



# 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

$\text{princ}, \text{msg}, \text{shK } A \ B, \dots$

- Term

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

- Fact

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

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

- State

➤ Multiset of facts

# MSR Rules

- Transition rule

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

- Check constraints  $C_1, \dots, C_i$
- Check that  $F_1, \dots, F_j$  contained in state
- Obtain next state by
  - deleting  $F_1, \dots, F_j$
  - adding  $G_1, \dots, G_k$  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  $\rho_i$





# 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





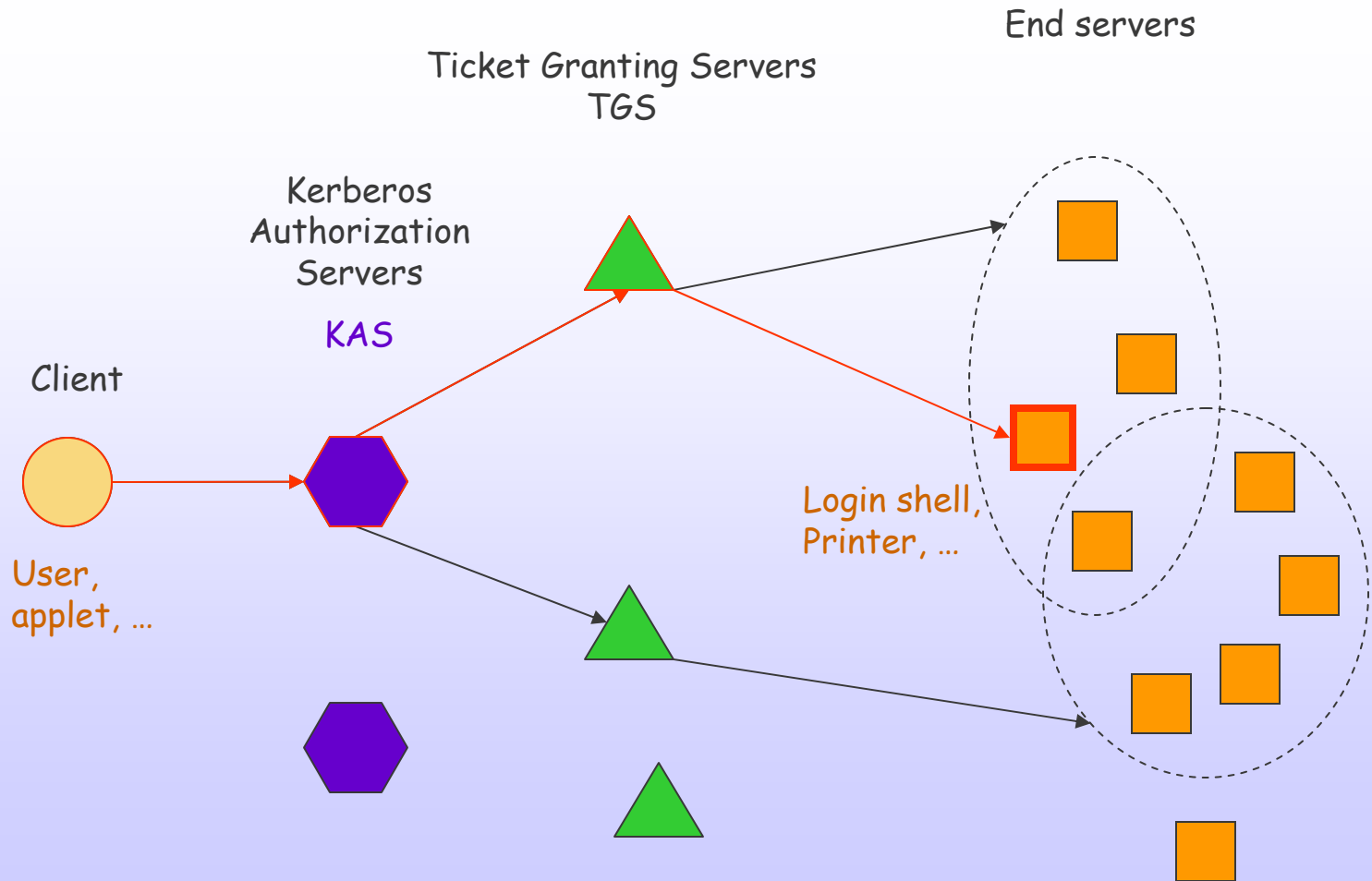
# Achievements

- 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



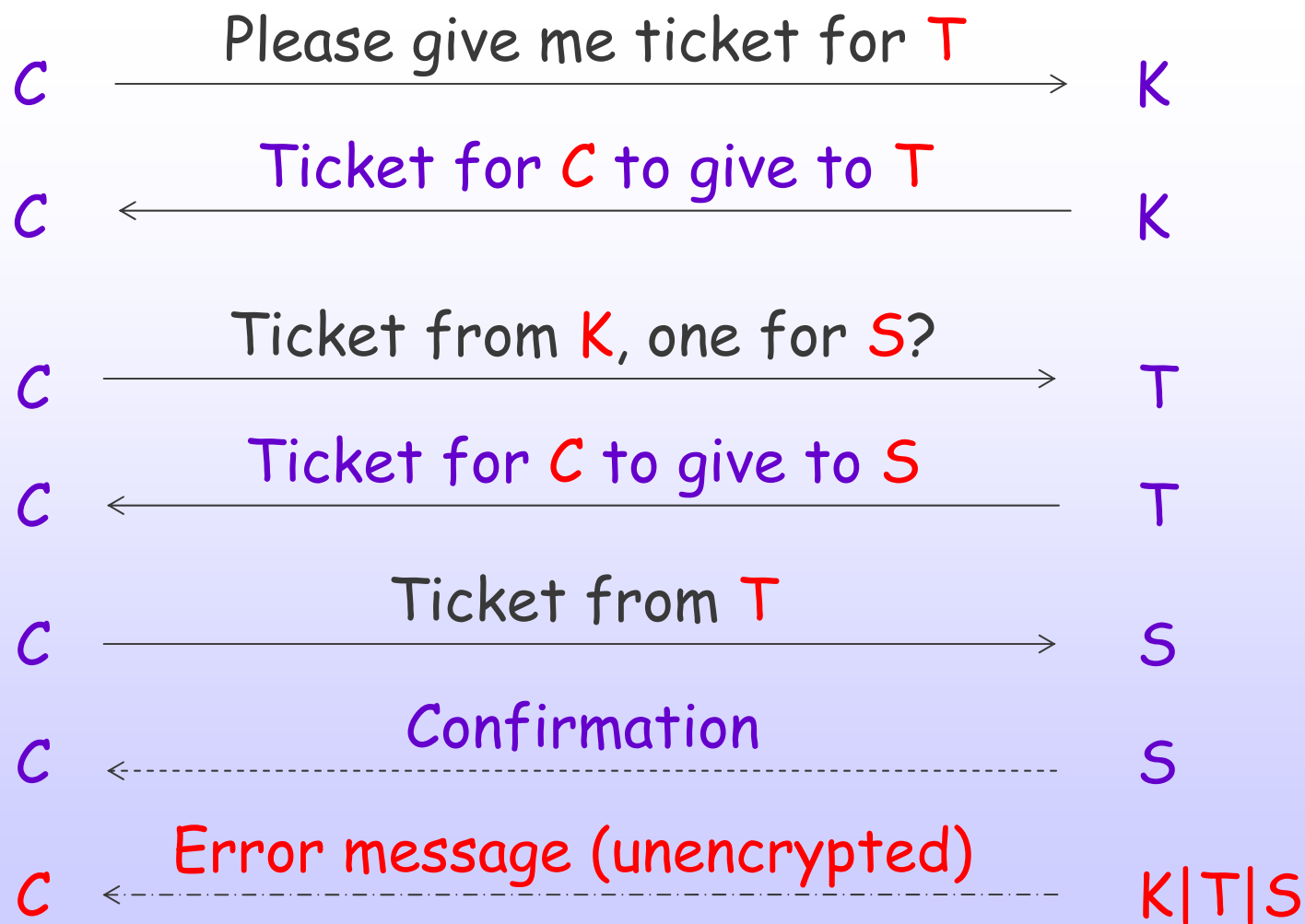
# Kerberos 5

Repeatedly authenticate client  $C$  to server  $S$

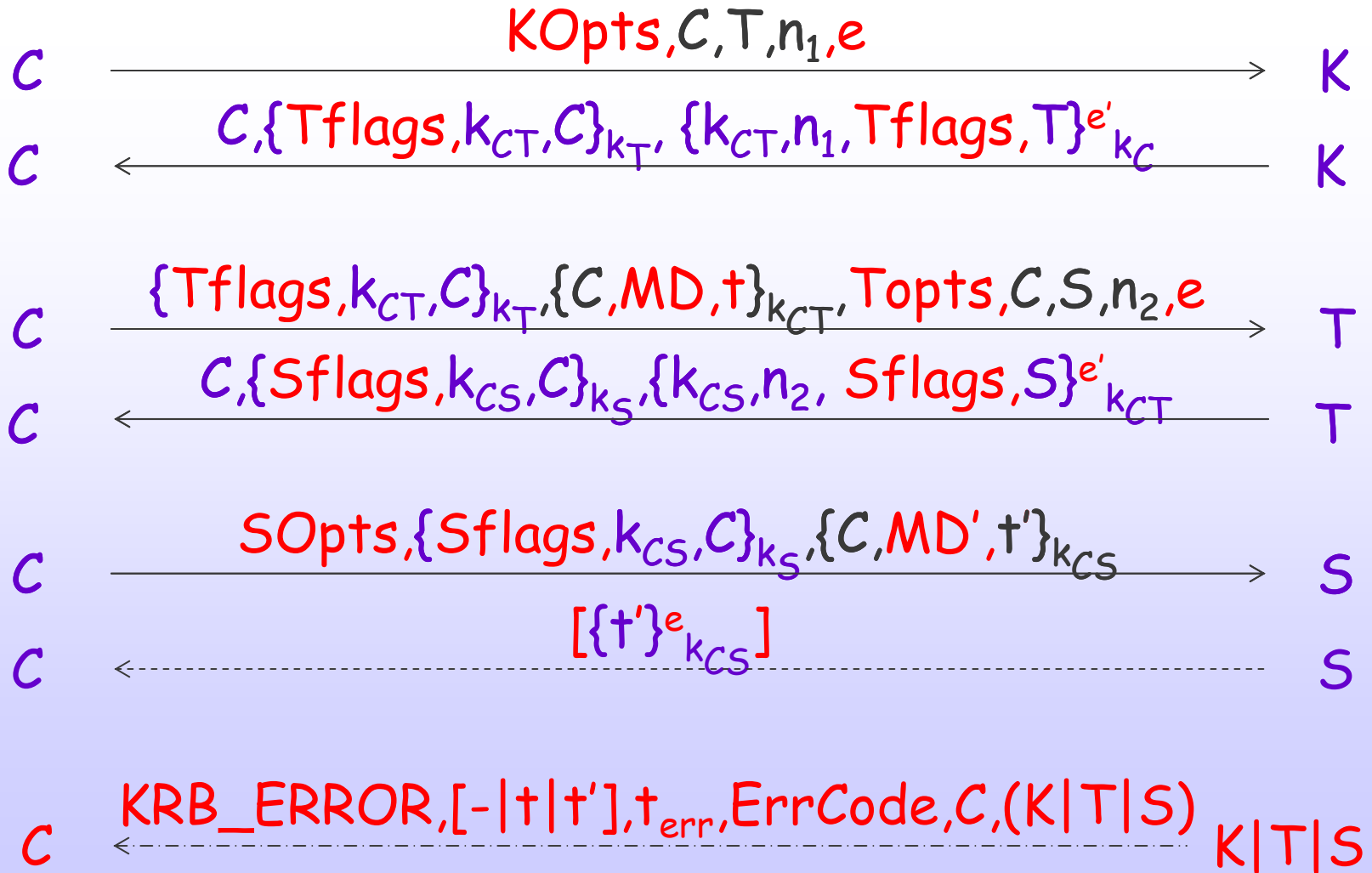
1.  $C$  obtains long term (eg, 1 day) ticket from KAS
  - Makes use of  $C$ 's long term key
  - Ticket encrypted - unreadable by  $C$
2.  $C$  obtains short-term (eg 5 min) ticket from TGT
  - Based on long term ticket from KAS
  - $C$  sends this ticket to  $S$



# Main Kerberos Exchange



# Abstract and Detailed Messages






# 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



$$\left( \begin{array}{lcl}
 \exists L : \text{client} \times \text{KOpt} \times \text{TGS} \times \text{nonce} \times \text{etype}. & & \\
 \forall T : \text{TGS} & . & \\
 \forall K : \text{KAS} & . & \\
 \forall K\text{Opts} : \text{KOpt}. & & \\
 \forall e : \text{etype} & . & \\
 \\
 \forall \dots & . & \\
 \forall k_C : \text{dbK } C & . & \\
 \forall AKey : \text{shK } C T. & N(C, X, \{AKey, & \\
 & n_1, TFlags, T\}_{k_C}) & \\
 \forall X : \text{msg} & . & \\
 \forall n_1 : \text{nonce} & . & \\
 \forall TFlags : \text{TFlag} & . & \\
 \\
 \forall \dots & . & N(\text{KRB\_ERROR}, t_{K,err}, \\
 \forall \text{ErrorCode} : \text{msg}. & \text{ErrorCode}, C, K) & \\
 \forall t_{K,err} : \text{time} & . & L(C, K\text{Opts}, T, n_1, e)
 \end{array} \right) \quad \forall C : \text{client}$$

$$\begin{array}{lcl}
 & \xrightarrow{\alpha\delta_{1.1}} & \begin{array}{l} \exists n_1 : \text{nonce} \\ N(K\text{Opts}, C, T, n_1, e) \\ L(C, K\text{Opts}, T, n_1, e) \end{array} \\
 \\
 & \xrightarrow{\alpha\delta_{1.2}} & \begin{array}{l} Auth_C(X, TFlags, \\ T, AKey) \end{array} \\
 \\
 & \xrightarrow{\delta_{1.2'}} & \begin{array}{l} ASError_C(\text{KRB\_ERROR}, \\ t_{K,err}, \text{ErrorCode}, K) \end{array}
 \end{array}$$

**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

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

# The E-Corank of $t$ Relative to $m_0$

Work needed to extract  $m_0$  using keys in  $E$

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

# (Co)Rank of Facts and States

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



# Effect of Rules on (Co)Rank

For a transition rule R

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

- Compare possible values of
  - $\rho_k(\{F_1, \dots, F_j\}; m_0)$  and  $\rho_k(\{G_1, \dots, G_k\}; m_0)$
- Compare possible values of
  - $cp_E(\{F_1, \dots, F_j\}; m_0)$  and  $cp_E(\{G_1, \dots, G_k\}; m_0)$
- 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  $\text{lhs}(R)$  contains  $I(k)$ 
  - Intruder knows the key  $k$
- If intruder rule  $R$  decreases  $\text{cp}_E(\_; m_0)$ , then  $\text{lhs}(R)$  contains  $I(k)$  for some  $k$  in  $E$ 
  - Intruder decrypts his way to  $m_0$or  $\text{rhs}(R)$  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  $cp_E(F; m_0) > 0$  for every fact in initial state, no intruder rule can decrease  $cp_E(\_; m_0)$ , and no honest principal creates a fact  $F$  with  $cp_E(F; m_0) = 0$ , then  $m_0$  is secret
    - $cp_E(I(m_0); m_0) = 0$

## 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



	<b>Confidentiality</b>	<b>Authentication</b>
<b>Ticket Granting Exchange</b>	Abstract & Detailed	Abstract & Detailed
<b>Client Server Exchange</b>	Abstract	Abstract

# Abstract Authentication Theorem



If TGT  $T$  receives the message

$\{k_{CT}, C\}_{k_T}, \{C\}_{k_{CT}}, C, S, n_2$

then some KAS  $K$  created  $k_{CT}$  and sent

$C, \{k_{CT}, C\}_{k_T}, \{k_{CT}, n_1, T\}_{k_C}$

and client  $C$  sent some

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

- 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

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

then some KAS **K** created  $k_{CT}$  and sent

$C, \{TFlags, k_{CT}, C\}_{k_T}, \{k_{CT}, n_1, TFlags, T\}_{k_C}$

and client **C** sent some

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

with

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



# Proving Authentication

## Authenticate data origin using rank

- Show ticket  $\{TFlags, k_{CT}, C\}_{k_T}$  originates with some  $K$
- Show authenticator  $\{C, ck, t\}_{k_{CT}}$  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

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

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

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

# Ticket Anomaly



$C \xleftarrow{\text{Ticket for } C \text{ to give to } T} K$

- Kerberos 4:

- Ticket is enclosed in another encryption

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

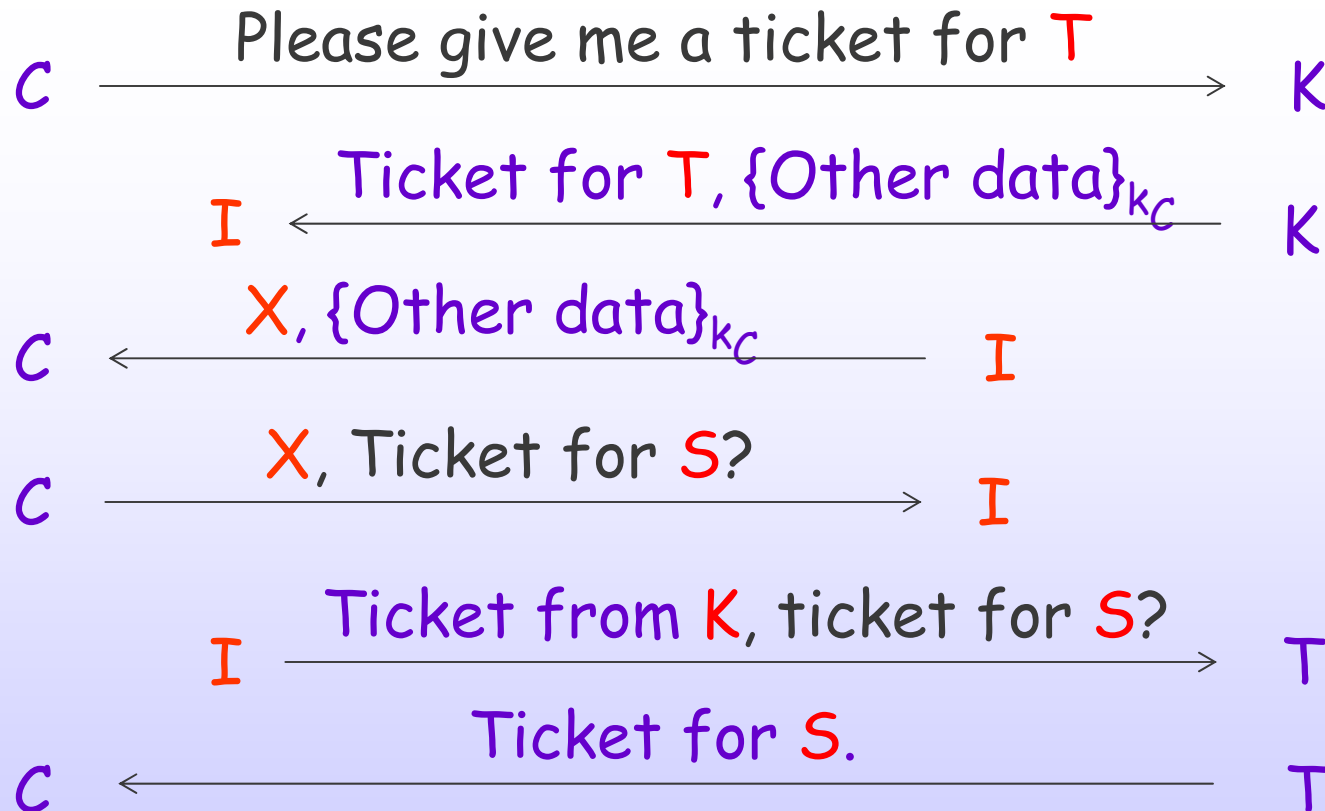
- Kerberos 5:

- Ticket is separate from other encryption

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



# Ticket Anomaly



# 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

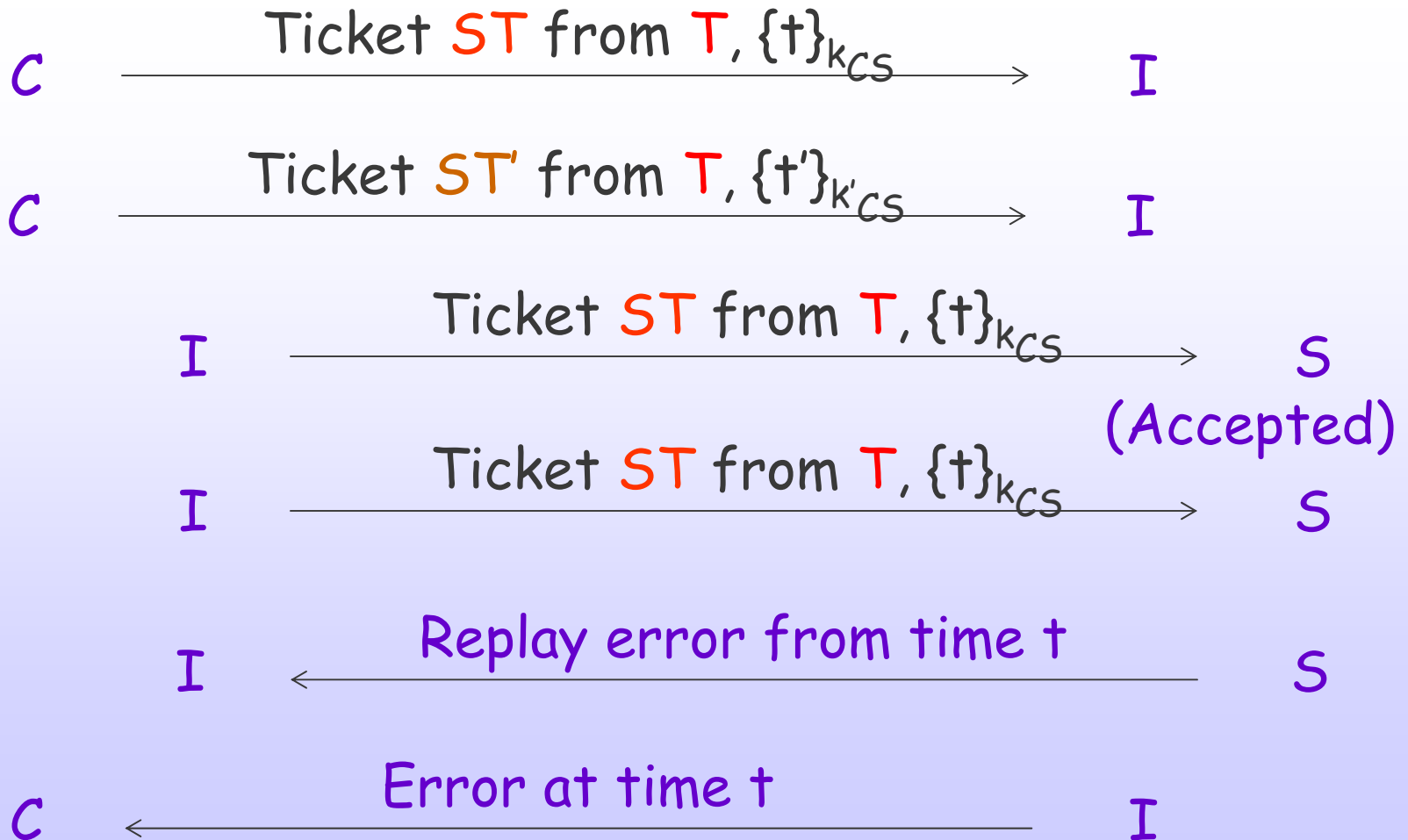


# Ticket Option Anomaly

- $C$  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



# 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

