Sharad Dixit
CMSC 487 Introduction to Network Security

Assignment 3

- ## OBJECTIVE
  The objective of this assignment is to analyze protocols for flaws and then verify their correctness properties using Cryptographic Protocol Shapes Analyzer(CPSA). We analyze two protocols Blanchet and Kerberos with CPSA and then verify their correctness.

- ## METHODOLOGY
  The CPSA tool analyze the protocol and verify their correctness by providing output shapes to analyze. Shapes provided are either unrealized and realized in which the realized shapes are the final output of a skeleton of a defined protocol, therefore in order to check and verify a protocol the realized shapes are analyzed.
  Further, In the realized shape nodes may be blue, red or black where a black node represents transmission and blue, red represents receptions. Also, a blue node represents an explainable reception and whereas a red node represents an unexplainable reception.
  Arrows in the shapes represents ordering of events in the skeleton, where a solid arrow represents the same message getting received which was send from the transmission point. The dashed arrow represents that the message received and transmitted do not agree.

Note: Throughout the document I have represented the screenshots of the analyzes of both protocols respectively that is first Blanchet and second Kerberos. The screenshots are aligned in such a way that the first screenshot represents the original protocol and the below it represents the screenshot for the corrected in protocol.

- ## ANALYSIS
  The analysis of both the protocols is done using CPSA tool and the flaws are found in the protocol and are rectified. On each page the first screenshot belong to the original protocol and the second screenshot below it represents the corrected protocol that is after fix. The detailed explanation for each screenshot is give adjacent to it.

Blanchet Protocol:

(a.) Original Protocol

```
(defprotocol blanchet basic
  (defrole init
    (vars (a b akey) (s skey) (d data))
    (trace
      (send (enc ( enc s (invk a)) b))
      (recv (enc d s))))
  (defrole resp
    (vars (a b akey) (s skey) (d data))
    (trace
      (recv (enc ( enc s (invk a)) b))
      (send (enc d s))))
```

In the original protocol the flaw is that the responder does not to whom he is receiving the data and whether the data was meant for the corresponding receiver or not. Therefore, we need to rectify this and correct flaw in the protocol. Further, also the responder when transmits the data to initiator then that data can be taken by the listener and hence this is another flaw which needs to be dealt with.

(b.) Flaw corrected in Protocol

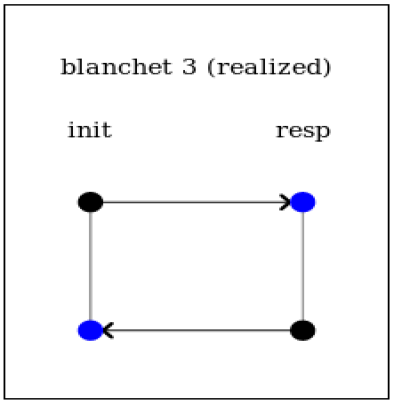
```
(defprotocol blanchet-corrected basic
  (defrole init
    (vars (a b akey) (s skey) (d data))
    (trace
      (send (enc ( enc s b (invk a)) b))
      (recv (enc (enc d s) a))))
  (defrole resp
    (vars (a b akey) (s skey) (d data))
    (trace
      (recv (enc ( enc s b (invk a)) b))
      (send (enc (enc d s) a)))))
```

The flaw is corrected in the original protocol by adding ‘b’ in the initiator strand under the encryption so that responder knows that the data was meant only for the responder.
Further also from responder to initiator we encrypt the data using public key of ‘a’ so that the data can be decrypted by only the initiator and hence both flaws are dealt with to obtain the correct protocol.

Analysis of shapes of Original Protocol and Corrected Protocol for Blanchet

1.) Initiator Perspective-

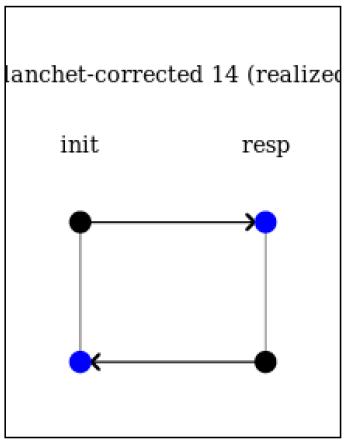
1.a) Original Protocol




```
(defskeleton blanchet
  (vars (d data) (s skey) (a b akey))
  (defstrand init 2 (d d) (s s) (a a) (b b))
  (defstrand resp 2 (d d) (s s) (a a) (b b))
  (precedes ((0 0) (1 0)) ((1 1) (0 1)))
  (non-orig (invk a) (invk b))
  (uniq-orig s)
  (operation nonce-test (contracted (a-0 a) (b-0 b)) s (1
    (enc (enc s (invk a)) b))
  (label 3)
  (parent 1)
  (unrealized)
  (shape)
  (maps ((0) ((a a) (b b) (s s) (d d))))
  (origs (s (0 0))))
```

The graph to the left shows initiator skeleton perspective of the original protocol, and It can be seen that the shape gives correct output as there is red node and as well as no dashed arrow and also it is to be noted that this is realized shape and hence final shape of initiator perspective Therefore, this means that our goal is satisfied.

1.b) Flaw Corrected in Protocol

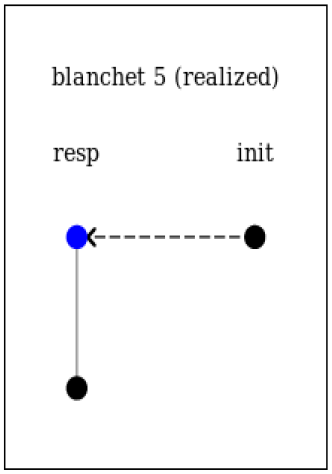



```
(defskeleton blanchet-corrected
  (vars (d data) (s skey) (a b akey))
  (defstrand init 2 (d d) (s s) (a a) (b b))
  (defstrand resp 2 (d d) (s s) (a a) (b b))
  (precedes ((0 0) (1 0)) ((1 1) (0 1)))
  (non-orig (invk a) (invk b))
  (uniq-orig s)
  (operation nonce-test (contracted (a-0 a) (b-0 b)) s (1 0)
    (enc (enc s b (invk a)) b))
  (label 14)
  (parent 12)
  (unrealized)
  (shape)
  (maps ((0) ((a a) (b b) (s s) (d d))))
  (origs (s (0 0))))
```

The graph to the left shows the initiator perspective of the corrected protocol, and it can be seen that there is no red node as well no dashed arrow and hence our goal is satisfied from initiator perspective.

2.) Responder Perspective-

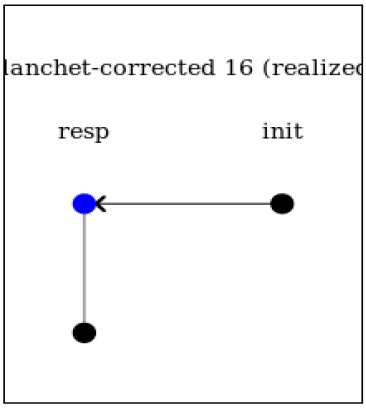
2.a) Original Protocol



The left shape represents the responder's perspective of the original protocol where it can be seen that the transmission between initiator and responder is dashed that means the message transmitted does not agree with responder with the initiator. Therefore, this means that the data sent by initiator cannot by verified by the responder that whether that data was meant for the responder.

```
(defskeleton blanchet
  (vars (d data) (s skey) (a b b-0 akey))
  (defstrand resp 2 (d d) (s s) (a a) (b b))
  (defstrand init 1 (s s) (a a) (b b-0))
  (precedes ((1 0) (0 0)))
  (non-orig (invk a) (invk b))
  (uniq-orig d)
  (operation encryption-test (added-strand init 1) (enc s (invk a))
    (0 0))
  (label 5)
  (parent 4)
  (unrealized)
  (shape)
  (maps ((0) ((a a) (b b) (s s) (d d))))
  (origs (d (0 1))))
```

2.b) Flaw Corrected in Protocol

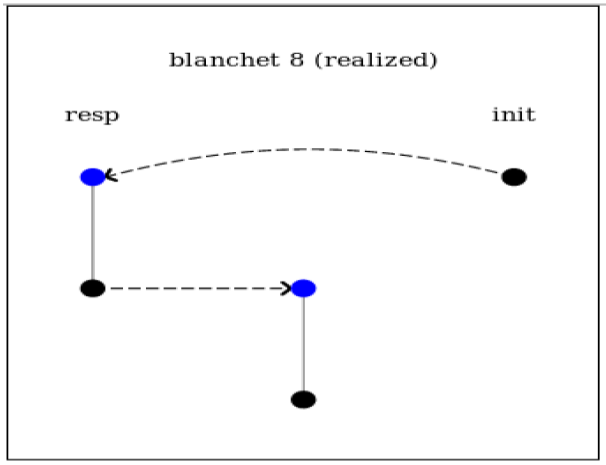


The left shape represents the responder's perspective of the corrected protocol where it can be seen that there is no dashed arrow as in the above case and as well as there is no red arrow. The above original protocol is rectified by adding b to the original protocol under the encryption of private key of a(initiator). Hence it solves the problem of original protocol

```
(defskeleton blanchet-corrected
  (vars (d data) (s skey) (a b akey))
  (defstrand resp 2 (d d) (s s) (a a) (b b))
  (defstrand init 1 (s s) (a a) (b b))
  (precedes ((1 0) (0 0)))
  (non-orig (invk a) (invk b))
  (uniq-orig d)
  (operation encryption-test (added-strand init 1) (enc s b (i
    (0 0))
  (label 16)
  (parent 15)
  (unrealized)
  (shape)
  (maps ((0) ((a a) (b b) (s s) (d d))))
  (origs (d (0 1))))
```

3.) Responder Perspective with Listener-

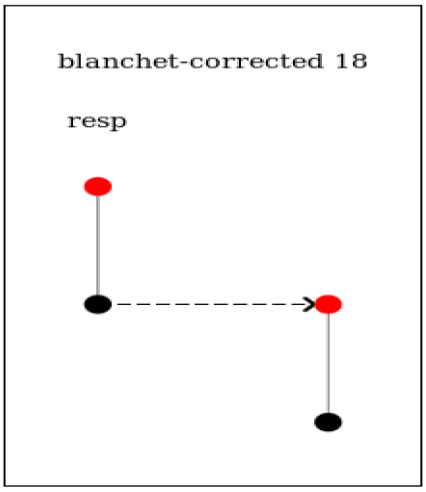
3.a) Original Protocol




```
(defskeleton blanchet
  (vars (d data) (s skey) (a b b-0 akey))
  (defstrand resp 2 (d d) (s s) (a a) (b b))
  (deflistener d)
  (defstrand init 1 (s s) (a a) (b b-0))
  (precedes ((0 1) (1 0)) ((2 0) (0 0)))
  (non-orig (invk a) (invk b))
  (uniq-orig d)
  (operation encryption-test (added-strand init 1) (enc s (
    (0 0))
  (label 8)
  (parent 7)
  (unrealized)
  (shape)
  (maps ((0 1) ((a a) (b b) (s s) (d d))))
  (origs (d (0 1))))
```

The left shape represents the responder perspective with the listener in the original protocol, in which the listener strand effectively listens the data d. Therefore, our protocol should be corrected in such way that the listener cannot obtain the data d.

3.b) Flaw Corrected in Protocol

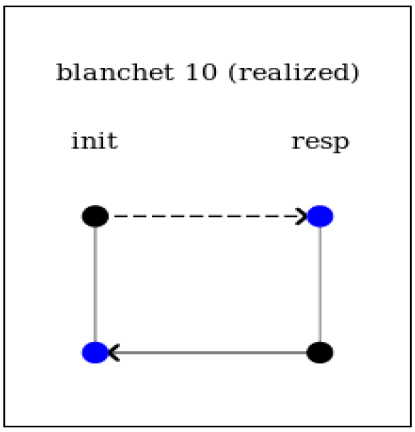



```
(defskeleton blanchet-corrected
  (vars (d data) (s skey) (a b akey))
  (defstrand resp 2 (d d) (s s) (a a) (b b))
  (deflistener d)
  (precedes ((0 1) (1 0)))
  (non-orig (invk a) (invk b))
  (uniq-orig d)
  (comment "Analyze from the responder's with a liste
  (label 18)
  (parent 17)
  (unrealized (0 0) (1 0))
  (origs (d (0 1)))
  (comment "empty cohort"))
```

The left shape represents the responder perspective with the listener of the corrected protocol, where the flaw of listener being able to listen to data. This flaw is corrected by applying encryption to the value by public key of “a” so that only the initiator can decrypt the message. But the shape in the corrected protocol as shown on the left is not realized which shows that we were able to correct the flaw and the listener is not able to fetch the data and there is no meaning to red node and dashed arrow.

4.) Skeleton where initiator and responder disagree on the values of b

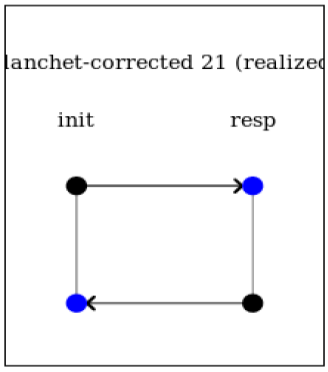
4.a) Original Protocol




```
(defskeleton blanchet
  (vars (d data) (s skey) (a b b-0 akey))
  (defstrand init 2 (d d) (s s) (a a) (b b))
  (defstrand resp 2 (d d) (s s) (a a) (b b-0))
  (precedes ((0 0) (1 0)) ((1 1) (0 1)))
  (non-orig (invk a))
  (uniq-orig d s)
  (comment "Analyze with disagree on values of b")
  (label 10)
  (parent 9)
  (unrealized)
  (shape)
  (maps ((0 1) ((a a) (b b) (s s) (d d) (b-0 b-0))))
  (origs (s (0 0)) (d (1 1))))
```

The left shape represents the last part of the question where initiator and responder disagree on the values of b. As seen in the left shape which is of original protocol has dashed arrow from initiator to responder and hence they do not agree on the message and hence needs to be fixed.

4.b) Flaw Corrected in protocol




```
(defskeleton blanchet-corrected
  (vars (d data) (s skey) (a b akey))
  (defstrand init 2 (d d) (s s) (a a) (b b))
  (defstrand resp 2 (d d) (s s) (a a) (b b))
  (precedes ((0 0) (1 0)) ((1 1) (0 1)))
  (non-orig (invk a))
  (uniq-orig d s)
  (operation encryption-test (displaced 2 0 init 1) (enc s b-0 (invk a
    (1 0))
  (label 21)
  (parent 20)
  (unrealized)
  (shape)
  (maps ((0 1) ((a a) (b b) (s s) (d d) (b-0 b))))
  (origs (s (0 0)) (d (1 1))))
```

The left shape represents the corrected part of the last question where there is no red node as well as no dashed arrow representing no flaw in the corrected protocol. This flaw is corrected by the same which was done in part-2 of Blanchet where the ‘b’ was added to the initiator strand under the encryption so that it knows that messaged was meant only for responder.

KERBEROS Protocol:

(a.) Original Protocol

```
(defprotocol kerberos basic

(defrole client

  (vars (c as t name) (k ak skey) (n1 n2 crtc crtas tgt data) (tc tas text))

  (trace (send (cat crtc c t (enc tc n2 (privk c)) n1))

         (recv (cat c tgt (enc crtas (enc k n2 (privk as)) (pubk c)) (enc t ak tas n1 k))))

(defrole kasas

  (vars (c as t name) (k ak skey) (n1 n2 crtc crtas tgt data) (tc tas text))

  (trace (recv (cat crtc c t (enc tc n2 (privk c)) n1))

         (send (cat c tgt (enc crtas (enc k n2 (privk as)) (pubk c)) (enc t ak tas n1 k))))
```

The left side shows the original protocol where we know from the shapes below that the client and as(Authentication Server) cannot confirm on the identity of each other from initiator perspective. The analysis of the Kerberos protocol is done below where the screenshot above is from the original protocol and following the screenshot below it is of the corrected protocol.

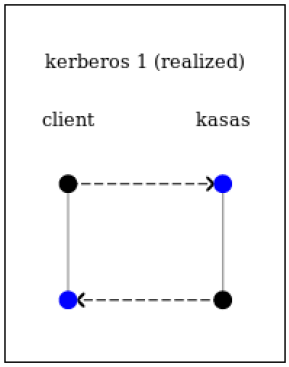
(b.) Flaw in Corrected Protocol

```
(defprotocol kerberos_corrected basic

(defrole client

  (vars (c as t name) (k ak skey) (n1 n2 crtc crtas tgt data) (tc tas text))

  (trace (send (cat c (enc crtc t as tc n2 n1 (privk c))))

         (recv (enc (enc tgt crtas k c crtc n1 n2 (enc t ak tas n1 k) (privk as)) (pubk c)))))

(defrole kasas

  (vars (c as t name) (k ak skey) (n1 n2 crtc crtas tgt data) (tc tas text))

  (trace (recv (cat c (enc crtc t as tc n2 n1 (privk c))))

         (send (enc (enc tgt crtas k c crtc n1 n2 (enc t ak tas n1 k) (privk as)) (pubk c))))

         (uniq-orig k)))
```

The left side shows the corrected protocol where flaws of protocol are corrected from my perspective. To rectify the protocol, I have made the follow changes which can be seen in the left side attached figure.

1.) Client Perspective-

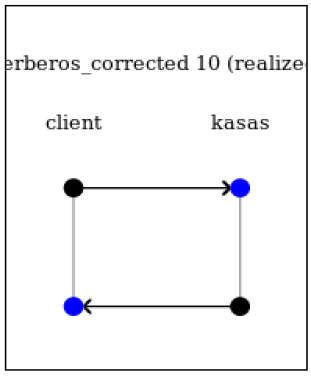
1.a) Original Protocol




```
(defskeleton kerberos
  (vars (tc tas tc-0 tas-0 text)
    (n1 n2 crtc crtas tgt n1-0 crtc-0 crtas-0 tgt-0 data)
    (c as t c-0 t-0 name) (k ak ak-0 skey))
  (defstrand client 2 (tc tc) (tas tas) (n1 n1) (n2 n2) (crtc crtc)
    (crtas crtas) (tgt tgt) (c c) (as as) (t t) (k k) (ak ak))
  (defstrand kasas 2 (tc tc-0) (tas tas-0) (n1 n1-0) (n2 n2)
    (crtc crtc-0) (crtas crtas-0) (tgt tgt-0) (c c-0) (as as) (t t-0)
    (k k) (ak ak-0))
  (precedes ((0 0) (1 0)) ((1 1) (0 1)))
  (non-orig (privk c) (privk as))
  (uniq-orig n1 n2)
  (operation encryption-test (added-strand kasas 2)
    (enc k n2 (privk as)) (0 1))
  (label 1)
  (parent 0)
  (unrealized)
  (shape)
  (maps
    ((0)
      ((c c) (as as) (t t) (k k) (ak ak) (n1 n1) (n2 n2) (crtc crtc)
        (crtas crtas) (tgt tgt) (tc tc) (tas tas))))
  (origs (n1 (0 0)) (n2 (0 0))))
```

The left shape represents the client's perspective where it can be seen that the transmission from client to as and as to client is dashed arrows which describes that the messages received do not agree on both the client as well as the as side. To rectify this, we needed to make changes in the protocol, where 'crctc', 'n1' and 't1' didn't agree on both the sides

1.b) Flaw corrected in protocol



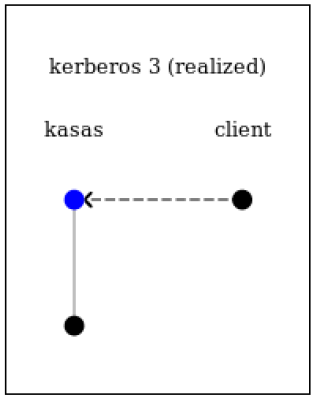

```
(defskeleton kerberos_corrected
  (vars (tc tas text) (n1 n2 crtc crtas tgt data) (c as t name)
    (k ak skey))
  (defstrand client 2 (tc tc) (tas tas) (n1 n1) (n2 n2) (crtc crtc
    (crtas crtas) (tgt tgt) (c c) (as as) (t t) (k k) (ak ak))
  (defstrand kasas 2 (tc tc) (tas tas) (n1 n1) (n2 n2) (crtc crtc)
    (crtas crtas) (tgt tgt) (c c) (as as) (t t) (k k) (ak ak))
  (precedes ((0 0) (1 0)) ((1 1) (0 1)))
  (non-orig (privk c) (privk as))
  (uniq-orig n1 n2 k)
  (operation encryption-test (displaced 2 0 client 1)
    (enc crtc t as tc-0 n2 n1 (privk c)) (1 0))
  (label 10)
  (parent 9)
  (unrealized)
  (shape)
  (maps
    ((0)
      ((c c) (as as) (t t) (k k) (ak ak) (n1 n1) (n2 n2) (crtc crt
        (crtas crtas) (tgt tgt) (tc tc) (tas tas))))
  (origs (k (1 1)) (n1 (0 0)) (n2 (0 0))))
```

Therefore, in order to rectify the above 'crctc','n1' and 't1' were moved to digital signature of c so that it's authentication is verified by as.

Further 'as' also has been moved to c in order for Authentication Server to know that the messaged was meant for it.

AS Perspective-

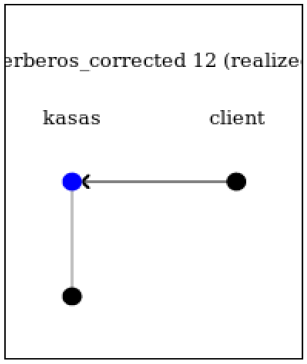
2.a) Original Protocol




```
(defskeleton kerberos
  (vars (tc tas text) (n1 n2 crtc crtas tgt n1-0 crtc-0 data)
    (c as t t-0 name) (k ak skey))
  (defstrand kasas 2 (tc tc) (tas tas) (n1 n1) (n2 n2) (crtc crtc
    (crtas crtas) (tgt tgt) (c c) (as as) (t t) (k k) (ak ak))
  (defstrand client 1 (tc tc) (n1 n1-0) (n2 n2) (crtc crtc-0) (c
    (t t-0))
  (precedes ((1 0) (0 0)))
  (non-orig (privk c) (privk as))
  (uniq-orig k ak)
  (operation encryption-test (added-strand client 1)
    (enc tc n2 (privk c)) (0 0))
  (label 3)
  (parent 2)
  (unrealized)
  (shape)
  (maps
    ((0)
      ((c c) (as as) (t t) (k k) (ak ak) (n1 n1) (n2 n2) (crtc cr
        (crtas crtas) (tgt tgt) (tc tc) (tas tas))))
  (origs (k (0 1)) (ak (0 1))))
```

The left side shape represents the AS perspective of the original protocol where the transmission between client to AS is dashed arrow which shows that the messages do not agree on both the sides. Therefore, this was needed to be rectified.

2.b) Flaw corrected in protocol




```
(defskeleton kerberos_corrected
  (vars (tc tas text) (n1 n2 crtc crtas tgt data) (c as t name)
    (k ak skey))
  (defstrand kasas 2 (tc tc) (tas tas) (n1 n1) (n2 n2) (crtc crtc)
    (crtas crtas) (tgt tgt) (c c) (as as) (t t) (k k) (ak ak))
  (defstrand client 1 (tc tc) (n1 n1) (n2 n2) (crtc crtc) (c c) (as
    (t t))
  (precedes ((1 0) (0 0)))
  (non-orig (privk c) (privk as))
  (uniq-orig k ak)
  (operation encryption-test (added-strand client 1)
    (enc crtc t as tc n2 n1 (privk c)) (0 0))
  (label 12)
  (parent 11)
  (unrealized)
  (shape)
  (maps
    ((0)
      ((c c) (as as) (t t) (k k) (ak ak) (n1 n1) (n2 n2) (crtc crt
        (crtas crtas) (tgt tgt) (tc tc) (tas tas))))
  (origs (k (0 1)) (ak (0 1))))
```

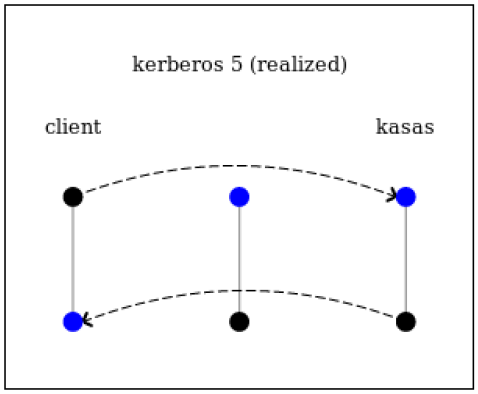
The left side shape represents the AS perspective of the corrected protocol where the transmission is rectified between AS and client. To rectify the above problem “c”, “crtas” and “tgt” under the digital signature of AS.

Further in order to verify the authenticity of the message which was sent by AS, I have moved block (enc t ak tas n1 k) block to digital signature.

This above solution rectifies the authenticity of the protocol.

3.) Client's Perspective with listener: -

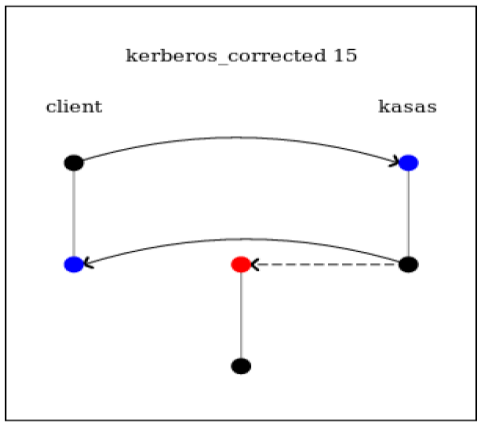
3.a) Original Protocol




```
(defskeleton kerberos
  (vars (tc tas tc-0 tas-0 text)
    (n1 n2 crtc crtas tgt n1-0 crtc-0 crtas-0 tgt-0 data)
    (c as t c-0 t-0 name) (k ak ak-0 skey))
  (defstrand client 2 (tc tc) (tas tas) (n1 n1) (n2 n2) (crtc crtc
    (crtas crtas) (tgt tgt) (c c) (as as) (t t) (k k) (ak ak))
  (deflistener k)
  (defstrand kasas 2 (tc tc-0) (tas tas-0) (n1 n1-0) (n2 n2)
    (crtc crtc-0) (crtas crtas-0) (tgt tgt-0) (c c-0) (as as) (t t
    (k k) (ak ak-0))
  (precedes ((0 0) (2 0)) ((2 1) (0 1)))
  (non-orig (privk c) (privk as))
  (uniq-orig n1 n2)
  (operation encryption-test (added-strand kasas 2)
    (enc k n2 (privk as)) (0 1))
  (label 5)
  (parent 4)
  (unrealized)
  (shape)
  (maps
    ((0 1)
      ((c c) (as as) (t t) (k k) (ak ak) (n1 n1) (n2 n2) (crtc crt
        (crtas crtas) (tgt tgt) (tc tc) (tas tas))))
  (origs (n1 (0 0)) (n2 (0 0))))
```

The left side shape represents the client's perspective with listener added in order to obtain the value of k. It is easily visible from the shape that in the original protocol the listener is able to obtain the value of k, therefore this needs to be corrected.

3.b) Flaw Corrected in Protocol

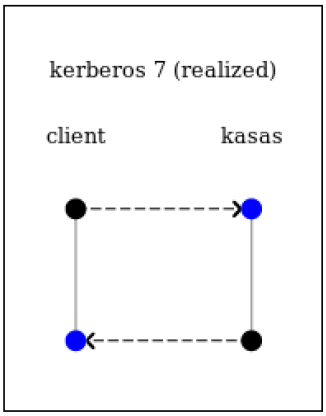



```
(defskeleton kerberos_corrected
  (vars (tc tas text) (n1 n2 crtc crtas tgt data) (c as t name)
    (k ak skey))
  (defstrand client 2 (tc tc) (tas tas) (n1 n1) (n2 n2) (crtc crt
    (crtas crtas) (tgt tgt) (c c) (as as) (t t) (k k) (ak ak))
  (deflistener k)
  (defstrand kasas 2 (tc tc) (tas tas) (n1 n1) (n2 n2) (crtc crtc
    (crtas crtas) (tgt tgt) (c c) (as as) (t t) (k k) (ak ak))
  (precedes ((0 0) (2 0)) ((2 1) (0 1)) ((2 1) (1 0)))
  (non-orig (privk c) (privk as))
  (uniq-orig n1 n2 k)
  (operation encryption-test (displaced 3 0 client 1)
    (enc crtc t as tc-0 n2 n1 (privk c)) (2 0))
  (label 15)
  (parent 14)
  (unrealized (1 0))
  (comment "empty cohort"))
```

The left side shape represents the client's perspective with listener in the corrected protocol where it can be seen from the shape that now the listener is not able to obtain the value of k as the listener transmission between the AS is dashed arrow and as well as the listener strand has red node. Therefore the dashed arrow and red node confirms that are fix in the protocol is correct as the listener is not able to obtain the value of k.

4.) Analysis of Client and Server don't agree values of c and ak:-

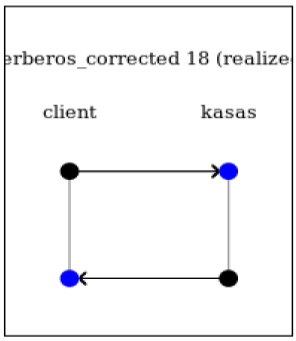
4.a) Original Protocol




```
(defskeleton kerberos
  (vars (tc tas text) (n1 n2 crtc crtas tgt data) (c as t c-0
    (k ak ak-0 skey))
  (defstrand client 2 (tc tc) (tas tas) (n1 n1) (n2 n2) (crtc
    (crtas crtas) (tgt tgt) (c c) (as as) (t t) (k k) (ak ak-0
  (defstrand kasas 2 (tc tc) (tas tas) (n1 n1) (n2 n2) (crtc c
    (crtas crtas) (tgt tgt) (c c-0) (as as) (t t) (k k) (ak ak
  (precedes ((0 0) (1 0)) ((1 1) (0 1)))
  (non-orig (privk as))
  (uniq-orig n1 n2 k ak)
  (label 7)
  (parent 6)
  (unrealized)
  (shape)
  (maps
    ((0 1)
      ((c c) (as as) (t t) (k k) (ak ak) (n1 n1) (n2 n2) (crtc
        (crtas crtas) (tgt tgt) (tc tc) (tas tas) (ak-0 ak-0)
        (c-0 c-0))))
  (origs (k (1 1)) (ak (1 1)) (n1 (0 0)) (n2 (0 0))))
```

The left side shape represents the last part of the question where client and server don't agree on the c and ak in the original protocol. It can be seen from the shape that the transmission from client to AS and AS to client is dashed arrow and which needs to be rectified.

4.b) Flaw Corrected in Protocol




```
(defskeleton kerberos_corrected
  (vars (tc tas text) (n1 n2 crtc crtas tgt data) (as t c name)
    (k ak skey))
  (defstrand client 2 (tc tc) (tas tas) (n1 n1) (n2 n2) (crtc crtc)
    (crtas crtas) (tgt tgt) (c c) (as as) (t t) (k k) (ak ak))
  (defstrand kasas 2 (tc tc) (tas tas) (n1 n1) (n2 n2) (crtc crtc)
    (crtas crtas) (tgt tgt) (c c) (as as) (t t) (k k) (ak ak))
  (precedes ((0 0) (1 0)) ((1 1) (0 1)))
  (non-orig (privk as))
  (uniq-orig n1 n2 k ak)
  (operation encryption-test (displaced 2 1 kasas 2)
    (enc tgt crtas k c-0 crtc n1 n2 (enc t ak-0 tas n1 k) (privk as))
    (0 1))
  (label 18)
  (parent 17)
  (unrealized)
  (shape)
  (maps
    ((0 1)
      ((c c) (as as) (t t) (k k) (ak ak) (n1 n1) (n2 n2) (crtc crtc)
        (crtas crtas) (tgt tgt) (tc tc) (tas tas) (ak-0 ak) (c-0 c))))
  (origs (k (1 1)) (ak (1 1)) (n1 (0 0)) (n2 (0 0))))
```

The left side shape represents the same case in corrected protocol were the steps taken above for rectified this too where the client and AS agree on the value of k but not on the value of c and ak.