```
! Synopsis of generic security wrapper as used in rs.exe as pseudo code in True BASIC

! The entire program is not included in an infinite do loop that exits by short circuit.
! Instead, with the exception handling and potential division by zero, it is straight line.

WHEN EXCEPTION IN                ! begin security wrapper

  WHEN EXCEPTION IN              ! begin error trap wrapper for all
    ! open and read key file to prove it exists locally
    OPEN  #21: NAME name_key_file$, ORG BYTE
    READ #21, BYTES len_record_key_file: input_buffer$
    CLOSE #21
  USE
    WHEN EXCEPTION IN
      ! if error in opening local key file, delete it if possible
      UNSAVE name_key_file$
    USE                  ! stops on error deleting file
      PRINT copyr_notice$
      STOP
    END WHEN
    PRINT copyr_notice$     ! stops after deleteing key file
    STOP
  END WHEN

  ! Decode and test security variables such as:
  !     expire_date > system_date > buy_date
  !
  ! Instead of logical tests using IF-THEN statements which are easy to find
  ! and hack in compiled code, it is possible to reduce the tests to numerical
  ! analysis where if the test fails, a division by zero occurs which is trapped
  ! by the exception handler wrapper.
  !
  ! For example here, if the dates above are out of whack this formula causes
  ! a division by zero.  Note SGN returns the plus or minus sign of a number.
  !
  ! LET test = 1 /  (    ( SGN( expire_date - system date) + 1)
  !                    * ( SGN( expire_date - buy_date) + 1)
  !                    * ( SGN( system_date - buy_date) + 1)   )
  !
  ! If any of the three parts of the denominator above are zero, the entire
  ! division operation for the test is undefined and trapped by the error handler.
  !
  ! Other logical tests may be reduced to numerical analysis such as testing for
  ! the integer multiple of a fixed licensing constant as the increment of 367 days.
```

```
    ! Helpful for that is not to store the number 367 as a number or a literal string so as
    ! to avoid hacking but stored rather as a calculated value using some square root
    ! function and other arithmentic.  The test is based on using the CEIL function
    ! that returns the ceiling integer of a floating point number and the FP function
    ! that returns the fractional part of a floating point number.  The logic is left
    ! to the reader.

    CALL mainline_processing     ! at this point, all security is passed

USE                              ! error trap and stop if division by zero or any other fatal error

    PRINT copyr_notice$
    STOP

END WHEN                         ! end security wrapper

END
```