page http://www.eecis.udel.edu/~ntp
  - NTP Version 3 release notes and HTML documentation
  - List of public NTP time servers (primary and secondary)
  - NTP newsgroup and FAQ compendium
  - Tutorials, hints and bibliographies
- NTP Version 3 implementation and documentation for Unix, VMS and Windows
  - Ported to over two dozen architectures and operating systems
  - Utility programs for remote monitoring, control and performance evaluation