Breaking and Fixing Public-Key Kerberos

Iliano Cervesato - Tulane University

(Joint work with A. D. Jaggard, A. Scedrov, J.-K. Tsay, and C. Walstad)

Partially supported by ONR and NSF

This Work

- Part of ongoing formal analysis of Kerberos 5
  - Previously studied
    - Detailed core protocol
    - Cross-realm authentication
  - Focus here on public-key extensions to Kerberos
- Attack on PKINIT
  - Breaks binding client's request and the response
  - Prevents full authentication and confidentiality
- Formal verification of fixes preventing attack
  - Close, ongoing interactions with IETF WG

Attack and Fixes (Overview)

- Man-in-the-middle attack on PKINIT
  - Kerberos KDC believes he is talking to the attacker
  - Client believes she is talking to the KDC
  - Attacker knows the key shared by C and KDC
- Possible because the KDC does not sign data identifying C
  - Attacker constructs request based on C's request
  - KDC signs data from request, sends in reply to attacker
  - Attacker forwards this to C after learning keys
  - Ran Canetti, consulted on details of spec., independently hypothesized the possibility of an "identity misbinding" attack
- PKINIT-27 is intended to defend against this attack
  - KDC signs data derived from C's identity

Consequences of the Attack

- **The attacker knows the keys C uses.**
  she may:
  - Impersonate Kerberos servers (in later rounds) to C
  - Monitor C's communications with end servers
- Notes
  - Attacker must be a legal user
  - C is authenticated to end server as attacker (not as C)
  - Applies to 1 of 2 PKINIT modes
    - "public-key encryption mode"
  - The "Diffie-Hellman" mode appears to avoid the attack
    - DH mode narrowly deployed
    - Still need to prove formally security for DH

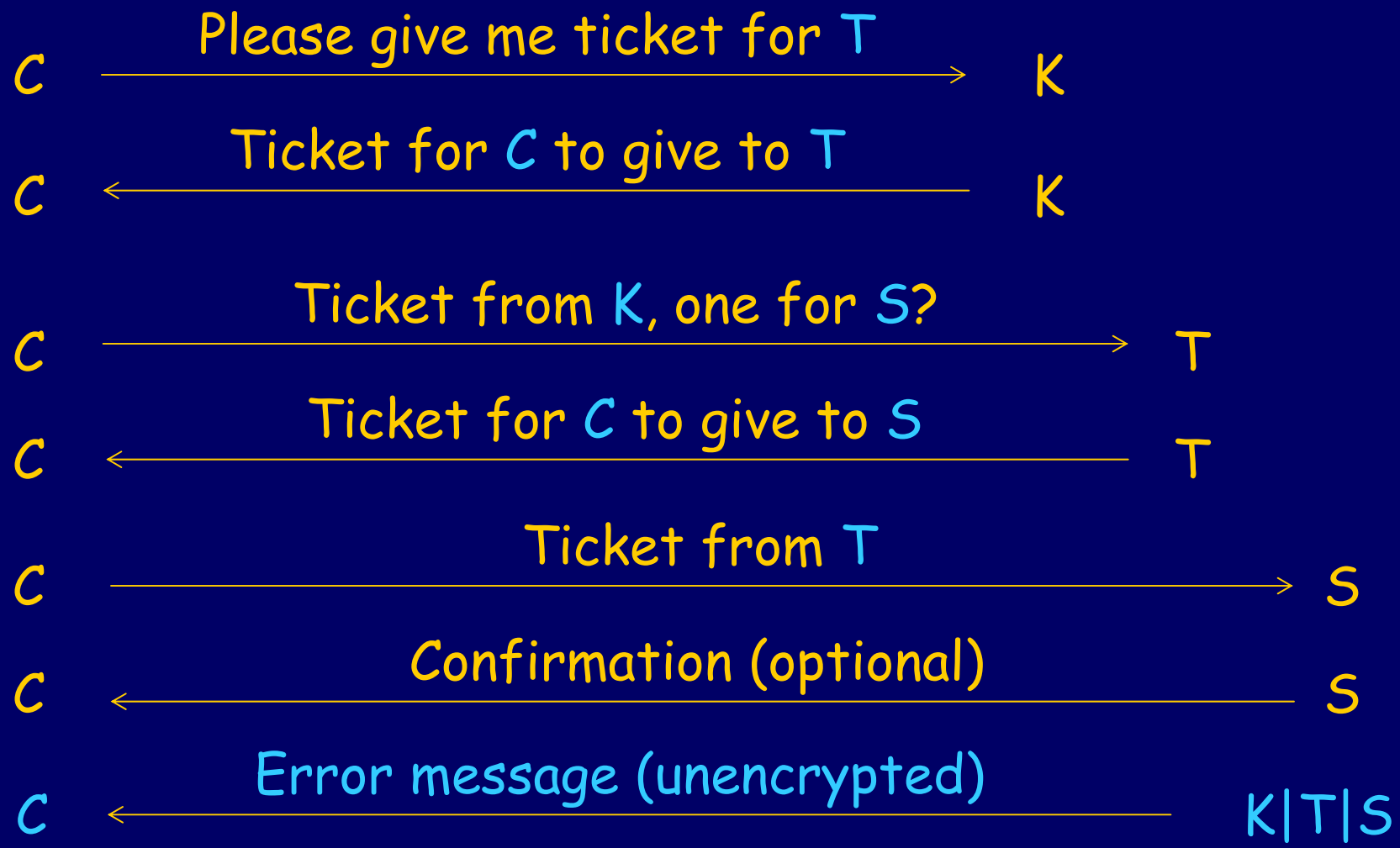
Kerberos Review

- Protocol goals
  - Repeatedly authenticate a client to multiple servers
  - Does not guard against DoS attacks
- Kerberos 4 - 1989
- Kerberos 5
  - Specified in RFC 1510 (1993), RFC 4120 (2005)
  - Extensions under development in IETF WG
- A real world protocol
  - Part of Windows, Linux, Unix and Mac OS
  - CableLabs implementation for cable TV
  - User login, file access, printing, email, etc.

Basic Kerberos 5

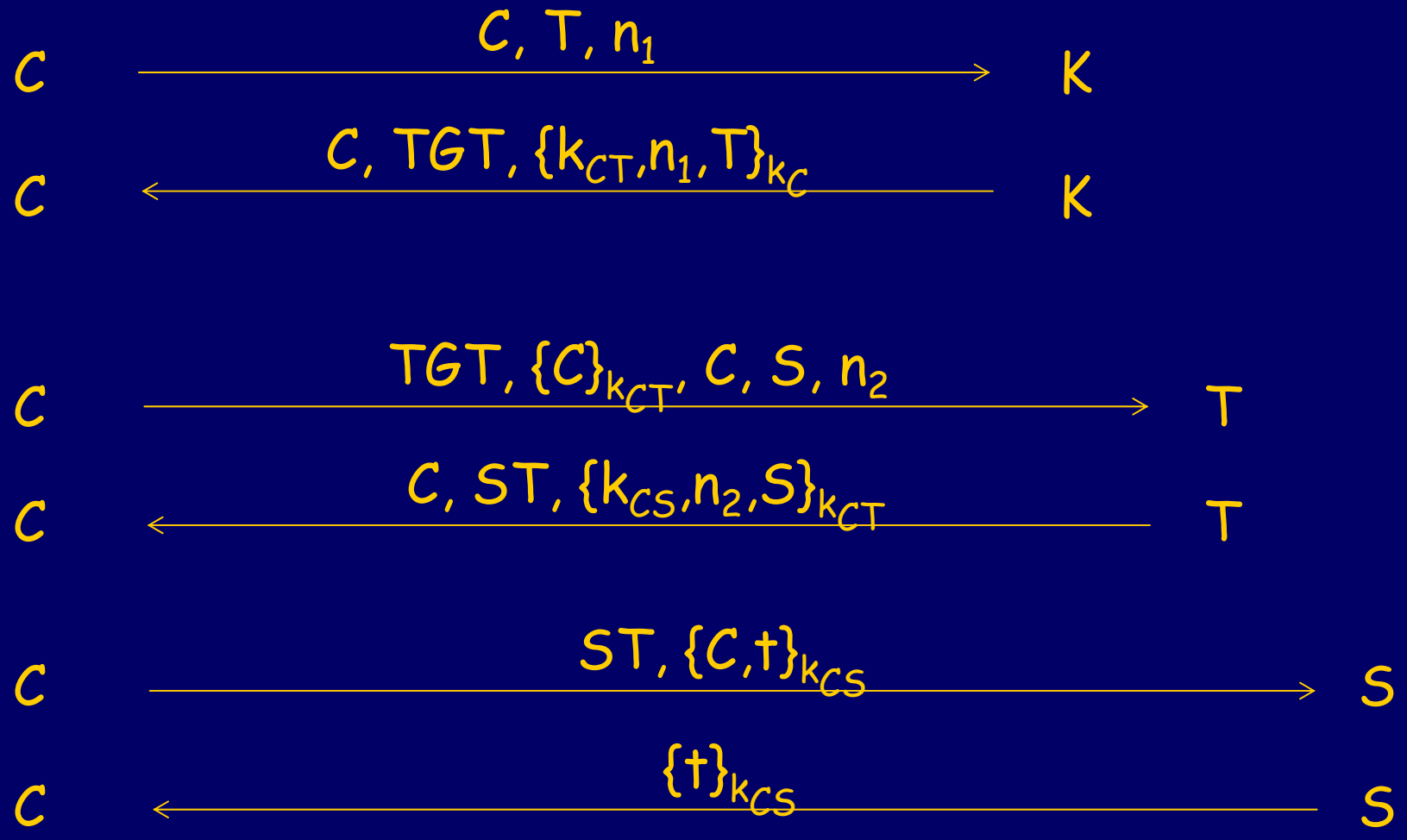
- Authentication
  - Repeatedly authenticate a client to multiple servers
- Client C wants ticket for end server S
  - Tickets are encrypted – unreadable by C
- C first obtains long term (*e.g.*, 1 day) ticket Ticket Granting Ticket (TGT) from a Kerberos Authentication Server K
  - Makes use of C's long term key
- C then obtains short term (*e.g.*, 5 min.) Service Ticket (ST) from a Ticket Granting Server T
  - Based on TGT from K
  - C sends this ticket to S

Basic Protocol Messages



Abstract Messages

$TGT = \{k_{CT}, C\}_{k_T}$
$ST = \{k_{CS}, C\}_{k_S}$



Results on Basic Kerberos 5

- Kerberos 5 does what it should
  - Authentication and confidentiality properties hold
  - Some anomalous behavior, but does not violate authentication

  (Butler, Cervesato, Jaggard, and Scedrov)

Public-Key Kerberos

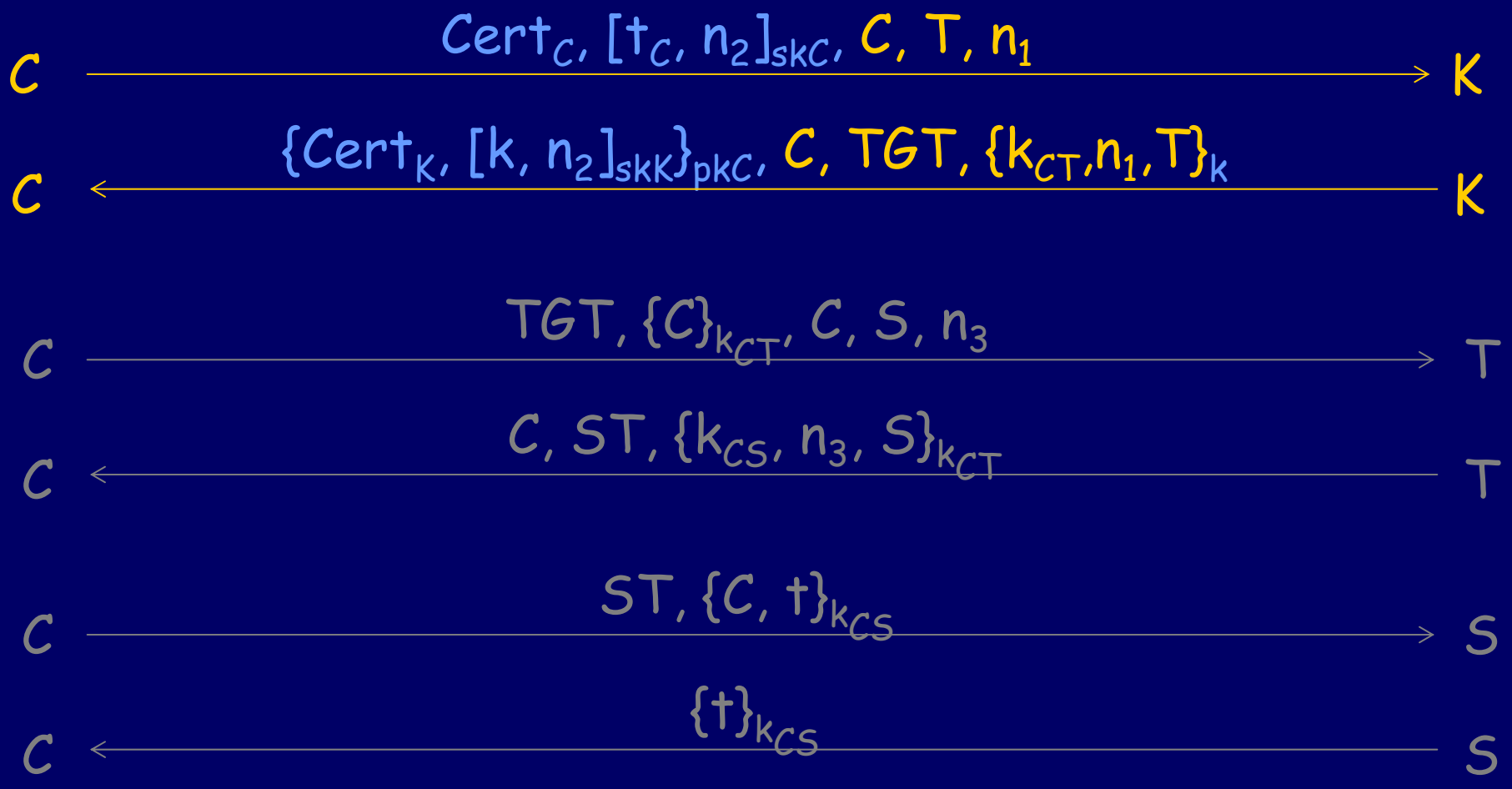
- **Extend basic Kerberos 5 to use PKI**
  - Change first round to avoid long-term shared keys
  - Originally motivated by security
    - If KDC is compromised, don't need to regenerate shared keys
    - Avoid use of password-derived keys
  - Current emphasis on administrative convenience
    - Avoid the need to register in advance of using Kerberized services
- This extension is called PKINIT
  - Current version is PKINIT-29
  - We found attack in -25;
    - We analyzed -26 (does not change the relevant design)
    - Traced back to -00
  - Attack fixed in -27
  - Versions included in Windows and Linux (called Heimdal)
  - Implementation developed by CableLabs (for cable boxes)
  - Not yet available in MIT version

Two Modes

- No more key k_C shared between C and K
  - Credentials for C instead encrypted under a temporary key k
    - How to generate and deliver k?
- Public-key encryption
  - k is generated by K
  - k encrypted under C's public key and is signed by K
  - Attack is against this mode
- Diffie-Hellman
  - k is derived from DH exchange using data from C and K
  - C and K each send signed data to contribute to DH key
    - Option for 'reuse' of the shared secret
  - CableLabs appears to be only implementation of DH mode
  - Initial inspection did not turn up attacks against this mode

Public-Key Encryption Mode

$TGT = \{k_{CT}, C\}_{kT}$
$ST = \{k_{CS}, C\}_{kS}$



Formalizing the Request

- Our formalization of pa-data includes
  - t_C = cusec/ctime (in pkAuthenticator)
  - n_2 = nonce (in pkAuthenticator)
  - $[t_C, n_2]_{skC}$ = signature (in signerInfos) over t_C, n_2 using C's secret key skC
- Our formalization of req-body includes
  - C = cname
  - T = sname
  - n_1 = nonce

$$[t_C, n_2]_{skC}, C, T, n_1$$

Formalizing the Reply

- Our formalization of pa-data includes
  - k = replyKey (in ReplyKeyPack)
  - n_2 = nonce (in ReplyKeyPack), from AS-REQ
  - $[k, n_2]_{skK}$ = signature with K's secret key skK
  - $\{\ldots\}_{pkC}$ is encryption with C's public key pkC
- C = cname in AS-REP
- TGT = ticket in AS-REP
- Our formalization of enc-part includes
  - AK = key
  - n_1 = nonce
  - t_K = authtime
  - T = sname
  - $\{\ldots\}_k$ is encryption with the reply key k

$$\{[k, n_2]_{skK}\}_{pkC}, C, TGT, \{AK, n_1, t_K, T\}_k$$

Attack and Fixes (Overview)

- MITM attack on PKINIT
  - KDC believes he is talking to the attacker
  - Client believes she is talking to the KDC
  - Attacker knows the key shared by the client and KDC
- Possible because the KDC does not sign data identifying the client
  - Attacker constructs request based on client's request
  - KDC signs data from client, sends in reply to attacker
  - Attacker forwards this to client after learning keys
  - Ran Canetti , consulted on details of spec., independently hypothesized the possibility of an "identity misbinding" attack
- PKINIT-27 is intended to defend against this attack
  - Kerberos server signs data derived from client's identity

The Attack

At time t_C, client C requests a ticket for ticket server T (using nonces n_1 and n_2):

The attacker I intercepts this, puts her name/signature in place of C's:

Kerberos server K replies with credentials for I, including: fresh keys k and AK, a ticket-granting ticket TGT, and K's signature over k,n_2:

(Ignore most of enc-part)

I decrypts, re-encrypts with C's public key, and replaces her name with C's:

- **I knows fresh keys k and AK**
- C receives K's signature over k,n_2 and assumes k, AK, etc., were generated for C (not I)

- Principal P has secret key skP, public key pkP
- $\{msg\}_{key}$ is encryption of msg with key
- $[msg]_{key}$ is signature over msg with key

Consequences of the Attack

- **The attacker knows the keys C uses**
  she may:
  - Impersonate servers (in later rounds) to the client C
  - Monitor C's communications with the end server
- Other notes
  - Attacker must be a legal user
  - C is authenticated to end server as attacker (not as C)
  - DH mode appears to avoid this attack
    - Still need to formally prove security for DH

After the First Round

- Both the attacker I and client C know the keys k and AK
  - C believes the KDC produced k and AK for C
- Attacker may monitor communications
  - Attacker must put her name into the request messages to match the tickets
  - Attacker learns keys in reply messages
- Attacker may impersonate servers
  - Instead of forwarding modified request messages, attacker may simply forge reply messages herself

Desired Authentication Property

If a client C processes a message containing KDC-generated public-key credentials, then some KAS K produced a set of such credentials **for C**.

- The attack shows this property does not hold in pk-init-00/-26
- We showed that this property holds if:
  - The KAS signs k, $F(C, n_i)$ for i=1,2

Preventing the Attack in General

- Sign data identifying client
  - The KDC signs k, $F(C, n_i)$
  - Assume $F(C, n) = F(C', n')$ implies $C = C'$ and $n = n'$
  - AS-REQ message now formalized as

$$\{ k, F(C, n_i)]_{skK}\}_{pkC}, C, TGT, \{AK, n_1, t_K, T\}_k$$

- We have a formal proof that this guarantees authentication
  - Does cname/crealm uniquely identify client?
  - Added secrecy properties if F(C, n) identifies pkC?

Fix Adopted in pk-init-27

- In the change implemented in pk-init-27:
  - The KDC signs k, cksum (*i.e.*, cksum in place of n_2)
    - k is replyKey
    - cksum is checksum over AS-REQ
    - Easier to implement than signing C, k, n_2
  - AS-REP now formalized as

$$\{k, cksum, [k, cksum]_{skK}\}_{pkC}, C, TGT, \{AK, n_1, t_K, T\}_k$$

- We have a formal proof that this guarantees authentication
  - Assume checksum is preimage resistant
  - Assume KDC's signature keys are secret
  - Plan to carry out a more detailed, cryptographic proof in the future

Proof Sketch for General Defense

- Assume
  - Client receives AS-REP with $[k, F(C, n_i)]_{skK}$
  - KAS's signature key is secret
  - Signatures are unforgeable
  - $F(C, n) = F(C', n')$ implies $C = C'$ and $n = n'$
- Proof sketch
  - Signature in reply must come from the KAS K
  - K would only produce this signature in response to a request containing C' such that $F(C', n') = F(C, n)$
  - Collision-freeness of F implies that K created the reply for C

Real-World Impact

- Our work cited in August MS security bulletin

  www.microsoft.com/technet/security/bulletin/MS05-042.mspx

- Although other vulnerabilities viewed as more pressing for IT managers, this attack has real-world effects and highlights a design vulnerability
  - Remote code execution, privilege elevation seem to arise from coding errors, not design flaws
  - No known exploit using our attack

Interactions with IETF

- Close collaboration with IETF Kerberos WG
  - Discussed possible fixes we were considering
  - Attack announced on WG list in July
  - We verified a fix the WG suggested
    - This was incorporated into PKINIT-27
  - Presented this work at IETF-63
    - Discussed possible fixes and our analysis of these
    - Useful discussions with WG participants on other areas for work
  - Participate in WG interim meeting in Sep and IETF-64 in Nov.
- Impact of formal methods in IETF security area
  - At security-area level, they want to see more interaction with formal methods

Conclusions

- Extended formalization of Kerberos 5 to PKINIT
- We found a MITM attack against public-key encryption mode in PKINIT-00/-26
  - Protocol attack with real-world effects (MS security bulletin)
  - We've given general fix defending against this
- Close collaboration with IETF WG
  - Discussion and analysis of possible fixes
    - We've analyzed the fix employed in PKINIT-27

Future Work

- Fully analyze and verify of PKINIT
  - Computational proofs
    - *E.g.*, signature strength
  - Look at DH mode
- Other parts of Kerberos suite
- Continue interactions with WG
- Timed analysis (Hasebe, Jaggard, Okada)