ارسال پیام دریافتی ازکاربر به اکسترنال CP

نسخه ۳,۰,۰,۰

تاریخ ۱۳۹۷/۰۲/۰٤

برای دریافت پیام کاربر باید یک post Api بر روی پروتکل http وجود داشته باشد. این آدرس Api باید در اختیار شرکت فناپ قرار بگیرد.

پس ازدریافت پیام کاربر, سامانه پیام رسان فناپ, اطلاعات زیر را در اختیار شرکت مشتری قرار می دهد.

Http request body:

بدنه پیام دارای object ی به فرمت آرایه ای تک عضوی از پارامترهای زیر است:

Muid	شناسه منحصر به فردی که توسط پیام رسان فناپ به پیام اختصاص داده شده است.
ReceiveTime	زمان دریافت پیام توسط سامانه (Iso 8601) به فرمت : universal time "yyyy-MM-ddTHH:mm:ss.fffZ"
AccountId	شناسه کاربر
ChannelType	درگاهی که پیام کاربر از آن دریافت شده است، شامل مقادیر: Pardis, Imi, Mtn, Rightel است.
Channel	سرشماره ای که پیام کاربر از آن دریافت شده است
MessageType	نوع پیام دریافتی که دارای مقادیر: Content, Subscription, Unsubscription, PremiumContent است.
Content	متن پیام ارسال شده توسط کاربر
Sid	شناسه منحصر به فرد مشتری که توسط شرکت فناپ در اختیار مشتری قرار گرفته و به منظور شناسایی مشتری از آن استفاده میکند.
Signature	پیام امضا شده توسط شرکت فناپ

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

شماره تلفن کاربر	UserPhoneNumber

مثال:


```
[
    {
        “Muid”: “74c925a6211f483fafb29650feb821c7”,
        “Sid”: ” d45987d89490432990f4af64ee2c3cd6”,
        “ReceiveTime”: "2018-04-23T10:22:21.028Z",
        “ChannelType”: “Imi”,
        “Channel”: “983048”,
        “AccountId”: “VL6DUI5T5TKUBJKGOSOB47P7XOUQ”,
        “UserPhoneNumber”: “98990***4656”,
        “MessageType”: “Content”,
        “Content”: “test”,
        Signature:
LSrRIM9Jh8HA9C6WtOZHXiRd4jt24vpALJr4FFvhda4TA2A4MO+xYtm93bxUcI3LANHDd5fMs2ruRUqAadB
xpDWRG+AVOLDR8uQHOyRNszvUYKdoDnnahRx6f3GI0abx6Lw1xUxzSUTr1Dk6PywllkVL2pmbaM6mL5PR
+tBO2Ps=
    }
]
```


پارامتر نوع پیام دریافتی دارای یکی از مقادیر زیر است :

- Content عادی
- Subscription عضویت

- unSubscription لغو عضویت

- PremiumContent پیام پولی

پارامتر Signature:

برای احراز هویت و اثبات صحت پیام می‌باشد. فناپ با توجه به پارامترهای بدنه‌ی پیام ، پیام فرمت شده‌ای را می‌سازد که همان پیام را توسط کلید خصوصی خود امضا می کند و در پارامتر Signature قرار می‌دهد . شرکت خصوصی محتوای این پیام را با کلید عمومی فناپ (کلید عمومی فناپ را همراه داکیومنت دریافت کنید)، رمزگشایی می کند.

پیام فرمت شده بدین ترتیب ایجاد شده است :

(توضیح آنکه، مقادیر داخل براکت با توجه به مقادیر پارامترهای نظیر در بدنه ی پیام به صورت جدا شده با ویرگول، پر می شوند و ترتیب از چپ به راست حتما رعایت شود):

```
[ReceiveTime],[Sid],[ChannelType],[Channel],[Muid],[Content],[MessageType],[AccountId]
```

شرکت فناپ منتظر پاسخ فراخوانی مشتری نمی ماند و در صورتی که مشتری قصد ارسال پیام به کاربر را دارد باید وب سرویس ارسال پیام به مشتری پیاده سازی شود و از طریق آن پیام ها ارسال شود. سرویس ارسال پیام به مشتری نیازمند یک کلید نامتقارن است که برای امضا و احراز هویت مشتری مورد استفاده قرار میگیرد، بنابراین کلید عمومی آن در فرمت XML باید در اختیار شرکت فناپ قرار بگیرد.

**برای اینکه آن شرکت بتواند پیام دریافتی از سامانه‌ی پیام‌رسان فناپ را در اختیار سامانه‌ی پیکو (سامانه‌ی پرداخت فناپ) قرار دهد مقدار پارامتر signature و پیام فرمت شده ای را که از بدنه پیام به منظور احراز هویت ساخته اید به پیکو ارسال کنید.

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)
    {
        using (var rsaProvider = new RSACryptoServiceProvider(CspParams))
        {
            rsaProvider.FromXmlString(key);

            var plainBytes = Encoding.UTF8.GetBytes(text);

                var encryptedBytes = rsaProvider.SignData(plainBytes, new
                                    SHA1CryptoServiceProvider());

            return Convert.ToBase64String(encryptedBytes);
        }
    }
```

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(), encryptedB
ytes);
            return check;
         }
         catch (Exception exception)
         {
            Log.Error(exception.Message, exception);
            return false;
         }

}
```