```
From 813838f6d08034b6a265a70e53b59b941b5d3e6d Mon Sep 17 00:00:00 2001
From: Tokuhiro Matsuno <tokuhirom@gmail.com>
Date: Fri, 26 Jan 2018 04:54:50 +0000
Subject: [PATCH] Validate session id

---
 lib/HTTP/Session2/ServerStore.pm |  3 ++-
 t/01_server_side.t               | 30 ++++++++++++++++++++++++++++++-
 2 files changed, 31 insertions(+), 2 deletions(-)

diff --git a/lib/HTTP/Session2/ServerStore.pm b/lib/HTTP/Session2/ServerStore.pm
index c99a2ba..c0d8c2c 100644
--- a/lib/HTTP/Session2/ServerStore.pm
+++ b/lib/HTTP/Session2/ServerStore.pm
@@ -49,6 +49,8 @@ sub load_session {
     # Load from cookie.
     my $cookies = Cookie::Baker::crush_cookie($self->env->{HTTP_COOKIE});
     if (my $session_id = $cookies->{$self->session_cookie->{name}}) {
+        # validate session_id
+        return if $session_id =~/[\x00-\x20\x7f-\xff]/ || length($session_id) > 40;
         my $data = $self->store->get($session_id);
         if (defined $data) {
             $self->{id}   = $session_id;
@@ -211,4 +213,3 @@ But, I recommend to use C<Cache::Memcached::Fast>.
 =head1 METHODS

 Methods are listed on L<HTTP::Session2::Base>.
-
diff --git a/t/01_server_side.t b/t/01_server_side.t
index d99d1ae..5396955 100644
--- a/t/01_server_side.t
+++ b/t/01_server_side.t
@@ -123,6 +123,35 @@ scenario 'In a login session' => sub {
     };
 };

+scenario 'Invalid session id' => sub {
+    Cache->new->set('invalid char in session id' => { user_id => 5963 });
+
+    my $session;
+    step 'client -> server: request without cookie' => sub {
+        $session = HTTP::Session2::ServerStore->new(
+            env => {
+                HTTP_COOKIE => 'hss_session=invalid%20char%20in%20session%20id',
+            },
+            get_store => sub { Cache->new() },
+            secret => 's3cret',
+        );
+    };
+    step 'server -> store: set more data' => sub {
+        $session->set('foo' => 'bar');
+    };
+    step 'server -> client: response with new session/xsrf cookie' => sub {
+        my $res = empty_res();
+        $session->finalize_psgi_response($res);
+        is $res->[1]->[0], 'Set-Cookie';
+        my ($sess_id) = ($res->[1]->[1] =~ qr{\Ahss_session=([^;]*); path=/; HttpOnly\z});
+        ok $sess_id;
+        isnt $sess_id, 'invalid char in session id';
+        is_deeply $Cache::STORE{$sess_id}, {
+            foo => 'bar',
+        };
+    };
+};
```

```
 +
  scenario 'Logout' => sub {
      Cache->new->set(SsEeSsIiOoNn => { foo => 'bar' });

 @@ -153,4 +182,3 @@ scenario 'Logout' => sub {
  };

  done_testing;
 -
```