```
    if (j <= 9)
        Hex[i*2+1] = (j + '0');
    else
        Hex[i*2+1] = (j + 'a' - 10);
    };
    Hex[HASHHEXLEN] = '\0';
    };

/* calculate H(A1) as per spec */
void DigestCalcHA1(
    IN char * pszAlg,
    IN char * pszUserName,
    IN char * pszRealm,
    IN char * pszPassword,
    IN char * pszNonce,
    IN char * pszCNonce,
    OUT HASHHEX SessionKey
    )
{
    MD5_CTX Md5Ctx;
    HASH HA1;
    MD5Init(&Md5Ctx);
    MD5Update(&Md5Ctx, pszUserName, strlen(pszUserName));
    MD5Update(&Md5Ctx, ":", 1);
    MD5Update(&Md5Ctx, pszRealm, strlen(pszRealm));
    MD5Update(&Md5Ctx, ":", 1);
    MD5Update(&Md5Ctx, pszPassword, strlen(pszPassword));
    MD5Final(HA1, &Md5Ctx);
    if (stricmp(pszAlg, "md5-sess") == 0) {
        MD5Init(&Md5Ctx);
        MD5Update(&Md5Ctx, HA1, HASHLEN);
        MD5Update(&Md5Ctx, ":", 1);
        MD5Update(&Md5Ctx, pszNonce, strlen(pszNonce));
        MD5Update(&Md5Ctx, ":", 1);
        MD5Update(&Md5Ctx, pszCNonce, strlen(pszCNonce));
        MD5Final(HA1, &Md5Ctx);
    };
    CvtHex(HA1, SessionKey);
};
/* calculate request-digest/response-digest as per HTTP Digest spec */
void DigestCalcResponse(
    IN HASHHEX HA1, /* H(A1) */
    IN char * pszNonce, /* nonce from server */
    IN char * pszNonceCount, /* 8 hex digits */
    IN char * pszCNonce, /* client nonce */
    IN char * pszQop, /* qop-value: "", "auth", "auth-int" */
    IN char * pszMethod, /* method from the request */
    IN char * pszDigestUri, /* requested URL */
    IN HASHHEX HEntity, /* H(entity body) if qop= "auth-int" */
    OUT HASHHEX Response /* request-digest or response-digest */
    )
{
    MD5_CTX Md5Ctx;
    HASH HA2;
    HASH RespHash;
    HASHHEX HA2Hex;
    // calculate H(A2)
```

578