This document describes a notation that I use for describing security protocols. I call this “Security Protocol Notation” or SPN for short. SPN was created by me and is thus not likely to be found, in this exact form, elsewhere. However, standardizing on some notation simplifies and clarifies the presentation of a protocol. It also promotes a formal approach to analysis. At some point it might make sense to use some other notation that is more widely accepted. 

TODO: Such as?

- **Principals** $A, B, C, \ldots, T, \ldots$
  
The principals in a protocol are represented by upper case Latin letters starting at $A$. The name $T$ is used for trusted intermediaries. Certain upper case letters are also used for other purposes described below. These special uses take priority unless there is no possibility for confusion.

- **Keys** $\kappa_1, \kappa_2, \kappa_A^{(p)}, \kappa_A^{(r)}$
  
  Keys are represented with the Greek letter $\kappa$. Keys are subscribed to identify them. Uppercase Latin letter subscripts associate a key with a particular principal. Public keys are shown with the superscript $(p)$. Private keys are shown with the superscript $(r)$.

- **Encryption algorithms** $E(\kappa, m)$
  
  Encryptions are shown as a function $E$ taking a key and a message $m$. The result is the encrypted version of $m$. The nature of the encryption (symmetric or public/private) is implied by the kind of key used. Subscripts on $E$ are used to identify different encryption algorithms in cases where that matters.
  
  Messages are understood to be sequences of octets of any length (unless otherwise specified). I assume there is some means, unspecified here but documented in practice, for converting SPN expressions into octet sequences. Thus syntactically $m$ can be any SPN expression.

- **Signature algorithms** $S(\kappa^{(r)}, m)$
Signatures are shown as a function $S$ taking a private key and a message $m$. The result is $m$ concatenated with the signature over $m$ (in other words, a new octet sequence). Subscripts on $S$ are used to identify different signature algorithms in cases where that matters. As a special case $S_d$ is used to denote a “detached” signature over $m$. The result of $S_d$ is just the signature and does not have $m$ concatenated to the signature.

- **Hash algorithms $H(m), H(κ, m)$**
  Hashes are shown as a function $H$ taking a message $m$ and returning the hash as an octet sequence. Message authentication codes are shown as a function $H$ that also takes a key. Subscripts on $H$ are used to identify different hash algorithms in cases where that matters.

- **Message transfer $A → m = e → B$**
  The transfer of a message from $A$ to $B$ is shown using right pointing arrows with the nature of the message transferred specified in the middle of the arrow as $e$. That message can be optionally given a name $m$ by which it can be referenced in later protocol steps.

- **Binary concatenation $m_1 || m_2$**
  The octet sequence resulting from concatenating $m_2$ onto the end of $m_1$. 