Verifying Confidentiality and Authentication in Kerberos 5

Joint work with Frederic Butler, Aaron Jaggard, and Andre Scedrov

(Adapted from original slides by Aaron Jaggard)

Iliano Cervesato iliano@itd.nrl.navy.mil

ITT Industries, inc @ NRL Washington, DC

http://theory.stanford.edu/~iliano

Outline

- MSR
- Kerberos 5
  - Main exchange
  - MSR 2.0 formalizations
- Proof method
  - Rank / corank functions
  - General approach
- Verification of Kerberos 5
  - Authentication properties
  - Anomalies

MSR Facts and States

Fix a first order signature for the protocol

- Types

  princ, msg, shK A B, …

- Term

  $t \ ::= \ a \mid x \mid f(t_1, \ldots, t_n)$

- Fact

  $F \ ::= \ P(t_1, \ldots, t_n)$

  - Predicate describes network, intruder knowledge, internal states, or stored data

- State

  - Multiset of facts

MSR Rules

- Transition rule

  ρ: $C_1, \ldots, C_i; F_1, \ldots, F_j \rightarrow \exists x_1 \ldots \exists x_m. \; G_1, \ldots, G_k$

  - Check constraints $C_1, \ldots, C_i$
  - Check that $F_1, \ldots, F_j$ contained in state
  - Obtain next state by
    - deleting $F_1, \ldots, F_j$
    - adding $G_1, \ldots, G_k$ with fresh symbols in place of the x_i
  - Free variables in rule universally quantified
- Trace
  - Sequence of states with M_{i+1} obtained from M_i via some rule ρ_i

Verification and MSR

MSR is a specification framework

- Open-ended
- Method-independent

- Tested approaches
  - CIL connectors
    - NPA, Maude, PVS, …
  - Model checking
    - Bozzano & Delzanno
      - Affine version of MSR 1.0
  - Theorem proving
    - This work (no automation)

The Kerberos Project

- Try MSR on a real world protocol
  - Kerberos 5
    - Will MSR scale up?
- Experiment with specification techniques
  - Multi-level specification
- Attempt verification on MSR specification
  - Variant of Paulson's inductive technique
- Formalize Kerberos 5
  - Precise statement of protocol
  - Identify and formalize protocol goals
  - Prove whether goals achieved by protocol
    - Note any anomalous behavior

Previous Work on Kerberos

- Kerberos 4
  - Analyzed using “inductive approach”
    - Bella & Paulson

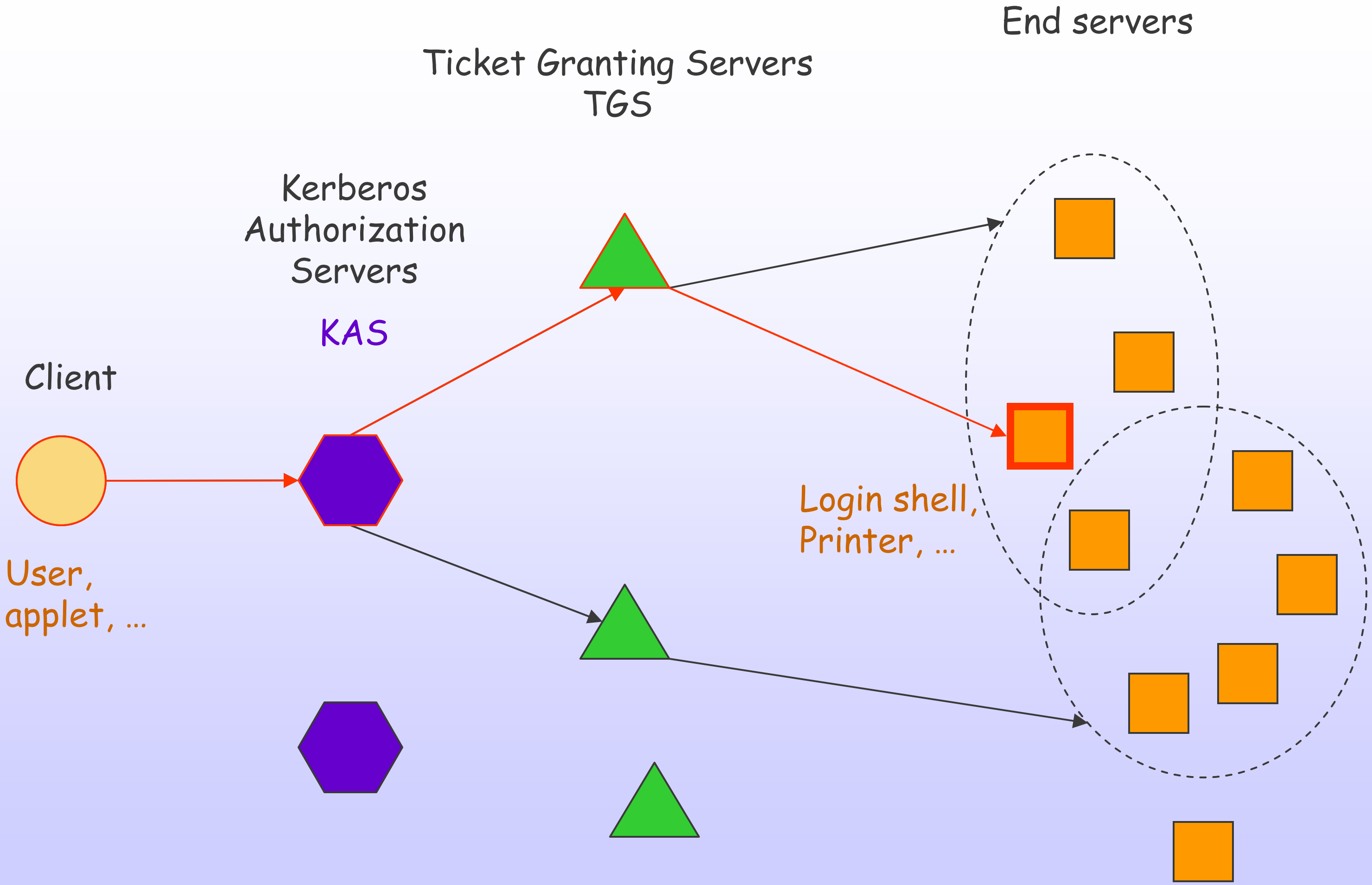
      Use Isabelle/HOL theorem prover
- Kerberos 5
  - Simplified version analyzed
    - Mitchell, Mitchell & Stern

      Use Murφ model checker

Achievements

- Formalizations of fragments of Kerberos 5
  - Three levels of formalizations (2 shown here)
  - Minimal adjustments to MSR 2.0's definition
    - Robust formalism
- Formal analysis of protocol
  - Proofs of protocol properties
    - Rank and corank functions
    - Properties and proofs show parallels between abstract and detailed formalizations
  - Curious behaviors observed
  - MSR 2.0 supports verification
    - Flexible formalism
- Interactions with Kerberos designers

Kerberos Principals



End servers

Ticket Granting Servers
TGS

Kerberos
Authorization
Servers

KAS

Client

User,
applet, …

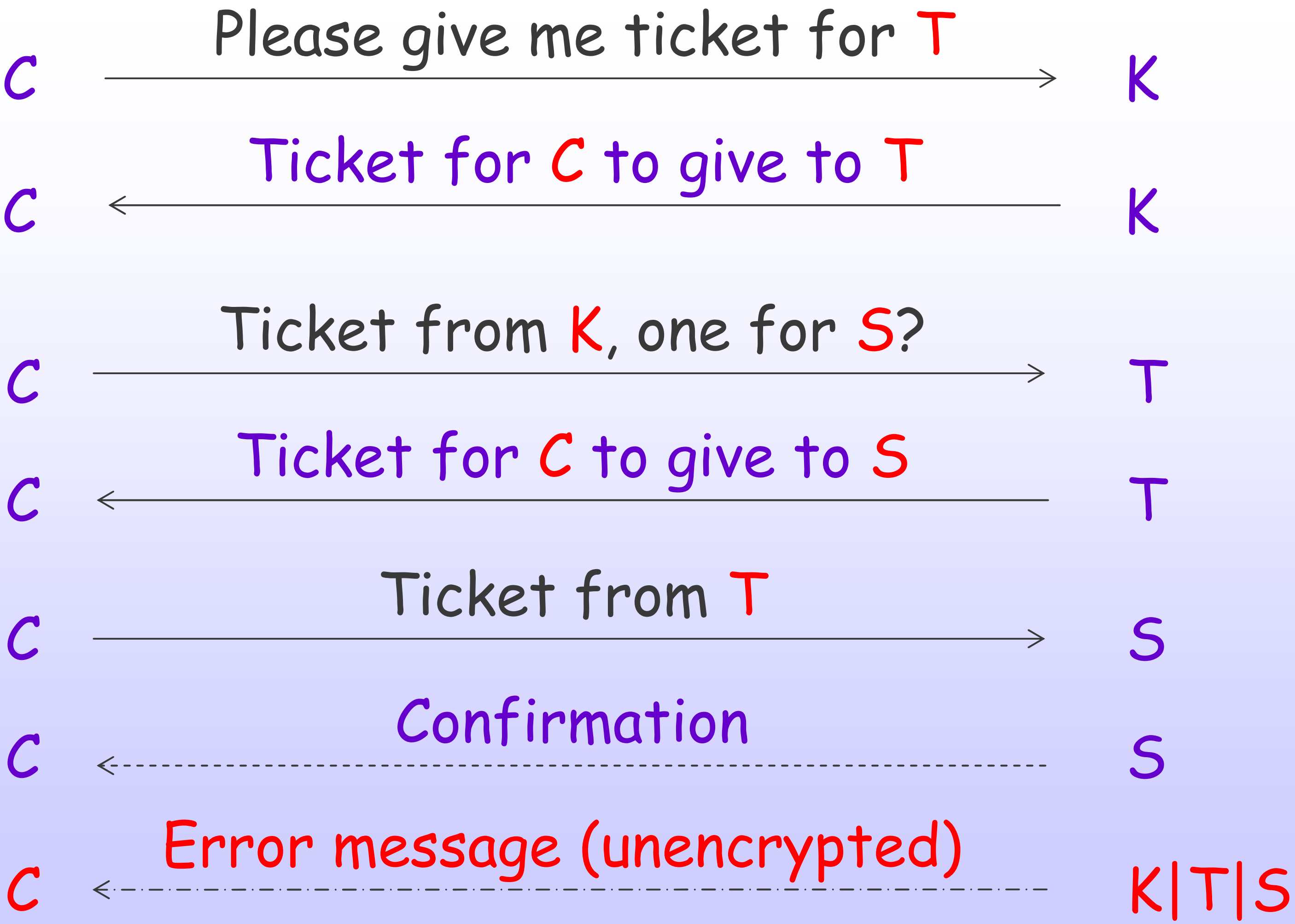
Login shell,
Printer, …

Kerberos 5

Repeatedly authenticate client C to server S

1. C obtains long term (eg, 1 day) ticket from KAS
   - Makes use of C's long term key
   - Ticket encrypted – unreadable by C
2. C obtains short-term (eg 5 min) ticket from TGT
   - Based on long term ticket from KAS
   - C sends this ticket to S

Main Kerberos Exchange



C — Please give me ticket for T → K

C ← Ticket for C to give to T — K

C — Ticket from K, one for S? → T

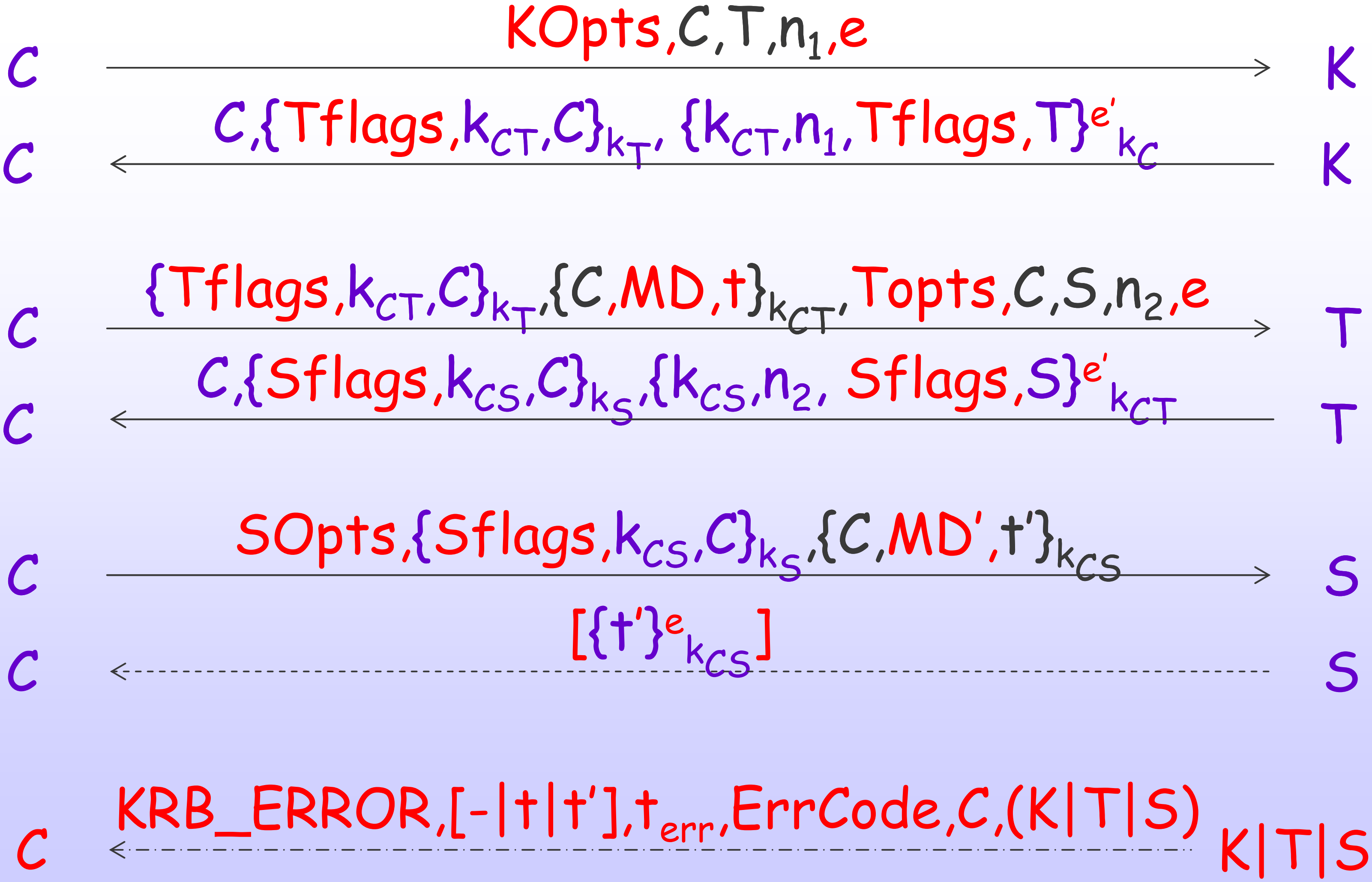
C ← Ticket for C to give to S — T

C — Ticket from T → S

C ← Confirmation — S

C ← Error message (unencrypted) — K|T|S

Abstract and Detailed Messages



$$C \xrightarrow{\text{KOpts},C,T,n_1,e} K$$

$$C \xleftarrow{C,\{\text{Tflags},k_{CT},C\}_{k_T},\ \{k_{CT},n_1,\text{Tflags},T\}^{e'}_{k_C}} K$$

$$C \xrightarrow{\{\text{Tflags},k_{CT},C\}_{k_T},\{C,\text{MD},t\}_{k_{CT}},\text{Topts},C,S,n_2,e} T$$

$$C \xleftarrow{C,\{\text{Sflags},k_{CS},C\}_{k_S},\{k_{CS},n_2,\ \text{Sflags},S\}^{e'}_{k_{CT}}} T$$

$$C \xrightarrow{\text{SOpts},\{\text{Sflags},k_{CS},C\}_{k_S},\{C,\text{MD}',t'\}_{k_{CS}}} S$$

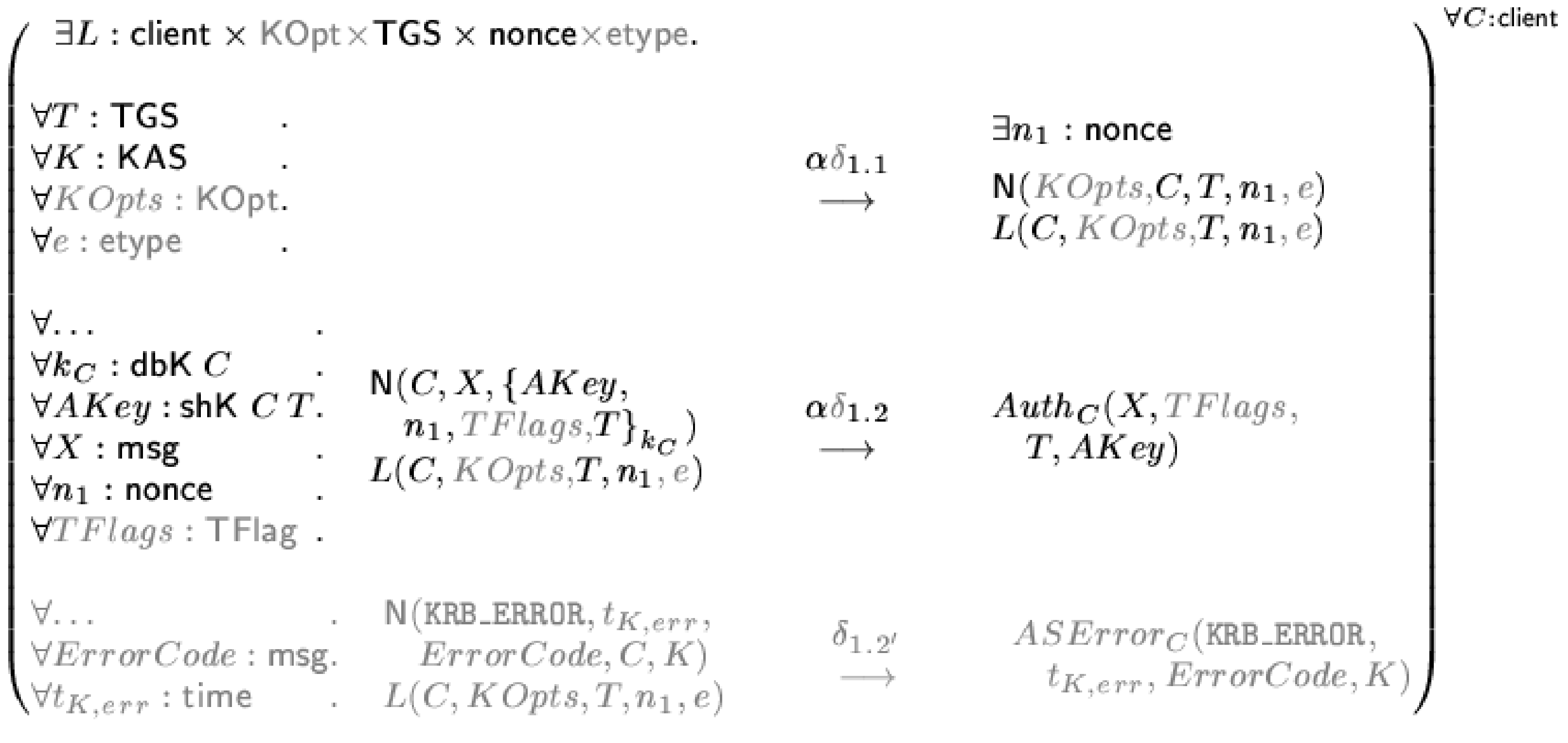
$$C \xleftarrow{[\{t'\}^{e}_{k_{CS}}]} S$$

$$C \xleftarrow{\text{KRB\_ERROR},[-|t|t'],t_{err},\text{ErrCode},C,(K|T|S)} K|T|S$$

Formalizations in MSR 2.0

- Abstract formalization
  - Core protocol
    - Enough detail to prove authentication and confidentiality
  - Exhibits some curious behavior (structural)
- Detailed formalization
  - Refines abstract formalization with
    - Options, encryption types, checksums
  - Exhibits additional curious behavior
- Timestamp-intensive formalization

Example MSR Rule



$$\left(\begin{array}{llll}
\exists L : \mathsf{client} \times \mathsf{KOpt} \times \mathsf{TGS} \times \mathsf{nonce} \times \mathsf{etype}. & & & \\
\begin{array}{l}\forall T : \mathsf{TGS} \quad .\\ \forall K : \mathsf{KAS} \quad .\\ \forall KOpts : \mathsf{KOpt}.\\ \forall e : \mathsf{etype} \quad .\end{array} & & \xrightarrow{\alpha\delta_{1.1}} & \begin{array}{l}\exists n_1 : \mathsf{nonce}\\ \mathsf{N}(KOpts, C, T, n_1, e)\\ L(C, KOpts, T, n_1, e)\end{array}\\
\begin{array}{l}\forall \ldots \quad .\\ \forall k_C : \mathsf{dbK}\ C \quad .\\ \forall AKey : \mathsf{shK}\ C\ T.\\ \forall X : \mathsf{msg} \quad .\\ \forall n_1 : \mathsf{nonce} \quad .\\ \forall TFlags : \mathsf{TFlag} \ .\end{array} & \begin{array}{l}\mathsf{N}(C, X, \{AKey,\\ \quad n_1, TFlags, T\}_{k_C})\\ L(C, KOpts, T, n_1, e)\end{array} & \xrightarrow{\alpha\delta_{1.2}} & \begin{array}{l}Auth_C(X, TFlags,\\ \quad T, AKey)\end{array}\\
\begin{array}{l}\forall \ldots \quad .\\ \forall ErrorCode : \mathsf{msg}.\\ \forall t_{K,err} : \mathsf{time} \quad .\end{array} & \begin{array}{l}\mathsf{N}(\mathtt{KRB\_ERROR}, t_{K,err},\\ \quad ErrorCode, C, K)\\ L(C, KOpts, T, n_1, e)\end{array} & \xrightarrow{\delta_{1.2'}} & \begin{array}{l}ASError_C(\mathtt{KRB\_ERROR},\\ \quad t_{K,err}, ErrorCode, K)\end{array}
\end{array}\right)^{\forall C : \mathsf{client}}$$

Figure 5. The client's role in the Authentication Service Exchange.

Formal Verification

- Define 2 classes of functions
  - k-Rank
    - Data origin authentication
    - Work done to encrypt a specific message with key k
  - E-Corank
    - Confidentiality
    - Work needed to extract information using keys from the set E

  Well-orders on which to build induction proofs

Inspired by work of Schneider

- Our corank functions parallel his rank functions

The k-Rank of t Relative to m_0

Work done to encrypt m_0 with key k

$\rho_k($ … $;m_0) =$

Atom	0	
$\{m_0\}_k$	1	
$\{m_1\}_k$	0	if $\rho_k(m_1; m_0) = 0, m_1 \neq m_0$
	$\rho_k(m_1; m_0) + 1$	if $\rho_k(m_1; m_0) > 0$
$\{m_1\}_{k'}$	$\rho_k(m_1; m_0)$	if $k' \neq k$
$[m_0]_k$	1	
$[m_1]_k$	0	if $\rho_k(m_1; m_0) = 0, m_1 \neq m_0$
	$\rho_k(m_1; m_0) + 1$	if $\rho_k(m_1; m_0) > 0$
$[m_1]_{k'}$	$\rho_k(m_1; m_0)$	if $k' \neq k$
t_1,t_2	$\max\{\rho_k(t_1; m_0), \rho_k(t_2; m_0)\}$	

The E-Corank of t Relative to m_0

Work needed to extract m_0 using keys in E

$cp_E($ t $; m_0) =$

m_0 (atomic)	0	
Atomic	∞	if $t \neq m_0$
$\{m_1\}_k$	$cp_E(m_1; m_0) + 1$	if $k \in E$
	$cp_E(m_1; m_0)$	if $k \notin E$
$[m_1]_k$	∞	
t_1,t_2	$\min\{cp_E(t_1; m_0), cp_E(t_2; m_0)\}$	

(Co)Rank of Facts and States

- Rank of a j-ary predicate P:
  - $\rho_k(P(t_1, \ldots, t_j); m_0) = \max\{\rho_k(t_1; m_0), \ldots, \rho_k(t_j; m_0)\}$
- Rank of a finite multiset M of facts
  - $\rho_k(M; m_0) = \max_{F \in M} \{\rho_k(F; m_0)\}$
- Corank of a j-ary predicate P:
  - $c\rho_k(P(t_1, \ldots, t_j); m_0) = \min\{c\rho_k(t_{i1}; m_0), \ldots, c\rho_k(t_{in}; m_0)\}$,
  - where $t_{i1}, \ldots, t_{in}$ are the 'public' terms
    - Look at terms that may be placed on the network later
    - In particular, $c\rho_E(I(m_0); m_0) = 0$
- Corank of a finite multiset M of facts
  - $c\rho_k(M; m_0) = \min_{F \in M} \{c\rho_k(F; m_0)\}$

Effect of Rules on (Co)Rank

For a transition rule R

$$\chi;\ F_1, \ldots, F_j \rightarrow \exists x_1 \ldots \exists x_m.\ G_1, \ldots, G_k$$

- Compare possible values of
  - $\rho_k(\{F_1, \ldots, F_j\}; m_0)$ and $\rho_k(\{G_1, \ldots, G_k\}; m_0)$
- Compare possible values of
  - $c\rho_E(\{F_1, \ldots, F_j\}; m_0)$ and $c\rho_E(\{G_1, \ldots, G_k\}; m_0)$
- Determine whether or not R can
  - Increase rank
  - Decrease corank

Dolev-Yao Intruder's Use of Keys

Apply this approach to intruder rules

- If intruder rule R increases $\rho_k(\_; m_0)$, then lhs(R) contains I(k)
  - ➢ Intruder knows the key k
- If intruder rule R decreases $c\rho_E(\_; m_0)$, then lhs(R) contains I(k) for some k in E
  - ➢ Intruder decrypts his way to m_0

  or rhs(R) contains $\exists m_0$
  - ➢ Intruder creates m_0

General Approach

- If $\rho_k(F; m_0) = 0$ for every fact in initial state and no intruder rule can increase $\rho_k(\_; m_0)$, then a fact F with $\rho_k(F; m_0) > 0$ implies that some honest principal created $\{m_0\}_k$
  - Show that it must have been a certain principal
- If $c\rho_E(F; m_0) > 0$ for every fact in initial state, no intruder rule can decrease $c\rho_E(\_; m_0)$, and no honest principal creates a fact F with $c\rho_E(F; m_0) = 0$, then m_0 is secret
  - $c\rho_E(I(m_0); m_0) = 0$

Analogs of Schneider's Rank Theorem

Summary: Using Rank and Corank

- Construct (co)rank function applicable to the desired property
- Inspect protocol rules
  - Determine which can
    - raise rank
    - lower corank
- Look at intruder rules
  - Find conditions ensuring that the intruder cannot raise rank/lower corank
    - Usually secrecy of certain key(s)

Properties Proved

	Confidentiality	Authentication
Ticket Granting Exchange	Abstract & Detailed	Abstract & Detailed
Client Server Exchange	Abstract	Abstract

Abstract Authentication Theorem

If TGT T receives the message

$\{k_{CT},C\}_{kT}, \{C\}_{kCT},C,S,n_2$

then some KAS K created k_{CT} and sent

$C,\{k_{CT},C\}_{kT}, \{k_{CT},n_1,T\}_{kC}$

and client C sent some

$X,\{C\}_{kCT},C,S',n'_2$

- In Kerberos 4
  - C must have sent the ticket and not generic X
- Similar result for Client/Server exchange
  - Ticket came from T, authenticator from C

Detailed Authentication Theorem

- Add details to obtain theorem for detailed formalization
  - Structure of abstract level proof remains
    - Just add details

If TGT T processes the message

$\{TFlags,k_{CT},C\}_{kT}, \{C,ck,t\}_{kCT},TOpts,C,S,n_2,e$

then some KAS K created k_{CT} and sent

$C,\{TFlags,k_{CT},C\}_{kT}, \{k_{CT},n_1,TFlags,T\}_{kC}$

and client C sent some

$X,\{C,ck,t\}_{kCT},TOpts',C,S',n'_2,e'$

with

$ck = [TOpts',C,S',n'_2,e']_{kCT}$

Proving Authentication

Authenticate data origin using rank

- Show ticket $\{TFlags,k_{CT},C\}_{kT}$ originates with some K
- Show authenticator $\{C,ck,t\}_{kCT}$ originates with C
  - Relies on the confidentiality of k_{CT}
- Prove confidentiality of k_{CT} using $\{k_C,k_T\}$-corank
  - No proper subset of $\{k_C,k_T\}$ protects k_{CT}
- Abstract level proofs follow same outline

Anomalies

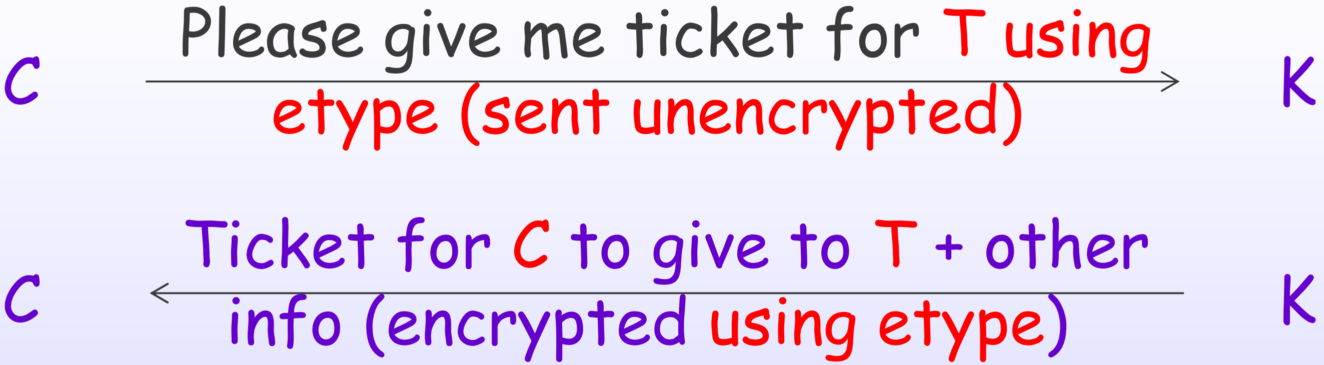
Interesting curiosities, but don't appear dangerous

- We've just seen that authentication does hold

- Encryption type anomaly
  - Difficult to recover from lost long term key
- Ticket switch anomaly
  - Client has incorrect beliefs about data in her possession
    - Application to anonymous tickets
    - Anonymous option under review by Working Group
- Ticket option anomaly
  - Effects similar to ticket switch anomaly

Encryption Type Anomaly

- Kerberos 5 allows C to specify encryption types that she wants used in K's response

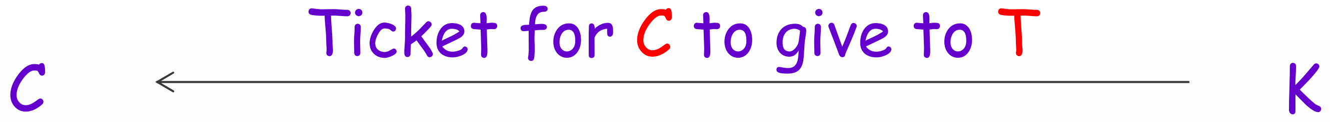


C $\xrightarrow{\text{Please give me ticket for T using etype (sent unencrypted)}}$ K

C $\xleftarrow{\text{Ticket for C to give to T + other info (encrypted using etype)}}$ K

- C's key of etype e_{bad} is k_{bad}
  - Intruder learns k_{bad}
  - C knows this and attempts to avoid e_{bad}/k_{bad}
  - I can still force k_{bad} to be used

Ticket Anomaly



C

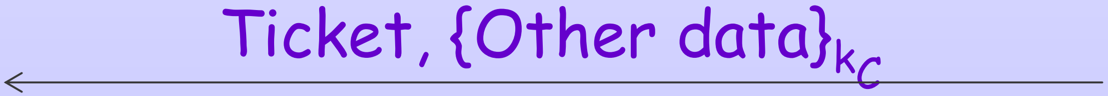
Ticket for C to give to T

K

- Kerberos 4:
  - ➢ Ticket is enclosed in another encryption

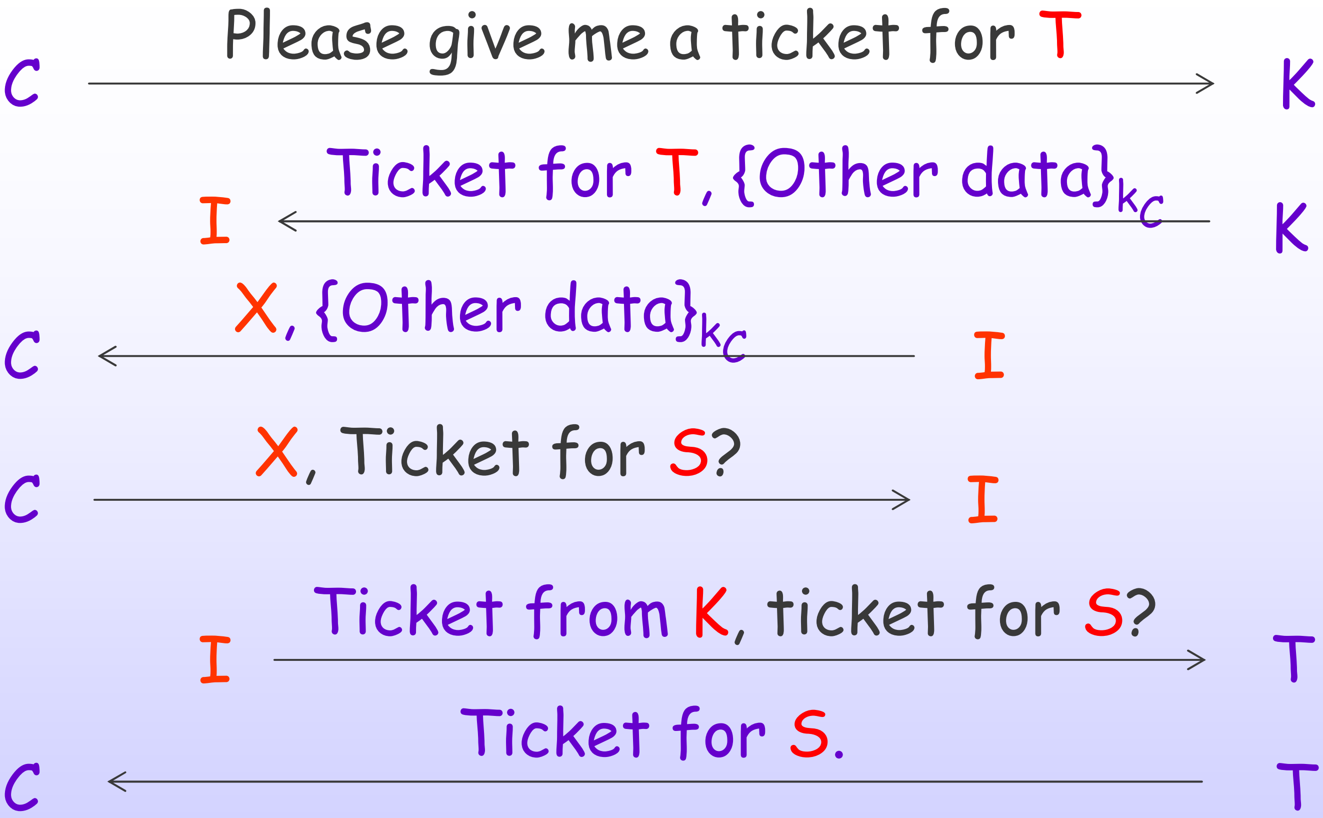
$\{\text{Ticket, Other data}\}_{k_C}$

- Kerberos 5:
  - ➢ Ticket is separate from other encryption



$\text{Ticket, }\{\text{Other data}\}_{k_C}$

Ticket Anomaly



C → K: Please give me a ticket for T

I ← K: Ticket for T, {Other data}$_{k_C}$

C ← I: X, {Other data}$_{k_C}$

C → I: X, Ticket for S?

I → T: Ticket from K, ticket for S?

C ← T: Ticket for S.

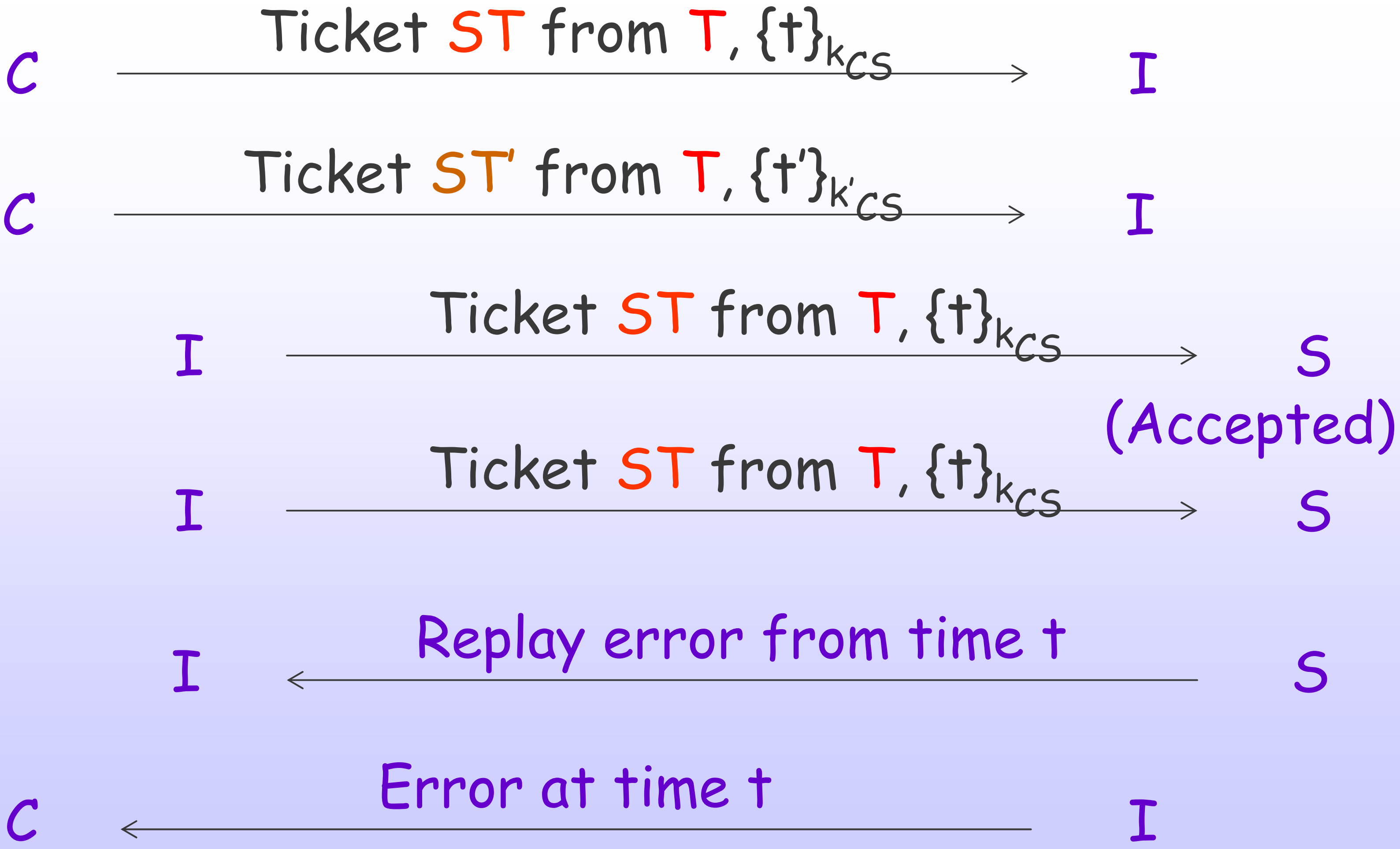
Ticket Anomaly

- T grants C a ticket for S
- But
  - C never has the ticket for T
  - C thinks she has sent a proper request
  - C's view of the world is inaccurate
  - Some properties of Kerberos 4 don't hold here
- Seen in both formalizations
  - Variations possible using added detail
    - Anonymous tickets

Ticket Option Anomaly

- C obtains tickets ST and ST' with different options for use with same server S
- C does not request mutual authentication from S
  - No response expected
- Assume that S can detect replays
  - Saves authenticators in a cache (following RFC 1510)

Ticket Option Anomaly



$C \xrightarrow{\text{Ticket ST from T, } \{t\}_{k_{CS}}} I$

$C \xrightarrow{\text{Ticket ST' from T, } \{t'\}_{k'_{CS}}} I$

$I \xrightarrow{\text{Ticket ST from T, } \{t\}_{k_{CS}}} S$ (Accepted)

$I \xrightarrow{\text{Ticket ST from T, } \{t\}_{k_{CS}}} S$

$I \xleftarrow{\text{Replay error from time t}} S$

$C \xleftarrow{\text{Error at time t}} I$

Ticket Option Anomaly

- C's request at time t is accepted, but her request at time t' is never seen by S
- C sees an error message with the timestamp t
  - Might assume request at t not accepted, request at t' accepted
  - I uses the replay to unpack the encrypted timestamp t
  - S's use of a replay cache allows this to occur
- Effects are similar to those of ticket switch found before but for more ticket options
  - Replay cache not yet formalized

Possible Future Developments

- Systematize definition and use of (co)rank functions
  - Need to determine 'public terms' for corank
- Analysis
  - Investigate temporal checks
  - Properties in more detailed formalizations
  - Anomalies – what can we still prove? Fix? Accept?
- Extend formalizations
  - Add structure and functionality
- Continue interaction with Kerberos designers