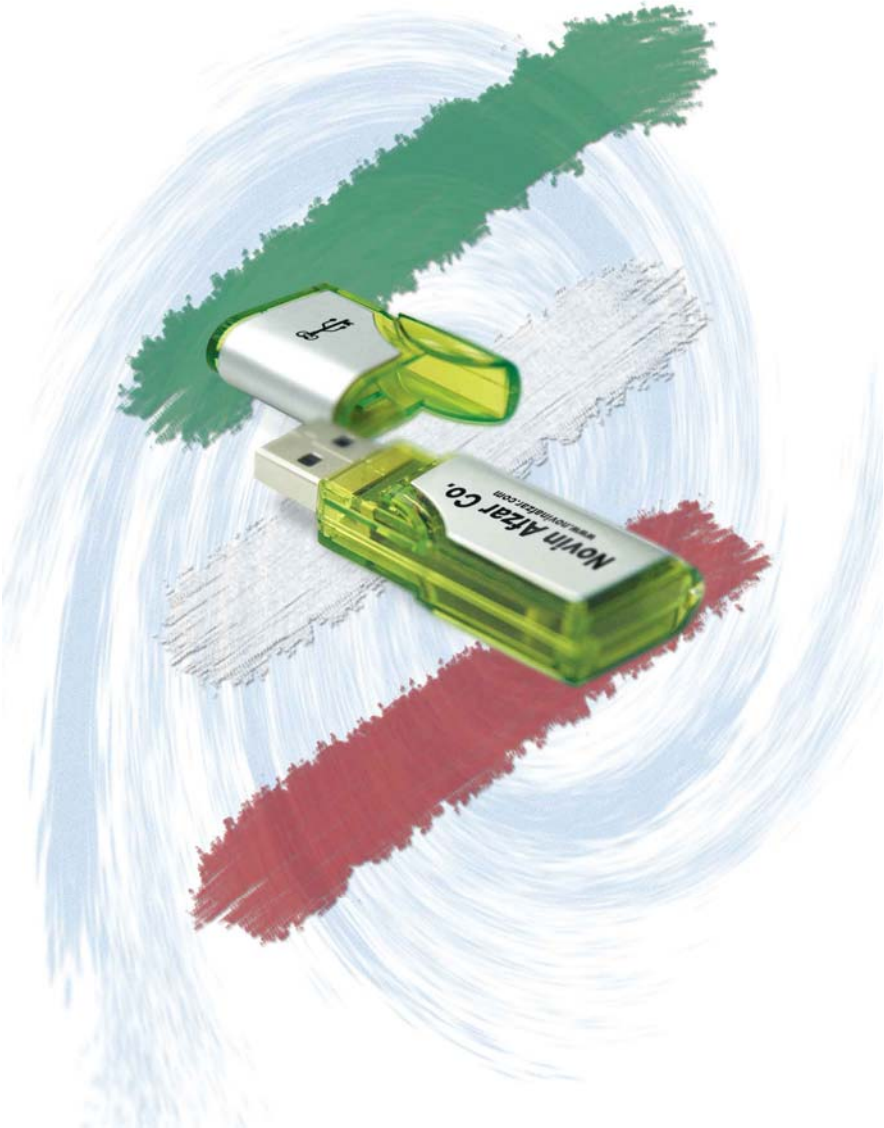
راهنمای استفاده از شناسه نوین

نسخه مخصوص برنامه نویسان



نگارش ۱-۰۸

تابستان ۱۳۸۷

فهرست مطالب

فصل اول: مشخصات شناسه نوین ۱
۱-۱- تنظیمات سمت تولید کننده ۱
۱-۱-۱- شماره سریال شناسه ۱
۱-۱-۲- کد اختصاصی نمایندگی یا مشتری (VID: Vendor ID) ۱
۱-۲- تنظیمات عمومی ۲
۱-۲-۱- رمز عبور مدیریت ۲
۱-۲-۲- رمز عبور برنامه نویس ۲
۱-۲-۳- بخش داده آزاد ۳
۱-۲-۴- تنظیمات سطوح دسترسی برای بخش داده ۳
۱-۳- تنظیمات کاربر نهایی ۴
۱-۳-۱- رمز عبور کاربر (پین کد) ۴
۱-۳-۲- کلید رمزنگاری AES کاربر ۴
۱-۳-۳- کلیدهای رمزنگاری RSA ۴

فصل دوم: نرم افزار برنامه ریزی (Builder) ۶
۲-۱- ساختار ظاهری ۶
۲-۱-۱- روال یک: تولید فایل تنظیمات شناسه (NTC) ۸
۲-۱-۲- روال دو: تغییر فایل تنظیمات شناسه (NTC) ۱۴
۲-۱-۳- روال سه: برنامه ریزی شناسه با (NTC) ۱۷
۲-۱-۴- روال چهار: برنامه ریزی دستی (مرحله به مرحله) ۱۹
۲-۱-۵- ریست/فعالسازی ۲۴

فصل سوم: کتابخانه رابط (dll) ۲۶
۳-۱- ارتباط با شناسه در برنامه نویسی ۲۶
۳-۲- تعریف روالها ۲۶

فصل چهارم: کاربردهای شناسه نوین ۴۸
۴-۱- احراز هویت کاربران به روش OTP ۴۸
۴-۲- رمزنگاری نامتقارن RSA و امضای دیجیتال ۴۹

پیوست ۱: توضیح نمونه کد ۵۰
توضیح نمونه کد به زبان PHP ۵۰
توضیح نمونه کد به زبان ASP ۵۹

پیوست ۲: کد خطاها ۶۸

فصل اول: مشخصات شناسه نوین

شناسه نوین در واقع یک میکروکنترلر کوچک هست که از طریق پورت USB به کامپیوتر وصل می‌شود . این شناسه یک نرم افزار داخل میکروکنترلر خود دارد که سرویس های مختلفی را در اختیار کامپیوتر قرار می‌دهد.

برای تنظیم سرویسها پارامترهای (ثابت ها) مختلفی داخل شناسه تنظیم می‌شود که از لحاظ روال تنظیم آنها می‌توان در سه دسته طبقه بندی کرد.

۱-۱- تنظیمات سمت تولید کننده

دسته ای از تنظیمات را شرکت تولید کننده (نوین افزار) روی شناسه تنظیم می‌کند که این پارامترهای فقط توسط تولید کننده قابل ایجاد و دستکاری می‌باشد.

۱-۱-۱- شماره سریال شناسه

این عدد توسط شرکت سازنده به شناسه اختصاص می‌یابد و ساختاری شبیه آدرس IP شبکه را دارد. شماره سریال یک عدد یکتا است که می‌توان از آن در شناسائی کاربران انتهایی (End-User) از آن استفاده کرد. یعنی هیچ دو شناسه ای وجود ندارد که شماره سریال یکسان داشته باشند. در شماره سریال (به طور مثال 8.0.0.1) اولین قسمت سال تولید آنرا نشان می‌دهد. بطور مثال ۸ نشاندهنده سال ۲۰۰۸ می‌باشد.

۱-۱-۲- کد اختصاصی نمایندگی یا مشتری (VID: Vendor ID)

برای افرادی که شناسه خام خریداری می‌کنند، یک کد اختصاصی یا VID توسط شرکت سازنده بطور رایگان تخصیص داده می‌شود که در تمام مراحل استفاده از شناسه این کد لازم است، این کد از سه یا چهار عدد بین (۲۵۵-۰) که با نقطه از هم جدا شده اند (شبیه به آدرس IP شبکه) تشکیل شده است. این کد جزو سطح امنیت شناسه ها به حساب آمده و فقط توسط شرکت سازنده قابل اختصاص است. پس از دریافت اولین سری از شناسه های خریداری شده، در خریدهای بعدی برای دریافت شناسه های خام با همان VID قبلی مدارک قانونی خریدار (مانند درخواست با سربرگ شرکت) لازم است. بدین گونه شناسه ها با VID مشخص فقط به صاحب قانونی آن VID تحویل داده می‌شود. به طور معمول یک شماره VID به هر مشتری اختصاص می‌یابد ولی مشتری می‌تواند بنا به درخواست خویش چندین VID را برای خود اختصاص دهد.

در صورتی که طرف خریدار مشتری اصلی شناسه باشد VID چهار قسمتی ارائه می‌شود ولی VID مربوط به شناسه هایی که به صورت نمایندگی تولید کننده هستند سه قسمتی می‌باشد که قسمت چهارم

توسط نماینده تنظیم و به مشتریان تحویل می‌گردد. توجه داشته باشید که سه قسمت اول VID (از سمت چپ) VID ریشه[1] و قسمت آخر زیر VID[2] نامیده می‌شود.

۱-۲- تنظیمات عمومی

این تنظیمات شامل پارامترهایی می‌شود که تغییر یا تنظیم آن فقط توسط سطح دسترسی مدیریت شناسه امکانپذیر است و پس از تنظیم سمت کاربر نهایی غیرقابل تغییر است.

۱-۲-۱- رمز عبور مدیریت[3]

سطح دسترسی برای تنظیم کردن یا تغییر پارامترهای شناسه رمز عبور Admin است که حداکثر ۱۶ رقم می‌باشد. به دلایل امنیتی، این سطح دسترسی را نباید به غیر از نرم افزار برنامه ریز شناسه (Builder) در جای دیگر استفاده کرد. همچنین نباید در اختیار سایر افراد مانند برنامه نویسها قرار داد. این رمز را حتما به خاطر بسپارید، زیرا که در صورت disable شدن شناسه، این رمز برای ریست کردن یا فعالسازی شناسه لازم است. همچنین در صورت نیاز به ریست کردن رمزعبور کاربر (PIN Code) داشتن رمزعبور مدیریت الزامی است.

۱-۲-۲- رمز عبور برنامه نویس[4]

برای استفاده از متدهای شناسه در برنامه نویسی از این سطح امنیت می‌توان استفاده کرد. که ساختار آن مانند رمز عبور قبلی می‌باشد. در خواندن یا نوشتن قسمت داده ممکن است این رمز عبور لازم شود.

نکته ۱: برای رمزهای عبور مدیریت و برنامه نویس حتما باید دو کلمه مجزا استفاده کرد.

نکته ۲: توجه داشته باشید چهار بار استفاده مکرر از رمز عبور و VID نادرست باعث می‌شود شناسه disable شود. یعنی زمانی که VID، رمزعبور مدیریت یا برنامه نویس چهار بار یا رمز عبور کاربر نهایی چهار بار اشتباه زده شود این اتفاق می‌افتد. برای خروج از این حالت می‌توان شناسه را با داشتن عبور مدیریت توسط نرم افزار برنامه ریزی ریست کرد. در صورتی که این عمل نیز چهار بار با رمزعبور مدیریت یا VID اشتباه انجام گردد شناسه به حالت مسدود (Blocked) رفته و خروج از این حالت فقط توسط شرکت سازنده طی روال گارانتی امکانپذیر است.

نکته ۳: در صورتی که رمزهای عبور را در برنامه نویسی استفاده می‌کنید از پاس کردن این کلمات به صورت رشته ساده به سرویسهای شناسه خودداری کنید. زیرا که فرم ارسال به سرویسها به صورت Delimitered String (جدا شده با نقطه) است، که این کار را با استفاده از متدهای تبدیل به راحتی قابل

---

[1] Root VID
[2] Sub VID
[3] Admin Password
[4] Developer Password

انجام است. رمز عبور باید به صورت کد ASCII استفاده شود به طور مثال رمز عبور '123' باید به صورت '49.50.51.0.0.0.0.0.0.0.0.0.0.0.0.0' استفاده شود.(کد ASCII کاراکتر 1 برابر 49 ، 2 برابر 50 و 3 برابر 51 است). برای تبدیل این دو مدل به همدیگر یا به مدل HEX از توابع تبدیل که با Conv شروع می‌شود استفاده کنید.

۱-۲-۳- بخش داده آزاد

این بخش ۳۲ بایت حافظه قابل خواندن و نوشتن را در اختیار قرار می‌دهد. که می‌توان در برنامه نویسی با توجه به تظیمات سطوح دسترسی آن با یک سطح دسترسی مناسب در آن مقادیری را نوشت یا از آن خواند.

۱-۲-۴- تنظیمات سطوح دسترسی برای بخش داده

بخش داده ۳۲ بایتی شناسه را می‌توان به حداکثر ۳ بخش تقسیم کرد و برای هر بخش سطح دسترسی لازم برای نوشتن یا خواندن را تعیین کرد. تنظیمات برای ۲ بخش اول قابل اعمال است و بخش سوم مازاد بخش داده ای هست که ممکن است در ۲ بخش اول استفاده نشده باشد. برای بخش آخر به صورت پیشفرض سطح دسترسی (۲و۲) تعلق می‌گیرد. یعنی برای نوشتن و خواندن سطح دسترسی ۲ در نظر گرفته می‌شود. سطوح دسترسی به صورت زیر تعریف می‌شود:

سطح صفر: همگانی (everyone) که در دسترس همه قرار دارد. یعنی به هر چهار نوع دسترسی جواب می‌دهد.

سطح یک: برای مدیریت هست و فقط به سطح دسترسی مدیریت جواب می‌دهد.

سطح دو: برای برنامه نویس می‌باشد که علاوه بر سطح برنامه نویس، سطح مدیریت را نیز قبول می‌کند.

سطح سه: برای کاربر نهایی که فقط در دسترس وی می‌باشد

توجه داشته باشید برای استفاده از سطوح دسترسی مختلف با VID و رمز عبور را به سرویسها پاس کرد که برای چهار سطح به گونه زیر تعریف می‌شود:

سطح دسترسی	VID	رمز عبور
مدیریت	VID شناسه	رمز عبور مدیریت
برنامه نویس	VID شناسه	رمز عبور برنامه نویس
کاربر نهایی	'0.0.0.0'	رمز عبور کاربر (پین کد)
همگانی	'0.0.0.1'	هر چیزی

۱-۳- تنظیمات کاربر نهایی

این تنظیمات مربوط به کاربر نهایی بوده و برای هر کاربر معمولا منحصر بفرد می‌باشد. این تنظیمات توسط کاربر نهایی قابل تنظیم و تغییر می‌باشد و ممکن است مدیریت مقادیر اولیه را برای این پارامترها را برای کاربر در نظر گرفته باشد.

۱-۳-۱- رمز عبور کاربر[1] (پین کد)

رمزعبور کاربر یا پین کد مهمترین عامل دسترسی به سرویس سمت کاربر می‌باشد، این رمز عبور توسط کاربر قابل تغییر است و برای موارد زیر استفاده می‌شود:

- رمزنگاری AES[2] اختصاصی کاربر در GetEncryption
- رمزنگاری RSA[3] و امضای دیجیتال در متدهای مربوطه
- خواندن یا نوشتن قسمتی از حافظه شناسه توسط توابع مربوطه در صورتی که اجازه داده شده باشد
- ایجاد کلیدهای رمزنگاری RSA سمت کاربر در صورتی که مدیریت اجازه تغییر کلیدهای را داده باشد

۱-۳-۲- کلید رمزنگاری AES کاربر

همانطور که قبلا ذکر شد در شناسه کلیدی برای رمزنگاری AES می‌توان ذخیره کرد. که این کلید توسط کاربر نهایی قابل تنظیم و تغییر می‌باشد. کاربرد اصلی این کلید در مبحث شناسه کاربر از طریق رمزهای عبور یکبار مصرف (OTP) می‌باشد.

۱-۳-۳- کلیدهای رمزنگاری RSA

برای هر شناسه می‌توان کلیدهای عمومی و خصوصی رمزنگاری RSA را تعریف کرد تا بتوان از شناسه برای بحثهای امضای دیجیتال و نامه محرمانه استفاده کرد. کلید رمزنگاری RSA از ۳ قسمت E، D و N تشکیل می‌شود که هر کدام از آنها در شناسه نوین می‌تواند ۵۱۲ و ۱۰۲٤ باشد بنابر این با شناسه نوین می‌توان RSA در انواع ۵۱۲ و ۱۰۲٤ بیتی داشت. این کلیدهای مخصوص کاربر بوده و استفاده از کلید

---

[1] User Password (PIN code)

[2] Advanced Encryption Standard

برای اطلاع بیشتر در مورد این الگوریتم رمزنگاری به آدرس http://en.wikipedia.org/wiki/Advanced_Encryption_Standard مراجعه کنید

[3] Ron **R**ivest, Adi **S**hamir, Leonard **A**dleman

این الگوریتم رمزنگاری به صورت کلید نامتقارن عمل می‌کند و توسط سه دانشمند فوق مطرح گردیده است. برای اطلاع بیشتر به آدرس http://en.wikipedia.org/wiki/RSA مراجعه نمایید

خصوصی در سرویسهای مربوطه نیاز به داشتن رمز عبور کاربر (PIN) می‌باشد. کلیدهای رمزنگـاری RSA اگر برای اولین بار با سطح دسترسی مدیریت ایجاد و ذخیره شود نمی توان با سطح دسترسی کـاربر آنهـا را تغییر داد. ولی در غیر اینصورت اگر با سطح کاربر این عمل انجام پذیرد امکان تغییر آن سمت کاربر وجـود دارد.

فصل دوم: نرم افزار برنامه ریزی (Builder)

این نرم افزار برای مدیر یا سرپرست سیستمی طراحی شده است که می‌خواهد شناسه ها را جهت استفاده در نرم افزار یا وب سایت خاصی آماده کند.

۲-۱- ساختار ظاهری

از نظر ظاهری این نرم افزار شامل بخشهای زیر است:

۱- بخش لیست دستگاه ها و مشخصات دستگاه انتخاب شده: در این بخش لیستی شامل شماره سریال و نگارش شناسه های متصل به کامپیوتر نمایش داده می‌شود. با انتخاب هر یک از دستگاههای موجود در لیست مشخصات دستگاه انتخاب شده در زیر لیست نمایش داده می‌شود. علامت ✓ نشان دهنده ثبت شدن آن پارامتر در شناسه و ✗ نشان دهنده خام (خالی) بودن آن پارامتر در شناسه است.

علاوه بر این مشخصات دستگاه انتخاب شد (شماره سریال/ نگارش/ وضعیت) در قسمت نوار وضعیت (StatusBar) نمایش داده می‌شود.

۲- قسمت فرم های مراحل مختلف کار که در وسط صفحه می‌باشد: در مراحل مختلف کار تنظیم و انتخابهای متناسب با نوع کار نمایش داده می‌شود.

۳- کلید های کنترل مرحله که شامل سه کلید «صفحه اول»، «قبلی» و «بعدی» است، مراحل حرکت بین آنها را کنترل می‌کند.

٤- قسمت راهنما که با کلید بیشتر باز و بسته می‌شود.

۵- نوار وضعیت که وضعیت شناسه انتخاب شده را نمایش می‌دهد.

توجه داشته باشید قسمت سوم این نوار، وضعیت کاری شناسه انتخاب شده را نمایش می‌دهد. با انتخاب وسیله سه حالت احتمالی برای شناسه را نشان می‌دهد.

حالت اول: «فعال» که نشان دهنده وضعیت نرمال کاری است .

حالت دوم: «غیر فعال» نشان دهنده غیر فعال بودن شناسه را دارد که مشخص کننده استفاده نادرست از سطوح دسترسی شناسه این وضعیت پیش آماده است . در این حالت نرم افزار به طور خودکار فرم فعالسازی شناسه را باز می‌کند که با داشتن VID و رمز عبور مدیریت می‌توان شناسه را فعال کرد یا با زدن کلید ریست به حالت خام اولیه درآورد.

حالت سوم: «مسدود بودن» شناسه است که این نیز نشان دهنده استفاده نادرست از سطح دسترسی مدیریت در هنگام فعال سازی یا ریست کردن شناسه می‌باشد. دراین حالت بازگرداندن شناسه به حالت اول توسط مشتری امکان پذیر نبوده و باید این مشکل طی روال گارانتی توسط شرکت سازنده مرتفع گردد.

برای سهولت دسترسی به قسمتهای مختلف برنامه کلیدهای میانبری نیز در نرم افزار وجود دارد که به شرح زیر است:

F1: راهنما	F2: شروع کار
F3: ریست وفعال سازی	F10: خروج
Alt+1: روال یک	Alt+2: روال دو
Alt+3: روال سه	Alt+4: روال چهار

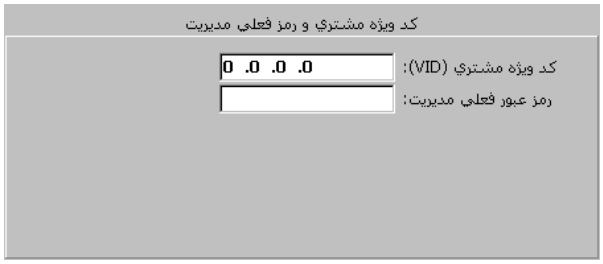
برای شروع کار با این نرم افزار پس از زدن شروع کار فرم زیر دیده می‌شود.

چنانچه مشاهده می‌کنید چهار روال اصلی برای کار با این نرم افزار وجود دارد. همچنین یک روال جانبی برای ریست کردن یا فعالسازی شناسه نیز از طریق همین فرم در دسترس می‌باشد.

۲-۱-۱- روال یک: تولید فایل تنظیمات شناسه (NTC)

تولید فایل NTC که برای ذخیره پارامترهای مورد نظر در یک فایل استفاده می‌شود. فایل NTC محتوی پارامترهای شناسه می‌باشد که اطلاعات در این فایل بصورت رمز شده نگهداری می‌شوند .

- **مرحله۱ :**



در فرم بالا باید VID شناسه و رمز اولیه مدیریت شناسه را باید وارد کنید.
توجه داشته باشید که VID رشته ای شبیه آدرس IP هست که از طرف شرکت سازنده به مشتری اختصاص داده می‌شود. این کد برای تمام شناسه های یک مشتری در تمام خریدها ثابت می‌باشد مگر اینکه برای تغییر آن درخواستی از طرف مشتری داد شده باشد. در اولین خرید به صورت نمونه (تکی) VID برای مشتری اختصاص داده نمی شود و از VID آزمایشی (عمومی) استفاده می‌شود و در دفعات بعدی خرید VID اختصاصی صادر و ارائه می‌شود .
VID آزمایشی عبارتست از 109.232.151.192 که با کلیک کردن روی برچسب VID روی فرم فوق به صورت خودکار در محل VID تایپ می‌شود.
رمز عبور شناسه ها به صورت پیش فرض خالی می‌باشد .

- **مرحله ۲ :**

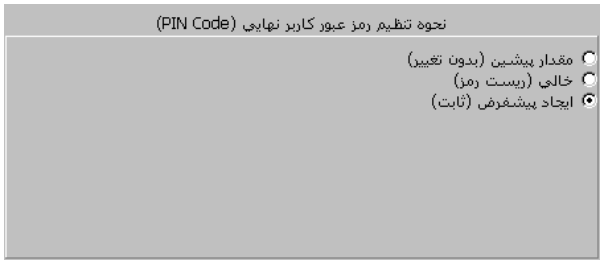
رمز عبور Admin شناسه که حداکثر می‌تواند ۱۶ کاراکتر باشد برای برنامه ریزی شناسه استفاده می‌شود و جایگاه استفاده آن فقط در نرم افزار برنامه ریزی شناسه می‌باشد.
این رمز برای تغییر در پارامترهای شناسه بعدا مورد استفاده قرار می‌گیرد و وجود آن به جلوگیری از دستکاری در پارامترهای شناسه کمک می‌کند. همچنین دانستن آن برای Reset کردن یا خارج کردن شناسه از حالت غیر فعال (disable) لازم است.

- **مرحله ۳ :**

رمز عبور برنامه نویس همانطور که از اسمش پیداست جهت استفاده از برخی سرویسهای شناسه در برنامه نویسی مورد استفاده قرار می‌گیرد.
این رمز برای سرویسهای زیر استفاده می‌شود:

- خواندن قسمتی از حافظه شناسه با استفاده از سرویسهای GetByte، GetChar، GetBlockstr و GetBlockHexstr
- نوشتن در قسمتی از حافظه شناسه با استفاده از SetByte ، SetChar ، SetBlockstr و SetBlockHexStr

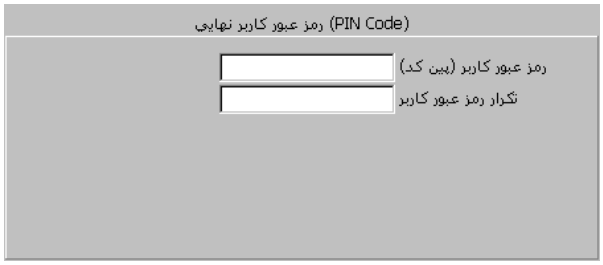
- **مرحله ۴ :**



- رمزعبور کاربر یا پین کد مهمترین عامل دسترسی به سرویس سمت کاربر می‌باشد، این رمز عبور توسط کاربر قابل تغییر است و برای موارد زیر استفاده می‌شود:
- رمزنگاری AES کاربری (با استفاده از کلید کاربر) در GetEncryption

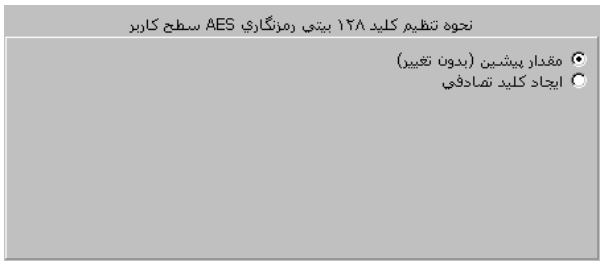
- رمزنگاری RSA و امضای دیجیتال در متدهای مربوطه
- خواندن یا نوشتن قسمتی از حافظه شناسه توسط توابع مربوطه
- ایجاد کلیدهای رمزنگاری RSA سمت کاربربا استفاده از تابع SetRSA
- برای تنظیم پین کد باید یکی از روالهای فوق را انتخاب کنید. روالها کاملا گویا می‌باشد، فقط توجه داشته باشید در صورت انتخاب روال سوم (ثابت) با زدن کلید بعدی در مرحله بعد پین کد از شما گرفته می‌شود.
- ضمنا به یاد داشته باشید که ثبت یا تغییر پین کد لازم نیست که حتما از طریق این نرم افزار صورت گیرد. زیرا که در توابع ActiveX تابع SetUserPWD این کار را برای شما می‌تواند انجام دهد.

- **مرحله ۵ :**



رمزعبور کاربر یا پین کد مانند سایر رمزهای شناسه حداکثر ۱۶ کاراکتری می‌باشد و از طریق فرم بالا می‌توانید آنرا تعیین کنید. ثبت یا تغییر پین کد سمت کاربر نیز از طریق تابع SetUserPWD از توابع ActiveXنیز قابل انجام است.

- **مرحله ۶:**



دانستن کلید کاربر برای نرم افزار چندان پر اهمیت نیست بدین جهت حق انتخاب در مورد کمیت و کیفیت کلید کاربر محدود است.

علاوه بر این می‌توانید بدون ثبت کلید کاربر کار برنامه ریزی را به پایان رسانده و ثبت کلید کاربر را در مراحل بعدی اجرا، از طریق سرویس SetUserKey از توابع ActiveX شناسه انجام دهید.

کلید کاربر مهمترین نقش خود را در پیاده سازی احراز هویت بوسیله رمز عبورهای یک بار مصرف (OTP) ایفا می‌کند .

- **مرحله ۷:**

در شناسه معمولا ۳۲ بایت از قسمت داده بعنوان داده آزاد مطرح است که می‌توان از طریق سرویس های مربوطه آنرا خواند یا در آن چیزی نوشت.

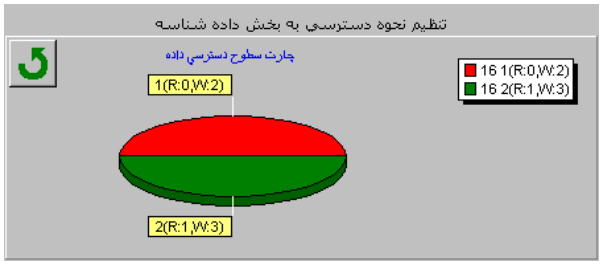
با توجه به این که در شناسه ها چند نوع سطح دسترسی وجود دارد (مدیریت، برنامه نویس، کاربر نهایی و همگانی) می‌توان روال دسترسی به حافظه را برای خواندن یا نوشتن از طریق این فرم تعریف کرد.

فضای داده را می‌توان به ۳ قسمت جدا در ابعاد ۸، ۱٦ یا ۳۲ بایتی تقسیم و برای قسمتهای اول ودوم سطح دسترسی خواندن یا نوشتن را تعریف کرد.

توجه داشته باشید برای استفاده از سطوح دسترسی مختلف در توابع با توجه به جدول زیر VID و رمز عبور مناسب را باید به تابع پاس داد.

سطح دسترسی	VID	رمز عبور
مدیریت	VIDشناسه	رمز عبور مدیریت
برنامه نویس	VID شناسه	رمز عبور برنامه نویس
کاربر نهایی	'0.0.0.0'	رمز عبور کاربر (پین کد)
همگانی	'0.0.0.1'	هر چیزی

سطح دسترسی مدیریت ، سطح دسترسی برنامه نویس را تحت پوشش قرار می‌دهد .هم چنین برای استفاده از قسمتهای همگانی هر چهار سطح دسترسی قابل قبول است، درغیر اینصورت هنگام کار به خطای عدم دسترسی (Access denied) برخورد می‌کنیم .

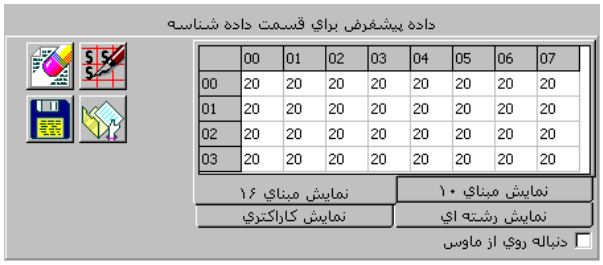


چارت فوق نمایش دهنده وضعیت تقسیم بخش داده و تخصیص سطوح دسترسی نویسنده و خواننده برای بخشهای مختلف است.

توجه داشته باشید R نشاندهنده خواننده (Reader) و W نشاندهنده نویسنده (Writer) است. عددهای ۰ ، ۱ ، ۲ و ۳ به ترتیب نشاندهنده سطح دسترسی هرکس، مدیر، برنامه نویس و کاربر نهائی است.

با زدن کلید می‌توانید به فرم اصلی تعیین سطوح دسترسی برگردید.

- **مرحله ۸:**



در این فرم می‌توانید داده پیش فرض شناسه را تعریف کنید. این داده را که می‌تواند حداکثر ۳۲ کاراکتر باشد در این فرم به صورت رشته ای، کاراکتری یا به صورت کد اسکی در مبنای ۱۰ یا ۱۶ می‌توان دید.

امکانات اضافی شامل:

- کلید برای پر کردن خانه ها با یک عدد (کاراکتر) خاص استفاده می‌شود. توسط کلید تمام خانه ها با کد ۳۲ (فاصله) پر می‌شود .
- با کلید می‌توانید این داده را از فایلی بخوانید یا با کلید در فایلی ذخیره کنید .
- در صورتیکه روال مرحله به مرحله را استفاده می‌کنید دو کلید نیز به فرم اضافه می‌شود. از طریق کلید داده موجود در شناسه انتخاب شده خوانده شده و با زدن اطلاعات موجود در جدول داده پیش فرض به داخل شناسه انتقال می‌یابد .

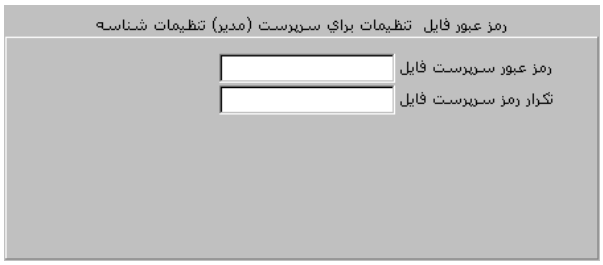
- **مرحله ۹:**

برای فعال کردن رمزنگاری نامتقارن RSA باید از طریق فرم فوق باید طول کلید رمزنگاری را انتخاب کنید. شناسه قابلیت استفاده از کلید ها تا طول ۱۰۲٤ بیتی را دارد. برای ایجاد و ثبت کلیدهای RSA در شناسه از دو سطح دسترسی می‌توان استفاده کرد. سطح دسترسی مدیریت یا سطح دسترسی کاربر نهائی.

در صورتیکه اولین ثبت کلید با سطح مدیریت باشد سمت کاربر توسط سطح کاربر نهائی کلید قابل تغییر نخواهد بود، ولی در صورتیکه اولین بار این کار با سطح دسترسی کاربر نهائی انجام شود امکان تغییر کلید سمت کاربر فراهم است.

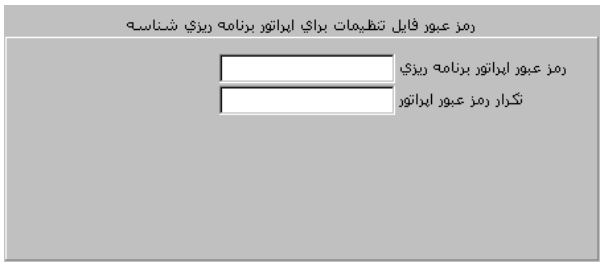
این مورد را از طریق تیک زدن «قابل تغییر سمت کاربر» می‌توان تعیین کرد .هنگام استفاده از روال مرحله به مرحله زدن کلید «تولید و ثبت» باعث می‌شود که کلیدها برای شناسه انتخاب شده تولید و ثبت شود. در صورتیکه RSA ثبت و فعال شود یکی از موارد رمزنگاری RSA در زیر لیست دستگاههای موجود فعال می‌شود.

- **مرحله ۱۰:**



رمز عبور سرپرست فایل به هیچ یک از سطوح دسترسی شناسه ربطی ندارد و فقط جهت دسترسی به پارامترهای ذخیره شده در فایل NTC مورد استفاده قرار می‌گیرد. توجه داشته باشید برای تغییر فایل NTC (روال دوم) داشتن این رمز عبور لازم است.

- **مرحله ۱۱:**



رمز عبور اپراتور برنامه ریزی نیز مانند رمز عبور سرپرست هیچ ربطی به رمز های شناسه ندارد. این رمز عبور در هنگام استفاده از روال سوم (برنامه ریزی شناسه با NTC) لازم است.

توجه داشته باشید هنگامی که تعداد زیادی شناسه را قرار است برنامه ریزی کنید می‌توانید چند اپراتور برای این کار اختصاص دهید و با در اختیار گذاشتن رمز عبور اپراتوری آنها می‌توانند برنامه ریزی اولیه شناسه ها را بدون آنکه از پارامترهای تنظیم شده با خبر شوند انجام دهند.

- **مرحله ۱۲:**

در این فرم باید فایل مقصد را برای ذخیره کردن تنظیمات انجام شده انتخاب نمایید.

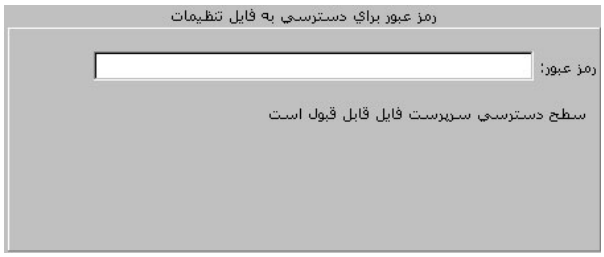
پسوند فایل تنظیمات NTC[1] است که مخفف می‌باشد. در این فایل اطلاعات به صورت رمز شده ذخیره می‌شود که مشاهده یا استفاده از آن بدون داشتن رمز عبور فایل امکانپذیر نخواهد بود.

۲-۱-۲- روال دو: تغییر فایل تنظیمات شناسه (NTC)

تغییر فایل NTC جهت تغییر در پارامترهای ذخیره شده در یک فایل NTC مورد استفاده قرار می‌گیرد، برای تغییر فایل NTC داشتن رمز عبور سرپرست (Supervisor) فایل نیاز است.

[1] Novin Token Configuration

- **مرحله ۱:**



از طریق فرم فوق می‌توانید فایل مورد نظر خود را انتخاب کنید تا اطلاعات و تنظیمات ذخیره شده از آن فایل خوانده شود.

نوع فایلNTC می‌باشد . این فایل باید توسط همین نرم افزار قبلا تولید شده باشد.

- **مرحله ۲:**

برای دسترسی به اطلاعات ذخیره شده در فایل NTC رمز عبور فایل را باید وارد کنید.

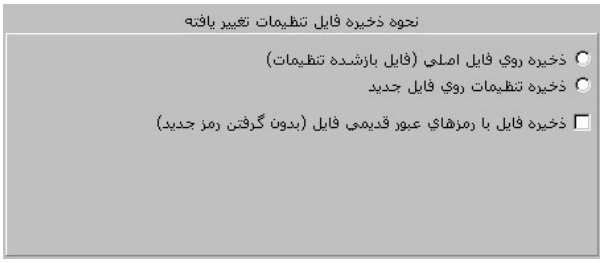
در صورتی که می‌خواهید تنظیمات موجود در فایل را تغییر دهید (روال دوم) حتما باید رمزعبور سرپرست فایل را داشته باشید و وارد کنید. ولی زمانی که می‌خواهید شناسه ها را از روی فایل برنامه ریزی کنید (روال سوم) می‌توانید از رمز عبور سرپرست وهم از رمز عبور اپراتور برنامه ریزی استفاده کنید.

اگر رمزهای عبور فایل را ندارید آنرا باید از ایجاد کننده فایل دریافت کنید.

- **مرحله های ۳ الی ۱۰:**

تکرار مراحل ۱ تا ۸ روال ۱ (تولید فایل تنظیمات شناسه(NTC))

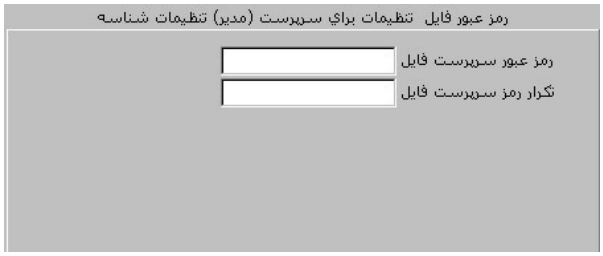
- **مرحله ۱۱:**



پس ازتغییرات مورد نظر در فایل باز شده میتوانید این تغییرات را روی فایل قبلی یادر یک فایل جدید ذخیره کنید. بدین منظور فایل مورد نظر خود را برای ذخیره فایل انتخاب نمایید .

در صورتی که می‌خواهید از رمزهای عبور فایل باز شده استفاده کنید می‌توانید گزینه ذخیره با رمزهای قدیمی را تیک نزنید که در غیر اینصورت رمزهای عبور جدید برای ذخیره فایل در مراحل بعدی پرسیده خواهد شد.

- **مرحله ۱۲:**



رمز عبور سرپرست فایل به هیچ یک از سطوح دسترسی شناسه ربطی ندارد و فقط جهت دسترسی به پارامترهای ذخیره شده در فایل NTC مورد استفاده قرار می‌گیرد. توجه داشته باشید برای تغییر فایل NTC (روال دوم) داشتن این رمز عبور لازم است.

- **مرحله ۱۳:**

رمز عبور اپراتور برنامه ریزی نیز مانند رمز عبور سرپرست هیچ ربطی به رمز های شناسه ندارد. این رمز عبور در هنگام استفاده از روال سوم (برنامه ریزی شناسه با NTC) لازم است.

توجه داشته باشید هنگامی که تعداد زیادی شناسه را قرار است برنامه ریزی کنید می‌توانید چند اپراتور برای این کار اختصاص دهید و با در اختیار گذاشتن رمز عبور اپراتوری ، آنها می‌توانند برنامه ریزی اولیه شناسه ها را بدون آنکه از پارامترهای تنظیم شده با خبر شوند انجام دهند.

- **مرحله ۱۴:**

در این فرم باید فایل مقصد را برای ذخیره کردن تنظیمات انجام شده انتخاب نمایید.

نوع فایل تنظیمات NTC است که مخففNovin Token Configuration می‌باشد. در این فایل اطلاعات به صورت رمز شده ذخیره می‌شود که مشاهده یا استفاده از آن بدون داشتن رمز عبور فایل امکانپذیر نخواهد بود.

۲-۱-۳- روال سه: برنامه ریزی شناسه با (NTC)

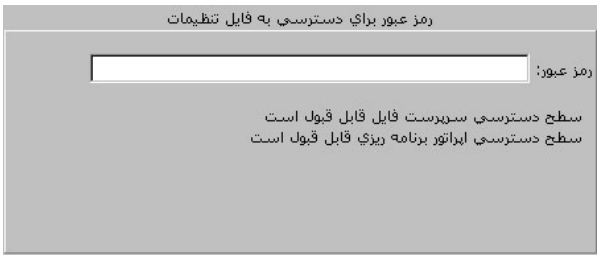
برنامه ریزی شناسه با استفاده از NTC برای انتقال پارامترهای تنظیم شده در یک فایل NTC به شناسه، مورد استفاده قرار می‌گیرد.

- **مرحله ۱:**

از طریق فرم فوق می‌توانید فایل مورد نظر خود را انتخاب کنید تا اطلاعات و تنظیمات ذخیره شده از آن فایل خوانده شود.

نوع فایلNTC می‌باشد که این فایل قبلا باید توسط همین نرم افزار تولید شده باشد.

- **مرحله ۲:**

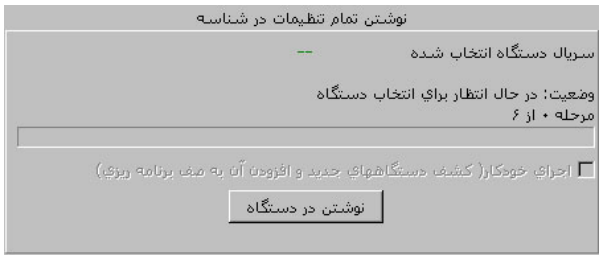


برای دسترسی به اطلاعات ذخیره شده در فایل NTC رمز عبور فایل را باید وارد کنید .

در صورتی که می‌خواهید تنظیمات موجود در فایل را تغییر دهید (روال دوم) حتما باید رمزعبور سرپرست فایل را داشته باشید و وارد کنید. ولی زمانی که می‌خواهید شناسه ها را از روی فایل برنامه ریزی کنید (روال سوم) می‌توانید از رمز عبور سرپرست و هم رمز عبور اپراتور برنامه ریزی استفاده کنید .

اگر رمزهای عبور فایل را ندارید آنرا باید از ایجاد کننده فایل دریافت کنید.

- **مرحله ۳:**



برای انتقال تنظیمات خوانده شده از فایل NTC به داخل شناسه باید از فرم فوق استفاده کرد. بدین منظور دو روال را می‌توان استفاده کرد.

- اجرای دستی: زمانیکه تیک مورد اجرای خودکار زده نشده است، پس از انتخاب شناسه مورد نظر از لیست شناسه ها یافته شده کلید «نوشتن در دستگاه» را بدین منظور می‌توانید بزنید.
- اجرای خودکار: با تیک زدن اجرای خودکار با پیدا کردن اولین شناسه نرم افزار شروع به برنامه ریزی آن می‌نماید. در صورتی که بیش از یک شناسه وصل شود به ترتیب شناسایی، شماره سریال شناسه ها در لیستی قرار می‌گیرد که به ترتیب آنها را برنامه ریزی کند.

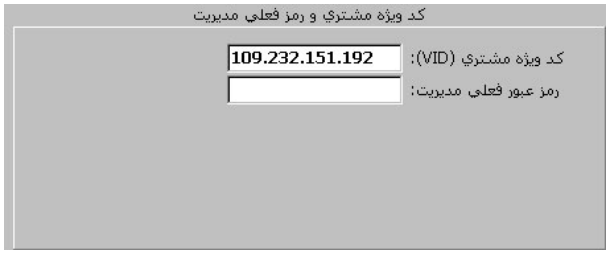
- استفاده از این مورد برای افراد مبتدی پیشنهاد نمی شود مخصوصا زمانیکه که شناسه ها فاقد LED نشاندهنده وضعیت فعالیت می‌باشد. زیرا که تشخیص اینکه کدام شناسه در حال برنامه ریزی هست یا کدام شناسه ها برنامه ریزی شده اند اندکی دشوار است.
- می توانید شناسه ها را به صورت گروهی به سیستم وصل کنید و تا اتمام برنامه ریزی همه شناسه ها هیچ شناسه ای را از کامپیوتر نکشید و پس از اتمام کار می‌توانید با انتخاب تک تک شناسه ها از برنامه ریزی شدن آنها مطمئن شوید و پس از کشیدن همه شناسه از کامپیوتر، گروه بعدی را برنامه ریزی کنید.

طی عملیات برنامه ریزی پیشرفت مراحل و نوع کار در حال انجام از طریق همین فرم کاملا مشهود است.

۲-۱-۴- روال چهار: برنامه ریزی دستی (مرحله به مرحله)

برنامه ریزی مرحله به مرحله برای ثبت پارامترهای شناسه انتخاب شده در هر مرحله مورد استفاده قرار می‌گیرد یعنی صفحه به صفحه پارامترها وارد و در شناسه ثبت می‌گردد.

- **مرحله ۱:**



در فرم بالا باید VID شناسه و رمز اولیه مدیریت شناسه را باید وارد کنید.

توجه داشته باشید که VID رشته ای شبیه آدرس IP هست که از طرف شرکت سازنده به مشتری اختصاص داده می‌شود. این کد برای تمام شناسه های یک مشتری برای همه مراحل خرید ثابت می‌باشد مگر اینکه برای تغییر آن درخواستی از طرف مشتری شود. این VID در اولین بار خرید به صورت آزمایشی (عمومی) ارائه می‌شود و در مراحل بعدی خرید کد اختصاصی ارائه می‌شود.

VID آزمایشی عبارتست از 109.232.151.192 که با کلیک کردن روی برچسب VID روی فرم فوق به صورت خودکار در محل VID تایپ می‌شود.

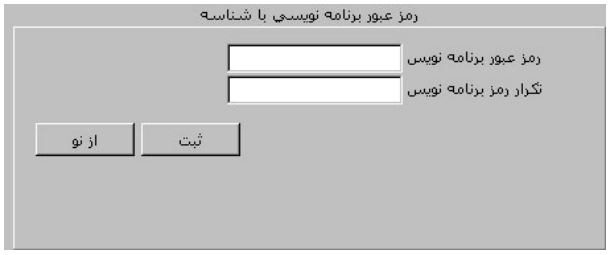
رمزهای عبور شناسه های خام به صورت پیش فرض خالی می‌باشد.

- **مرحله ۲:**

رمز عبور Admin شناسه که حداکثر می‌تواند ۱۶ کاراکتر باشد برای برنامه ریزی شناسه استفاده می‌شود و جایگاه استفاده آن فقط در نرم افزار برنامه ریزی شناسه می‌باشد.

این رمز برای تغییر در پارامترهای شناسه بعدا مورد استفاده قرار می‌گیرد و وجود آن به جلوگیری از دستکاری در پارامترهای شناسه کمک می‌کند. همچنین دانستن آن برای Reset کردن یا خارج کردن شناسه از حالت غیر فعال (disable) لازم است.

- **مرحله ۳:**

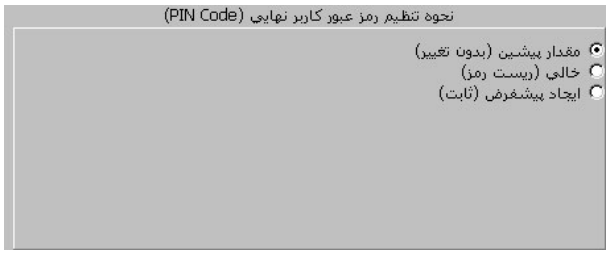


رمز عبور برنامه نویس همانطور که از اسمش پیداست جهت استفاده از برخی سرویسهای شناسه در برنامه نویسی مورد استفاده قرار می‌گیرد.

این رمز برای سرویسهای زیر استفاده می‌شود:

- خواندن قسمتی از حافظه شناسه با استفاده از سرویسهای GetByte، GetChar، GetBlockstr و GetBlockHexstr
- نوشتن در قسمتی از حافظه شناسه با استفاده از SetByte، SetChar، SetBlockstr و SetBlockHexStr

- **مرحله ۴:**

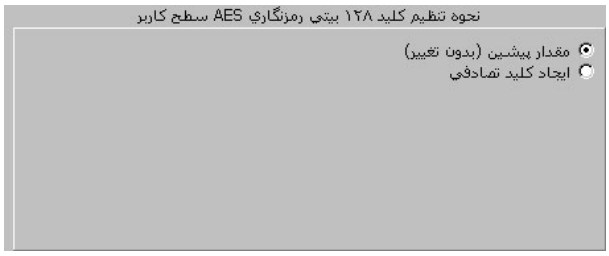


رمزعبور کاربر یا پین کد مهمترین عامل دسترسی به سرویس سمت کاربر می‌باشد، این رمز عبور توسط کاربر قابل تغییر است و برای موارد زیر استفاده می‌شود:

- رمزنگاری RSA و امضای دیجیتال در متدهای مربوطه
- خواندن یا نوشتن قسمتی از حافظه شناسه توسط توابع مربوطه
- ایجاد کلیدهای رمزنگاری RSA سمت کاربر با استفاده از تابع SetRSA

برای تنظیم پین کد باید یکی از روالهای فوق را انتخاب کنید. روالها کاملا گویا می‌باشد، فقط توجه داشته باشید در صورت انتخاب روال سوم (ثابت) با زدن کلید بعدی در مرحله بعد پین کد از شما گرفته می‌شود. ضمنا به یاد داشته باشید که ثبت یا تغییر پین کد لازم نیست که حتما از طریق این نرم افزار صورت گیرد. زیرا که در توابع ActiveX تابع SetUserPWD این کار را برای شما می‌تواند انجام دهد.

- **مرحله ۵:**



دانستن کلید کاربر برای استفاده از AES در نرم افزار چندان پر اهمیت نیست بدین جهت امکان انتخاب کلید به صورت دستی وجود ندارد.

علاوه بر این می‌توانید بدون ثبت کلید کاربر کار برنامه ریزی را به پایان رسانده و ثبت کلید کاربر را از داخل نرم افزار خودتان، از طریق سرویس SetUserKey از توابع ActiveX شناسه انجام دهید.

کلید کاربر مهمترین نقش خود را در پیاده سازی احراز هویت بوسیله رمز عبورهای یک بار مصرف (OTP) ایفا می‌کند .

- **مرحله ۶:**

در شناسه معمولا ۳۲ بایت از قسمت داده بعنوان داده آزاد مطرح است که می‌توان از طریق سرویس های مربوطه آنرا خواند یا در آن چیزی نوشت.

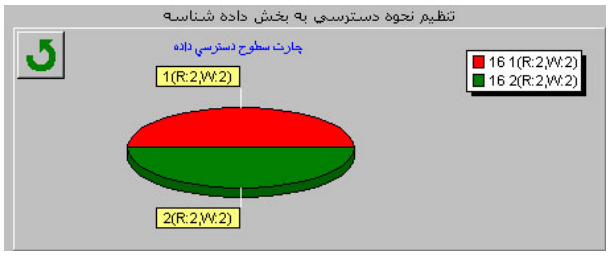
با توجه به این که در شناسه ها چند نوع سطح دسترسی وجود دارد (مدیریت، برنامه نویس، کاربر نهایی و همگانی) می‌توان روال دسترسی به حافظه را برای خواندن یا نوشتن از طریق این فرم تعریف کرد.

فضای داده را می‌توان به ۳ قسمت جدا در ابعاد ۸، ۱۶ یا ۳۲ بایتی تقسیم و برای قسمتهای اول ودوم سطح دسترسی خواندن یا نوشتن را تعریف کرد.

توجه داشته باشید برای استفاده از سطوح دسترسی مختلف در توابع با توجه به جدول زیر VID و رمز عبور مناسب را باید به تابع پاس داد.

سطح دسترسی	VID	رمز عبور
مدیریت	VIDشناسه	رمز عبور مدیریت
برنامه نویس	VID شناسه	رمز عبور برنامه نویس
کاربر نهایی	'0.0.0.0'	رمز عبور کاربر (پین کد)
همگانی	'0.0.0.1'	هر چیزی

سطح دسترسی مدیریت ، سطح دسترسی برنامه نویس را تحت پوشش قرار می‌دهد .هم چنین برای استفاده از قسمتهای همگانی هر چهار سطح دسترسی قابل قبول است، درغیر اینصورت هنگام کار به خطای عدم دسترسی (Access denied) برخورد می‌کنیم .



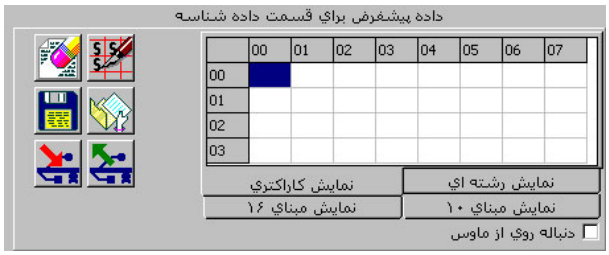
چارت فوق نمایش دهنده وضعیت تقسیم بخش داده و تخصیص سطوح دسترسی نویسنده و خواننده برای بخشهای مختلف است.

توجه داشته باشید R نشاندهنده خواننده (Reader) و W نشاندهنده نویسنده (Writer) است.

عددهای ۰ ، ۱ ، ۲ و ۳ به ترتیب نشاندهنده سطح دسترسی هرکس، مدیر، برنامه نویس و کاربر نهائی است.

با زدن کلید می‌توانید به فرم اصلی تعیین سطوح دسترسی برگردید.

- **مرحله ۷:**



در این فرم می‌توانید داده پیش فرض شناسه را تعریف کنید. این داده را که می‌تواند حداکثر ۳۲ کاراکتر باشد می‌توان در این فرم به صورت رشته ای، کاراکتری یا به صورت کد اسکی در مبنای ۱۰ یا ۱٦ دید .

امکانات اضافی شامل:

- کلید برای پر کردن خانه ها با یک عدد (کاراکتر) خاص استفاده می‌شود .
- توسط کلید تمام خانه ها با کد ۳۲ (فاصله) پر می‌شود .
- با کلید می‌توانید این داده را از فایلی بخوانید یا با کلید در فایلی ذخیره کنید.
- در صورتیکه روال مرحله به مرحله را استفاده می‌کنید دو کلید نیز به فرم اضافه می‌شود. از طریق کلید داده موجود در شناسه انتخاب شده خوانده شده و با زدن اطلاعات موجود در جدول داده پیش فرض به داخل شناسه انتقال می‌یابد.

- **مرحله ۸:**

برای فعال کردن رمزنگاری نامتقارن RSA باید از طریق فرم فوق طول کلید رمزنگاری را انتخاب کنید. شناسه قابلیت استفاده از کلید ها تا طول ۱۰۲٤ بیتی را دارد.

برای ایجاد و ثبت کلیدهای RSA در شناسه از دو سطح دسترسی می‌توان استفاده کرد. سطح دسترسی مدیریت یا سطح دسترسی کاربر نهائی.

در صورتیکه اولین ثبت کلید با سطح مدیریت باشد سمت کاربر توسط سطح کاربر نهائی کلید قابل تغییر نخواهد بود، ولی در صورتیکه اولین بار این کار با سطح دسترسی کاربر نهائی انجام شود امکان تغییر کلید سمت کاربر فراهم است.

این مورد را از طریق تیک زدن «قابل تغییر سمت کاربر» می‌توان تعیین کرد.

هنگام استفاده از روال مرحله به مرحله زدن کلید «تولید و ثبت» باعث می‌شود که کلیدها برای شناسه انتخاب شده تولید و ثبت شود. در صورتیکه RSA ثبت و فعال شود یکی از موارد رمزنگاری RSA در زیر لیست دستگاههای موجود فعال می‌شود.

۲-۱-۵- ریست/فعالسازی

درصورتی که شناسه شما غیر فعال شده است و یا می‌خواهید آن را به صورت خام دربیاورید و یا پین کد کاربر آن را ریست (پاک) کنید پس از وارد کردن رمز عبورمدیریت از طریق کلیدهای مربوطه خواسته خود را بر آورده کنید.

فصل سوم: کتابخانه رابط (dll)

۳-۱- ارتباط با شناسه در برنامه نویسی

برای ارتباط با شناسه نوین از طریق نرم افزارهایی که می‌نویسیم یا از طریق صفحه وب وجود فایل DLL رابط شناسه لازم است. این فایل تحت نام NovinToken.dll در CD مربوط به شناسه موجود است. دسترسی به متدهای این فایل از دو طریق امکانپذیر است.

۱. ارتباط از طریق توابع export شده که می‌توان به طور مستقیم به DLL لینک شد و استفاده کرد.

۲. یک کلاس اتومات مشتق شده از NovinAfzar به نام clsLocalDevice که در صورت ثبت (register) کردن dll در ویندوز قابل دسترس می‌شوند.

برای فراخوانی شناسه از داخل نرم افزار خود باید فایل مربوط به روتین های شناسه (NovinToken.dll) را به پروژه اضافه شود.

۳-۲- تعریف روالها

روال های موجود در clsLocalDevice عبارتند از:

Methods:

Init	آماده سازی کلاس برای کار
GetFirstDevice	گرفتن مشخصات اولین شناسه شناسایی شده
GetNextDevice	گرفتن مشخصات شناسه بعدی شناسایی شده
GetDeviceCount	دریافت تعداد دستگاه های شناسایی شده
SelectDevice	انتخاب یک دستگاه برای اجرای دستورات
GetDeviceReady	تست وجود شناسه
GetDeviceStatus	تست وضعیت شناسه
GetSerial	دریافت شماره سریال شناسه
GetVersion	دریافت شماره نسخه شناسه
GetMemorySize	دریافت مقدار حافظه قابل دسترس
GetMemorySizeEx	دریافت مقدار کاراکتر قابل دسترس
SetDataByte	نوشتن یک بایت در حافظه شناسه

SetDataBlockStr	نوشتن اطلاعات در حافظه بصورت رشته ای
SetDataHexBlock	نوشتن بلوک HEX در بخش داده
GetDataByte	خواندن یک بایت از حافظه شناسه
GetDataBlockStr	خواندن اطلاعات از حافظه بصورت رشته ای
GetDataHexBlock	خواندن بلوک HEX از بخش داده
GetEncryption	رمزنگاری
GetDecryption	رمزگشایی
ConvDelimiteredStringToString	تبدیل رشته از DNString به رشته کاراکتری
ConvStringToDelimiteredString	تبدیل رشته از رشته کاراکتری[1] به DNString[2]
ConvHexStringToDelimiteredString	تبدیل رشته HEX[3] به DNString
ConvDelimiteredStringToHexString	تبدیل DNString به رشته HEX
ConvStringToHEXString	تبدیل رشته معمولی به رشته HEX
ConvHEXStringToString	تبدیل رشته HEX به رشته معمولی
RSAEncrypt	رمزنگاری نامتقارن RSA
RSADecrypt	رمزگشایی نامتقارن RSA
GetPublicKey	گرفتن کلید عمومی از بانک کلیدها
GetHashStr	ایجاد Hash یک داده (درهم ساز)
GetSignature	ایجاد امضای دیجیتال یک داده
VerifySignature	تست اعتبار امضای یک داده
SetRSA	ایجاد و تنظیم کلیدهای RSA
SetUserPWD	تنظیم و تغییر سطح سوم دسترسی
SetUserKEY	تنظیم کلید رمزنگاری شخصی (کاربر)
SetDeviceType	تعیین نوع شناسه

[1] رشته کاراکتری یکی از سه نوع داده رشته ای است که به صورت معمولی (کاراکتری) و از نوع WideString است

[2] رشته Delimitered یا DNString از انواع دیگر داده رشته ای است که به صورت اعداد جدا شده با نقطه (همانند آدرس IP) است

[3] نوع رشته ای HEX نوع داده ای است که کد اسکی کاراکترها به صورت اعداد پشت سر هم در مبنای ۱۶ می‌باشد

Properties:

ErrNo	کد خطای حاصل از آخرین متد اجرا شده
ErrDescr	پیام خطای متناسب با کد خطا فوق (انگلیسی)
ErrDescrFA	پیام خطای متناسب با کد خطا فوق (فارسی)
UserPwd	رمز عبور سطح سه (کاربر)
DeviceName	نام وسیله در نتیجه جستجو
DeviceVer	نگارش وسیله در نتیجه جستجو
DeviceSerial	شماره سریال وسیله در نتیجه جستجو
SelectedName	نام وسیله انتخاب شده
SelectedVer	نگارش وسیله انتخاب شده
SelectedSerial	شماره سریال وسیله انتخاب شده

Init	آماده سازی کلاس برای کار
Procedure Init;	
برخی مواقع هنگام استفاده از **DLL** رابط شناسایی **HID** درست آماده سازی نمی شود و به خطای **nErr_HID** برخورد می‌کنید. برای رفع این مشکل باید هنگام اتصال به کتابخانه این تابع را فراخوانی کنید	
هیچ پارامتری ندارد	
هیچ مقدار بازگشتی ندارد	
خطا ندارد.	

GetFirstDevice	گرفتن مشخصات اولین شناسه شناسایی شده
Procedure GetFirstDevice;	
با توجه به اینکه کتابخانه قابلیت کار با بیش از یک دستگاه وصل شده به کامپیوتر را داراست. این روال مشخصات اولین دستگاه شناسایی شده را از طریق **DeviceSerial**، **DeviceVer** و **DeviceName** نمایش می‌دهد.	
هیچ پارامتری ندارد	
هیچ مقدار بازگشتی ندارد	
انواع خطاها: **nErr_OK**، **nErr_OTE**، **nErr_DNF**	

GetNextDevice	گرفتن مشخصات شناسه بعدی شناسایی شده
Procedure GetNextDevice;	
با این روتین مشخصات دستگاه بعدی شناسایی شده نمایش داده می‌شود.	
هیچ پارامتری ندارد	
هیچ مقدار بازگشتی ندارد	
انواع خطاها: **nErr_OK**، **nErr_OTE**، **nErr_DNF**	

GetDeviceCount	دریافت تعداد دستگاه های شناسایی شده
Function GetDeviceCount: Word;	
این تابع تعداد دستگاههای شناسایی شده سازگار با کتابخانه را نمایش می‌دهد.	
هیچ پارامتری ندارد	
مقدار بازگشتی: تعداد دستگاهها	
انواع خطاها:**nErr_OK**، **nErr_OTE**، **nErr_DNF**، **nErr_BSY**	

SelectDevice	انتخاب یک دستگاه برای اجرای دستورات
Procedure SelectDevice(SerialNo: WideString);	
برای استفاده از توابع که در ادامه توضیح داده می‌شود باید دستگاهی انتخاب شود .سا ین عمل از طریق این روال انجام می‌پذیرد. در صورت عدم استفاده از این تابع حین کار به خطای **nErr_DNS** برخورد خواهید کرد.	
SerialNo: شماره سریال دستگاهی که می‌خواهید انتخاب کنید.	
هیچ مقدار بازگشتی ندارد	
انواع خطاها: انواع خطاها:**nErr_OK**، **nErr_DNF**	

GetDeviceReady	تست وجود شناسه
Procedure GetDeviceReady;	
ساده ترین روش برای چک کردن شناسه در پورت **USB**.	
هیچ پارامتری ندارد	
هیچ مقدار بازگشتی ندارد	
انواع خطاها: **nErr_BSY**، **nErr_DNS**، **nErr_DNF**، **nErr_IOE**،**nErr_OK**، **nErr_OTE**	

GetDeviceStatus	تست وضعیت شناسه
Procedure GetDeviceStatus;	
تست وضعیت: فعال یا یکی از وضعیتهای غیرفعال (**Disabled, Blocked or Catastrophic Disabled**)	
هیچ پارامتری ندارد	
هیچ مقدار بازگشتی ندارد. وضعیت موجود قفل در **ErrNo** با خطاهای متناسب برگشت می‌شود.	
انواع خطاها:**nErr_DNS**، **nErr_BLK**، **nErr_DIS**، **nErr_SUS**، **nErr_OK**، **nErr_CAT**، **nErr_OTE**، **nErr_BSY**	

GetSerial	دریافت شماره سریال شناسه
Function GetSerial: WideString;	
هر شناسه دارای یک سریال یکتا می‌باشد که در موقع تولید، توسط تولید کننده در شناسه ذخیره می‌گردد با استفاده از این روتین می‌توان این سریال را بدست آورد.	
هیچ پارامتری ندارد	
مقدار برگشتی : در صورت اجرای موفقیت آمیز شماره سریال را که از نوع **WideString** می‌باشد بازمیگرداند.	
انواع خطاها: **nErr_OTE**، **nErr_OK**، **nErr_WRP**، **nErr_IOE**، **nErr_DNF**، **nErr_BSY**، **nErr_DNS**	

GetVersion	دریافت شماره نسخه شناسه
Function GetVersion: WideString;	
هر شناسه دارای نرم افزارهای داخلی می‌باشد که با توسعه شناسه این نرم افزارها تغییر می‌کند. لذا هر شناسه دارای شماره نسخه مربوط به خود است که از طریق این روتین می‌توان آنرا بدست آورد.	
هیچ پارامتری ندارد	
مقدار برگشتی : در صورت اجرای موفقیت آمیز شماره نسخه از نوع **WideString** برگشت داده می‌شود.	
انواع خطاها: **nErr_IOE**، **nErr_OK**، **nErr_WRP**، **nErr_OTE**، **nErr_DNF**، **nErr_BSY**، **nErr_DNS**	

GetMemorySize	دریافت مقدار حافظه قابل دسترس
Function GetMemorySize: Word;	
با توجه به نوع شناسه، حافظه قابل دسترس متغیر می‌باشد، با استفاده از این سرویس، می‌توان مقدار این حافظه را به بایت بدست آورد.	
هیچ پارامتری ندارد	
مقدار برگشتی : از نوع **WORD** می‌باشد و بعد از اجرا مقدار حافظه قابل دسترس را برگشت می‌دهد.	
انواع خطاها: **nErr_OTE**، **nErr_OK**، **nErr_WRP**، **nErr_IOE**، **nErr_DNF**، **nErr_BSY**، **nErr_DNS**	

GetMemorySizeEx	دریافت مقدار کاراکتر قابل دسترس
Function GetMemorySizeEx: WideString;	
با توجه به نوع شناسه، حافظه قابل دسترس و **charset** فضای داده ، متغیر می‌باشد، با استفاده از این سرویس، می‌توان مقدار این حافظه را برحسب کاراکتر و نوع **charset** بدست آورد.	
هیچ پارامتری ندارد	
مقدار برگشتی : از نوع **Widestring** می‌باشد، تعداد کاراکترهای حافظه و نوع **charset** را برگشت می‌دهد.	
انواع خطاها: **nErr_DNF**، **nErr_BSY**، **nErr_DNS**، **nErr_CS**، **nErr_CNS**، **nErr_DNS**	

SetDataByte	نوشتن یک بایت در حافظه شناسه
Procedure SetDataByte(const sVID: WideString; const sPWD: WideString; Address: Word;DataByte: Byte);	
برای نوشتن اطلاعات در شناسه بصورت بایت به بایت از این روتین استفاده می‌شود این روتین در آدرس مشخص شده اطلاعات یک بایت را می‌نویسد.	
sVID: این پارامتر **VID** می‌باشد که قبل از اجرای روتین باید مقدار دهی شود و بصورت **WideString** است. **sPWD**: این پارامتر رمز عبور شناسه می‌باشد که قبل از اجرای روتین باید مقدار دهی شود و بصورت **WideString** باشد. **Address**: آدرس مورد نظر جهت نوشتن اطلاعات و از نوع **Word** می‌باشد. **DataByte**: داده مورد نظر در این پارامتر ذخیره و بعد از اجرای روتین ، در آدرس مشخص شده ذخیره می‌گردد. نوع آن **byte** می‌باشد.	
هیچ مقدار بازگشتی ندارد	
انواع خطاها: **nErr_OK**، **nErr_WRP**، **nErr_IOE**، **nErr_IVP**، **nErr_INP**، **nErr_DNF**، **nErr_BSY**	

SetDataBlockStr	نوشتن اطلاعات در حافظه بصورت رشته ای
Procedure SetDataBlockStr(const sVID:WideString; const sPWD: WideString; Start: Word;DataLen: Word; const DataBlockStr:WideString;const defChar: WideString=' ');	
در صورتیکه بخواهید بیش از یک بایت را بصورت رشته ای در حافظه ذخیره نمائید باید از این روتین استفاده نمائید.	
sVID: این پارامتر **VID** می‌باشد که قبل از اجرای روتین باید مقدار دهی شود و بصورت **WideString** است. **sPWD**: این پارامتر رمز عبور شناسه می‌باشد که قبل از اجرای روتین باید مقدار دهی شود و بصورت **WideString** باشد. **Start**: آدرس شروع برای ذخیره اطلاعات و از نوع **Word** **DataLen**: این متغیر که از نوع **Word** می‌باشد اندازه بلوک مورد نظر برای نوشتن را مشخص می‌نماید. **DataBlockStr**: اطلاعات مورد نظر دراین متغیر که یک آرایه از نوع **WideString** می‌باشد ذخیره و پس از اجرای روتین در شناسه نوشته می‌شود. **defChar**: در صورتیکه رشته مورد نظر کمتر از طول ذکر شده باشد بقیه حافظه با این کاراکتر پر می‌شود.	
هیچ مقدار بازگشتی ندارد	
انواع خطاها: **nErr_IVP**، **nErr_OTE**، **nErr_WRP**، **nErr_IOE**، **nErr_INP**	

SetDataHexBlock	نوشتن بلوک HEX در بخش داده
Procedure SetDataHexBlock(const sVID: WideString; const sPWD: WideString; Start: Word;DataLen: Word; const DataHexBlock: WideString);	
در صورتیکه بخواهید بیش از یک بایت را بصورت رشته **HEX** در حافظه ذخیره نمائید باید از این روتین استفاده نمائید.	
sVID: پارامتر **VID** می‌باشد که قبل از اجرای روتین باید مقدار دهی شود وباید بصورت **WideString** باشد. **sPWD**: این پارامتر رمز عبور شناسه می‌باشد که قبل از اجرای روتین باید مقدار دهی شود و بصورت **WideString** باشد. **Start**: آدرس شروع برای ذخیره اطلاعات و از نوع **Word** **DataLen**: این متغیر که از نوع **Word** می‌باشد اندازه بلوک مورد نظر برای نوشتن را مشخص می‌نماید. **DataHexBlock**: اطلاعات مورد نظر دراین متغیر که یک آرایه از نوع **WideString** می‌باشد ذخیره و پس از اجرای روتین در شناسه نوشته می‌شود.	
هیچ مقدار بازگشتی ندارد	
انواع خطاها: **nErr_OTE**، **nErr_WRP**، **nErr_IOE**، **nErr_IVP**، **nErr_DNF**، **nErr_BSY**	

GetDataByte	خواندن یک بایت از حافظه شناسه
Function GetDataByte(const sVID: WideString; const sPWD: WideString; Address: Word): Byte;	
برای خواندن اطلاعات از شناسه بصورت بایت به بایت از این روتین استفاده می‌شود این روتین از آدرس مشخص شده اطلاعات یک بایت را می‌خواند.	
sVID: پارامتر **VID** می‌باشد که قبل از اجرای روتین باید مقدار دهی شود وباید بصورت **WideString** باشد. **sPWD**: این پارامتر رمز عبور شناسه می‌باشد که قبل از اجرای روتین باید مقدار دهی شود و بصورت **WideString**می باشد. **Address**: آدرس مورد نظر جهت خواندن اطلاعات و از نوع **Word** می‌باشد.	
مقدار برگشتی : در صورت اجرای موفقیت مقدار موجود در آدرس مورد نظر برگشت داده می‌شود و نوع آن **byte** می‌باشد.	
انواع خطاها: **nErr_OK**، **nErr_WRP**، **nErr_IOE**، **nErr_IVP**، **nErr_INP**، **nErr_DNF**، **nErr_BSY**	

GetDataBlockStr	خواندن اطلاعات از حافظه بصورت رشته ای
Function GetDataBlockStr(const sVID: WideString; const sPWD: WideString; Start: Word;DataLen: Word): WideString;	
در صورتیکه بخواهید بیش از یک بایت را بصورت رشته ای از حافظه بخوانید باید از این روتین استفاده نمائید.	
sVID: پارامتر **VID** می‌باشد که قبل از اجرای روتین باید مقدار دهی شود وباید بصورت **WideString** باشد. **sPWD**: این پارامتر رمز عبور شناسه می‌باشد که قبل از اجرای روتین باید مقدار دهی شود و بصورت **WideString** باشد. **Start**: آدرس شروع برای خواندن اطلاعات و از نوع **Word** **DataLen**: این متغیر که از نوع **Word** می‌باشد اندازه بلوک مورد نظر برای خواندن را مشخص می‌نماید.	
مقدار برگشتی : اطلاعات مورد نظر پس از اجرای موفقیت آمیز روتین بصورت یک **widestring** برگشت داده می‌شود.	
انواع خطاها: **nErr_OTE**، **nErr_CNS**، **nErr_CS**، **nErr_IVP**، **nErr_INP**، **nErr_WRP**، **nErr_IOE** ، **nErr_IVP**، **nErr_INP**، **nErr_DNF**، **nErr_BSY**، **nErr_DNS**	

GetDataHexBlock	خواندن بلوک **HEX** از بخش داده
Function GetDataHexBlock(const sVID: WideString; const sPWD: WideString; Start: Word;DataLen: Word): WideString;	
در صورتیکه بخواهید بیش از یک بایت را بصورت رشته **HEX** از حافظه بخوانید از این روتین استفاده نمائید.	
sVID: این پارامتر **VID** می‌باشد که قبل از اجرای روتین باید مقدار دهی شود وباید بصورت **WideString** باشد. **sPWD**: این پارامتر رمز عبور شناسه می‌باشد که قبل از اجرای روتین باید مقدار دهی شود و بصورت **WideString** باشد. **Start**: آدرس شروع برای خواندن اطلاعات و از نوع **Word** **DataLen**: این متغیر که از نوع **Word** می‌باشد اندازه بلوک مورد نظر برای خواندن را مشخص می‌نماید.	
مقدار برگشتی : اطلاعات مورد نظر پس از اجرای موفقیت آمیز روتین بصورت یک **widestring** برگشت داده می‌شود.	
انواع خطاها: **nErr_INP**، **nErr_DNF**، **nErr_BSY**، **nErr_OTE**، **nErr_WRP**، **nErr_IOE**، **nErr_IVP**، **nErr_DNS**	

رمزنگاری	**GetEncryption**
Function GetEncryption(const sVID: WideString; const sPWD: WideString; const sPData: WideString;Repetition: Word): WideString;	
یکی از روتین های مهم و کاربردی در شناسه عمل رمزنگاری (**Encryption**) می‌باشد. این روتین با استفاده از کلید ذخیره شده در خود شناسه ، اطلاعات ارسالی را رمزنگاری می‌نماید ، این عمل می‌تواند بصورت متوالی انجام شود به این صورت که پارامتر ارسالی بعد از رمزنگاری شدن ، دوباره رمزنگاری شود.	
sVID: این پارامتر **VID** می‌باشد که قبل از اجرای روتین باید مقدار دهی شود و بصورت **WideString** است. **sPWD**: این پارامتر رمز عبور شناسه می‌باشد که قبل از اجرای روتین باید مقدار دهی شود و بصورت **WideString** باشد. **sPData** اطلاعات مورد نظر برای رمزنگاری بصورت **WideString** در این پارامتر قرار می‌گیرد. **Repetition**: این متغیر که از نوع **Word** می‌باشد تعداد دفعات برای انجام عمل رمزنگاری را مشخص می‌کند این عدد حداقل باید یک باشد.	
مقدار برگشتی : اطلاعات رمزنگاری شده بعد از اجرای موفقیت آمیز روتین و از نوع **WideString** برگشت داده می‌شود.	
انواع خطاها:**nErr_OK**، **nErr_IOE**، **nErr_IVP**، **nErr_DNF**، **nErr_BSY**	

رمزگشایی	**GetDecryption**
Function GetDecryption(const sVID: WideString; const sPWD: WideString; const sCData: WideString;Repetition: Word): WideString;	
برای رمزگشایی اطلاعات رمزنگاری شده از این روتین استفاده می‌شود. به این صورت که در صورتیکه عددی را رمزنگاری و سپس عدد بدست آمده را رمزگشایی نمائیم ، عدد اول بدست می‌آید به این شرط که تعداد تکرار در هر دو مرحله یکسان باشد.	
sVID: این پارامتر **VID** می‌باشد که قبل از اجرای روتین باید مقدار دهی شود و بصورت **WideString** است. **sPWD**: این پارامتر رمز عبور شناسه می‌باشد که قبل از اجرای روتین باید مقدار دهی شود و بصورت **WideString** باشد. **sCData**: اطلاعات رمزنگاری شده که می‌خواهیم رمزگشایی شود و به صورت **WideString** می‌باشد. **Repetition**: این متغیر که از نوع **Word** می‌باشد تعداد دفعات برای انجام عمل رمزگشایی را مشخص می‌کند.	
مقدار برگشتی : در صورت اجرای موفقیت آمیز : اطلاعات رمزگشایی شده برگشت داده می‌شود که از نوع **WideString** می‌باشد.	

انواع خطاها: **nErr_OK**، **nErr_IOE**، **nErr_IVP**، **nErr_INP**، **nErr_DNF**، **nErr_BSY**

ConvDelimiteredStringToString	تبدیل رشته از **DNString** به رشته کاراکتری
Function ConvDelimiteredStringToString(const strDCnv: WideString;const ALen: Word=16;const Sep: WideString='.'): WideString;	
با استفاده از این روتین می‌توان رشته کاراکتری را به **DN** تبدیل کرد	
strDCnv: این پارامتر رشته ورودی می‌باشد که می‌خواهیم تبدیل کنیم و به صورت **WideString** می‌باشد. **ALen**: طول رشته ای که قرار است تبدیل شود را مشخص می‌کند. **Sep**: کاراکتر واسط در خروجی را تعیین می‌کند که به صورت پیشفرض **'.'** است	
مقدار برگشتی : در صورت اجرای موفقیت آمیز رشته تبدیل شده بصورت کاراکتری و **WideString** برگشت داده می‌شود.	
خطا ندارد.	

ConvStringToDelimiteredString	تبدیل رشته از رشته کاراکتری به **DNString**
Function ConvStringToDelimiteredString(const strCnv: WideString;const ALen: Word=16;const Sep: WideString='.'): WideString;	
با استفاده از این روتین می‌توان رشته کاراکتری را به **DN** تبدیل کرد	
strCnv: این پارامتر رشته ورودی می‌باشد که می‌خواهیم تبدیل کنیم و رشته ای از نوع **WideString** باشد. **Sep**: کاراکتر واسط در خروجی را تعیین می‌کند که به صورت پیشفرض **'.'** است	
مقدار برگشتی : در صورت اجرای موفقیت آمیز رشته تبدیل شده بصورت کاراکتری و **WideString** برگشت داده می‌شود.	
خطا ندارد.	

ConvHexStringToDelimiteredString	تبدیل رشته HEX به DNString
Function ConvHexStringToDelimiteredString(strHCnv: WideString;const ALen: Word=16;const Sep: WideString='.'): WideString;	
با استفاده از این روتین می‌توان رشته مبنای ۱۶ (**HEX**) را به **DN** تبدیل کرد	
strHCnv: این پارامتر رشته ورودی می‌باشد که می‌خواهیم تبدیل کنیم و به صورت **WideString** می‌باشد. **ALen**: طول رشته ای که قرار است تبدیل شود را مشخص می‌کند. **Sep**: کاراکتر واسط در خروجی را تعیین می‌کند که به صورت پیشفرض '.' است	
مقدار برگشتی : در صورت اجرای موفقیت آمیز رشته تبدیل شده بصورت کاراکتری و **WideString** برگشت داده می‌شود.	
خطا ندارد.	

ConvDelimiteredStringToHexString	تبدیل DNString به رشته HEX
Function ConvDelimiteredStringToHexString(const strDCnv: WideString;const ALen: Word=16;const DSep: WideString='.';const HSep: WideString=' '): WideString;	
با استفاده از این روتین می‌توان رشته **DN** را به رشته مبنای ۱۶ (**HEX**) تبدیل کرد	
strDCnv: این پارامتر رشته ورودی می‌باشد که می‌خواهیم تبدیل کنیم و به صورت **WideString** می‌باشد. **ALen**: طول رشته ای که قرار است تبدیل شود را مشخص می‌کند. **Dsep** و **HSep**: کاراکتر واسط در ورودی و خروجی را تعیین می‌کند که به صورت پیشفرض '.' و ' ' است	
مقدار برگشتی : در صورت اجرای موفقیت آمیز رشته تبدیل شده بصورت کاراکتری و **WideString** برگشت داده می‌شود.	
خطا ندارد.	

ConvStringToHEXString	تبدیل رشته معمولی به رشته HEX
Function ConvStringToHEXString(const strCnv: WideString;const ALen: Word=16;const Sep: WideString=' '): WideString;	
با استفاده از این روتین می‌توان رشته معمولی را به رشته مبنای ۱۶ (**HEX**) تبدیل کرد	
strCnv: این پارامتر رشته ورودی می‌باشد که می‌خواهیم تبدیل کنیم و به صورت **WideString** می‌باشد. **ALen**: طول رشته ای که قرار است تبدیل شود را مشخص می‌کند. **HSep**: کاراکتر واسط در خروجی را تعیین می‌کند که به صورت پیشفرض **' '** است	
مقدار برگشتی : در صورت اجرای موفقیت آمیز رشته تبدیل شده بصورت کاراکتری و **WideString** برگشت داده می‌شود.	
خطا ندارد.	

ConvHEXStringToString	تبدیل رشته HEX به رشته معمولی
Function ConvHEXStringToString(strHCnv: WideString;const ALen: Word=16): WideString;	
با استفاده از این روتین می‌توان رشته مبنای ۱۶ (**HEX**) را به رشته معمولی تبدیل کرد	
strHCnv: این پارامتر رشته ورودی می‌باشد که می‌خواهیم تبدیل کنیم و به صورت **WideString** می‌باشد. **ALen**: طول رشته ای که قرار است تبدیل شود را مشخص می‌کند.	
مقدار برگشتی : در صورت اجرای موفقیت آمیز رشته تبدیل شده بصورت کاراکتری و **WideString** برگشت داده می‌شود.	
خطا ندارد.	

SetUserPWD	تنظیم و تغییر سطح سوم دسترسی
Procedure SetUserPWD(sOUPWD,sUPWD:WideString);	
رمز عبور سطح سوم (کاربر) توسط این روتین قابل تنظیم و تغییر است.	
sOUPWD: رمز عبور فعلی که برای اولین بار خالی است. **sUPWD**: رمز عبور جدید که قرار است تنظیم شود. هر دو پارامتر بالا از نوع **widestring** است و باید به فرمت **DNString** به تابع داده شود	
هیچ مقدار بازگشتی ندارد	
خطا ندارد.	

SetUserKEY	تنظیم کلید رمزنگاری شخصی (کاربر)
Procedure SetUserKEY(sUPWD,sNKEY:WideString);	
برای انجام رمزنگاری شخصی یک کلید ۱۶ بایتی (۱۲۸ بیتی) در شناسه تنظیم می‌شود توجه داشته باشید این کلید فقط می‌تواند یک بار در شناسه ست شود و هنگام تنظیم مجدد خطا خواهد داد.	
sUPWD: رمز عبور فعلی کاربر (سطح سوم) **sNKEY**: کلیدی که قرار است تنظیم شود. هر دو پارامتر بالا از نوع **widestring** است و باید به فرمت **DNString** به تابع داده شود	
هیچ مقدار بازگشتی ندارد	
انواع خطاها: **SetUserKEY**، **nErr_BSY**، **nErr_OK**، **nErr_WRP**، **nErr_IOE**، **nErr_DNF**	

GetTimerCounter	دریافت مقدار شمارنده/زمانسنج
Function GetTimerCounter: integer;	
در صورتی که تنظیمات خاصی برای محدود کردن دفعات یا مدت زمان استفاده از قفل تعیین شده باشد با استفاده از این روتین می‌توان از آن تنظیمات مطلع شد.	
هیچ پارامتری ندارد	
مقدار بازگشتی مقدار عدد موجود در پارامتر شمارنده قفل است. در صورتی که کاهنده خودکار (**Timer**) ست نشده باشد خطای **Nlevt_TimerNS** در پارامتر خطا مشاهده خواهد شد.	
انواع خطاها:**nErr_OK**، **nErr_WRP**، **nErr_IOE**، **nErr_DNF**، **nErr_BSY**	

Activate	فعالسازی مجدد قفل
Procedure Activate(ActivationCode: WideString);	
در صورتی که در طول استفاده از شناسه مقدار شمارنده به صفر برسد، شناسه به حالت **Suspend** می‌رود که برای خروج از این حالت و غیر فعال کردن شمارنده/زمانسنج می‌توان با استفاده از یک کد فعال سازی صحیح شناسه را از طریق این روتین فعال کرد.	
ActivationCode: رشته فعالسازی است که توسط **NLBuilder** ایجاد می‌گردد.	
هیچ مقدار بازگشتی ندارد	
انواع خطاها:**nErr_INP**، **nErr_OK**، **nErr_WRP**، **nErr_IOE**، **nErr_INP**، **nErr_BSY**، **nErr_DNS**	

DecreaseCounter	کاهش یک واحد از شمارنده
Procedure DecreaseCounter;	
در طول استفاده از شناسه می‌توان مقداری را به عنوان دفعات استفاده از شناسه تعیین و با استفاده از این روتین در مقاطع مورد نظر آنرا یک واحد کاهش داد.	
هیچ پارامتری ندارد	
هیچ مقدار بازگشتی ندارد	
انواع خطاها:**nErr_OTE**، **nErr_OK**، **nErr_IOE**، **nErr_DNF**، **nErr_BSY**	

RSAEncrypt	رمزنگاری نامتقارن RSA
Function RSAEncrypt(KeyType: Integer; const sPData: WideString; const sUPWD: WideString;const sKey: WideString): WideString;	
با استفاده از این روتین می‌توانید اطلاعات را با الگوریتم **RSA** با استفاده از کلید ذخیره شده در شناسه یا کلید خارجی رمزنگاری کنید.	
KeyType: نوع کلید است که می‌تواند ۱ به معنی کلید عمومی یا ۲ به معنی کلید خصوصی باشد **sPData**: رشته ای است که قرار است رمزنگاری شود. این رشته باید به صورت معمولی (کاراکتری) پاس شود **sUPWD**: در صورت استفاده از کلید خصوصی ذخیره شده در شناسه باید پین کد در این پارامتر قرار گیرد. **sKey**: پارامتری است اختیاری برای پاس کردن کلید رمزنگاری خارجی به صورت رشته **HEX** که می‌توان کلیدی با طول ۲۵۶، ۵۱۲ یا ۱۰۲۴ بیت استفاده کرد.	
مقدار بازگشتی: رمز شده داده است که به صورت **HEX** برمی گردد. در صورتی که طول رشته دریافتی نسبت به کلید طولانی باشد، خروجی به صورت بخشهای جداشده با **','** بازگشت داده می‌شود	
انواع خطاها: **nErr_INVKL**، **nErr_IOE**، **nErr_RSANS**، **nErr_DNF**، **nErr_BS** ، **nErr_OTE**، **nErr_DNS**	

RSADecrypt	رمزگشایی نامتقارن RSA
Function RSADecrypt(KeyType: Integer; const sCData: WideString; const sUPWD: WideString;const sKey: WideString): WideString;stdcall export;	
با استفاده از این روتین می‌توانید اطلاعات را با الگوریتم **RSA** با استفاده از کلید ذخیره شده در شناسه یا کلید خارجی رمزگشایی کنید.	
KeyType: نوع کلید است که می‌تواند ۱ به معنی کلید عمومی یا ۲ به معنی کلید خصوصی باشد. **sCData**: رشته ای است که قرار است رمزگشایی شود. این رشته باید به صورت **HEX** پاس شود . **sUPWD**: در صورت استفاده از کلید خصوصی ذخیره شده در شناسه باید رمز عبور سوم در این پارامتر قرار گیرد. **sKey**: پارامتری است اختیاری برای پاس کردن کلید رمزنگاری خارجی به صورت رشته **HEX** که می‌توان کلیدی با طول ۵۱۲ یا ۱۰۲۴ بیت استفاده کرد.	
مقدار بازگشتی: رمز شده داده است که به صورت **HEX** برمی گردد. در صورتی که طول رشته دریافتی نسبت ب کلید طولانی باشد، خروجی به صورت بخشهای جداشده با **','** بازگشت داده می‌شود	
انواع خطاها:**nErr_OTE**، **nErr_INVKL**، **nErr_IOE**، **nErr_OK**، **Err_IOE** ، **nErr_RSANS** ، **nErr_BSY**، **nErr_DNF**، **nErr_NSD**	

GetPublicKey	گرفتن کلید عمومی از CA
Function GetPublicKey(const SerialNo: WideString; const ConsoleURL: WideString): WideString;stdcall export;	
در صورتی که کلید عمومی شناسه ها لازم باشد با استفاده از این تابع می‌توان کلید عمومی را از سرویس دهنده مربوطه از وب سایت مورد نظر دریافت نمود	
SerialNo: شماره سریال شناسه مورد نظر است. **ConsoleURL**: پارامتری است اختیاری مربوط به صفحه سرویس دهنده شناسه، این صفحه می‌تواند در **WebConfig** در داخل شناسه تنظیم شده باشد.	
مقدار بازگشتی: در صورتی که عملیات با موفقیت انجام شود کلید عمومی مربوط به شناسه مورد نظر به صورت رشته **HEX** بازگشت داده می‌شود.	
انواع خطاها: **nErr_OK**، **nErr_IOE**، **nErr_RSANS**، **nErr_DNF**، **nErr_BSY**، **nErr_OTE**	

GetHashStr	ایجاد Hash یک داده
Function GetHashStr(HashType: Integer; const sPData: WideString): WideString;stdcall export;	
از این روتین برای درهمسازی اطلاعات (**Hashing**) استفاده می‌شود.	
HashType: نوع درهمسازی می‌توانید اعدادی از ۱، ۲، ۳ یا ۴ باشد که به معنی الگوریتم **MD2**، **MD4**، **MD5** یا **SHA-1** است. **sPData**: داده ای است که قرار است **Hash** شود، که به صورت رشته معمولی پاس می‌شود.	
مقدار بازگشتی: درهم شده رشته ورودی است که به صورت رشته **HEX** باز می‌گردد.	
انواع خطاها: **nErr_OK**، **nErr_INVHT**، **nErr_OTE**	

GetSignature	ایجاد امضای دیجیتال یک داده
Function GetSignature(const sUPWD: WideString; const sPData: WideString; HashType: Integer;const sPrivateKey: WideString): WideString;stdcall export;	
ایجاد امضای دیجیتال یه داده را می‌توان با روتین فوق انجام داد. این عمل با انتخاب یک گلید خصوصی و یک روش درهمسازی انجام می‌پذیرد.	
sUPWD: رمز عبور سطح سوم شناسه است که به صورت رشته **DN** پاس می‌شود **sPData**: داده است که قرار است امضا شود که به صورت رشته معمولی پاس می‌شود. **HashType**: نوع درهمسازی می‌توانید اعدادی از ۱، ۲، ۳ یا ۴ باشد که به معنی الگوریتم **MD2**،**MD4** ، **MD5** یا **SHA-1** است. **sPrivateKey**: پارامتری است اختیاری برای پاس کردن کلید رمزنگاری خصوصی خارجی به صورت رشته **HEX** که می‌تواند کلیدی با طول ۵۱۲ یا ۱۰۲۴ بیت باشد	
مقدار بازگشتی: امضای داده پاس شده است که به صورت رشته **HEX** می‌باشد.	
انواع خطاها: **nErr_OK, nErr_INVHT, nErr_DNF, nErr_BSY, nErr_OTE, nErr_DNS, nErr_RSANS, nErr_IOE, nErr_INVKL, nErr_INSKL,**	

VerifySignature	تست اعتبار امضای یک داده
procedure VerifySignature(HashType: Integer; const sPData: WideString;const sSignature: WideString; const sPublicKey: WideString);stdcall export;	
با داشتن کلید عمومی شناسه امضا کننده، می‌توان بر اساس داده و نوع **hash** امضای آن را مورد ارزیابی قرار داد. در صورت بروز هر مشکل مقدار خطا را می‌توان در **ErrNo** دریافت کرد	
HashType: نوع درهمسازی می‌توانید اعدادی از ۱، ۲، ۳ یا ۴ باشد که به معنی الگوریتم **MD2**، **MD4**، **MD5** یا **SHA-1** است. **sPData**: داده اصلی است که به صورت رشته معمولی پاس می‌شود. **sSignature**: امضای ارائه شده برای داده فوق است که به صورت رشته **HEX** است. **sPublicKey**: پارامتر اختیاری کلید عمومی است که به صورت **HEX** پاس می‌شود. در صورت استفاده نکردن از این کلید، امضا با استفاده از کلید عمومی داخل شناسه عمل فوق را انجام می‌دهد.	
هیچ مقدار بازگشتی ندارد	
انواع خطاها:**nErr_OTE**، **nErr_INSKL**، **nErr_INVKL**، **nErr_IOE**، **nErr_OK**، **nErr_RSANS** ، **nErr_DNF**، **nErr_BSY**	

SetRSA	ایجاد و تنظیم کلیدهای RSA
function SetRSA(const sUPWD: WideString; KeyLength: Integer): WideString;	
در صورتی که بخواهیم جفت کلید برای رمزنگاری **RSA** ایجاد و در شناسه ذخیره کنیم باید از این تابع استفاده کنیم. تابع در صورتی اجازه این کار را خواهید داشت که مدیریت شناسه این دسترسی را هنگام برنامه ریزی (توسط **Builder**) داده باشد.	
sUPWD: رمز عبور کاربر نهایی (**PIN code**) **KeyLength**: طول کلید رمزنگاری (۵۱۲ یا ۱۰۲۴)	
مقدار بازگشتی کلید عمومی تنظیم شده در شناسه است.	
انواع خطاها: **nErr_OTE**، **nErr_IOE**، **nErr_OK**، **nErr_INVKL**، **nErr_INP**، **nErr_BSY**، **nErr_DNS**	

ErrNo	کد خطای حاصل از آخرین متد اجرا شده
property ErrNo: Word	
توابع فوق در صورت اجرای موفق باعث بازگشت مقدار –۰– از طریق این **property** می‌شوند ولی در صورت بروز خطا این پارامتر عدد غیر صفر بر می‌گرداند.	
این مشخصه به طور یکطرفه خواندنی است و مقدار عددی از نوع **word** بر می‌گرداند	

ErrDescr	پیام خطای متناسب با کد خطا (انگلیسی)
property ErrDescr: WideString	
توابع در صورت اجرای موفق باعث بازگشت کلمه **'OK'** از طریق این **property** می‌شوند ولی در صورت بروز خطا پیام متناسب با نوع خطا از این مشخصه قابل دریافت است.	
این مشخصه به طور یکطرفه خواندنی است و مقدار عددی از نوع **widestring** بر می‌گرداند	

ErrDescrFA	پیام خطای متناسب با کد خطا (فارسی)
property ErrDescrFA: WideString	
توابع در صورت اجرای موفق باعث بازگشت کلمه «تایید» از طریق این **property** می‌شوند ولی در صورت بروز خطا پیام متناسب با نوع خطا از این مشخصه قابل دریافت است.	
این مشخصه به طور یکطرفه خواندنی است و مقدار عددی از نوع **widestring** بر می‌گرداند	

DeviceName	نام وسیله در نتیجه جستجو
property DeviceName: WideString	
پس از استفاده از **GetFirstDevice** یا **GetNextDevice** این **property** نام وسیله شناسایی شده را نمایش می‌دهد.	
این مشخصه به طور یکطرفه خواندنی است و مقدار عددی از نوع **widestring** بر می‌گرداند	

DeviceVer	نگارش وسیله در نتیجه جستجو
property DeviceVer: WideString	
پس از استفاده از **GetFirstDevice** یا **GetNextDevice** این **property** نگارش وسیله شناسایی شده را نمایش می‌دهد.	
این مشخصه به طور یکطرفه خواندنی است و مقدار عددی از نوع **widestring** بر می‌گرداند	

DeviceSerial	شماره سریال وسیله در نتیجه جستجو
property DeviceSerial: WideString	
پس از استفاده از **GetFirstDevice** یا **GetNextDevice** این **property** شماره سریال وسیله شناسایی شده را نمایش می‌دهد.	
این مشخصه به طور یکطرفه خواندنی است و مقدار عددی از نوع **widestring** بر می‌گرداند	

SelectedName	نام وسیله انتخاب شده
property SelectedName: WideString	
پس از استفاده از **SelectDevice** این **property** نام وسیله شناسایی شده را نمایش می‌دهد.	
این مشخصه به طور یکطرفه خواندنی است و مقدار عددی از نوع **widestring** بر می‌گرداند	

SelectedVer	نگارش وسیله انتخاب شده
property SelectedVer: WideString	
پس از استفاده از **SelectDevice** این **property** نگارش وسیله شناسایی شده را نمایش می‌دهد.	
این مشخصه به طور یکطرفه خواندنی است و مقدار عددی از نوع **widestring** بر می‌گرداند	

SelectedSerial	شماره سریال وسیله انتخاب شده
property SelectedSerial: WideString	
پس از استفاده از **SelectDevice** این **property** شماره سریال وسیله شناسایی شده را نمایش می‌دهد.	
این مشخصه به طور یکطرفه خواندنی است و مقدار عددی از نوع **widestring** بر می‌گرداند	

فصل چهارم: کاربردهای شناسه نوین

شناسه نوین کاربردهای مختلفی دارد که در ادامه به چندین مورد اشاره خواهیم کرد.

۴-۱- احراز هویت[1] کاربران به روش OTP[2]

در این روش از امکانات رمزنگاری AES شناسه استفاده می شود.

برای فهم روش OTP مجموعه ای از رمزهای عبور را به صورت $P_0, P_1, P_2 ... P_i, P_{i+1} ... P_n$ را در نظر بگیرید. ارتباط رمز iام با رمز i+1ام یک رابطه ریاضی یا یک الگوریتم می تواند باشد.

برای این الگوریتم می توان از AES که مبتنی بر یک کلید ۱۲۸ بیتی است استفاده کرد. پس می توان با اختصاص کلید AES یکتا برای هر کاربر (درون شناسه) این ارتباط منطقی رمزها را یکتا کرد. برای اینکه کاربر را احراز هویت کنیم می توان رمزها را از ۰ تا n روی سرور ذخیره کرد. با ارسال رمز iام به کاربر و گرفتن جواب درست یعنی رمز i+1ام می توان کاربر را احراز هویت کرد. علاوه بر این می توان به جای ذخیره همه رمزها که ممکن است محدودیت هایی را در اجرا به وجود آورد فقط کلید کاربران را در سرور ذخیره کرد. برای اینکه بتوانیم این روش را با شناسه نوین پیاده سازی کنیم باید مراحل را به صورت زیر اجرا کنیم:

۱. تنظیم رمز عبور سوم (کاربر) و کلید رمزنگاری AES کاربر با استفاده از متدهای SetUserPWD و SetUserKey
۲. ایجاد یک بانک اطلاعاتی شامل فیلدهایی برای شماره سریال شناسه، اطلاعات آشکار، اطلاعات رمزنگاری شده و یک flag (boolean) برای عدم استفاده مجدد از رکورد است .
۳. پر کردن بانک اطلاعاتی با استفاده از رمزنگاری AES با شروع از یک رشته شانسی (P_0) وتولید رمزهای بعدی(P_i) به صورت پشت سر هم به تعداد دلخواه (مثلا ۵۰ عدد یعنی (n=50
٤. در هر مرحله از چک کردن شناسه مثلا در هنگام Log in به وب سایت اولین رمزی که تا به حال استفاده نشده است یا به عبارت بهتر آخرین رمزی که کاربر به سرور ارسال کرده است را بر اساس شماره سریال شناسه کاربر استخراج و به سمت کاربر ارسال می کنیم.
۵. دریافت رمز عبور سوم از کاربر و انجام عمل رمزنگاری و ارسال کردن جواب آن به سرور
٦. در صورتی که جواب رمزنگاری با جوابی که قبلا در بانک ذخیره شده بود (رمز بعدی سوال) برابر بود. آن رکورد برای عدم استفاده در آینده علامت می‌خورد و به کاربر اجازه ورود به وب سایت داده می‌شود.

---

[1] Authentication
[2] One-Time Password

۷. در صورتی که رمزنگاری ها در بانک تمام شد، یعنی همه موارد مورد استفاده قرار گرفت می‌توان با استفاده از روش کنسولی یک سری رمزنگاری های دیگر را جایگزین کرد.

توجه داشته باشید که این روش بیشتر در ورود به وب سایت یا در هنگام انجام یک عمل حساس استفاده می‌شود و در حرکت در لابلای صفحات استفاده نمی شود.

جهت درک بهتر موضوع به مثالی که در شاخه Samples/ASP/DS/OTP هست توجه نمایید.

۴-۲- رمزنگاری نامتقارن RSA و امضای دیجیتال

جدیدترین امکانی که به شناسه نوین اضافه گردیده، امکان استفاده از رمزنگاری RSA و امضای دیجیتال است که بر اساس رمزنگاری RSA و الگوریتمهای درهم سازی (Hashing) پیاده سازی شده است. برای فهم کامل موضوع به مستندات مربوطه در CD همراه شناسه مراجعه کنید.

پیوست ۱: توضیح نمونه کد

توضیح نمونه به زبان PHP

صفحات مربوطه عبارتند از:

Index.php :

صفحه ورودی که شامل لینکهای زیر است :

در قسمت مدیریت :

- ثبت نام
- اعتباردهی برای ورود (شارژ OTP)

در قسمت عمومی:

- ورود (به روش OTP)

در قسمت کاربران :

- تغییر PINCode (رمزعبور شناسه)
- ارسال نامه
- صندوق دریافت

اولین بار که وارد می‌شویم کاربر مهمان در نظر گرفته می‌شود. که در صورت ثبت نام کردن نام کاربر با شماره سریال در بالای صفحه ثبت می‌شود.

کد مربوطه :

```
<?
    if ($_SESSION["UserSerial"]==""){ ?>مهمان<?
    }
    else
          echo($_SESSION["UserFullname"]."(".$_SESSION["UserSerial"].")");
?>
```

Register.php:

این فرم برای ثبت نام کاربر شناسه است. در صورتی که چند شناسه به پورت متصل باشد شماره سریال مورد نظر را انتخاب می‌کنیم و با زدن کلید خواندن ، کلید عمومی داخل شناسه خوانده می‌شود در صورتی که شناسه قبلا ثبت شده باشد، مشخصات به روزرسانی می‌شود در غیر این صورت مشخصات وارد شده به بانک اضافه می‌گردد.

Check_Lock(): تابع javascript ای که تعداد شناسه های متصل به سیستم را چک می‌کند.

LoadPublic(): این تابع کلید عمومی داخل شناسه را می‌دهد که با زدن کلید خواندن فراخوانی می‌شود.

```
function LoadPublic()
{
    if (nl.GetDeviceCount()>1)
    {
        nl.SelectDevice(devSerials[RegForm.devs.selectedIndex]);
    }
    RegForm.PublicKey.value=nl.GetPublicKey(nl.GetSerial());
    if (nl.ErrNo) alert('Error in getting public key:'+nl.ErrDescr);
}
```

دستورات مربوط به ثبت مشخصات در بانک اگر شماره سریال در بانک وجود نداشته باشد، مشخصات به بانک افزوده می‌گردد اما اگر شماره سریال در بانک باشد مشخصات تنها به روز رسانی می‌شود.

```
<?
    if ( $_POST["Level"]=="next")
    {
            $Serial=$_POST["devSerial"];
            if ($Serial=="")
            $Serial=$_POST["devs"];
            $cn=odbc_connect("dssample","","");
            $rs=odbc_exec($cn,"Select * from tbPersons where TokenSerial='".$Serial."'");
            if (!odbc_fetch_row($rs))
            {
                    $sql="INSERT INTO tbPersons( FName, LName, TokenSerial, TokenPublicKey)
                    VALUES('".$_POST[Fname]."','".$_POST[Lname]."','".$Serial."','".$_POST[PublicKey]."')";
                    odbc_exec($cn,$sql);
                    $errs="شد انجام موفقیت با نام ثبت";
            }
            else
            {
                    $sql="Update tbPersons set FName='".$_POST[Fname]."', LName='".$_POST[Lname]."',
                    TokenPublicKey='".$_POST[PublicKey]."' Where TokenSerial='".$Serial."'";
                    odbc_exec($cn,$sql);
                    $errs="شد بروزرسانی مشخصات .بود شده ثبت قبلا سریال شماره";
            }
    }
?>
```

:Charge.php

شارژ اعتبار جهت ورود به روش OTP

در صورتی که شناسه موجود باشد این فرم نمایش داده می‌شود بایست رمز عبور شناسه (رمز عبور کاربر نباید رمز عبر مدیریت باشه؟) را وارد کرد تعداد اعتبار ورود به معنای حداکثر تعداد ورود کاربر است.

ورود به روش OTP با الگوریتم AES انجام پذیر است که در این جا می‌توانیم کلید اصلی آن را ایجاد کنیم. با کلیک روی دکمه شارژ عمل اعتبار دهی انجام می‌گیرد.

با زدن کلید شارژ تابع ()GoGenerate فرا خوانی میشود اگر تعیین کلید جدید برای AES تیک خورده باشد چک می‌کند که آیا این امر امکان پذیر است یا نه. سپس تابع ()GenerateNew از داخل این تابع فراخوانی می‌شود این تابع عمل اعتبار دهی را انجام می‌دهد.

```
function GoGenerate()
{
    doWork = true;
    OTPCH.doBTN.disabled=true;
    if (nl.GetDeviceCount()>1)
    {
        nl.SelectDevice(devSerials[OTPCH.devs.selectedIndex]);
        OTPCH.Serial.value =devSerials[OTPCH.devs.selectedIndex];
    }
    var Rnd16='';
    dtPWD=nl.ConvStringToDelimiteredString(OTPCH.PWD.value);
    if (OTPCH.NewAES.checked)
    {
        Rnd16+=Math.floor(Math.random()*256);
        for (var j=1;j<16;j++)
            Rnd16+='.'+Math.floor(Math.random()*256);
        nl.SetUserKEY(dtPWD,Rnd16);
        if (nl.ErrNo!=0)
        {
            alert('جدید کلید تعیین در خطا: '+nl.ErrDescrFA);
            return;
        }
    }
    cnt=25+OTPCH.Credit.selectedIndex*25;
    pRatio=200/cnt;
    Rnd16='';
    Rnd16+=Math.floor(Math.random()*256);
    for (var j=1;j<16;j++)
        Rnd16+='.'+Math.floor(Math.random()*256);
        pln=Rnd16;
        xml.open("GET","http://<? echo $_SERVER['SERVER_NAME']?>:8080/ds-php/OTP-PHP/SubmitOtp.php?tTime="+Date()+"&Serial="+
Serial.value+"&Action=DeleteAll",false);
        xml.send();
    if (xml.status!=200)
    {
        alert('قبلي هاي رمزينه حذف در خطا:'+xml.statusText);
        return;
    }

    tblLevel.width=2;
    i=1;
    GenerateNew();
}
```

```
function GenerateNew()
{

    ciph = nl.GetEncryption('0.0.0.0',dtPWD,pln,1);
    if (nl.ErrNo!=0)
    {
        alert('رمز تولید در خطا: '+nl.ErrDescrFA);
        return;
    }
    xml.open("GET","http://<? echo $_SERVER['SERVER_NAME']?>
    :8080/ds-php/OTP-PHP/SubmitOtp.php?Serial="+
    OTPCH.Serial.value+"&OtpQ="+pln+"&OtpA="+ciph,false);
    xml.send();
    if (xml.status!=200)
    {
        alert('ها رمزینه ثبت در خطا:'+xml.statusText);
        return;
    }
    pln = ciph;
    tblLevel.width=Math.floor(i*pRatio);
    if (i++<cnt) setTimeout('GenerateNew();',1);
    else
    {
        alert('رسید پایان به اعتباردهی');
        OTPCH.doBTN.disabled =false;
        doWork=false;
    }
}
```

:OTP/index.php

ورود به روش OTP

در صورتی که شناسه شناسایی شده باشد رمز عبور را وارد کرده و وارد سیستم می‌شویم. اگر ورود به سیستم با موفقیت انجام پذیرد، فرم زیر نمایش داده خواهد شد.

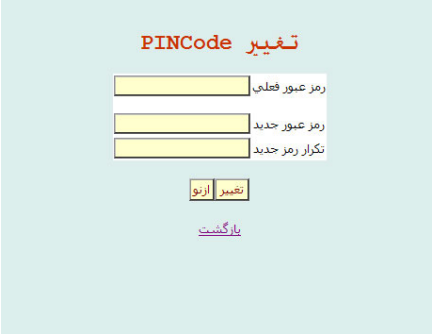
در این فرم تعداد اعتبار دهی ورود نمایش داده می‌شود در صورتی که تعداد آن کم باشد می‌توان با رفتن به فرم شارژ مقدار اعتبار دهی را افزایش داد. در صورتی که به صفحه اصلی برگردیم، در بالای صفحه Index نام کاربری و شماره سریال نمایش داده می‌شود.

با زدن دکمه ورود به سیستم این دستورات انجام میگیرد. ابتدا تابع Submitter() فراخوانی می‌شود این تابع چک می‌کند که آیا شماره سریال و تعداد رمزینه ها برای ورود معتبر است یا نه. در صورتی که مشخصات ورودی صحیح باشد جدول tbotp به روز رسانی می‌شود.

```
if ($Level==2 )
{
        $OTPans=$_POST["OTPreply"];
        $Serial=$_POST["Serial"];
        if ($Serial==="")
        {
                $Serial=$_POST["devs"];
        }
        $cn=odbc_connect("dssample","" ,"");
        $sql="SELECT otp.id, Per.id as PID,per.TokenPublicKey,per.FName, per.LName FROM tbPersons
        per INNER JOIN tbOTP otp ON per.ID = otp.PersonID WHERE (per.TokenSerial = '".$Serial."')
        AND (otp.NewPass = '".$OTPans."') and (otp.used=0)";
        $rs=odbc_exec($cn,$sql);
        if (!odbc_fetch_row($rs))
        {
                $Err=3;
                $Level=1;
        }
        else
        {
                $_SESSION["UserSerial"]=$Serial;
                $_SESSION["pub"]=odbc_result($rs,"TokenPublicKey");
                $_SESSION["UserID"]=odbc_result($rs,"PID");
                $_SESSION["UserFullname"]=odbc_result($rs,"FName")." ".odbc_result($rs,"LName");
                $sql="Update tbotp Set Used=1 Where ID=".odbc_result($rs,"id");
                odbc_exec($cn,$sql);
                $SQL="SELECT Count(*) as otpCNT FROM tbPersons per INNER JOIN tbOTP otp
                ON per.ID = otp.PersonID WHERE (per.TokenSerial= '".$Serial."') and (otp.used=0)";
                $rs=odbc_exec($cn,$SQL);
                $otpCNT=odbc_result($rs,"otpCNT");
```

changePIN.php:

تغییر رمز کاربر



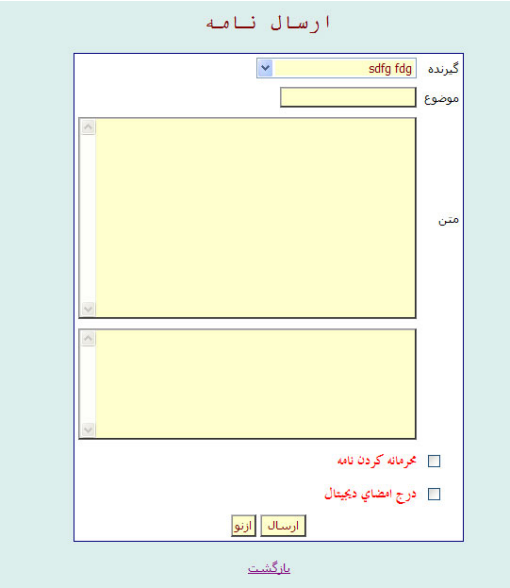
با وارد کردن رمز عبور فعلی و رمز عبور جدید و تکرار آن رمز کاربرتغییر می‌کند.

تابع ChangePIN() برای تغییر دادن رمز عبور

```
function ChangePIN()
{
    if (PINForm.NewPIN1.value!=PINForm.NewPIN2.value)
    {
        PINForm.NewPIN1.value='';
        PINForm.NewPIN2.value='';
        alert("نيستند برابر آن تكرار و جديد عبور رمز.");
    }
    else if (PINForm.NewPIN1.value==='')
    {
        if (confirm('نماييد؟ تنظيم خالي عبور رمز خواهيد مي واقعا آيا .دارد امنيتي مشكل خالي رمزعبور'))
        {
            nl.SetUserPWD(nl.ConvStringToDelimiteredString(PINForm.CurrPIN.value),
            nl.ConvStringToDelimiteredString(PINForm.NewPIN1.value));
            if (nl.ErrNo) alert('جديد رمزعبور ثبت در خطا:\n'+nl.ErrDescr);
            else alert('يافت تغيير رمزعبور');
        }
    }
    else
    {

        nl.SetUserPWD(nl.ConvStringToDelimiteredString(PINForm.CurrPIN.value),
        nl.ConvStringToDelimiteredString(PINForm.NewPIN1.value));
        if (nl.ErrNo) alert('جديد رمزعبور ثبت در خطا:\n'+nl.ErrDescr);
        else alert('يافت تغيير رمزعبور');
    }
}
```

Copmpose.php:

ارسال نامه



در این فرم کاربر می‌تواند نامه محرمانه یا امضادارا ارسال کند.

در صورتی که کاربر تیک درج امضای دیجیتال را بزند ، امضا در قسمت پایین نمایش داده می‌شود.

انتخاب گیرنده نامه با این کد صورت می‌گیرد.

```
odbc_connect("dssample","","");
$sql="Select * from tbPersons ";// where TokenSerial<>'".$_SESSION[UserSerial]."'";
$rs=odbc_exec($cn,$sql);

while (odbc_fetch_row($rs))
{
?>
    <option value="<?   ; echo odbc_result($rs,"id")?>">
    <? echo  odbc_result($rs,"Fname")." ".odbc_result($rs,"Lname")?>
    </option>
<?
}
```

تابعی که رمز نگاری متقارن را برای محرمانه کردن نامه انجام می‌دهد. از الگوریتم AES برای رمزگزاری استفاده می‌کنیم.

```
function N_AESEncrypt()
{
    var RndKey=new Array(32);
    for (var i=0;i<32;i++)
        RndKey[i]=Math.floor(Math.random()*256);
    DataForm.Key.value=byteArrayToHex(RndKey);

    AES_Init();
    AES_ExpandKey(RndKey);
    var CI='';
    var PL=DataForm.Text.value;
    var block = new Array(16);
    for (var i=0;i<=Math.floor((PL.length-1)/8);i++)
    {
        for(var j = 0; j < 8; j++)
        {
            block[j*2] = (i*8+j>PL.length?0:PL.charCodeAt(i*8+j)) % 256;
            block[j*2+1] = Math.floor((i*8+j>PL.length?0:PL.charCodeAt(i*8+j)) / 256);
        }
        AES_Encrypt(block, RndKey);
        CI+=byteArrayToHex(block);
    }
    DataForm.Text.value=CI;
```

اگر علاوه بر محرمانه بودن امضادارهم باشد.

```
    if (DataForm.chkSIGN.checked)
    {
        var CI='';
        var PL=DataForm.Sign.value;
        var block = new Array(16);
        for (var i=0;i<=Math.floor((PL.length-1)/16);i++)
        {
            for(var j = 0; j < 16; j++)
                block[j] = (i*16+j>PL.length?0:PL.charCodeAt(i*16+j));
            AES_Encrypt(block, RndKey);
            CI+=byteArrayToHex(block);
        }
        DataForm.Sign.value=CI;
    }
    AES_Done();
}
```

در صورتی که تنها امضا دیجیتال داشته باشیم در سمت سروراز توابع OpenSSL استفاده می‌کنیم و قبل از ثبت در بانک درستی امضا را چک می‌کنیم.

```
<?

    $verify=true;
    if(($_POST["chkSIGN"])and (!$_POST["chkSECRET"]))
    {
            $sig=pack("H*",$_POST["Sign"]);
            $pubkeyid = openssl_get_publickey($_SESSION["pub"]);
            $verify=openssl_verify($_POST["data"],$sig,$pubkeyid,OPENSSL_ALGO_MD5 );


    }

            $cn=odbc_connect("dssample","","");
            if ($_POST["Level"]=="next")
            {
                    if ($verify)
                    {
                            $sql="INSERT INTO tbMails( SenderID, ReceiverID, Subject, Body, Sign, AsKey, Secured, Signed)
                            VALUES(".$_SESSION[UserID].",".$_POST[rec].",'".$_POST[Subject]."','".$_POST[Text]."',
                            '".$_POST[Sign]."','".$_POST[Key]."',0".$_POST[chkSECRET].",0".$_POST[chkSIGN].")";
                            $rs=odbc_exec($cn,$sql);
                    }
                    else
                    {
                        $err="cannot verify";
                    }
            }
?>
```

:Inbox.php

دریافت نامه

نامه های دریافتی

فرستنده	موضوع	تاریخ	امضا دار	محرمانه
somayeh khoshnood	digital signature	2008-06-10 00:00:00	بلی	خیر
somayeh khoshnood	welcome	2008-06-10 00:00:00	خیر	خیر

بازگشت

نامه های دریافتی در این فرم نمایش داده می‌شوند در صورتی که نامه ای خوانده شود موضوع آن از حالت Bold خارج می‌شود.

```
<?
$sql="SELECT tbMails.ID, tbMails.Subject,tbMails.Secured, tbMails.Signed, tbMails.CreateDate, tbMails.Readed,
    tbPersons.FName, tbPersons.LName FROM tbMails INNER JOIN tbPersons ON tbMails.SenderID = tbPersons.ID WHERE
    tbMails.ReceiverID = ".$_SESSION[UserID]." order by CreateDate desc";
$cn=odbc_connect("dssample","","");
$rs=odbc_exec($cn,$sql);

while (odbc_fetch_row($rs))
{
?>
```

توضیح نمونه کد به زبان ASP

صفحات مربوطه عبارتند از:

Index.asp :

صفحه ورودی که شامل لینکهای زیر است :

در قسمت مدیریت :

- ثبت نام
- اعتباردهی برای ورود (شارژ OTP)

در قسمت عمومی:

- ورود (به روش OTP)

در قسمت کاربران :

- تغییر PINCode (رمزعبور شناسه)
- ارسال نامه
- صندوق دریافت

اولین بار که وارد می‌شویم کاربر مهمان در نظر گرفته می‌شود. که در صورت ثبت نام کردن نام کاربر با شماره سریال در بالای صفحه ثبت می‌شود.

کد مربوطه :

```
<%if session("UserSerial")="" then%>مهمان<%
    else
        response.write(Session("UserFullname")&"("&Session("UserSerial")&")")
    end if
%>
```

:Register.asp

این فرم برای ثبت نام کاربر شناسه است. در صورتی که چند شناسه به پورت متصل باشد شماره سریال مورد نظر را انتخاب می‌کنیم و با زدن کلید خواندن ، کلید عمومی داخل شناسه خوانده می‌شود در صورتی که شناسه قبلا ثبت شده باشد، مشخصات به روزرسانی می‌شود در غیر این صورت مشخصات وارد شده به بانک اضافه می‌گردد.

Check_Lock(): تابع javascript ای که تعداد شناسه های متصل به سیستم را چک می‌کند.

LoadPublic(): این تابع کلید عمومی داخل شناسه را می‌دهد که با زدن کلید خواندن فراخوانی می‌شود.

```
function LoadPublic()
{
    if (nl.GetDeviceCount()>1)
    {
        nl.SelectDevice(devSerials[RegForm.devs.selectedIndex]);
    }
    RegForm.PublicKey.value=nl.GetPublicKey(nl.GetSerial());
    if (nl.ErrNo) alert('Error in getting public key:'+nl.ErrDescr);
}
```

دستورات مربوط به ثبت مشخصات در بانک اگر شماره سریال در بانک وجود نداشته باشد، مشخصات به بانک افزوده می‌گردد اما اگر شماره سریال در بانک باشد مشخصات تنها به روز رسانی می‌شود.

```
<%
if request("Level")="next" then
    Serial=request("devSerial")
    if serial="" then Serial=request("devs")

            set cn=server.createobject("ADODB.connection")
            cn.open SQLcnl
            set rs=cn.execute("Select * from tbPersons where TokenSerial='"&Serial&"'")
            if rs.eof then
                    cn.execute("INSERT INTO tbPersons( FName, LName, TokenSerial, TokenPublicKey)
                    "&_" VALUES('"&request("FName")&"',
                    '"&request("LName")&"','"&Serial&"','"&request("PublicKey")+"')")
                    errs="شد انجام موفقیت با نام ثبت"
            else
                    cn.execute("Update tbPersons set FName='"&request("FName")&"', LName='"&request("LName")&"','"&_
                    " TokenPublicKey='"&request("PublicKey")+"' Where TokenSerial='"&Serial&"'")
                    errs="شد بروزرسانی مشخصات .بود شده ثبت قبلا سریال شماره"
            end if
    set rs=nothing
    set cn=nothing
    end if
%>
```

:OTP/Charge.asp

شارژ اعتبار جهت ورود به روش OTP

در صورتی که شناسه موجود باشد این فرم نمایش داده می‌شود و می‌توان رمز عبور کاربرنهایی (پین کد) را وارد کرد. تعداد اعتبار ورود به معنای حداکثر تعداد ورود کاربر پس از مرحله شارژ است.

ورود به روش OTP با الگوریتم AES انجام پذیر است که در این جا می‌توانیم کلید اصلی آن را ایجاد کنیم.

با کلیک روی دکمه شارژ عمل اعتبار دهی انجام می‌گیرد.

با زدن کلید شارژ تابع GoGenerate() فرا خوانی می‌شود اگر تعیین کلید جدید برای AES تیک خورده باشد چک می‌کند که آیا این امر امکان پذیر است یا نه. سپس تابع GenerateNew() از داخل این تابع فراخوانی می‌شود این تابع عمل اعتبار دهی را انجام می‌دهد.

```
function GoGenerate()
{
    doWork = true;
    OTPCH.doBTN.disabled=true;
    if (nl.GetDeviceCount()>1)
    {
        nl.SelectDevice(devSerials[OTPCH.devs.selectedIndex]);
        OTPCH.Serial.value =devSerials[OTPCH.devs.selectedIndex];
    }
    var Rnd16='';
    dtPWD=nl.ConvStringToDelimiteredString(OTPCH.PWD.value);
    if (OTPCH.NewAES.checked)
    {
        Rnd16+=Math.floor(Math.random()*256);
        for (var j=1;j<16;j++)
            Rnd16+='.'+Math.floor(Math.random()*256);
        nl.SetUserKEY(dtPWD,Rnd16);
        if (nl.ErrNo!=0)
        {
            alert('خطا در تعیین کلید جدید: '+nl.ErrDescrFA);
            return;
        }
    }
    cnt=25+OTPCH.Credit.selectedIndex*25;
    pRatio=200/cnt;
    Rnd16='';
    Rnd16+=Math.floor(Math.random()*256);
    for (var j=1;j<16;j++)
        Rnd16+='.'+Math.floor(Math.random()*256);
        pln=Rnd16;
        xml.open("GET","http://<%=Request.ServerVariables("server_name")%>/ds/otp/SubmitOtp.asp?tTime="+Date()+"&Serial="+OTP
Serial.value+"&Action=DeleteAll",false);
        xml.send();
    if (xml.status!=200)
    {
        alert('خطا در حذف رمزینه های قبلی:'+xml.statusText);
        return;
    }
```

:OTP/index.asp

ورود به روش OTP

در صورتی که شناسه شناسایی شده باشد رمز عبور را وارد کرده و وارد سیستم می‌شویم. اگر ورود به سیستم با موفقیت انجام پذیرد، فرم زیر نمایش داده خواهد شد.

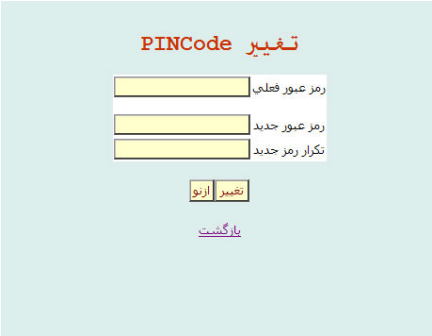
در این فرم تعداد اعتبار دهی ورود نمایش داده می‌شود در صورتی که تعداد آن کم باشد می‌توان با رفتن به فرم شارژ مقدار اعتبار دهی را افزایش داد. در صورتی که به صفحه اصلی برگردیم، در بالای صفحه Index نام کاربری و شماره سریال نمایش داده می‌شود.

با زدن دکمه ورود به سیستم این دستورات انجام می‌گیرد. ابتدا تابع Submitter() فراخوانی می‌شود این تابع چک می‌کند که آیا شماره سریال و تعداد رمزینه ها برای ورود معتبر است یا نه. در صورتی که مشخصات ورودی صحیح باشد جدول tbOtp به روز رسانی می‌شود.

```
<%
Err =0
Level=request("Level")
if Level=2 then

    OTPans=request("OTPreply")
    Serial= request("Serial")

    set Cn=Server.createobject("ADODB.connection")
    cn.open SQLcnl
    SQL="SELECT otp.id, Per.id as PID, per.FName, per.LName "& _
        "FROM tbPersons per INNER JOIN tbOTP otp ON per.ID = otp.PersonID"& _
        " WHERE (per.TokenSerial = '"&Serial&"') AND (otp.NewPass = '"&OTPans&"') and (otp.used=0)"
'   response.write SQL
    set Rs=cn.execute(SQL)
    if rs.eof then
        set Rs=nothing
        set Cn=nothing
        Err=3
        Level=1
    else
        session("UserSerial")=Serial
        session("UserID")=rs("PID")
        Session("UserFullname")=rs("FName")&" "&rs("Lname")
        cn.execute("Update tbotp Set Used=1 Where ID="&rs("ID"))
        SQL="SELECT Count(*) as otpCNT "& _
            "FROM tbPersons per INNER JOIN tbOTP otp ON per.ID = otp.PersonID"& _
            " WHERE (per.TokenSerial= '"&Serial&"') and (otp.used=0)"
        set Rs=cn.execute(SQL)

        otpCNT=rs("otpCNT")
        set Rs=nothing
        set Cn=nothing

%>
```

:changePIN.asp

تغییر رمز کاربر



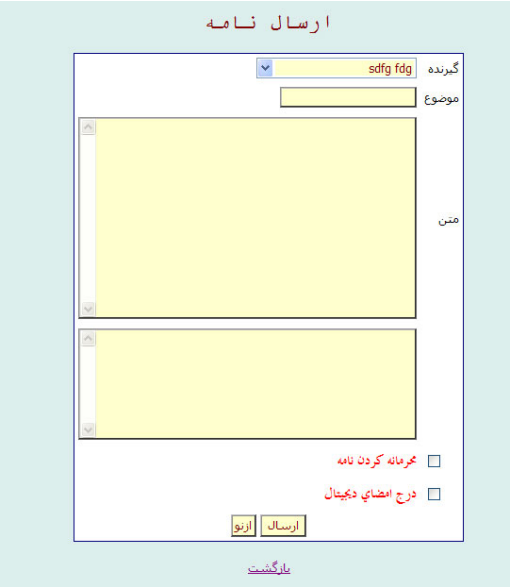
با وارد کردن رمز عبور فعلی و رمز عبور جدید و تکرار آن رمز کاربرتغییر می‌کند.

تابع ()ChangePIN برای تغییر دادن رمز عبور

```
function ChangePIN()
{
    if (PINForm.NewPIN1.value!=PINForm.NewPIN2.value)
    {
        PINForm.NewPIN1.value='';
        PINForm.NewPIN2.value='';
        alert("نیستند برابر آن تکرار و جدید عبور رمز.");
    }
    else if (PINForm.NewPIN1.value==='')
    {
        if (confirm('نمایید؟ تنظیم خالي عبور رمز خواهید مي واقعا آیا .دارد امنیتي مشکل خالي رمزعبور'))
        {
            nl.SetUserPWD(nl.ConvStringToDelimiteredString(PINForm.CurrPIN.value),
            nl.ConvStringToDelimiteredString(PINForm.NewPIN1.value));
            if (nl.ErrNo) alert('جدید رمزعبور ثبت در خطا:\n'+nl.ErrDescr);
            else alert('یافت تغییر رمزعبور');
        }
    }
    else
    {

        nl.SetUserPWD(nl.ConvStringToDelimiteredString(PINForm.CurrPIN.value),
        nl.ConvStringToDelimiteredString(PINForm.NewPIN1.value));
        if (nl.ErrNo) alert('جدید رمزعبور ثبت در خطا:\n'+nl.ErrDescr);
        else alert('یافت تغییر رمزعبور');
    }
}
```

Copmpose.asp:

ارسال نامه



در این فرم کاربر می‌تواند نامه محرمانه یا امضادارا ارسال کند.

در صورتی که کاربر تیک درج امضای دیجیتال را بزند ، امضا در قسمت پایین نمایش داده می‌شود.

انتخاب گیرنده نامه با این کد صورت می‌گیرد.

```
<%
SQL="Select * from tbPersons where TokenSerial<>'"&session("UserSerial")&"'"
set rs=cn.execute(SQL)
while not rs.eof
%>
    <option value="<%=rs("ID")%>"><%=rs("Fname")&" "&rs("Lname")%></option>
<%
    rs.movenext
wend
%>
```

تابعی که رمز نگاری متقارن را برای محرمانه کردن نامه انجام می‌دهد. از الگوریتم AES برای رمز گزاری استفاده می‌کنیم.

```
function N_AESEncrypt()
{
    var RndKey=new Array(32);
    for (var i=0;i<32;i++)
        RndKey[i]=Math.floor(Math.random()*256);
    DataForm.Key.value=byteArrayToHex(RndKey);

    AES_Init();
    AES_ExpandKey(RndKey);
    var CI='';
    var PL=DataForm.Text.value;
    var block = new Array(16);
    for (var i=0;i<=Math.floor((PL.length-1)/8);i++)
    {
        for(var j = 0; j < 8; j++)
        {
            block[j*2] = (i*8+j>PL.length?0:PL.charCodeAt(i*8+j)) % 256;
            block[j*2+1] = Math.floor((i*8+j>PL.length?0:PL.charCodeAt(i*8+j)) / 256);
        }
        AES_Encrypt(block, RndKey);
        CI+=byteArrayToHex(block);
    }
    DataForm.Text.value=CI;
```

اگر علاوه بر محرمانه بودن امضادارهم باشد.

```
    if (DataForm.chkSIGN.checked)
    {
        var CI='';
        var PL=DataForm.Sign.value;
        var block = new Array(16);
        for (var i=0;i<=Math.floor((PL.length-1)/16);i++)
        {
            for(var j = 0; j < 16; j++)
                block[j] = (i*16+j>PL.length?0:PL.charCodeAt(i*16+j));
            AES_Encrypt(block, RndKey);
            CI+=byteArrayToHex(block);
        }
        DataForm.Sign.value=CI;
    }
    AES_Done();
}
```

در صورتی که تنهاامضا دیجیتال داشته باشیم در سمت سروراز توابع openssl استفاده می‌کنیم و قبل از ثبت در بانک درستی امضا را چک میکنیم.

```
<%
set cn=server.createobject("ADODB.connection")
cn.open SQLcnl

if request("Level")="next" then

    SQL="INSERT INTO tbMails( SenderID, ReceiverID, Subject, Body, Sign, AsKey, Secured, Signed)"& _
        "VALUES("&Session("UserID")&","&Request("rec")&", '"&request("Subject")&"','"&request("Text")&"','"&_
        "'"&request("Sign")&"','"& _
        request("Key")&"',0"&request("chkSECRET")&",0"&request("chkSIGN")&")"
    set rs=cn.execute(SQL)

end if

%>
```

Inbox.asp:

دریافت نامه

نامه های دریافتی

فرستنده	موضوع	تاریخ	امضا دار	محرمانه
somayeh khoshnood	digital signature	2008-06-10 00:00:00	بلی	خیر
somayeh khoshnood	welcome	2008-06-10 00:00:00	خیر	خیر

بازگشت

نامه های دریافتی در این فرم نمایش داده می‌شوند در صورتی که نامه ای خوانده شود موضوع آن از حالت Bold خارج می‌شود.

در صورتی که نامه تنهاامضا دار باشد در بازگشایی نامه می‌توان امضا آن را با این کدچک کرد. اگر نامه دستکاری نشده باشد در Inbox نامه ارسالی نشان داده می‌شود.

```
<%
set cn=server.createobject("ADODB.connection")
set nl=server.createobject("NovinAfzar.clsLocalDevice")
cn.open SQLcnl
SQL="SELECT tbMails.ID, tbMails.Subject,tbMails.Secured, tbMails.Signed, tbMails.CreateDate,"&_
" tbMails.Readed, tbPersons.FName, tbPersons.LName "& _
"FROM tbMails INNER JOIN tbPersons ON tbMails.SenderID = tbPersons.ID "& _
"WHERE tbMails.ReceiverID = "&session("UserID")&" order by CreateDate desc"
set rs=cn.execute(SQL)


while not rs.eof
%>
```

پیوست ۲: کد خطاها

در جدول زیر تمام خطاهای ممکن در استفاده از کتابخانه رابط شناسه در برنامه نویسی آمده است:

شماره خطا	سمبل خطا	توضیح
۰	nErr_OK	بدون خطا
۱	nErr_DNF	شناسه پیدا نشد
۲	nErr_INP	پارامتر ورودی غیرمعتبر است
۳	nErr_IOE	خطای IO
۴	nErr_WRP	پارامتر ورودی اشتباه است
۵	nErr_CS	کد مورد نظر ست شده است
۶	nErr_CNS	کد مورد نظر ست نشده است
۸	nErr_BSY	شناسه مشغول است
۱۰	nErr_ICV	نگارش شناسه ناسازگار است
۱۱	nErr_ACD	دسترسی غیر مجاز است
۱۲	nErr_IVP	VID یا رمز عبور اشتباه است
۱۳	nErr_DIS	شناسه غیرفعال شده است
۱۴	nErr_BLK	شناسه در مسدود است
۱۶	nErr_OTE	خطای ناشناخته
۱۹	nErr_CAT	شناسه غیرفعال فجیع شده است
۲۰	nErr_NSD	سرویس مورد نظر پشتیبانی نمی شود
۲۱	nErr_HID	خطا در HID (رابط USB)
۳۲	nErr_WCHttpEr	خطای سرویس وب (http)
۵۶	nErr_TimerNS	کاهنده خودکار (زمانسنج ۵٫۵۹ ثانیه ای) فعال نیست

۶۰	nErr_RSANS	رمزنگاری نامتقارن RSA فعال نشده است
۶۱	nErr_INVHT	نوع Hash غیر معتبر است
۶۲	nErr_SNV	شماره سریال نامعتبر است
۶۳	nErr_INVKL	طول کلید رمزنگاری نامعتبر است
۶۴	nErr_INVKT	نوع کلید رمزنگاری نامعتبر است
۶۵	nErr_INSGN	امضای دیجیتال داده معتبر نیست
۶۶	nErr_INSKL	طول کلید برای امضای دیجیتال کافی نیست
۱۳۲	nErr_DNS	هیچ دستگاهی انتخاب نشده است