आउटलुक पासवर्डस्

पास्केप सोफ्टवेर (हिन्दी अनुवाद : धीरेन कुमार)

परिचय

1 परिचय

"मजबूत पासवर्ड का उपयोग करें जो अपरकेस अक्षरों, संख्याओं और प्रतीकों को मिलकर बनता हैं। कमजोर पासवर्ड इन तत्वों को नहीं मिलाते हैं। मजबूत पासवर्ड: Y6dh!et5। कमजोर पासवर्ड: House27।"

MS ऑफिस आउटलुक यूजर मैनुअल से अंश

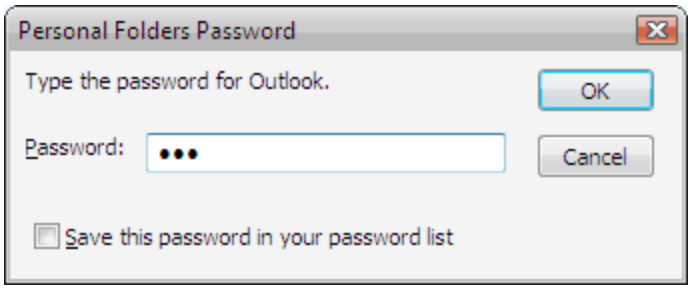
यह लेख मूल रूप से आपको आउटलुक की **PST** फाइलों में एक अजीब पासवर्ड टकराव के बारे में बताने के लिए था। बाद में, यह प्रदर्शित करने के लिए इसका विस्तार किया गया कि कमियों के बावजूद, आउटलुक के फायदे अपने निकटतम प्रतिस्पर्धियों से कहीं अधिक हैं, साथ ही साथ व्यक्तिगत डेटा को संग्रहीत करने के लिए उपयोग की जाने वाली तकनीकों की व्याख्या करने के लिए। इसके अलावा, उदाहरण के रूप में आउटलुक का उपयोग करके क्रिप्टोग्राफी के विकास का अनुसरण करना बहुत सुविधाजनक है। इसे आम तौर पर समग्र रूप से विंडोज ऑपरेटिंग सिस्टम की पूरी लाइन के विकास के लिए पेश किया जा सकता है।

PST सुरक्षा पासवर्ड

2 PST सुरक्षा पासवर्ड

तो, आइए इस बिंदु से शुरू करते हैं कि Microsoft Office Outlook की .PST फ़ाइल एक स्थानीय कंप्यूटर पर फ़ाइल-प्रकार डेटा संग्रहण है, जो एक निश्चित क्रम द्वारा व्यवस्थित संपर्कों, नोट्स, ई-मेल संदेशों और अन्य वस्तुओं को संग्रहीत करता है। ई-मेल संदेशों को वितरित करने के लिए एक .PST फ़ाइल का उपयोग डिफ़ॉल्ट स्थान के रूप में किया जा सकता है। इसका उपयोग डेटा को ऑर्डर करने और बैकअप करने के लिए भी किया जा सकता है।

किसी .PST फ़ाइल की सामग्री की सुरक्षा के लिए और तीसरे पक्ष द्वारा उस तक अनधिकृत पहुंच को प्रतिबंधित करने के लिए, कोई भी अधिकतम 15 वर्णों का पासवर्ड सेट कर सकता है (चित्र 1)। ऐसे मामले में, आमतौर पर यह सोचा जाता है कि .PST फ़ाइल तब तक नहीं खोली जा सकती जब तक कि कोई मूल पासवर्ड नहीं जानता। आइए देखें कि यह कितना सच है।



चित्र 1. आउटलुक PST पासवर्ड डायलोग।

A.PST फ़ाइल एक्सेस पासवर्ड को न तो याद किया जाता है और न ही स्पष्ट रूप में संग्रहीत किया जाता है। इसके बजाय, कंप्यूटर पासवर्ड के हैश वेल्यु की गणना करता है और इसे .PST फ़ाइल में संग्रहीत करता है या, यदि विकल्प 'इस पासवर्ड को अपनी पासवर्ड सूची में सहेजें' (चित्र 1) चुना गया है, तो **Windows रजिस्ट्री** में, अतिरिक्त रूप से एन्क्रिप्ट किया गया है।

सबसे दिलचस्प बात यह है कि पासवर्ड हैश गणना एल्गोरिदम वास्तविक हैशिंग एल्गोरिदम नहीं है - बल्कि यह एक साधारण **CRC32** चेकसम गणना रूटीन है। CRC32 एक अतिरेक जाँच एल्गोरिथ्म है, और यह निश्चित रूप से हैशिंग रूटीन नहीं है। एक या किसी अन्य कारण से, Microsoft ने **SHA-1** जैसे मजबूत एल्गोरिदम के बजाय उस एल्गोरिथ्म का उपयोग करने का निर्णय लिया था। इसे आउटलुक के पुराने वर्जनो से विरासत के रूप में रखते हुए, MS लंबे समय से एल्गोरिदम को नहीं बदल रहा है, जैसा कि लगता है, पिछड़े संगतता विचारों द्वारा नेतृत्व किया जा रहा है। दूसरी ओर, यह अभी भी स्पष्ट नहीं है कि आउटलुक 2003 जारी होने पर इसे क्यों नहीं बदला गया था (आउटलुक 2003 के .PST प्रारूप में 64-बिट आंतरिक एड्रेसिंग है और पिछले वर्जनो के साथ संगत नहीं है)।

इस प्रकार, एक पासवर्ड हैश उसके चेकसम के 32 बिट्स प्रतीत होता है। आइए CRC32 के प्रदर्शन को करीब से देखें:

```
DWORD CPstReader::Crc32( LPBYTE pPassword )
{
   assert( pPassword );

   //set initial crc to zero
   DWORD crc=0;

   //till the end of string
   while ( *pPassword )
      crc = (crc>>8) ^ pCRCTable[(BYTE)crc ^ (*pPassword++)];
```

```
    return crc;
}
```

जैसा कि स्रोत कोड के अंश से देखा जा सकता है, पासवर्ड (pPassword) इनपुट पर फीड किया जाता है, इसका चेकसम आउटपुट पर दिखाई देता है। CRC32 की सबसे कमजोर बात यह है कि पासवर्ड हैश की **32-बिट लंबाई** स्पष्ट रूप से पर्याप्त लंबी नहीं है, और इसलिए दो अलग-अलग पासवर्ड के चेकसम मेल खा सकते हैं! उदाहरण के लिए, पासवर्ड **1** और **orxgnm** या **mozart** और **2920347097** के लिए चेकसम समान है। यदि आपको लगता है कि इस तरह के टकराव दुर्लभ हैं, तो आप बहुत गलत हैं।

ये टकराव कितनी आश्चर्यजनक बातें हैं! रिवर्स निर्भरता अक्सर पर्याप्त होती है: पासवर्ड जितना लंबा होगा, उस पासवर्ड के लिए टकराव मिलान उतना ही आसान होगा। आइए आउटलुक के यूजर मैनुअल के उदाहरण पर एक नज़र डालें, जिसके अंश को इस लेख की प्रस्तावना के रूप में चुना गया था, जिसे जानबूझकर चुना गया था। एक आसान संयोजन है जो हमारे 'विश्वसनीय' पासवर्ड **Y6dh!et5** को आसानी से बदल सकता है - यह एक 5-वर्ण स्ट्रिंग **JlSfw** है।

फिर भी एक और दिलचस्प अवलोकन: यदि एक .PST फ़ाइल में संग्रहीत चेकसम शून्य के बराबर है, तो प्रोग्राम 'सोचता है' कि कोई पासवर्ड सेट नहीं है। हालाँकि, चूंकि हम जानते हैं कि समान चेकसम वाले पासवर्ड होते हैं, हम मान सकते हैं कि ऐसे पासवर्ड हैं, जो चेकसम भी शून्य के बराबर हैं। ऐसे पासवर्ड वास्तव में मौजूद हैं। यहां बड़ी सूची का एक छोटा हिस्सा दिया गया है: **1Rj78C, 5J8j84**, **ArTniW**।

Crc32("1Rj78C")=0, Crc32("5J8j84")=0, Crc32("ArTniW")=0. यदि हम अपनी .PST फ़ाइल की सुरक्षा के लिए इनमें से कोई एक पासवर्ड सेट करते हैं, तो हमारे पास वास्तव में वह फ़ाइल असुरक्षित होगी, और जब भी कोई अगली बार उस तक पहुँचने का प्रयास करेगा, तो वह पासवर्ड के लिए संकेत भी नहीं देगा। विश्वास नहीं है? - इसे स्वयं आज़माएं।

एक प्रयोग से पता चला है कि ब्रूट फोर्स अटैक का उपयोग करके आउटलुक हैश पासवर्ड को रिकवर करने में औसतन लगभग एक मिनट का समय लगता है। हालाँकि, CRC32 के क्रिप्टो विश्लेषण से पता चला है कि एल्गोरिथ्म छोटे पासवर्ड (4 वर्णों तक) के लिए पूरी तरह से प्रतिवर्ती है और अन्य सभी के लिए आंशिक रूप से प्रतिवर्ती है। इसका मतलब है कि, कोई भी मूल पासवर्ड या उसके CRC32 समकक्ष पासवर्ड को रिकवर कर सकता है, जो कि आउटलुक के लिए लगभग तुरंत ही अप्रभेद्य होगा। यह साबित हो चुका है कि टक्कर लेने के लिए 7 से अधिक वर्णों की आवश्यकता नहीं है (मूल पासवर्ड के समान चेकसम वाला पासवर्ड)।

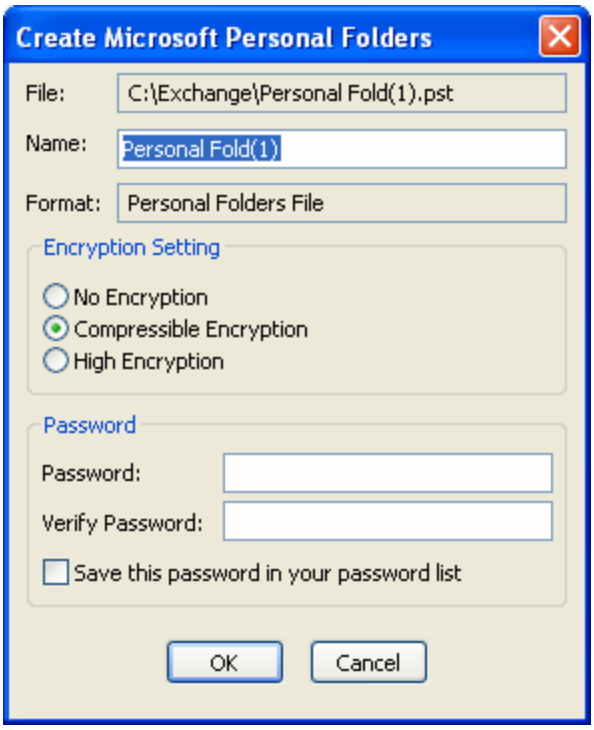
PST एन्क्रिप्शन

3 PST एन्क्रिप्शन

यदि हम PST फ़ाइल विकल्पों (चित्र 2) को देखें, तो हम देख सकते हैं कि आउटलुक, बाकी सब चीजों के अलावा, सामग्री को एन्क्रिप्ट करने की अनुमति देता है। ऐसे मामले में, पासवर्ड चेकसम को .PST फ़ाइल में खुले रूप में नहीं रखा जाता है; इसके बजाय, यह अतिरिक्त रूप से एक एन्क्रिप्शन एल्गोरिथम के साथ एन्क्रिप्ट किया गया है। आइए एन्क्रिप्शन के लिए उपयोग किए जाने वाले एल्गोरिदम की समीक्षा करें।

एक नई .PST फ़ाइल के निर्माण के दौरान, आउटलुक हमें 3 फ़ाइल प्रकारों में से चुनने के लिए प्रेरित करता है:

1. एन्क्रिप्ट नहीं किया गया
2. कोम्प्रेसिबल एन्क्रिप्शन
3. मजबूत एन्क्रिप्शन



चित्र 2. नई PST फ़ाइल बनाना।

यदि किसी PST फ़ाइल के लिए **कोई एन्क्रिप्शन नहीं** चुना गया है, तो यूजर के सभी डेटा: संपर्क, संदेश, पासवर्ड, आदि अन्य यूजर्स के लिए उपलब्ध खुले रूप में संग्रहीत किए जाएंगे। उस डेटा को, उदाहरण के लिए, एक टेक्स्ट एडिटिंग प्रोग्राम के साथ देखा जा सकता है।

कोम्प्रेसिबल एन्क्रिप्शन एल्गोरिथ्म इस तरह से बनाया गया है कि एन्क्रिप्ट किए जा रहे प्रत्येक वर्ण को एक विशेष टेबल से लिए गए एक अलग वर्ण के साथ प्रतिस्थापित किया जाता है। यहाँ एल्गोरिथ्म है:

```
BOOL CPstReader::Decrypt1( LPBYTE buf, int iSize )
{
   assert( buf );

   BYTE y=0;
   int x=0;
```

```
    //Check input buffer
    if ( buf==NULL )
        return FALSE;

    //Check encryption type
    if ( m_pst.encryption!=PST_ENCRYPT_COMPESSIBLE )
        return FALSE;

    //actual decryption
    while ( iSize-- )
    {
        y=buf[x];
        buf[x++]=m_pTable[y];
    }

    return TRUE;
}
```

प्रतिस्थापन तालिका (**m_pTable**) को अर्थपूर्ण तरीके से बनाया गया है जो इस एल्गोरिथम के साथ एन्क्रिप्ट किए गए टेक्स्ट के इष्टतम आगे कोम्प्रेसन की अनुमति देता है। हालाँकि, एल्गोरिथम स्वयं सामग्री को संपीड़ित नहीं करता है; यह केवल उसके लिए अनुकूल परिस्थितियों का निर्माण करता है।

मजबूत एन्क्रिप्शन भी पहले प्रतिस्थापन एल्गोरिथ्म का एक प्रकार है। हालांकि, पिछले एक के विपरीत, यह बहुत मजबूत एन्क्रिप्शन प्रदान करता है। इस एल्गोरिथम में एक और अंतर यह है कि इस एल्गोरिथम के साथ एन्क्रिप्टेड PST फ़ाइल को उतना कॉम्पैक्ट नहीं बनाया जा सकता जितना कि इसे पहली विधि से बनाया जा सकता है।

```
BOOL CPstReader::Decrypt2( LPBYTE buf, int iSize, DWORD id )
{
    assert( buf );

    int x=0;
    BYTE y=0;
    WORD wSalt;

    //Check input buffer
    if ( buf==NULL )
        return FALSE;

    //Check encryption type
    if ( m_pst.encryption!=PST_ENCRYPT_STRONG )
        return FALSE;

    //prepare encryption key from block ID
    wSalt=HIWORD(id) ^ LOWORD(id);

    //actual decryption
    while ( iSize-- )
    {
```

```
        y=buf[x];

        y+=LOBYTE(wSalt);
        y=m_pTable2[y];

        y+=HIBYTE(wSalt);
        y=m_pTable2[y+0x100];

        y-=HIBYTE(wSalt);
        y=m_pTable2[y+0x200];

        buf[x++]= y - LOBYTE(wSalt++);
    }

    return TRUE;
}
```

डेटा के एक ब्लॉक को रिकवर करने के लिए, उस ब्लॉक पहचानकर्ता का पता होना चाहिए। उस पहचानकर्ता के बिना रिकवरी प्रक्रिया कुछ कठिन प्रतीत होगी। फिर से, यदि हम एल्गोरिथम के स्रोत कोड पर करीब से नज़र डालते हैं, तो हम आसानी से देख सकते हैं कि केवल 16-बिट ब्लॉक पहचानकर्ता का उपयोग किया जाता है, और यह एन्क्रिप्शन की को ब्रूट फोर्स अटैक के साथ चुनने के लिए पुनरावृत्तियों की संख्या को बहुत कम करने की अनुमति देता है।

ई-मेल अकाउन्ट पासवर्ड संग्रहीत करने की तकनीक

4 ई-मेल अकाउन्ट पासवर्ड संग्रहीत करने की तकनीक

जैसा कि आप इस लेख को पढ़ रहे हैं, आपको कुछ हद तक गलत विचार मिल सकता है कि MS आउटलुक केवल कमजोर, अविश्वसनीय पासवर्ड एन्क्रिप्शन एल्गोरिदम का उपयोग करता है। ऐसा बिलकुल नहीं है। सत्य, जैसा कि कहा जाता है, तुलना में पाया जाएगा। इतना ही नहीं, हमने केवल PST का विश्लेषण किया है। आइए अब अन्य लोकप्रिय कार्यक्रमों के साथ ई-मेल खाते के लिए उपयोग किए जाने वाले आउटलुक के पासवर्ड एन्क्रिप्शन तंत्र की समीक्षा करें और तुलना करें।

अधिकांश लोकप्रिय ई-मेल क्लाइंट प्रोग्राम में संग्रहीत यूजर्स के पासवर्ड की सुरक्षा की परवाह नहीं करते हैं। उदाहरण के लिए, Eudora, TheBat! या नेटस्केप के पुराने वर्जन यूजर के डेटा को एन्क्रिप्ट करने के लिए पुरातन **BASE64** एल्गोरिथम या इसकी व्युत्पत्तियों का उपयोग करते हैं। IncrediMail केवल **XOR** गामा को मूल पासवर्ड पर लागू करता है (एक जीभ इस चीज़ को एन्क्रिप्शन कहने के लिए तैयार नहीं है), ओपेरा ब्राउज़र के पुराने वर्जनो ने ई-मेल पासवर्ड को बिल्कुल भी एन्क्रिप्ट नहीं किया था; उन्होंने उन्हें प्लेन टेक्स्ट के रूप में संग्रहीत किया!

लोकप्रिय ई-मेल क्लाइंट Thunderbird, Opera M2, और आउटलुक एक्सप्रेस के नए वर्जनो के साथ समस्या थोड़ी बेहतर है। ये सभी विश्वसनीय, समय-सिद्ध एल्गोरिदम का उपयोग करते हैं जिसमें माहिर की होती है। यह आम तौर पर **MD5 + RC4** या **SHA + 3DES** या उनके व्युत्पन्न का एक बंडल है। Thunderbird, Opera M2 और आउटलुक एक्सप्रेस एन्क्रिप्टेड पासवर्ड के साथ अपनी मास्टर की और एन्क्रिप्शन की स्टोर करते हैं। यह उन पासवर्ड को मूल रूप से (या लगभग मूल रूप से) वापस करने की अनुमति देता है।

आउटलुक से क्या संबंधित है (वैसे, इसे आउटलुक एक्सप्रेस के साथ न मिलाएं), यहां हम सबसे दिलचस्प तस्वीर देखते हैं। आउटलुक के ई-मेल खातों के लिए विकास पासवर्ड स्टोरेज तकनीकों के संपूर्ण कालक्रम को चार अवधियों में विभाजित किया जा सकता है:

- प्रागैतिहासिक काल
- पाषाण युग
- मध्य युग
- तकनीकी प्रगति आयु

अभी और विवरण।

4.1 प्रागैतिहासिक काल

प्रागैतिहासिक काल - यह प्रथम सोपानों का काल है। वे कहते हैं कि प्रोग्राम के पहले वर्जन **BASE64** एल्गोरिथम के साथ रजिस्ट्री में संग्रहीत पासवर्ड को एन्क्रिप्ट करने में सक्षम थे। यह पहले से ही उस समय के स्टेंडर्ड से एक उपलब्धि थी। इस तरह के पासवर्ड को रिकवर करने के लिए एक कैलकुलेटर और दिमाग में कुछ कनवल्शन की जरूरत होती है।

4.2 पाषाण युग

पाषाण युग - आउटलुक 9x - पहले प्रयोगों का समय। एन्क्रिप्शन की और अद्वितीय रिकॉर्ड पहचानकर्ता का उपयोग करने वाला एक नया एन्क्रिप्शन एल्गोरिथम इस समय प्रकट होता है। यह उम्र का ज्ञान है। इस एल्गोरिथम के साथ एन्क्रिप्ट किया गया डेटा की और रिकॉर्ड पहचानकर्ता के बिना रिकवर नहीं किया जा सकता है। आज के कई ई-मेल क्लाइंट आज भी

अपने शस्त्रागार में इस तरह के एल्गोरिथम होने के बारे में डींग नहीं मार सकते हैं। हालांकि, उस प्रकार के एन्क्रिप्शन के सामान्य विचार को एक महत्वपूर्ण कमजोरी से गहरा कर दिया गया था - एन्क्रिप्शन की और रिकॉर्ड पहचानकर्ता को एन्क्रिप्टेड डेटा के साथ रजिस्ट्री में संग्रहीत किया गया था।

4.3 मध्य युग

मध्य युग - आउटलुक 2000 - पहला स्टैंडर्ड। ई-मेल अकाउन्ट पासवर्ड अब विंडोज़ के प्रोटेक्टेड स्टोरेज में रखा गया था (आगामी लेखों में से एक प्रोटेक्टेड स्टोरेज को अधिक विस्तार से कवर करेगा)। प्रोटेक्टेड स्टोरेज में पासवर्ड डिक्रिप्शन एल्गोरिथम यहां दिया गया है:

1. Key1 को SHA(Salt) + SHA(SID) + SHA(Salt) का उपयोग करके मास्टर की को डिक्रिप्ट करने के लिए बनाया गया है। Salt एक वैश्विक कोन्सटन्ट है। SID यूजर पहचानकर्ता है।
2. Key2 को SHA(MKSalt) + SHA(key1) का उपयोग करके मास्टर की को डिक्रिप्ट करने के लिए बनाया गया है। MKSalt इसके साथ संग्रहीत प्रत्येक मास्टर की के लिए अद्वितीय बाइनरी डेटा है। Key1 पिछले चरण पर प्राप्त 20 बाइट्स डेटा है।
3. मास्टर की को DES एल्गोरिथम और Key2 के साथ डिक्रिप्ट किया गया है।
4. रिकवरी मास्टर की डेटा एन्क्रिप्शन की के डिक्रिप्शन में भाग लेती है। डेटा एन्क्रिप्शन की डेटा के साथ ही संग्रहीत होती है और प्रत्येक डेटा रिकॉर्ड के लिए अलग होती है। इसमें 16 बाइट्स होते हैं, जिनमें से पहले आधे भाग का उपयोग डिक्रिप्शन के लिए किया जाता है, और दूसरे आधे भाग का उपयोग वैधता जांच के लिए किया जाता है।
5. अब, इस डिक्रिप्टेड डेटा की का उपयोग करके कोई भी डेटा रिकॉर्ड को स्वयं डिक्रिप्ट कर सकता है (पासवर्ड, क्रेडेंशियल और अन्य संवेदनशील जानकारी)।

क्या बढ़िया और स्टाइलिश विचार है! इस एल्गोरिथ्म की प्रमुख विशेषताएं:

- सभी रिकॉर्ड के लिए एक ही मास्टर की है। इसलिए, इसे एक ही बार डिक्रिप्ट करना पर्याप्त है (चरण 1-3)। यह पूरी तरह से सुरक्षा स्तर को नीचे लाए बिना डिक्रिप्शन प्रक्रिया को गति देता है।
- यूजर SID मास्टर की के डिक्रिप्शन में भाग लेता है; इसलिए, प्रत्येक यूजर के पास एक अद्वितीय मास्टर की होगी। इस प्रकार, यह स्पष्ट हो जाता है कि डेटा को तब तक डिक्रिप्ट करना असंभव है जब तक कि कोई SID (जो प्रत्येक यूजर के लिए अद्वितीय है) को नहीं जानता। हालाँकि, SID को मास्टर की के साथ रजिस्ट्री में संग्रहीत किए जाने से यह कुछ हद तक काला हो जाता है।
- यह एल्गोरिथम, हालांकि यह 10 साल पहले बनाया गया था, फिरब भी इन दिनों रेन्बो-टेबल का उपयोग करके इतने लोकप्रिय हमलों के खिलाफ दृढ़ है।

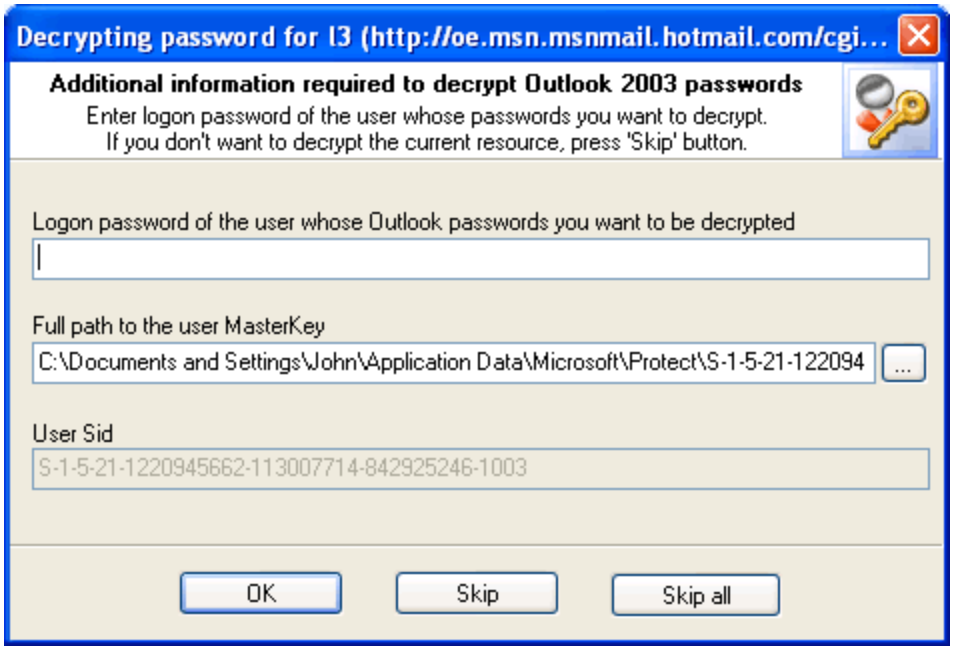
सुझाया गया एन्क्रिप्शन रूटीन, जो वैसे, पहले से ही इस्तेमाल किया जा रहा था जब इंटरनेट एक्सप्लोरर 4 जारी किया गया था, डेटा एन्क्रिप्शन में एक बड़ी सफलता थी। सभी आधुनिक ब्राउज़र (ओपेरा, मोज़िला, फ़ायरफ़ॉक्स, और इंटरनेट एक्सप्लोरर, वर्जन 6 तक) समान पासवर्ड एन्क्रिप्शन योजनाओं का उपयोग करते हैं। ध्यान दें कि एन्क्रिप्शन तकनीक के मध्य युग में सर्वश्रेष्ठ आउटलुक के सबसे मजबूत प्रतिस्पर्धियों की रचनात्मक कल्पना यहीं रुक गई है।

4.4 तकनीकी प्रगति आयु

तकनीकी प्रगति युग - आउटलुक 2003 दुनिया से आगे बढ़ रहा है। लोकप्रिय ई-मेल क्लाइंट का नया वर्जन एक नए एन्क्रिप्शन एल्गोरिथम का उपयोग करता है, जो जारी रहता है और पुराने को तार्किक रूप से विकसित करता है। यह एल्गोरिथम एक महत्वपूर्ण वस्तु पर आधारित है - यह यूजर के पासवर्ड के लिए बाध्य है। यह लेख एल्गोरिथम के कार्य के

विवरण का वर्णन करने के लिए नहीं था, इसके लिए पाठ के कुछ पृष्ठ लगेंगे। इसके बजाय, हम केवल यह उल्लेख करेंगे कि इस एल्गोरिथम के साथ एन्क्रिप्ट किए गए पासवर्ड को रिकवर करने के लिए कम से कम तीन चीजें जानने की जरूरत है (चित्र 3):

1. यूजर की मास्टर की
2. यूजर का SID
3. यूजर का पासवर्ड



चित्र 3. आउटलुक 2003 ई-मेल अकाउन्ट पासवर्ड रिकवरी।

संक्षेप में, नए DPAPI एल्गोरिथम के लाभ हैं:

- मास्टर की अब स्थानीय कंप्यूटर पर एक अलग फ़ोल्डर में संग्रहीत है। उस फ़ोल्डर तक पहुंच आंशिक रूप से प्रतिबंधित है।
- मास्टर की की लंबाई 512 बिट है, जो निकट भविष्य में एक मैच चुनने की संभावना को समाप्त करती है।
- उपयोग किए गए नए एन्क्रिप्शन एल्गोरिदम, विशेष रूप से SHA-HMAC, जो लूप में पुनरावृत्तियों की एक वेरिएबल संख्या (डिफ़ॉल्ट रूप से 4000) का उपयोग करते हैं।
- एन्क्रिप्शन एल्गोरिदम (मास्टर की और वास्तविक डेटा दोनों के लिए) पूरी तरह से अनुकूलन योग्य हैं। ऑपरेटिंग सिस्टम द्वारा समर्थित कोई भी गुण सेट कर सकता है।
- ऑपरेटिंग सिस्टम के अनुमति स्तर का उपयोग करके डेटा सुरक्षा को महसूस किया जा सकता है।
- एल्गोरिथ्म यूजर के लॉगऑन पासवर्ड के लिए बाध्य है।

लॉग इन यूजर के लिए पासवर्ड का एन्क्रिप्शन बिल्कुल पारदर्शी है। पासवर्ड केवल एक बार मांगा जाता है, जब यूजर ऑपरेटिंग सिस्टम पर लॉग ऑन करता है। ऑपरेटिंग सिस्टम बाकी का ख्याल रखता है। यहां तक कि अगर एक संभावित हैकर एन्क्रिप्टेड डेटा तक भौतिक पहुंच प्राप्त करता है, तो वह उस डेटा को तब तक डिक्रिप्ट नहीं कर पाएगा जब तक कि वह यूजर का पासवर्ड नहीं जानता।

निष्कर्ष

5 निष्कर्ष

हम देखते हैं कि इस लोकप्रिय ई-मेल क्लाइंट के प्रत्येक नए वर्जन के आगमन के साथ, Microsoft कुछ नया पेश करता है, जो उस समय तक ज्ञात नहीं था, इस प्रकार यह साबित करता है कि वे वास्तव में अंतिम-यूजर की सुरक्षा की परवाह करते हैं। नया आउटलुक 2003 का एन्क्रिप्शन तंत्र विशेष रूप से बढ़िया है।

क्या आप उत्सुक नहीं हैं, कि अगले वर्जन, आउटलुक 2007 का विमोचन क्या उजागर करेगा? आइए मान लें कि PST पासवर्ड जो अब विंडोज़ रजिस्ट्री में संग्रहीत हैं और इसके अलावा आउटलुक 9x (जब हम पाषाण युग के बारे में बात कर रहे थे, हमने इस विधि को कवर किया था) को एन्क्रिप्ट किया गया था क्योंकि वे अब एन्क्रिप्टेड हैं आउटलुक 2003 में। कम से कम, यह तार्किक होगा। ई-मेल अकाउंट पासवर्ड के लिए कुछ भविष्यवाणी करना मुश्किल है। हम नए आउटलुक 2003-आधारित DPAPI एन्क्रिप्शन एल्गोरिथम या प्रोटेक्टेड स्टोरेज + यूजर के पासवर्ड के एक नए बंडल के आगे के विकास को देखने की सबसे अधिक संभावना रखते हैं। एक तरह से या दूसरे, हम इस लोकप्रिय एप्लिकेशन के नए वर्जन के जारी होने की प्रतीक्षा करेंगे, और हम निश्चित रूप से आगामी लेखों में से एक में कार्यक्रम में उपयोग किए जाने वाले एन्क्रिप्शन तंत्र के बारे में बताने जा रहे हैं।