ارسال پیام دریافتی ازکاربر به اکسترنال CP

نسخه ۱.۵.۰.۱

تاریخ : ۹۶/۱۰/۱۲

برای ارتباط با سامانه‌ی پیام‌رسان فناپ باید یک API روی پروتکل HTTP به صورت زیر وجود داشته باشد.

Verb: POST

Input:

Message	String	Message sent to messaging system from end-user
Muid	String	UniqueCode For Message
FormattedMessage	String	Message that be formatted by aggregator
AccountId	String	User's account ID
RecieveDate	DateTime	Timestamp in which the message is received by the system (UTC)
Operator	string	Currently available as MCI, IMI, MTN, RIGHTEL,
SID	string	the service identifier code provided by aggregator to third-party

در صورتی که متقاضی دریافت شماره کاربر هستندinput، مقادیر زیر نیز ارسال می گردد:

PhoneNumber	String	User's Phone Number

این ورودی‌ها باید طبق استاندارد REST و در قالب یک JSON Object در Request Body قرار داشته باشند.

آدرس API فوق متعاقباً باید برای اعمال در سیستم در اختیار فناپ قرار گیرد.

پس از دریافت پیام از طرف کاربر، سامانه‌ی پیام‌رسان فناپ اطلاعات زیر را در اختیار آن نهاد قرار می‌دهد.

http request header:

appson-messaging-signature	پیام امضا شده سامانه برای احراز هویت و اثبات صحت پیام ارسالی
appson-messaging-message	پیام ارسالی به آن شرکت با ساختار JSON برای اعتبارسنجی با امضای دیجیتال

پیام سریالایز شده توسط فناپ در قسمت appson-messaging-message قرار گرفته که همین پیام را فناپ با کلید خصوصی خود sign کرده است و در appson-messaging-signature قرار داده است ، CP باید این مقدار را با کلید عمومی خود چک کند که آیا این پیام از طرف فناپ ارسال شده یا خیر اگر خروجی true بود یعنی پیام از طرف فناپ میباشد.

http request body:

an object in above format

** شرکت منتظر پاسخ فراخوانی سرویس ExternalCP نمی‌ماند. در صورتی که درخواست ارسال پیام به کاربر (در ازای پیام دریافتی از کاربر) را دارید، باید وب سرویس ارسال پیام به کاربر توسط ExternalCP پیاده‌سازی و از طریق آن ارسال پیام انجام شود. وب سرویس ارسال پیام به کاربر نیازمند یک کلید نامتقارن (Asymmetric Key) برای امضا محتوا دارد و **کلید عمومی آن (در فرمت XML) باید در اختیار فناپ قرار گیرد.**

** برای اینکه آن شرکت بتواند پیام دریافتی از سامانه‌ی پیام‌رسان فناپ را در اختیار سامانه‌ی پیکو (سامانه‌ی پرداخت فناپ) قرار دهد باید مقادیر موجود در header پیام ارسالی را نیز در ساختار ارسالی خود به سیستم پیکو ارسال کند.

****در آخر ادرس API ساخته شده در اختیار شرکت فناپ قرار گیرد.****

پیوست ۱:

نحوه sign کردن فناپ

الگوریتم مورد استفاده برای **asymmetric cryptography**، الگوریتم RSA است که الگوریتم hash آن نیز SHA1 است نمونه کد در زیر آمده است.

```
public static string Sign(string key, string text)
        {
            try
            {
                // Select target CSP
                var cspParams = new CspParameters { ProviderType = 1 };
                // PROV_RSA_FULL
                //cspParams.ProviderName; // CSP name
                var rsaProvider = new RSACryptoServiceProvider(cspParams);

                // Import public key
                rsaProvider.FromXmlString(key);

                // Encrypt plain text
                var plainBytes = Encoding.UTF8.GetBytes(text);

                var encryptedBytes = rsaProvider.SignData(plainBytes, new SHA1CryptoS-
erviceProvider());

                return Convert.ToBase64String(encryptedBytes);

                // Write encrypted text to file
            }
            catch (Exception exception)
            {
                Log.Error(exception.Message, exception);
                return null;
            }
        }
```

پیوست۲:

نمونه کد احراز هویت

```
public static bool Check(string key, string signedText, string text)
        {
            if (string.IsNullOrWhiteSpace(text)) return false;
            try
            {
                // Select target CSP
                var cspParams = new CspParameters { ProviderType = 1 };
                // PROV_RSA_FULL
                //cspParams.ProviderName; // CSP name
                var rsaProvider = new RSACryptoServiceProvider(cspParams);


                // Import private/public key pair
                rsaProvider.FromXmlString(key);

                var encryptedBytes = Convert.FromBase64String(signedText);
                var plainInput = Encoding.UTF8.GetBytes(text);

                // Decrypt text
                     var check = rsaProvider.VerifyData(plainInput, new
                     SHA1CryptoServiceProvider(), encryptedBytes);
                return check;
            }
            catch (Exception exception)
            {
                Log.Error(exception.Message, exception);
                return false;
            }
```