BAE SYSTEMS
INSPIRED WORK

Solutum cumulus mediocris

Blackhat Asia 2014

AGENDA

- Introduction
- Why
- What
- How

@WIREGHOUL

- Husband
- Father
- Penetration tester
- Geek
- Blogger – http://www.justanotherhacker.com
- Projects
  - htshells
  - Graudit
  - Doona and more
- Contributor
  - Nikto
  - Dotdotpwn
  - PadBuster and more

INTRODUCTION – PAYMENT GATEWAY

A payment gateway is an e-commerce application service provider service that authorizes payments online. It is the equivalent of a physical point of sale terminal. Payment gateways protect credit card details by encrypting sensitive information, such as credit card numbers, to ensure that information is passed securely between the customer and the merchant and also between merchant and the payment processor.

INTRODUCTION

- Actors
- Definitions
- Payment gateway APIs
- Design vulnerabilities
- Cryptography
- Implementation bugs

CUSTOMER

BAE SYSTEMS
INSPIRED WORK

MERCHANT

PAYMENT GATEWAY

ATTACKER

TESTING PAYMENT

- Use test card numbers

VISA	4111 1111 1111 1111
Mastercard	5555 5555 5555 4444
American Express	378282246310005

API

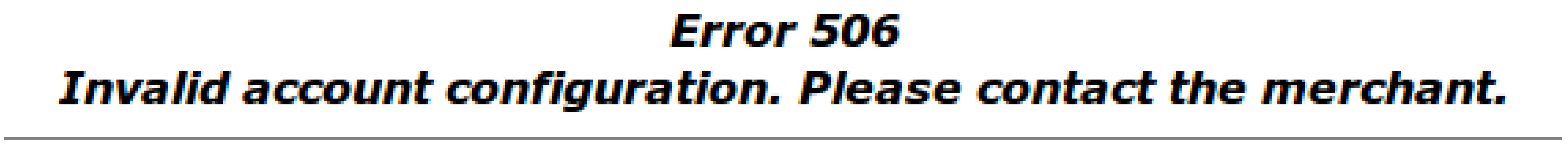
- Primary means of interaction between online payment form and payment gateway
- Typical operations include:
- Charge card
- Query payment status
- Manage recurring payments
- Refund payments

API ACCESS POINTS

- Production
- https://api.paymentgateway.url

- Sandbox
- https://test.paymentgateway.url

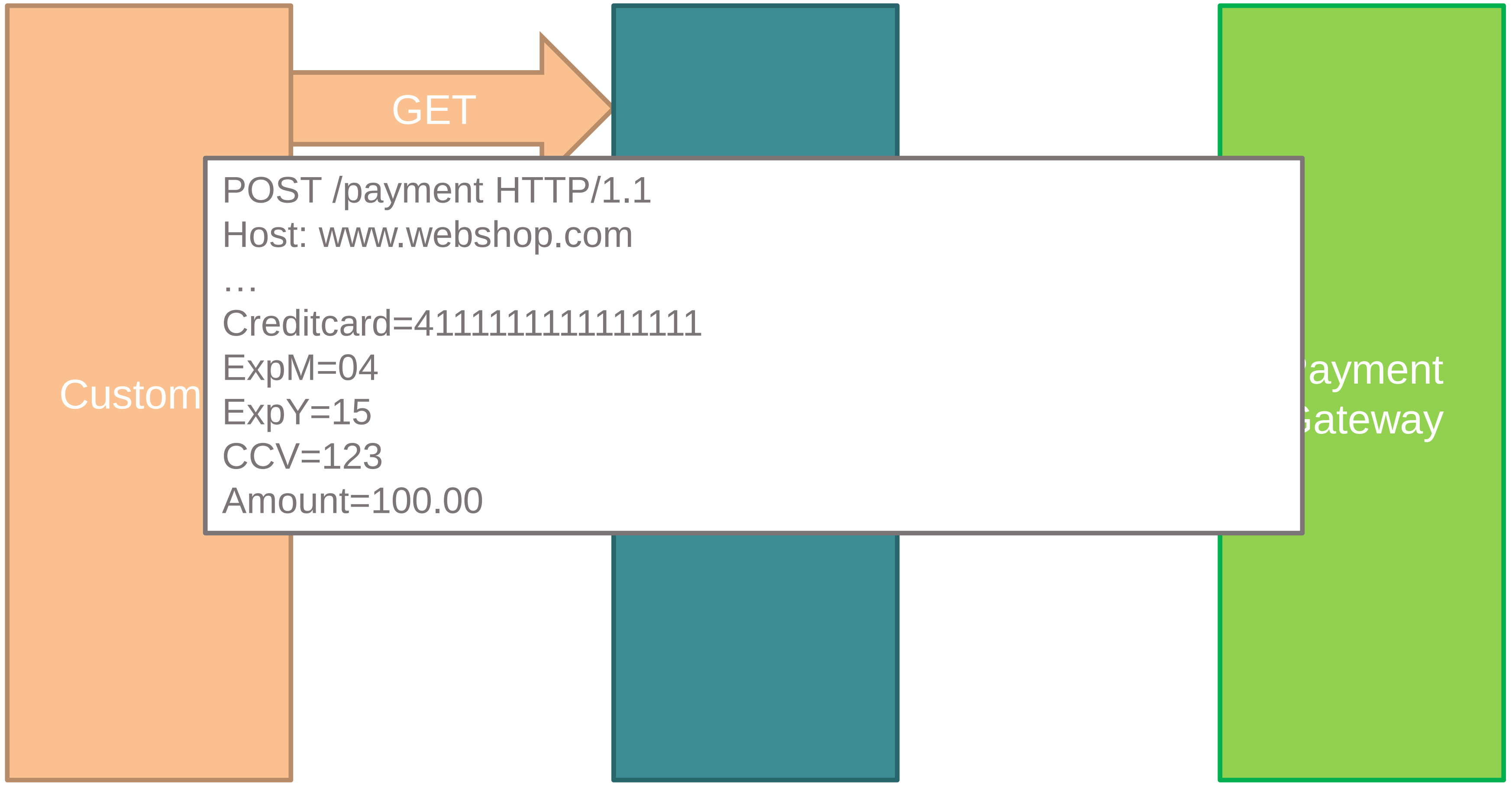
LEVERAGING THE SANDBOX


Error 506

Invalid account configuration. Please contact the merchant.

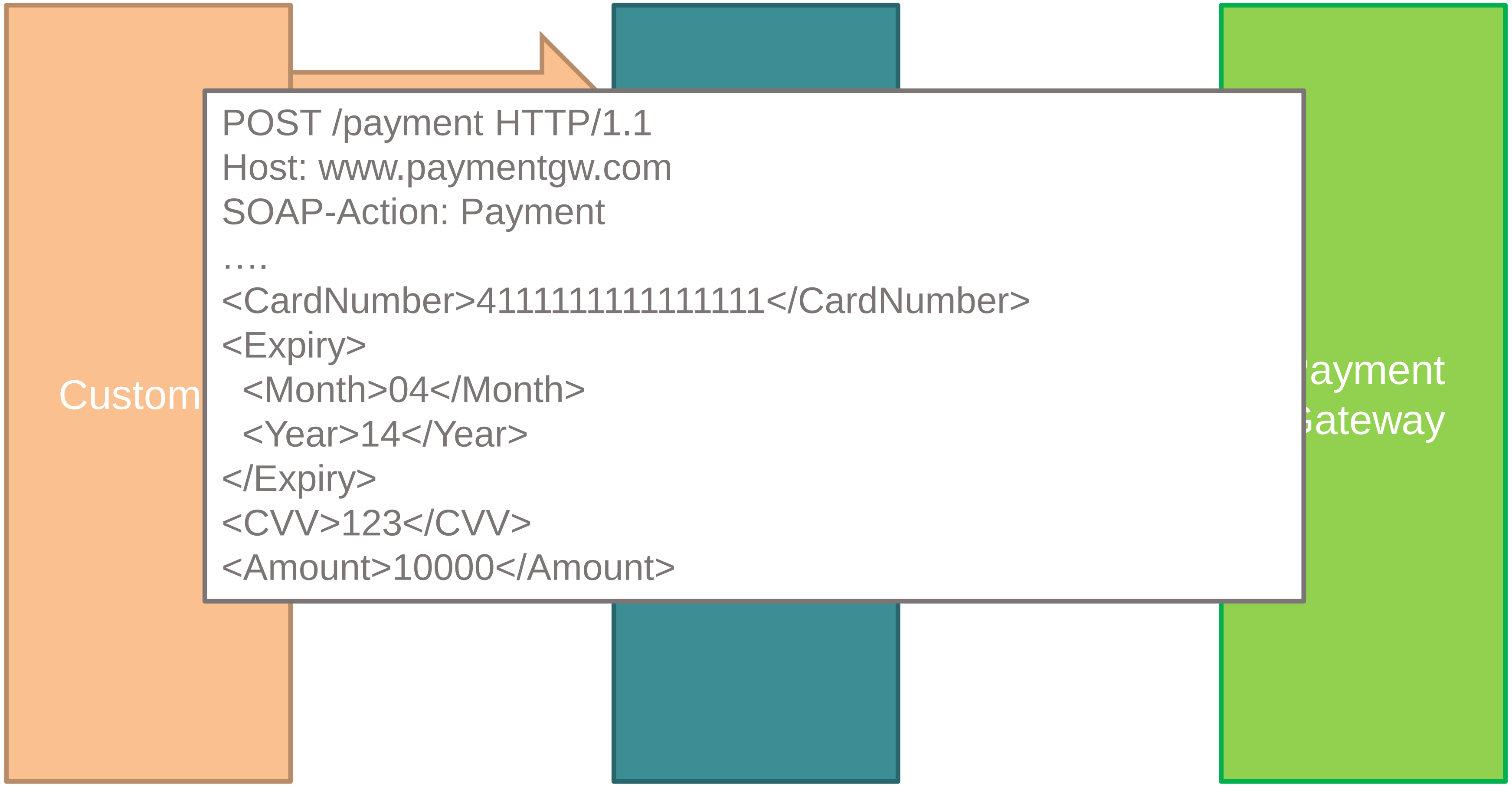
BAE SYSTEMS
INSPIRED WORK

API - DIRECT



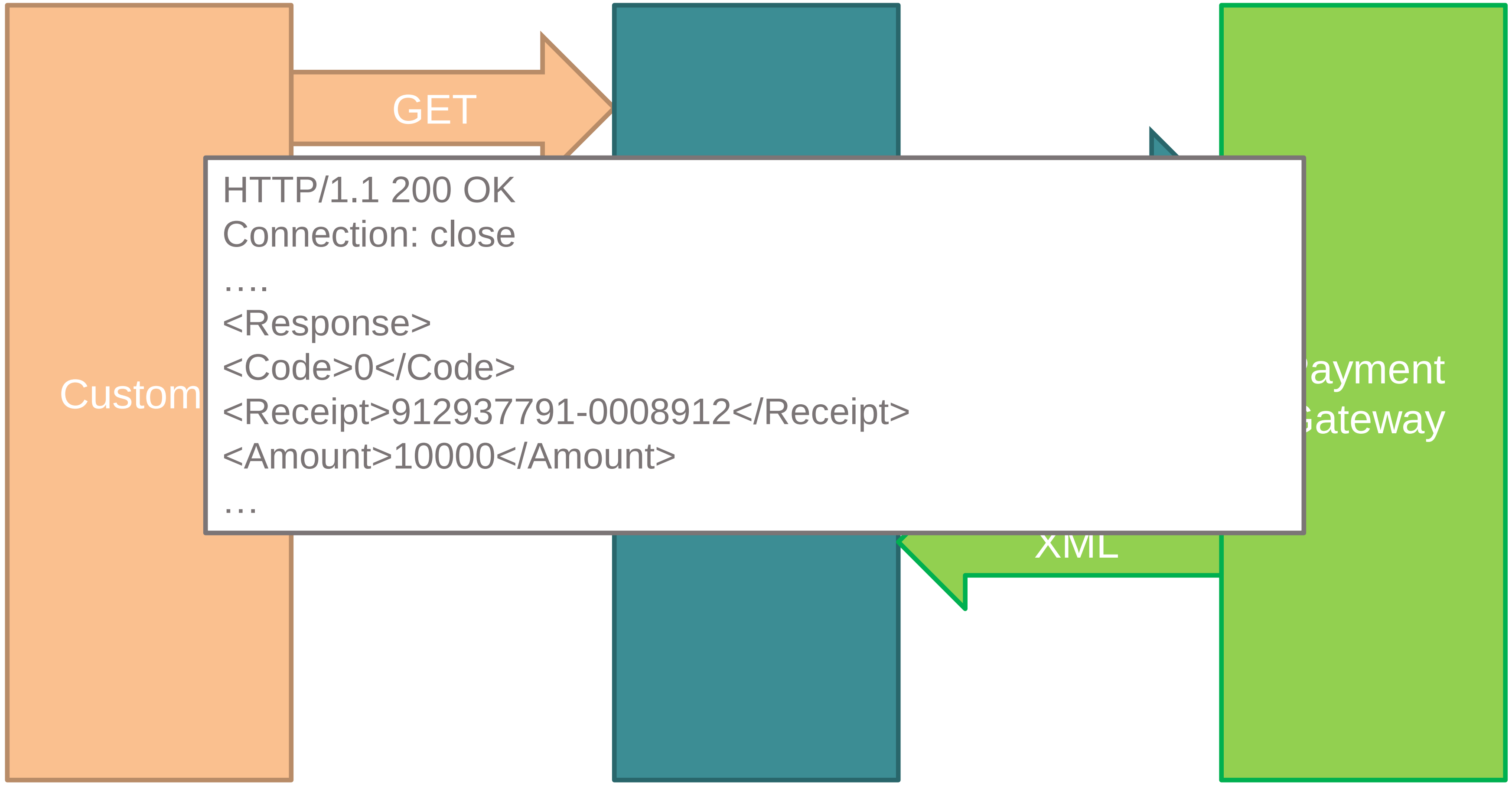
```
POST /payment HTTP/1.1
Host: www.webshop.com
…
Creditcard=4111111111111111
ExpM=04
ExpY=15
CCV=123
Amount=100.00
```

API - DIRECT



```
POST /payment HTTP/1.1
Host: www.paymentgw.com
SOAP-Action: Payment
….
<CardNumber>4111111111111111</CardNumber>
<Expiry>
  <Month>04</Month>
  <Year>14</Year>
</Expiry>
<CVV>123</CVV>
<Amount>10000</Amount>
```

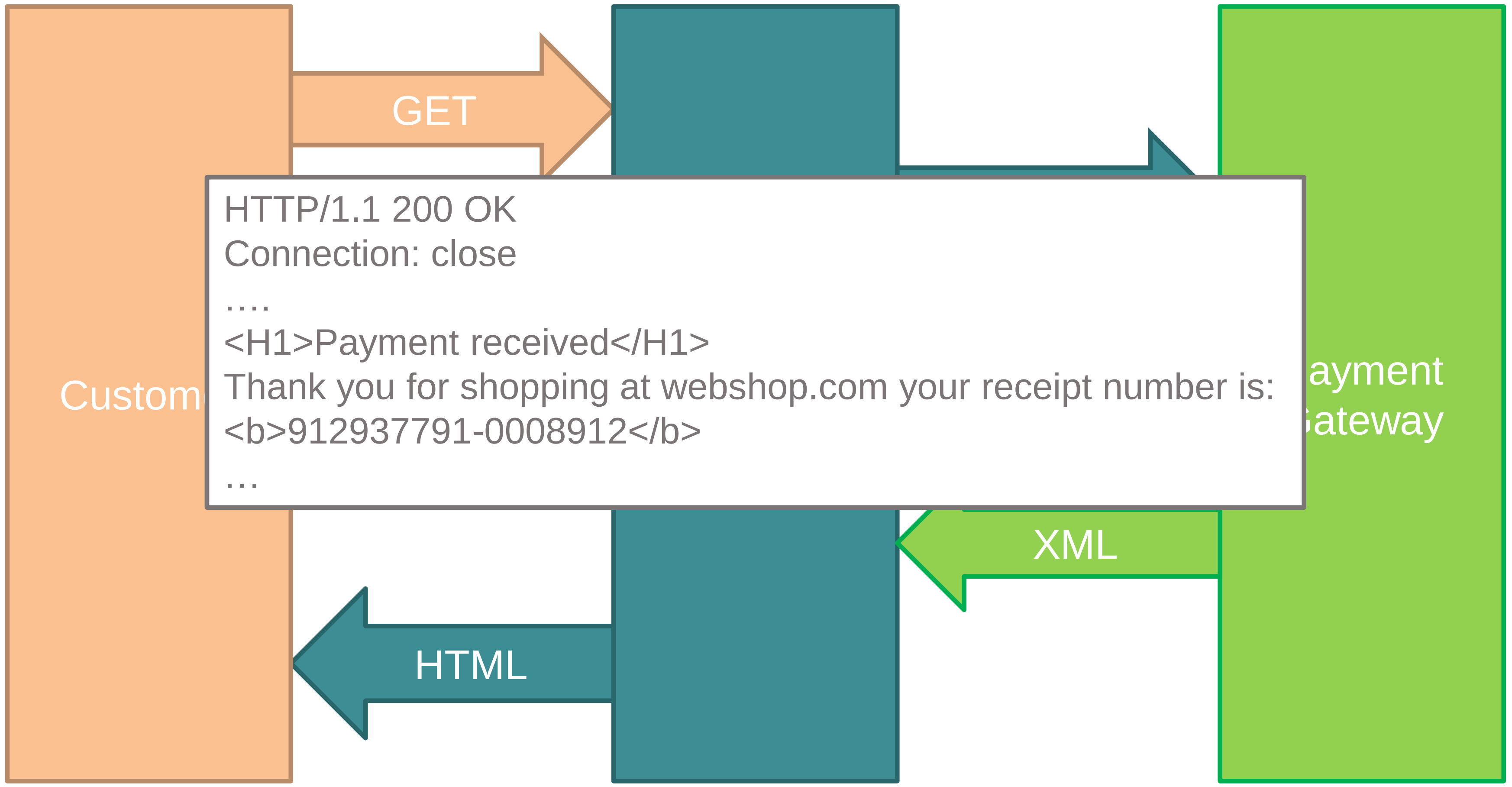
API - DIRECT



```
HTTP/1.1 200 OK
Connection: close
….
<Response>
<Code>0</Code>
<Receipt>912937791-0008912</Receipt>
<Amount>10000</Amount>
…
```

BAE SYSTEMS
INSPIRED WORK

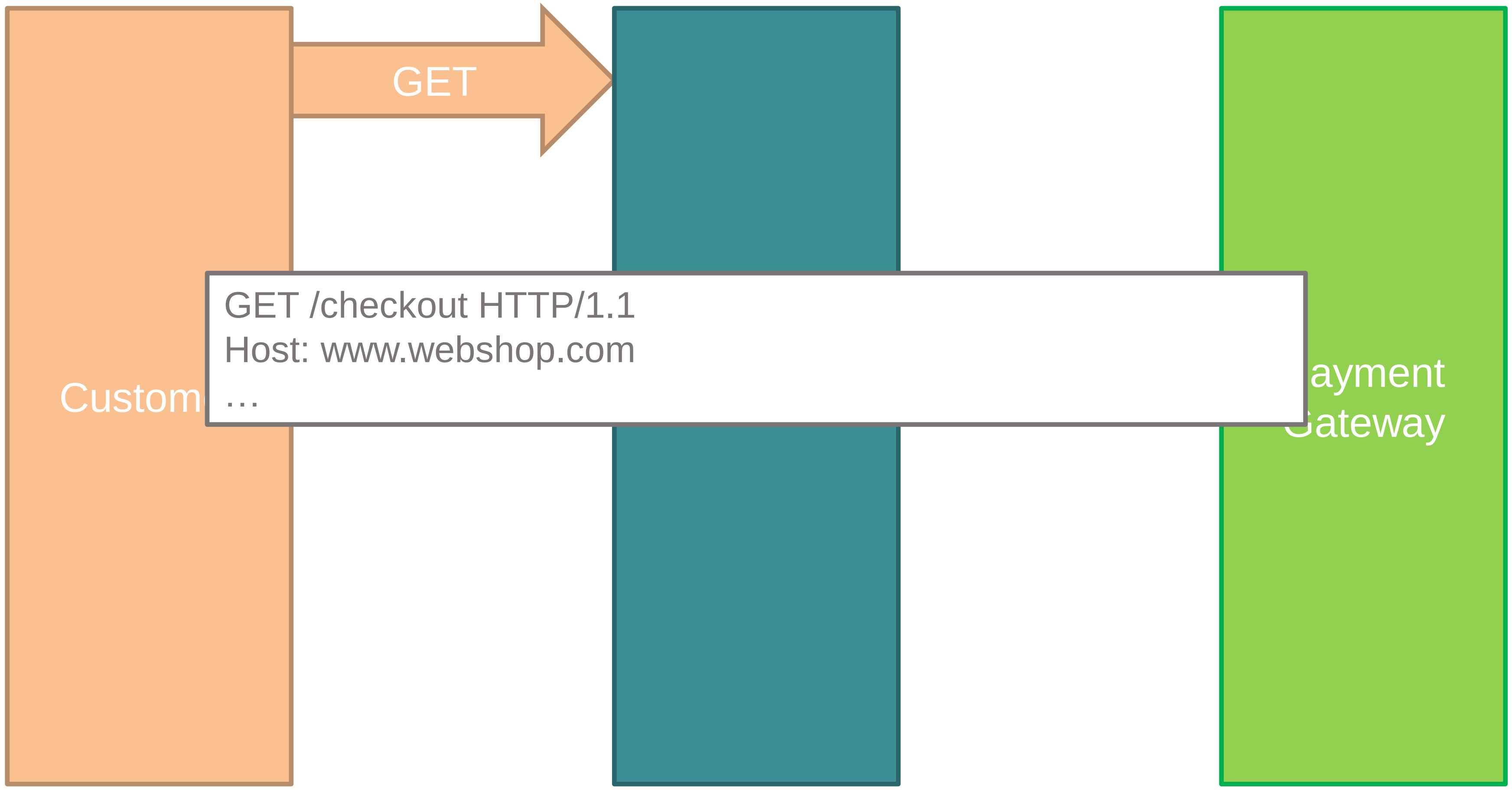
API - DIRECT



```
HTTP/1.1 200 OK
Connection: close
….
<H1>Payment received</H1>
Thank you for shopping at webshop.com your receipt number is:
<b>912937791-0008912</b>
…
```

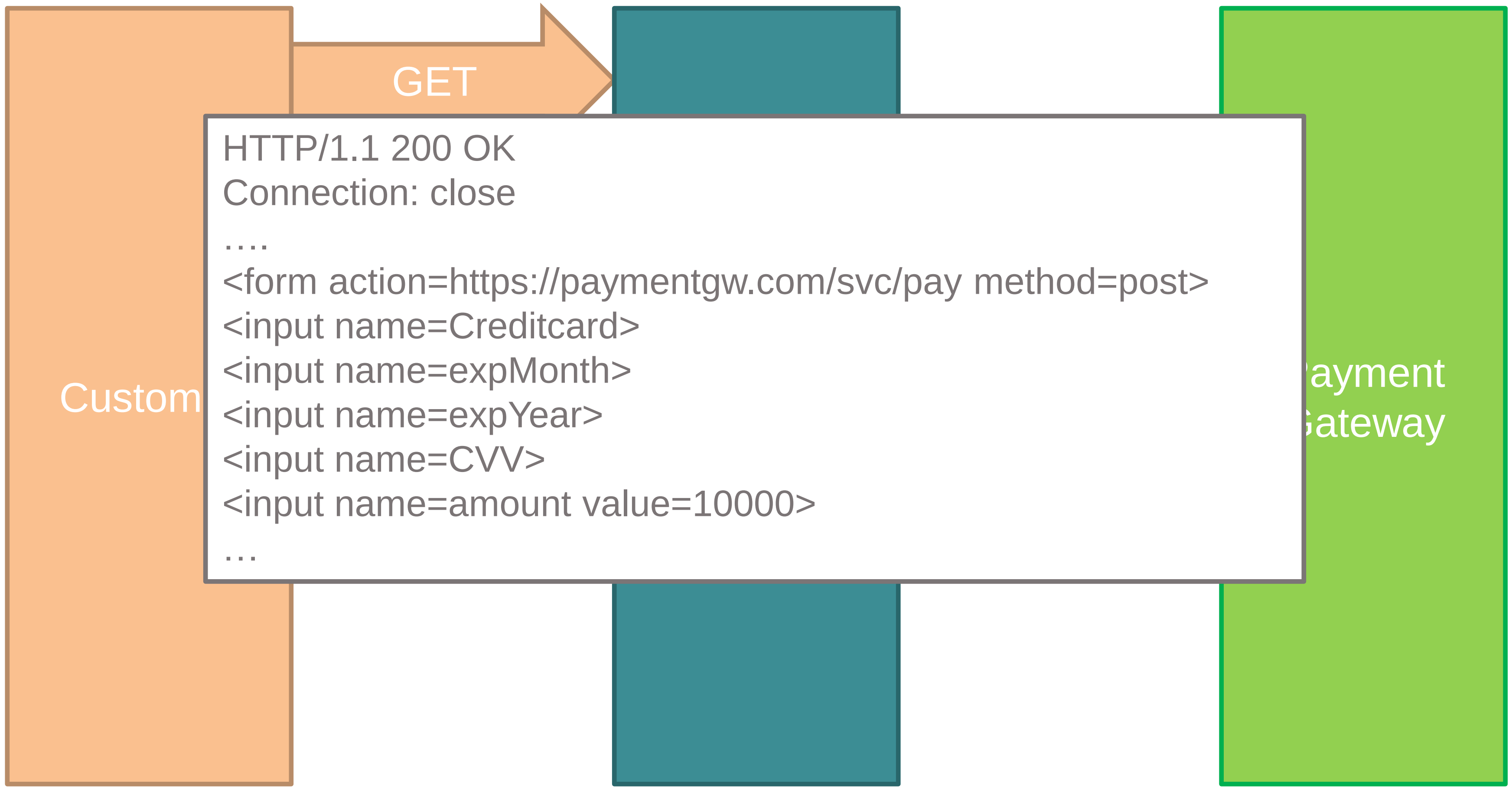
BAE SYSTEMS
INSPIRED WORK

API – HOSTED DIRECT POST



```
GET /checkout HTTP/1.1
Host: www.webshop.com
…
```

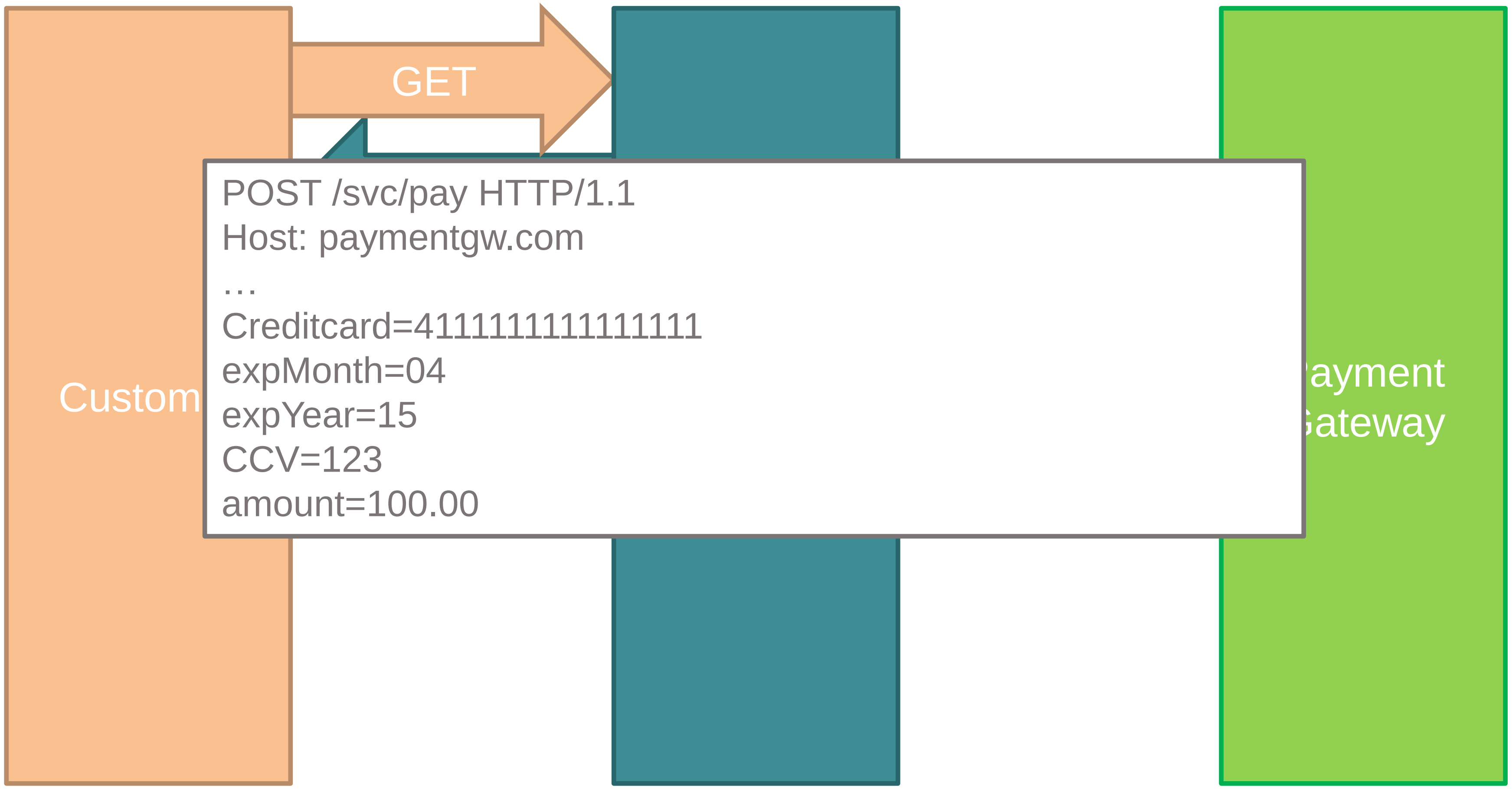
API – HOSTED DIRECT POST



```
HTTP/1.1 200 OK
Connection: close
….
<form action=https://paymentgw.com/svc/pay method=post>
<input name=Creditcard>
<input name=expMonth>
<input name=expYear>
<input name=CVV>
<input name=amount value=10000>
…
```

BAE SYSTEMS
INSPIRED WORK

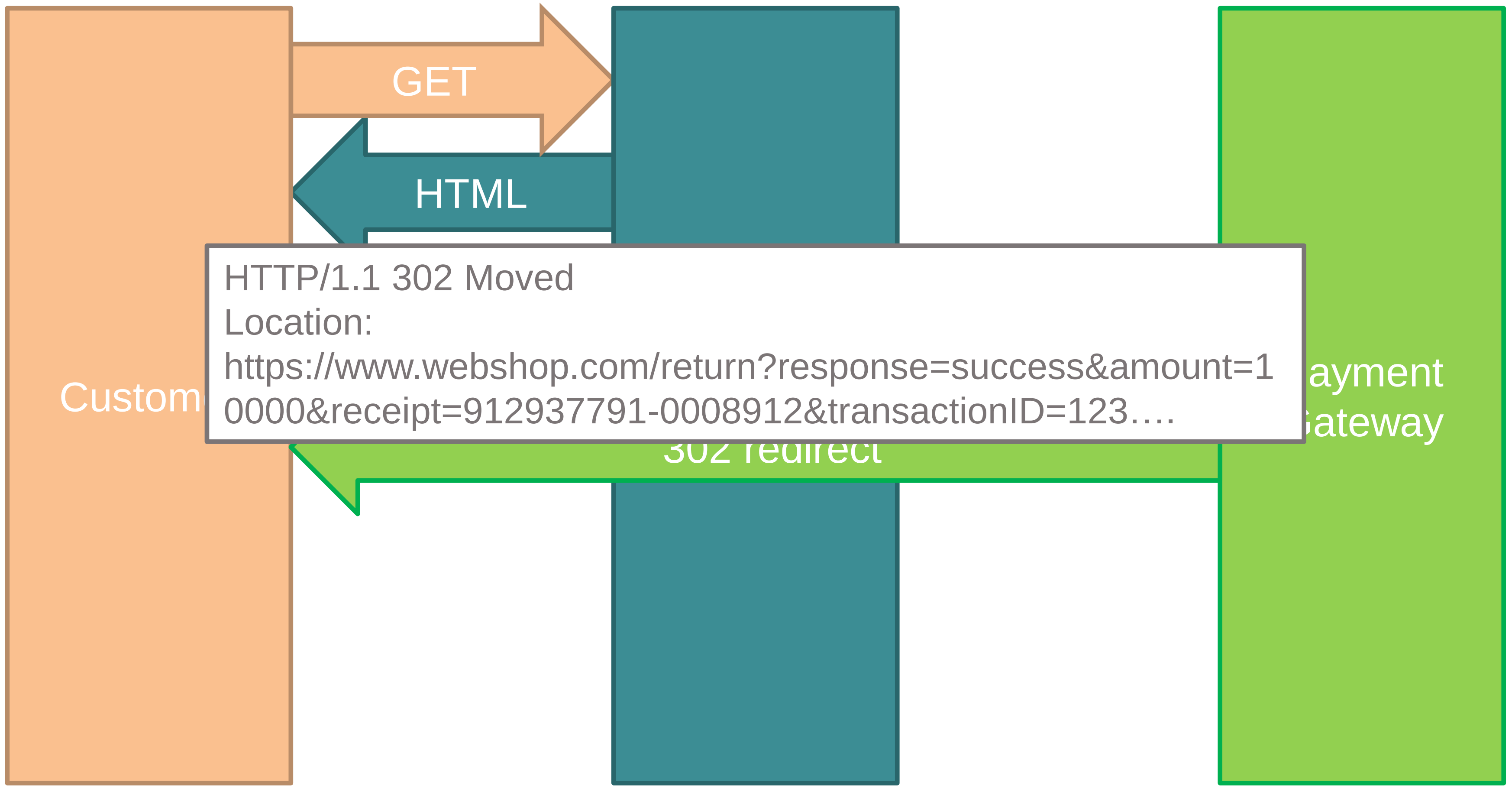
API – HOSTED DIRECT POST



```
POST /svc/pay HTTP/1.1
Host: paymentgw.com
…
Creditcard=4111111111111111
expMonth=04
expYear=15
CCV=123
amount=100.00
```

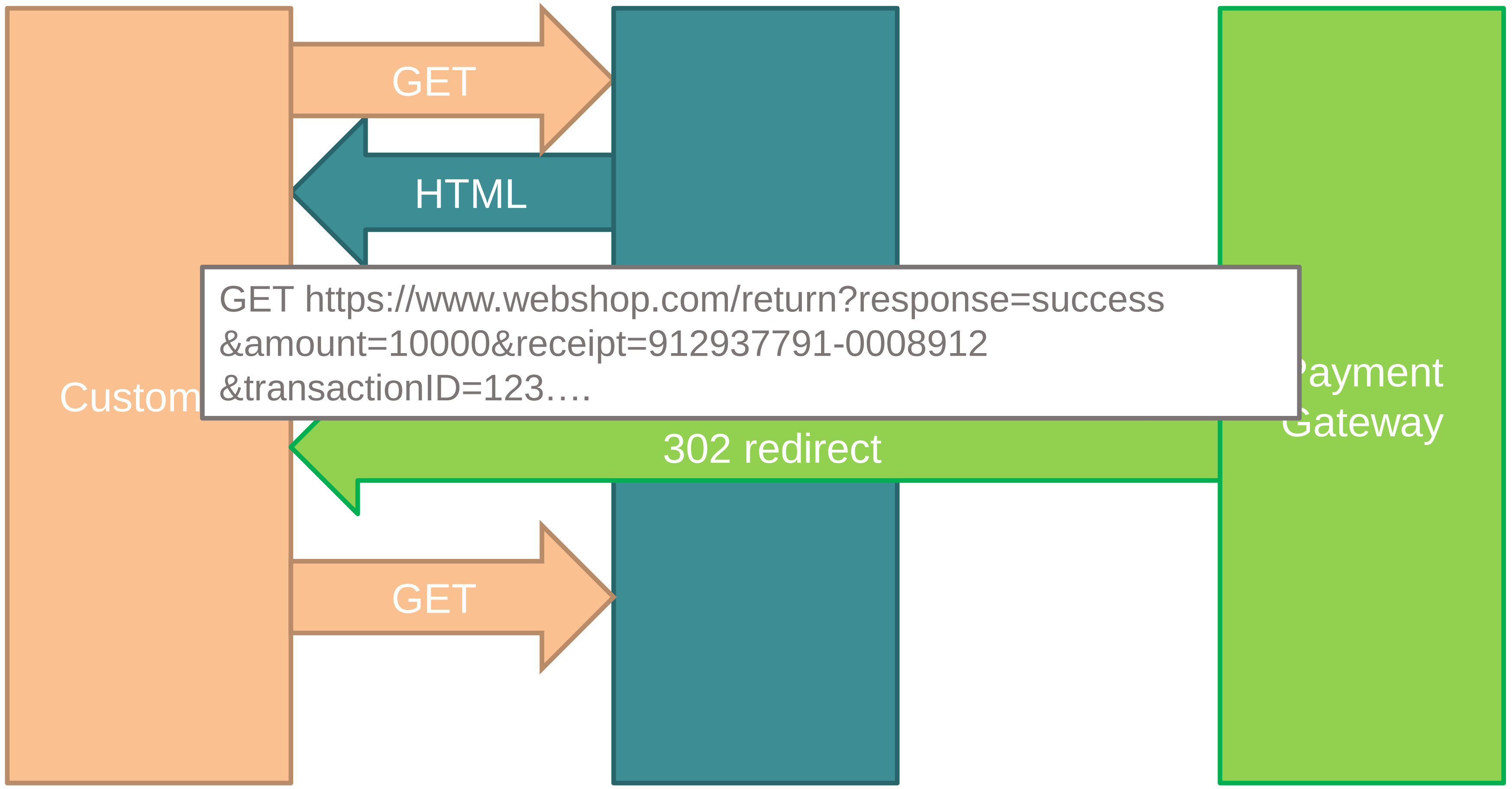
BAE SYSTEMS
INSPIRED WORK

API – HOSTED DIRECT POST



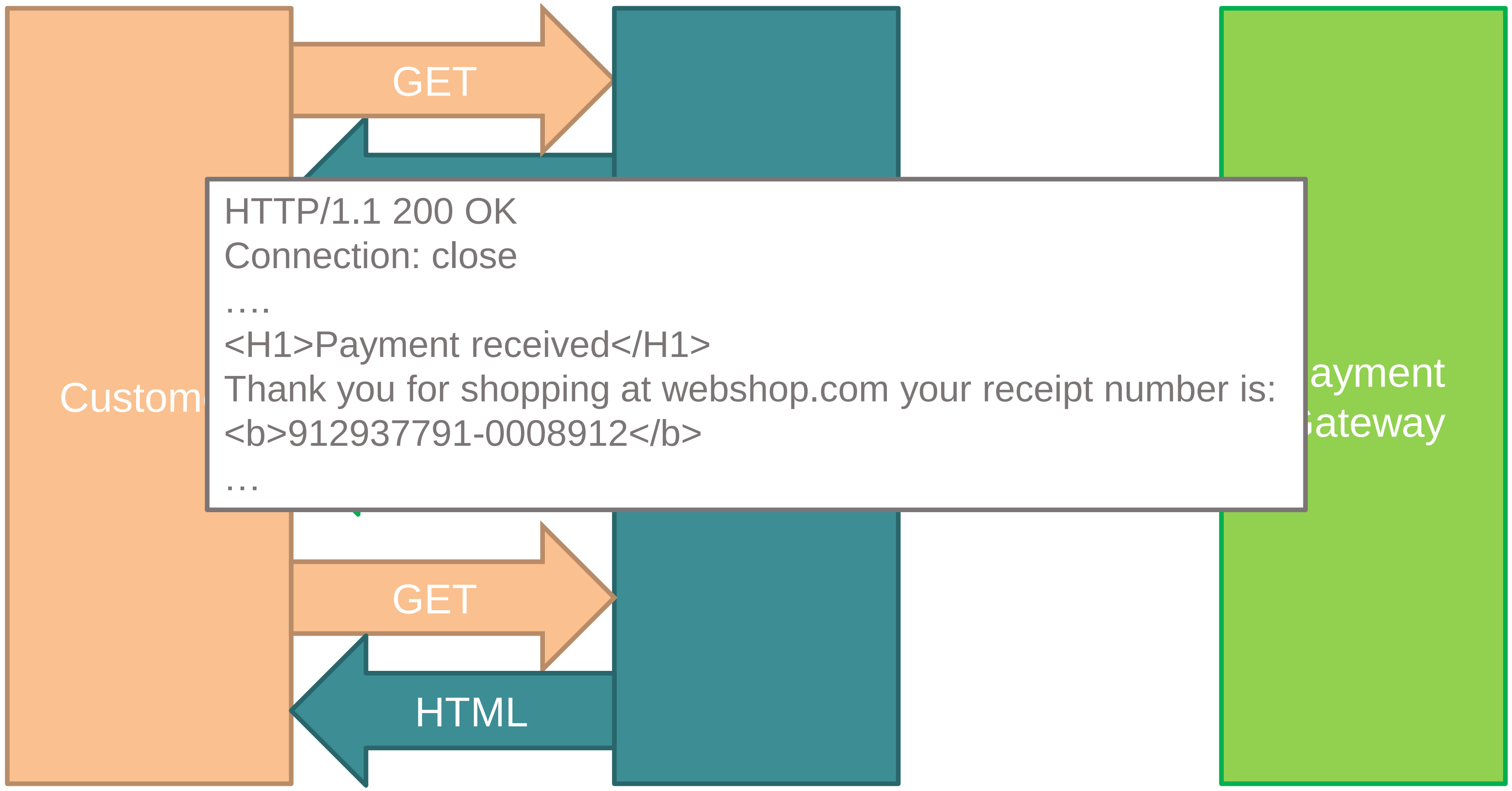
HTTP/1.1 302 Moved
Location:
https://www.webshop.com/return?response=success&amount=10000&receipt=912937791-0008912&transactionID=123….

API – HOSTED DIRECT POST



GET https://www.webshop.com/return?response=success
&amount=10000&receipt=912937791-0008912
&transactionID=123….

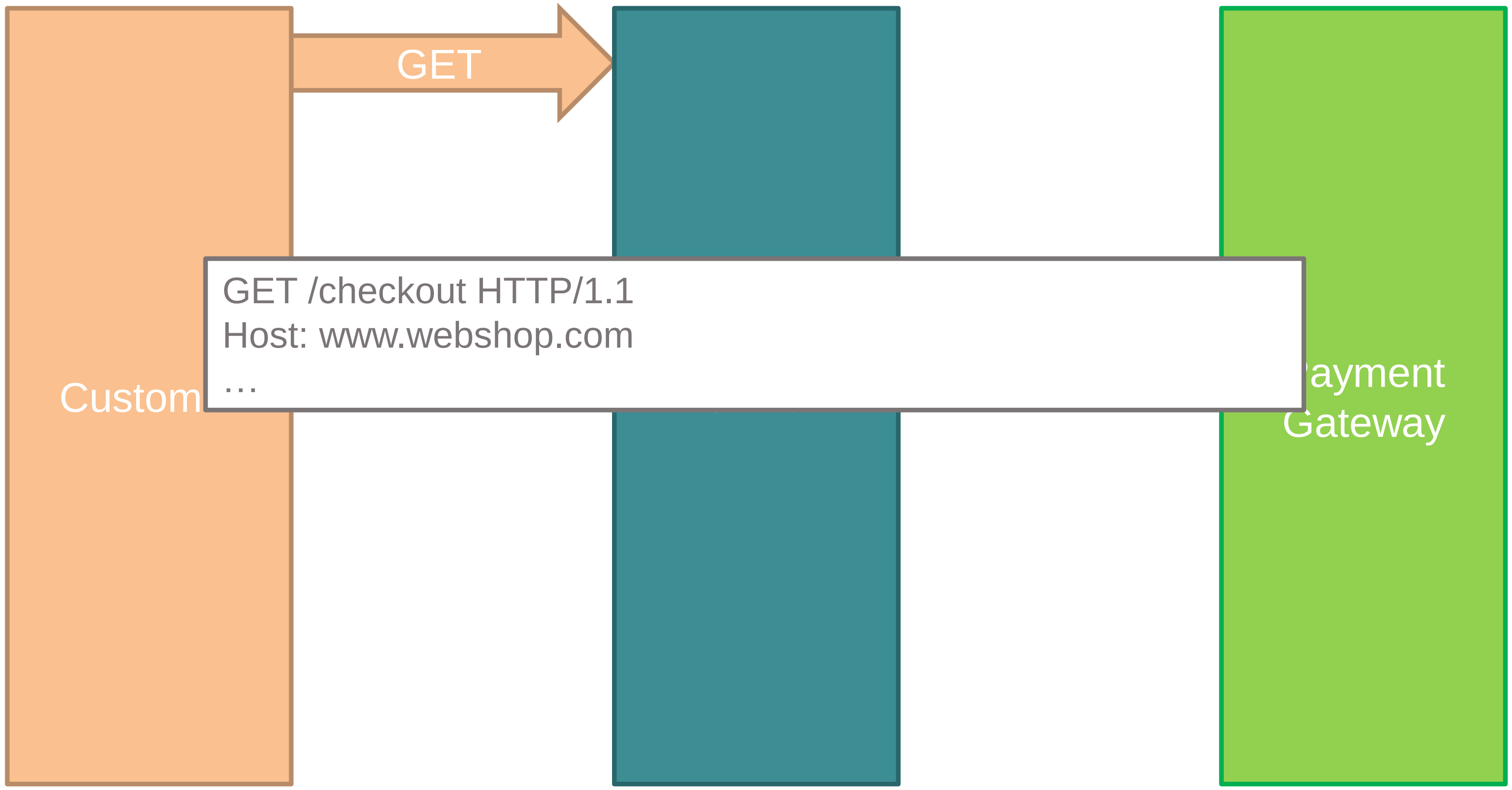
API – HOSTED DIRECT POST



```
HTTP/1.1 200 OK
Connection: close
….
<H1>Payment received</H1>
Thank you for shopping at webshop.com your receipt number is:
<b>912937791-0008912</b>
…
```

BAE SYSTEMS
INSPIRED WORK

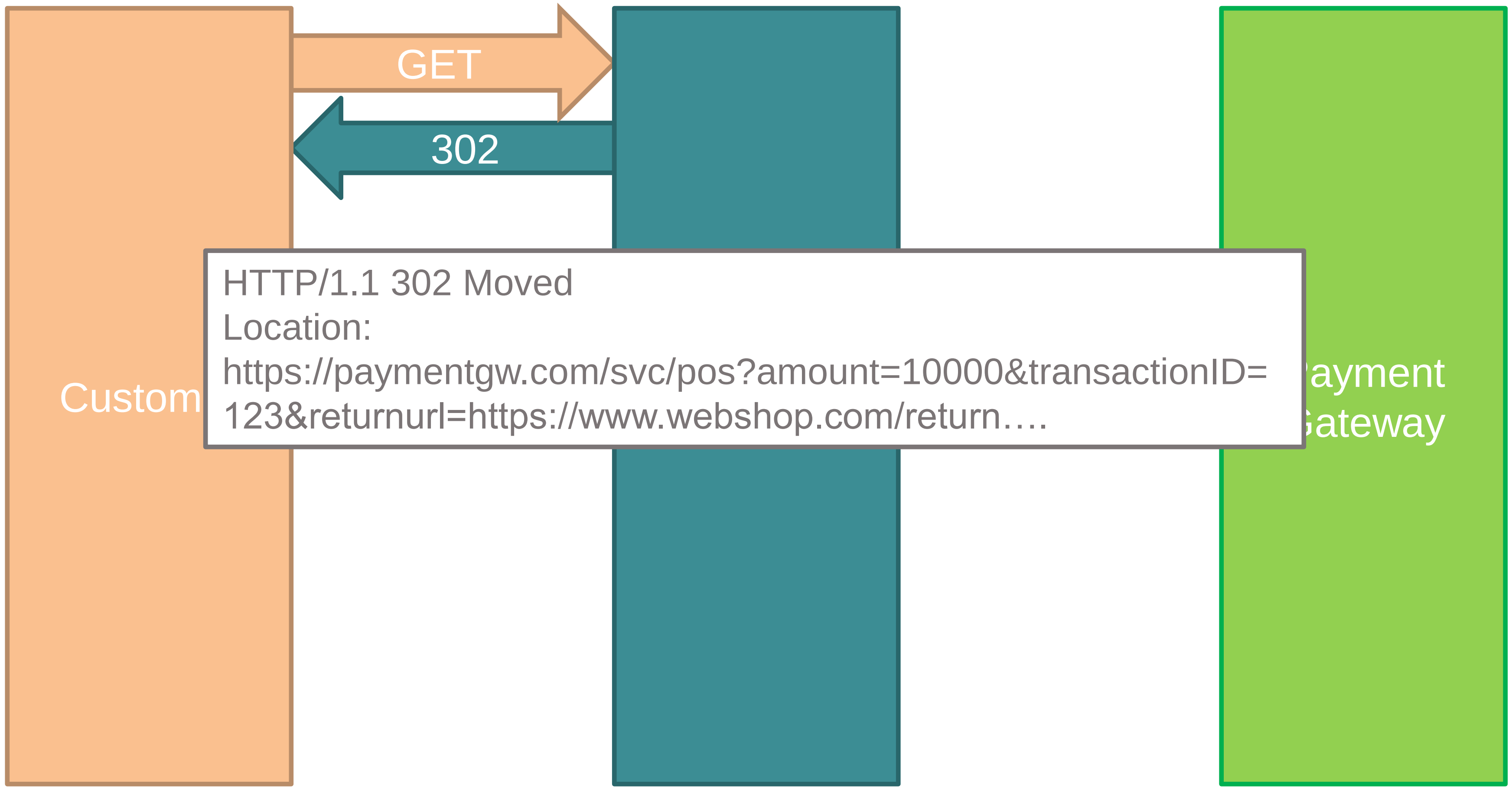
API – HOSTED REDIRECT



```
GET /checkout HTTP/1.1
Host: www.webshop.com
…
```

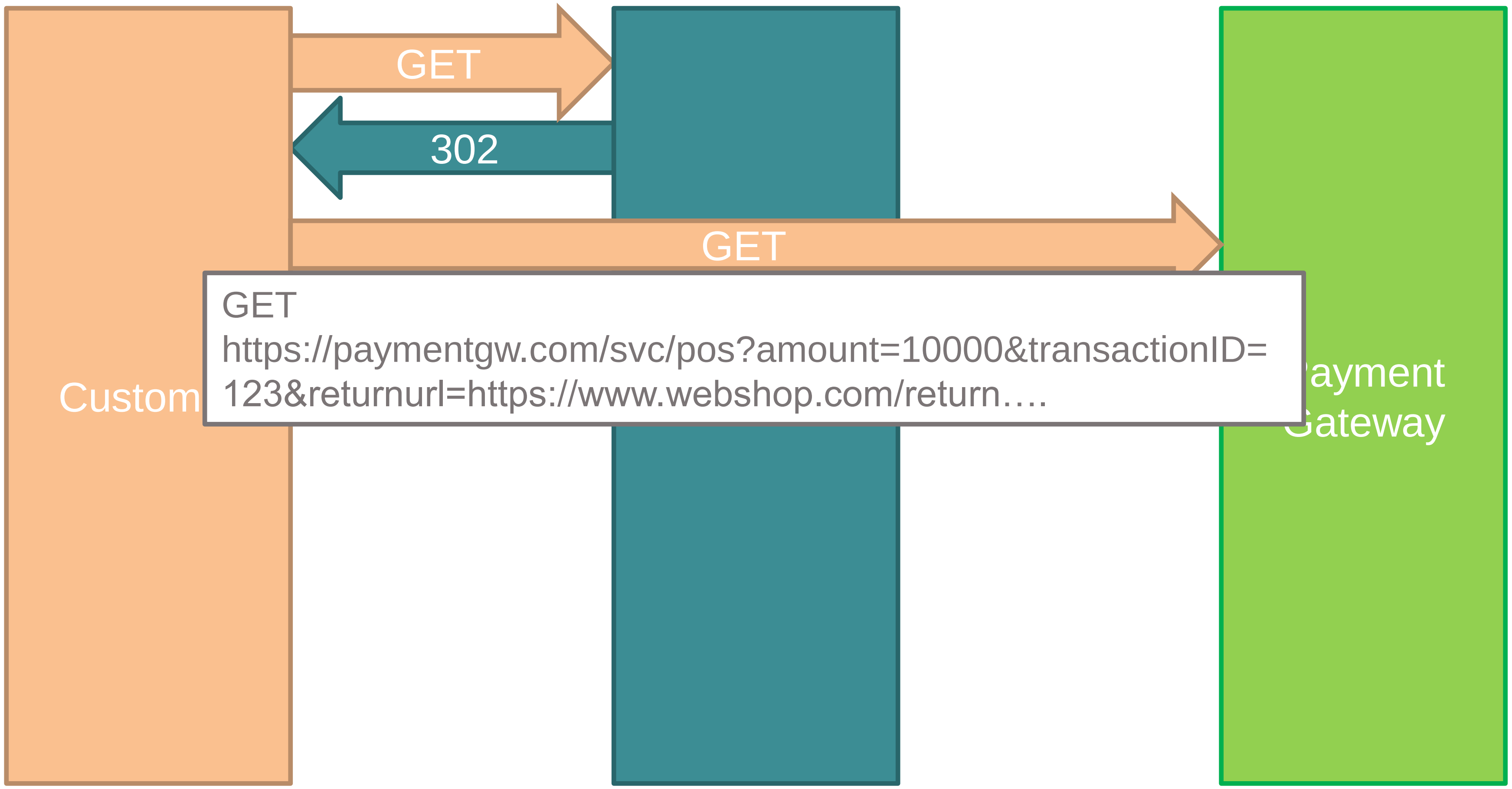
BAE SYSTEMS
INSPIRED WORK

API – HOSTED REDIRECT



```
HTTP/1.1 302 Moved
Location:
https://paymentgw.com/svc/pos?amount=10000&transactionID=
123&returnurl=https://www.webshop.com/return….
```

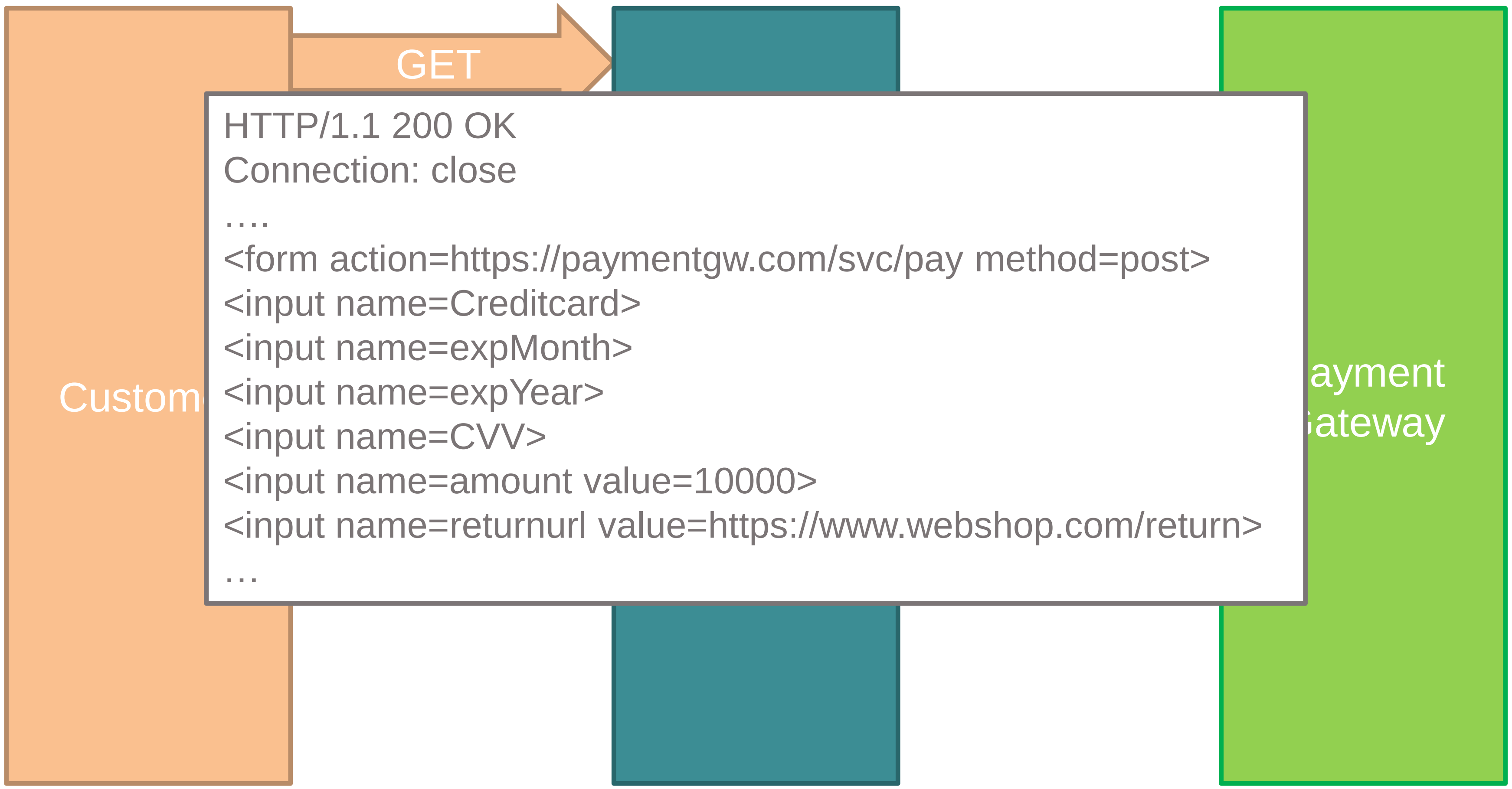
API – HOSTED REDIRECT



GET
https://paymentgw.com/svc/pos?amount=10000&transactionID=123&returnurl=https://www.webshop.com/return….

BAE SYSTEMS
INSPIRED WORK

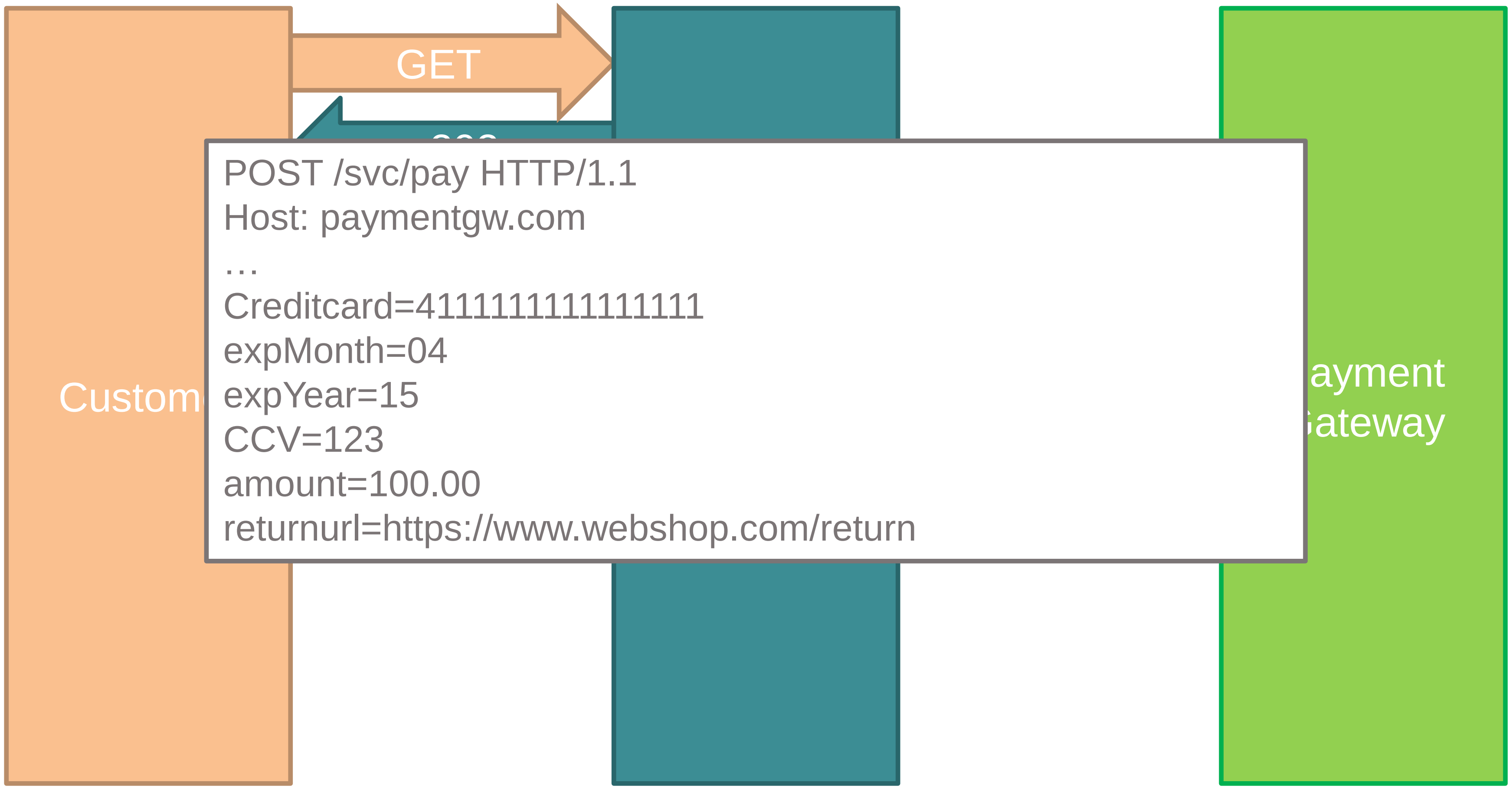
API – HOSTED REDIRECT



```
HTTP/1.1 200 OK
Connection: close
….
<form action=https://paymentgw.com/svc/pay method=post>
<input name=Creditcard>
<input name=expMonth>
<input name=expYear>
<input name=CVV>
<input name=amount value=10000>
<input name=returnurl value=https://www.webshop.com/return>
…
```

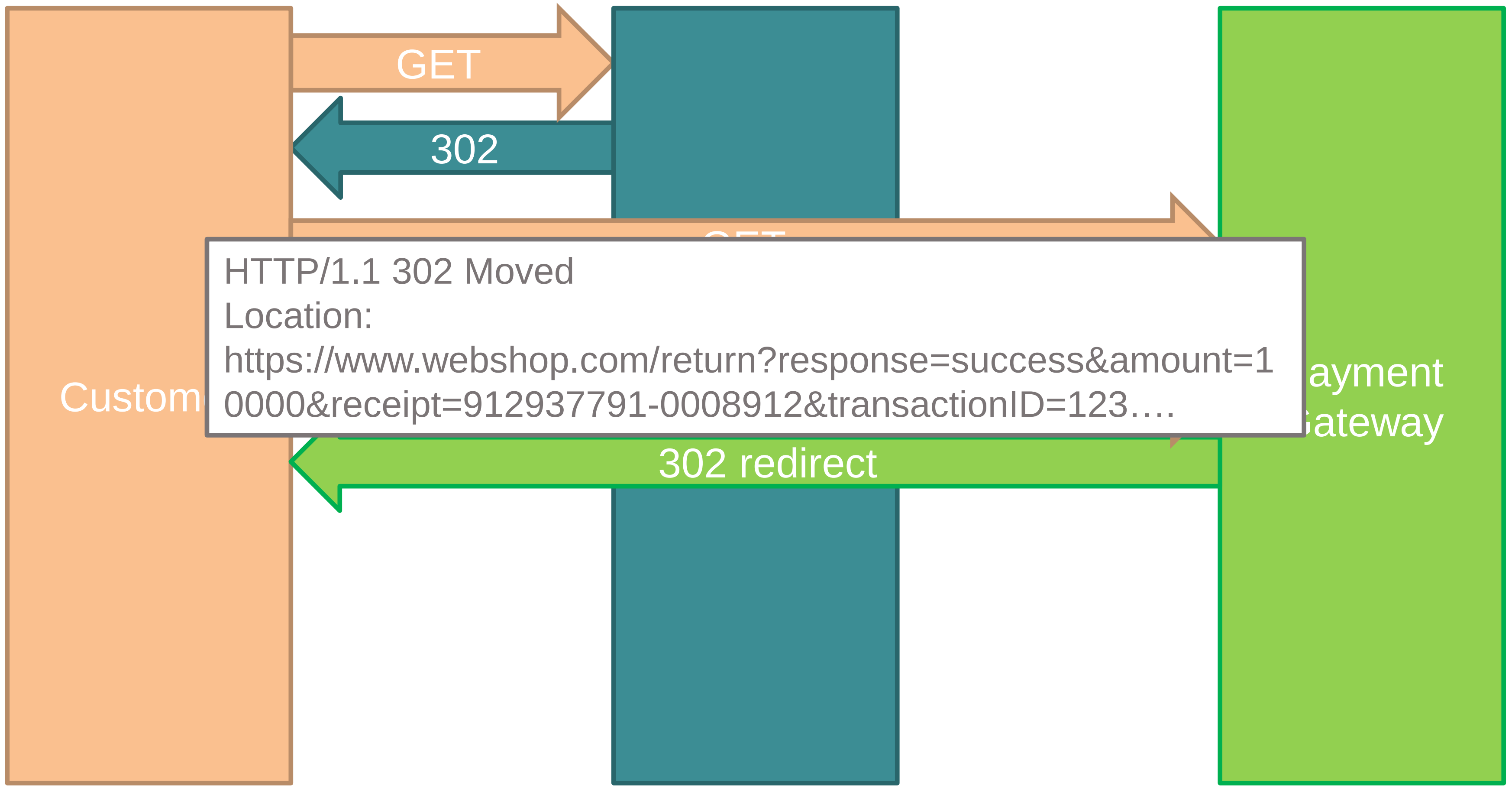
BAE SYSTEMS
INSPIRED WORK

API – HOSTED REDIRECT



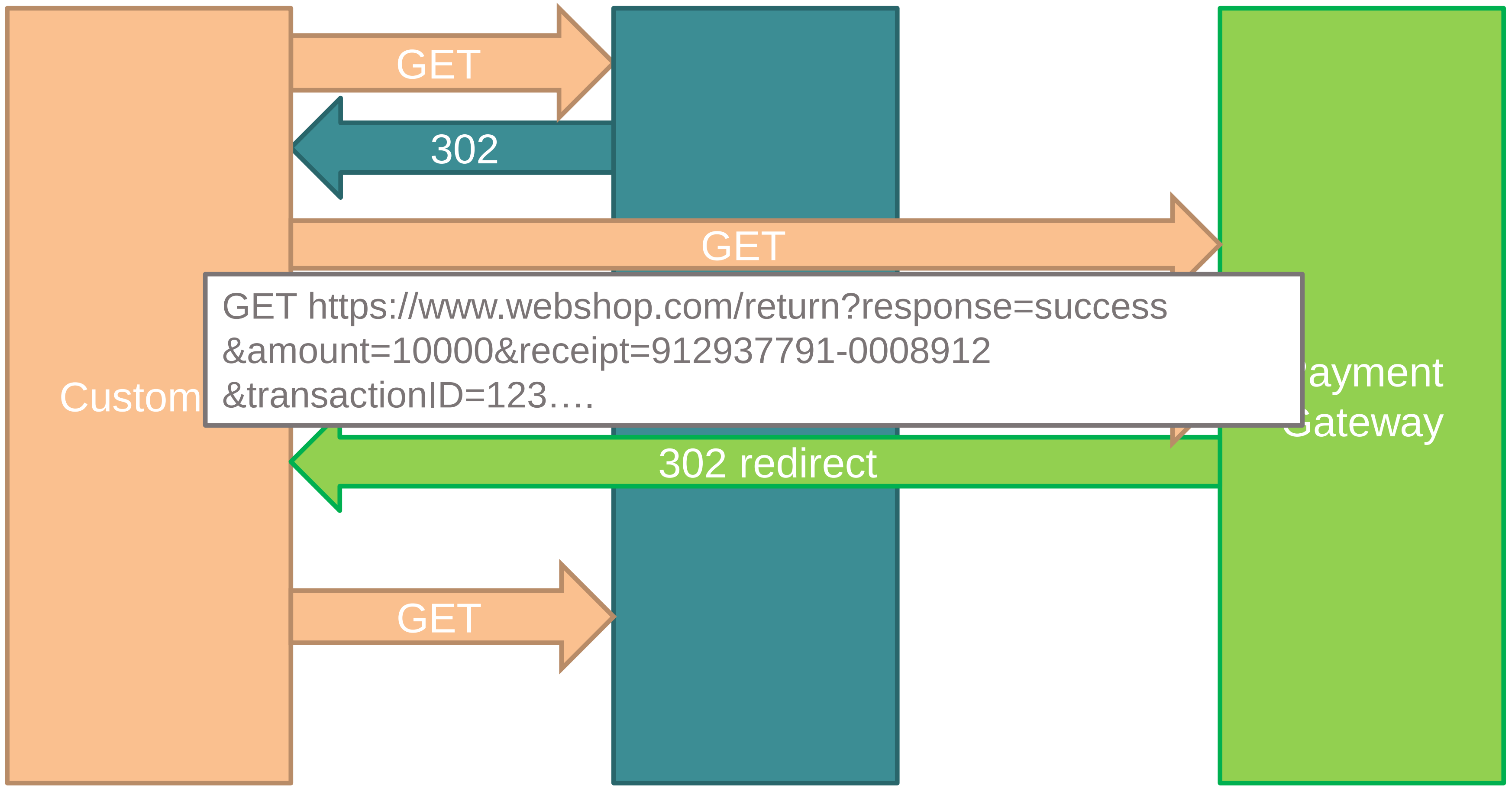
```
POST /svc/pay HTTP/1.1
Host: paymentgw.com
…
Creditcard=4111111111111111
expMonth=04
expYear=15
CCV=123
amount=100.00
returnurl=https://www.webshop.com/return
```

API – HOSTED REDIRECT



HTTP/1.1 302 Moved
Location:
https://www.webshop.com/return?response=success&amount=10000&receipt=912937791-0008912&transactionID=123….

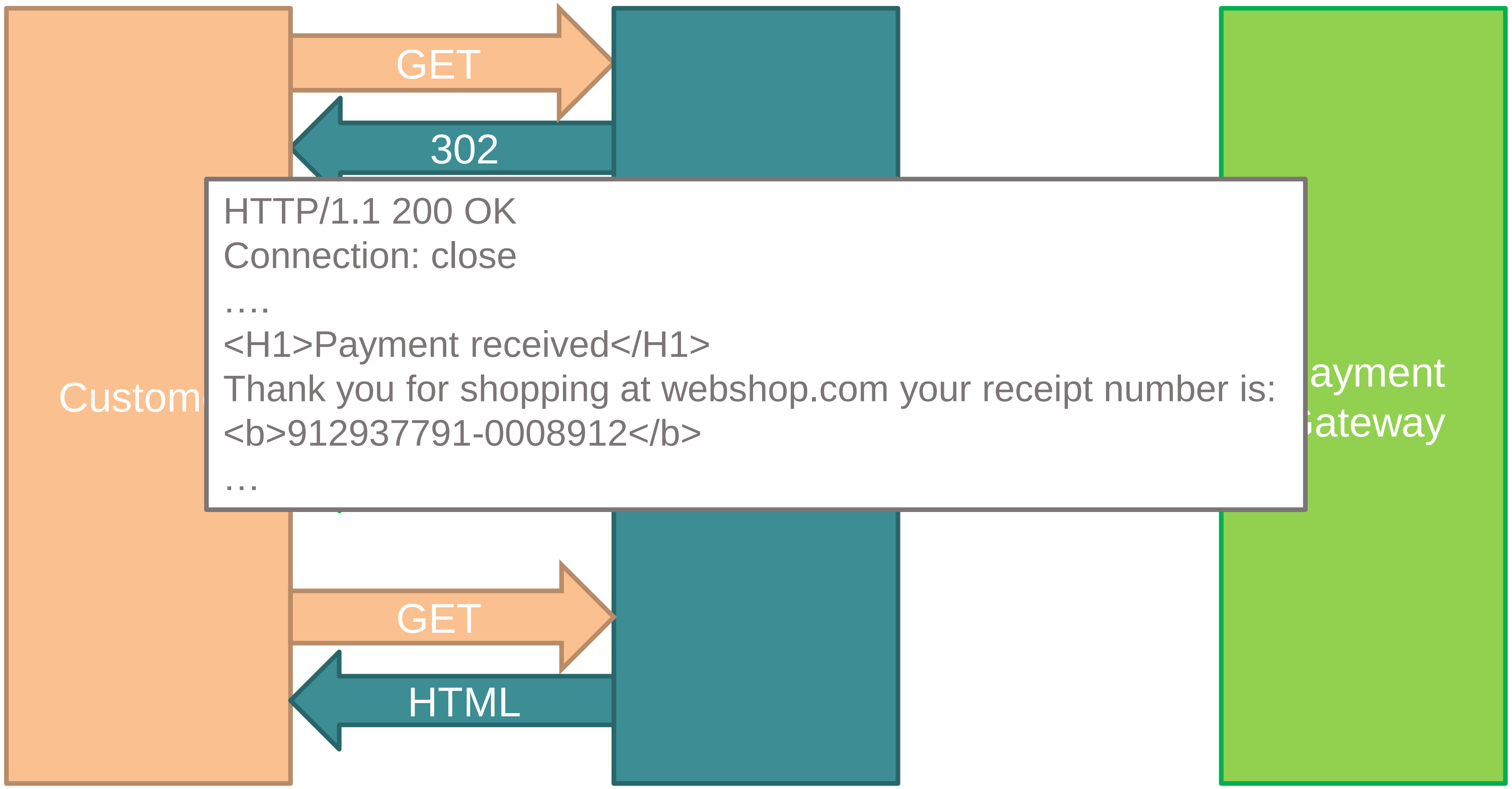
API – HOSTED REDIRECT



GET https://www.webshop.com/return?response=success
&amount=10000&receipt=912937791-0008912
&transactionID=123….

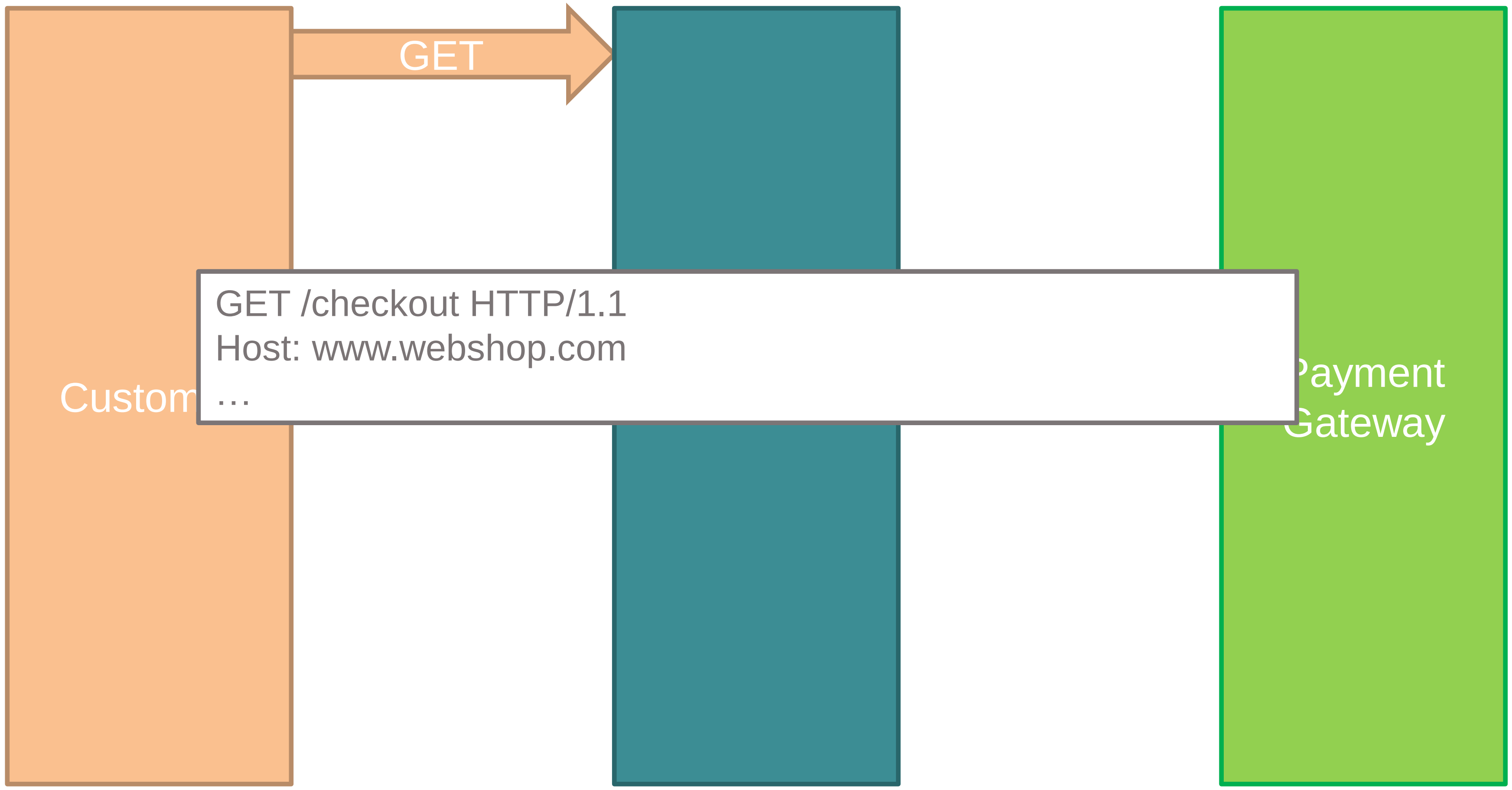
BAE SYSTEMS
INSPIRED WORK

API – HOSTED REDIRECT



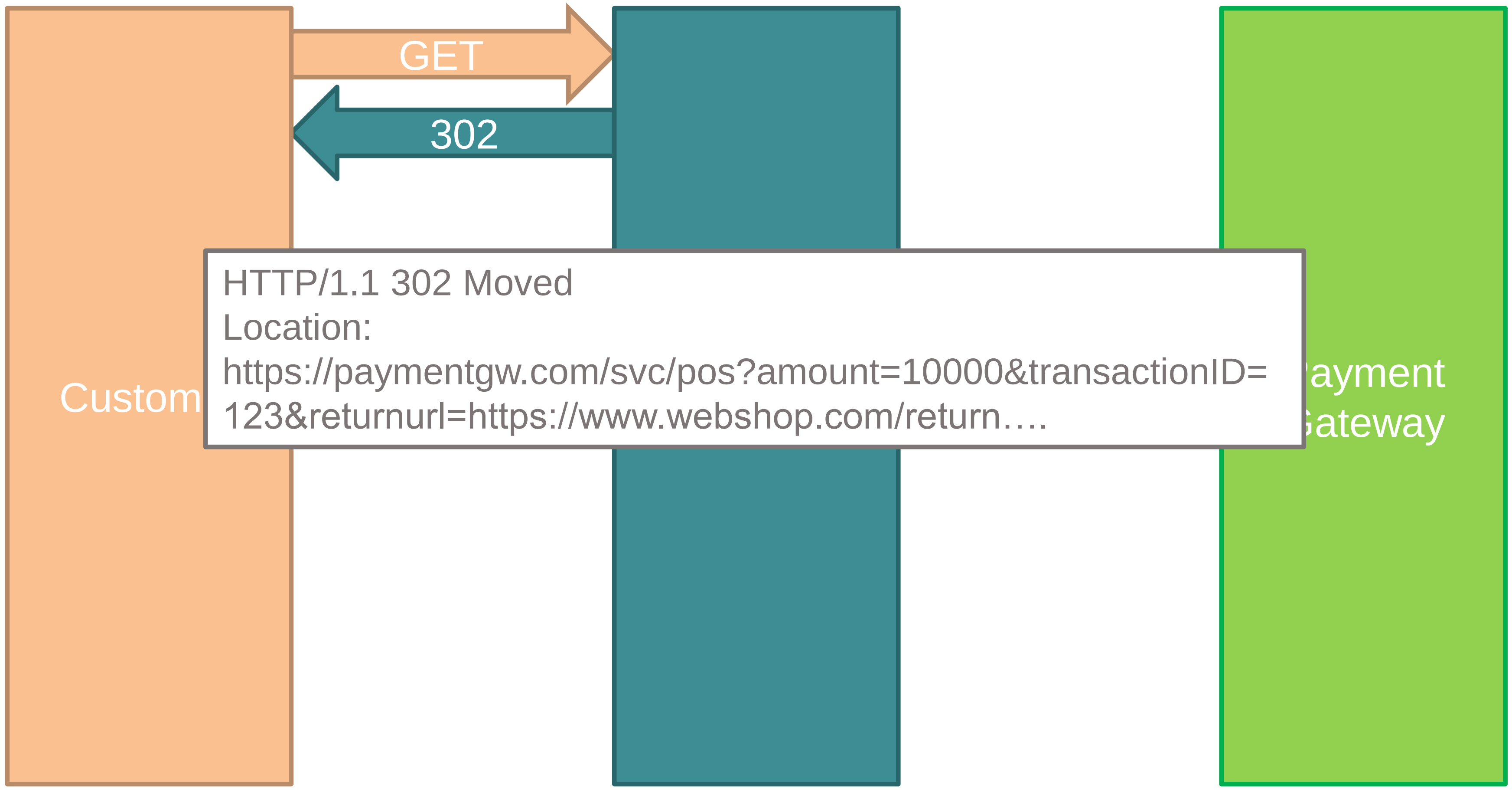
```
HTTP/1.1 200 OK
Connection: close
….
<H1>Payment received</H1>
Thank you for shopping at webshop.com your receipt number is:
<b>912937791-0008912</b>
…
```

API – HOSTED RE-DIRECT ALTERNATIVE



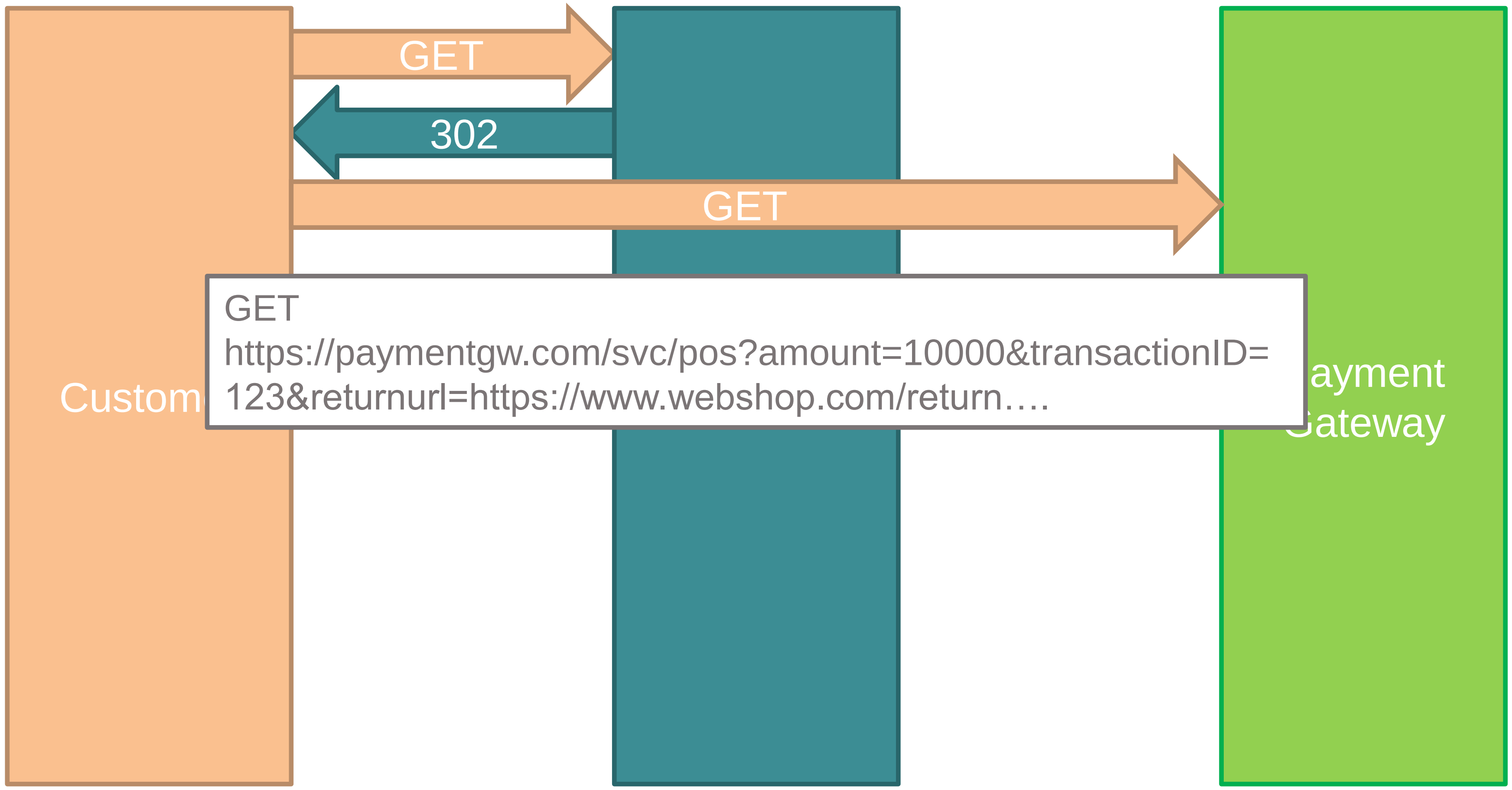
```
GET /checkout HTTP/1.1
Host: www.webshop.com
…
```

API – HOSTED RE-DIRECT ALTERNATIVE



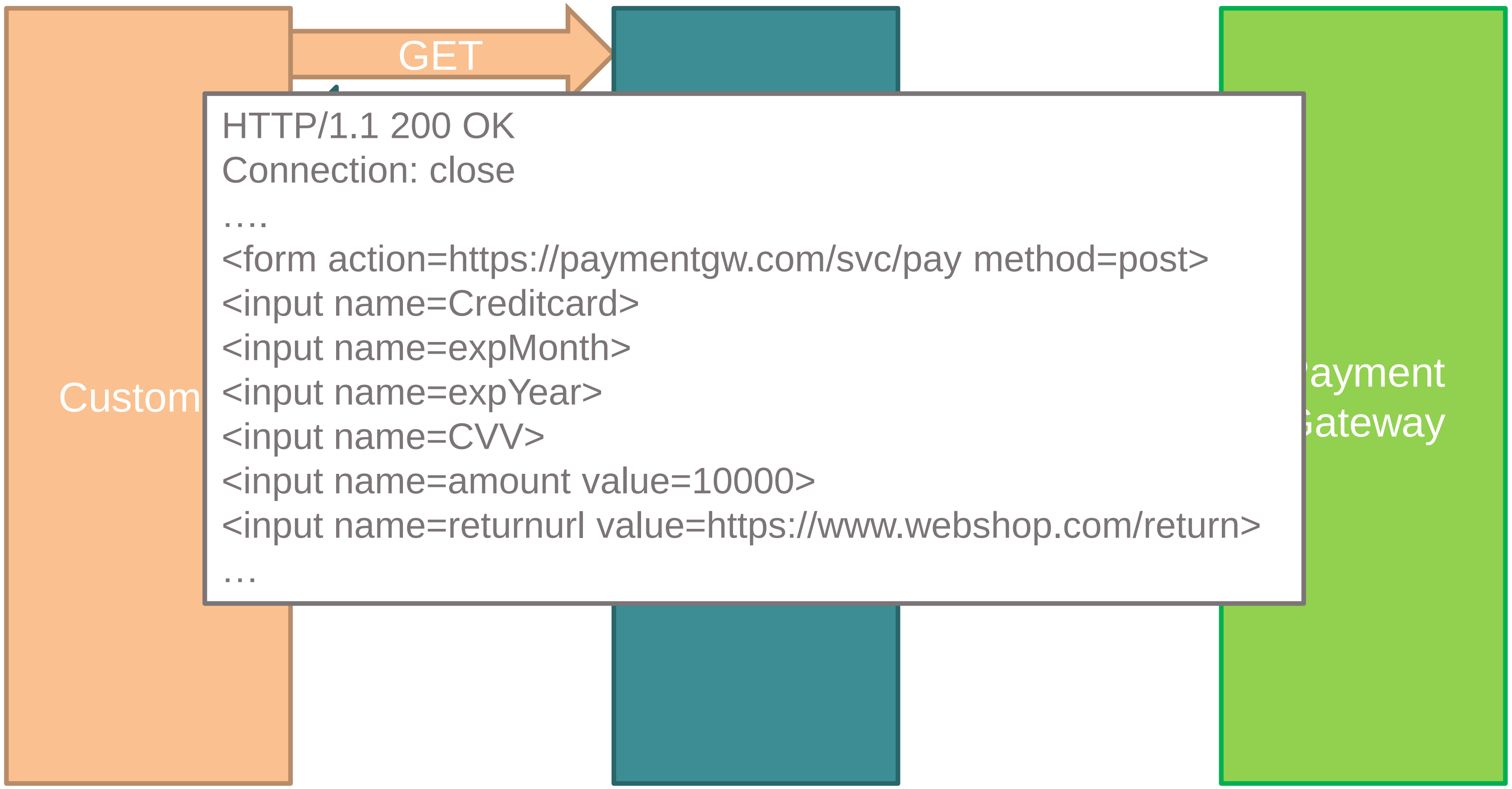
```
HTTP/1.1 302 Moved
Location:
https://paymentgw.com/svc/pos?amount=10000&transactionID=
123&returnurl=https://www.webshop.com/return….
```

API – HOSTED RE-DIRECT ALTERNATIVE



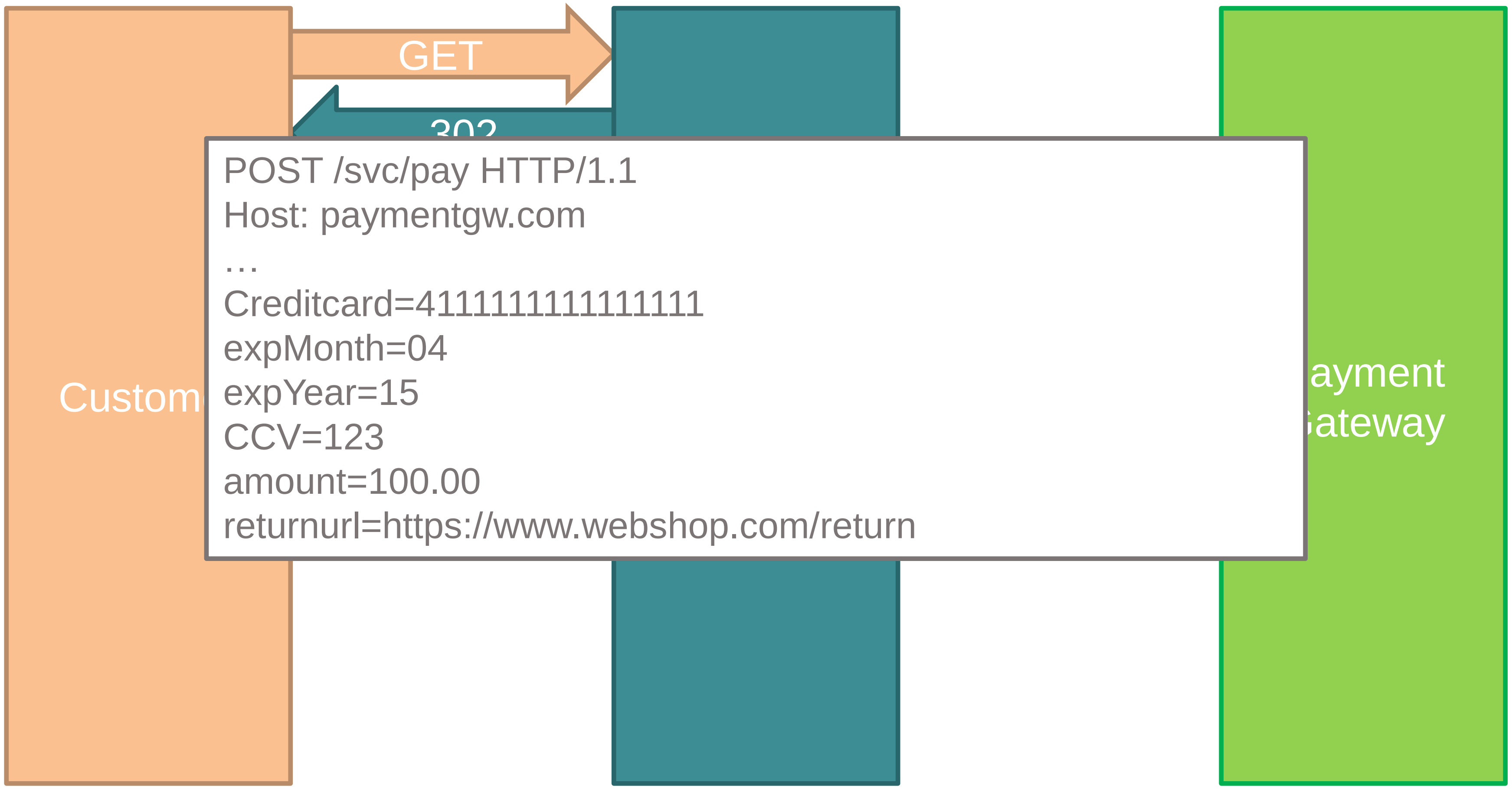
GET
https://paymentgw.com/svc/pos?amount=10000&transactionID=123&returnurl=https://www.webshop.com/return….

API – HOSTED RE-DIRECT ALTERNATIVE



```
HTTP/1.1 200 OK
Connection: close
….
<form action=https://paymentgw.com/svc/pay method=post>
<input name=Creditcard>
<input name=expMonth>
<input name=expYear>
<input name=CVV>
<input name=amount value=10000>
<input name=returnurl value=https://www.webshop.com/return>
…
```

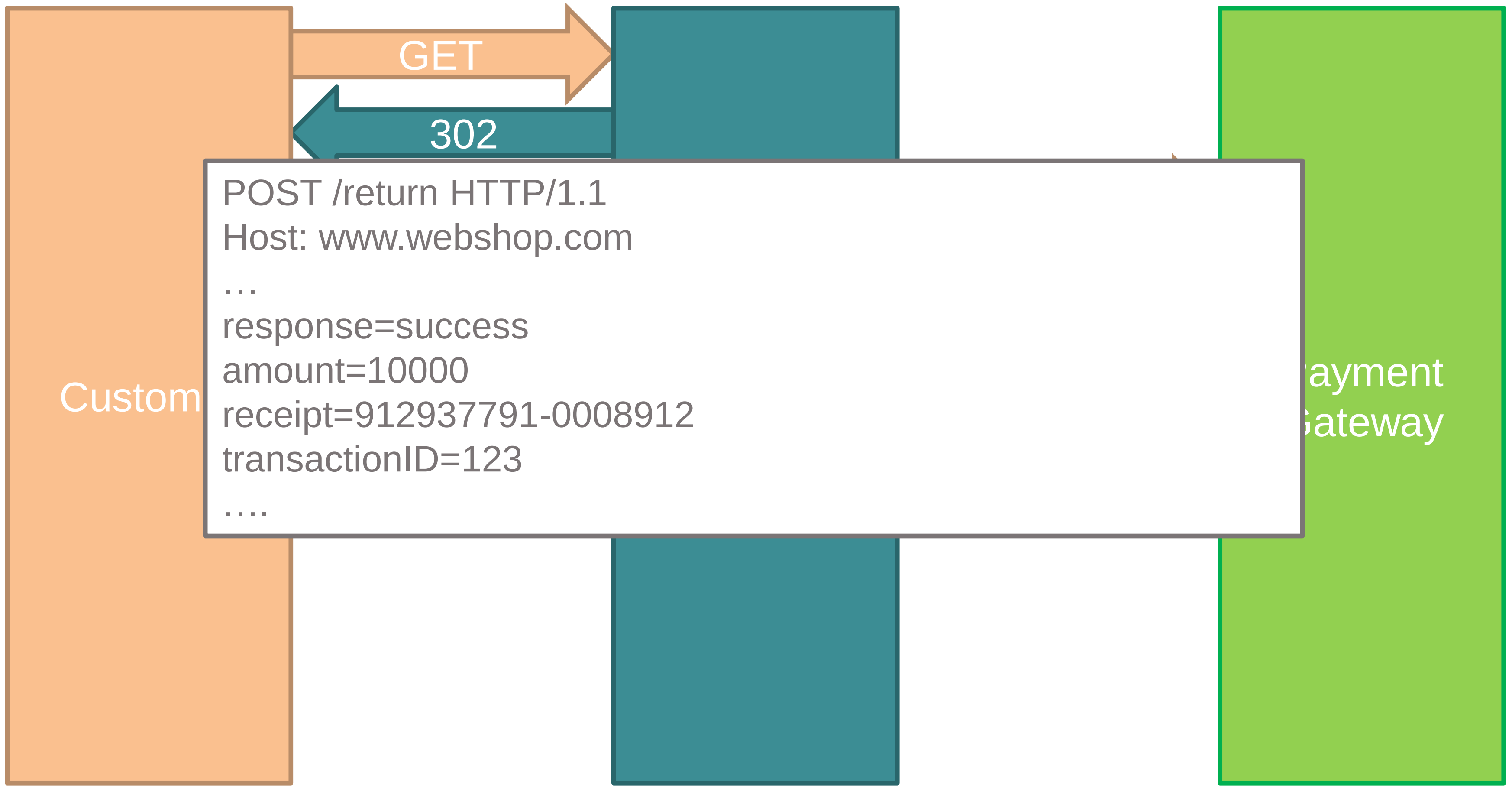
API – HOSTED RE-DIRECT ALTERNATIVE



```
POST /svc/pay HTTP/1.1
Host: paymentgw.com
…
Creditcard=4111111111111111
expMonth=04
expYear=15
CCV=123
amount=100.00
returnurl=https://www.webshop.com/return
```

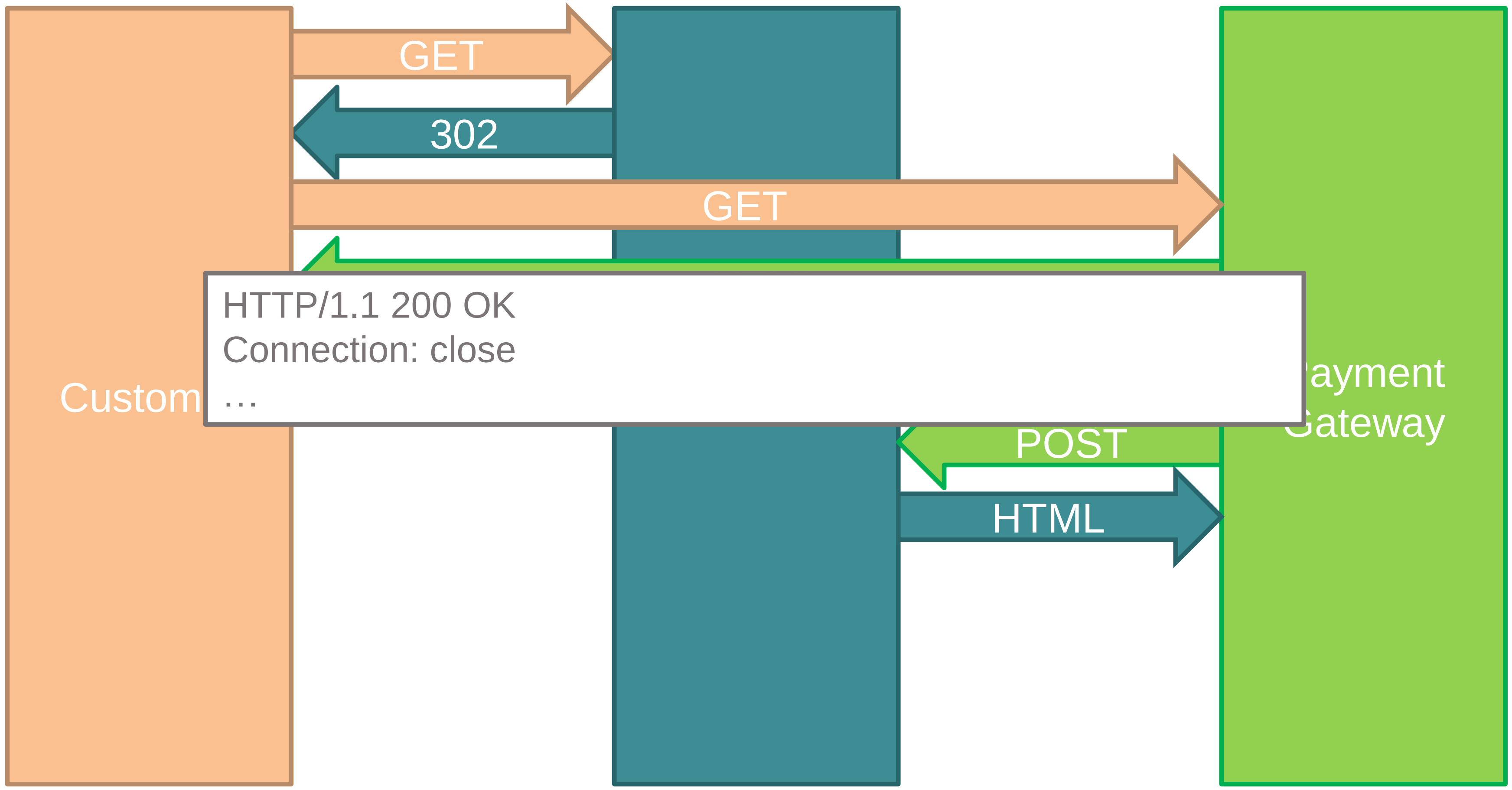
BAE SYSTEMS
INSPIRED WORK

API – HOSTED RE-DIRECT ALTERNATIVE



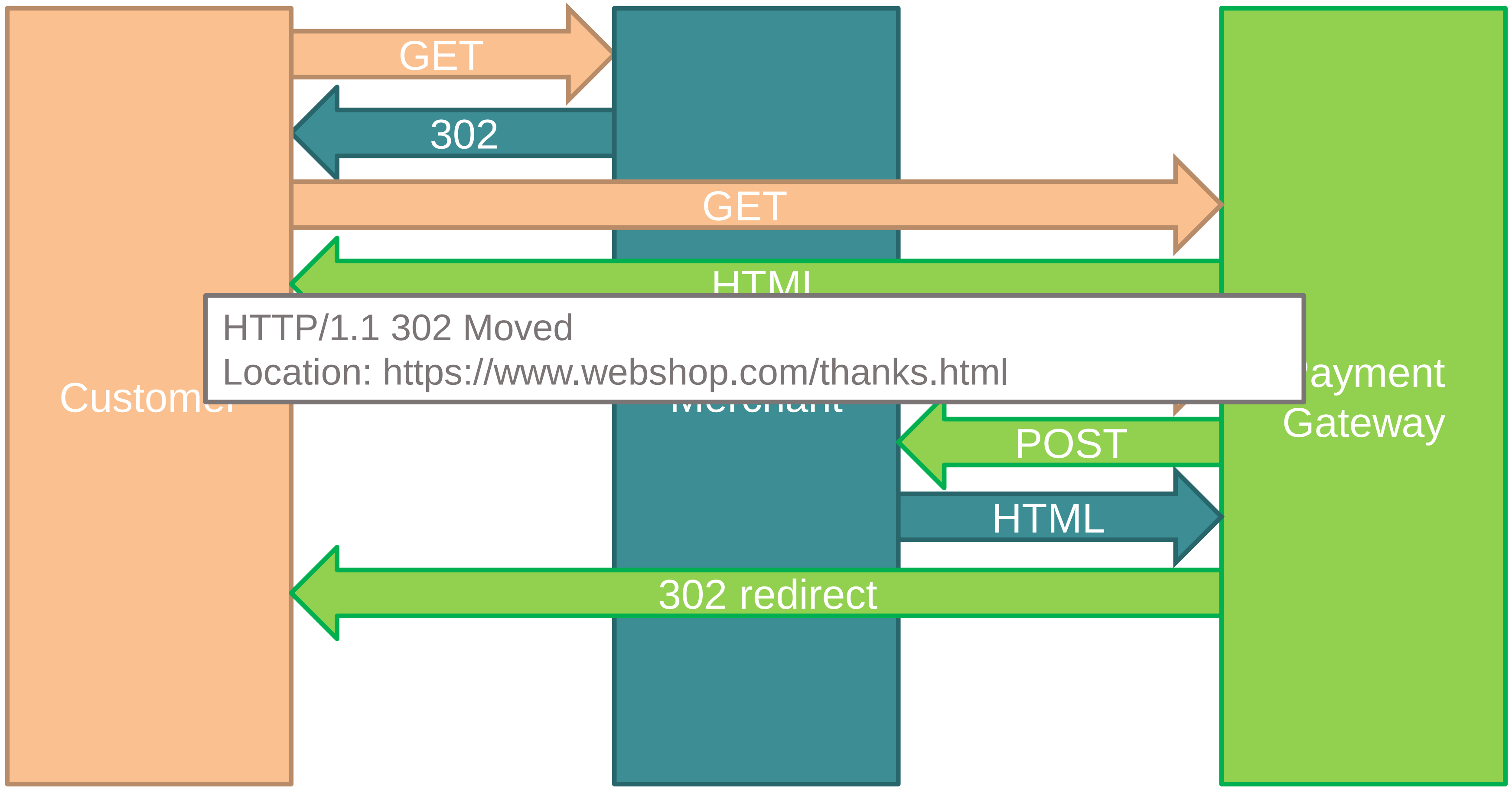
```
POST /return HTTP/1.1
Host: www.webshop.com
…
response=success
amount=10000
receipt=912937791-0008912
transactionID=123
….
```

API – HOSTED RE-DIRECT ALTERNATIVE



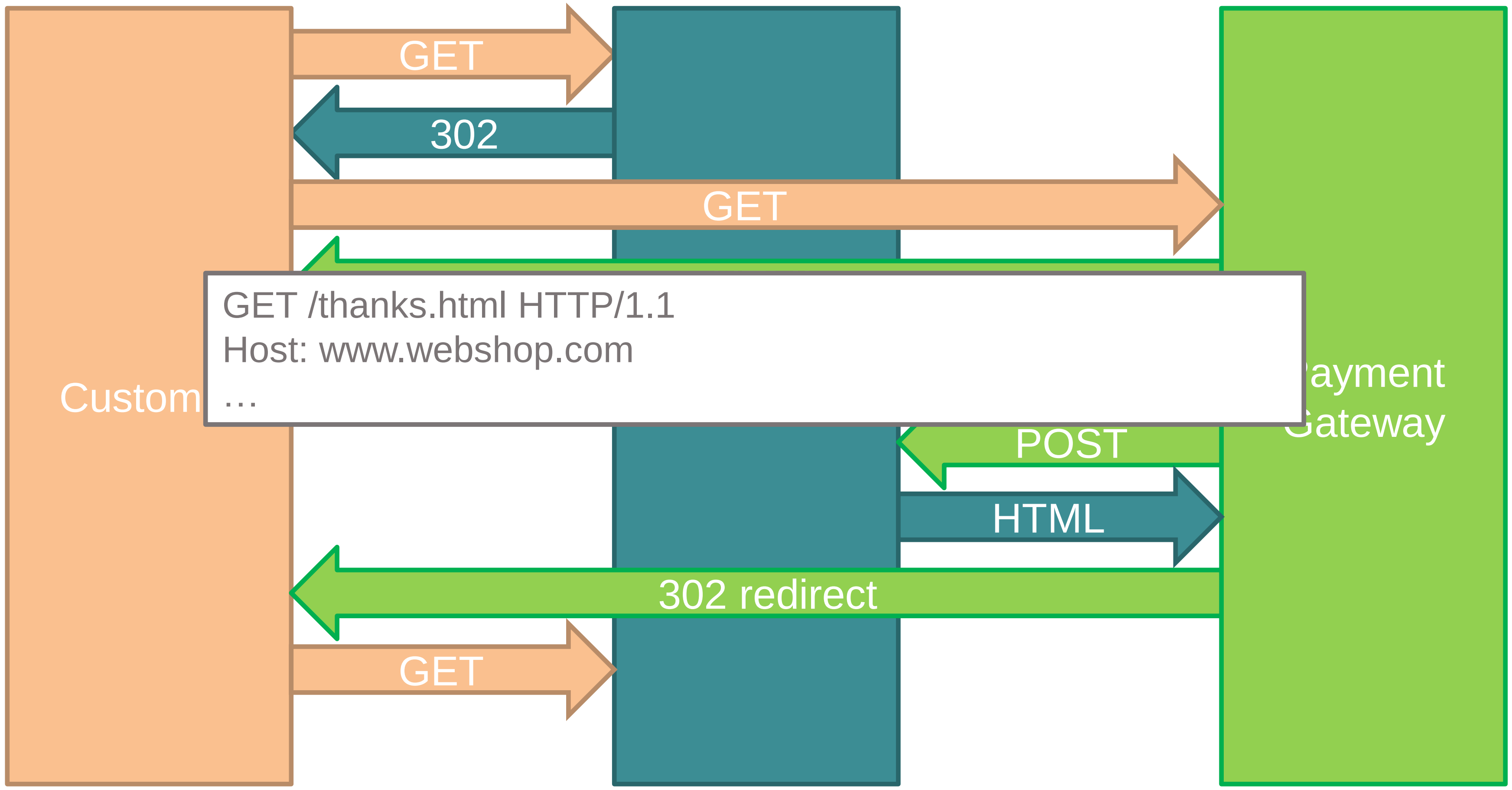
```
HTTP/1.1 200 OK
Connection: close
…
```

API – HOSTED RE-DIRECT ALTERNATIVE



```
HTTP/1.1 302 Moved
Location: https://www.webshop.com/thanks.html
```

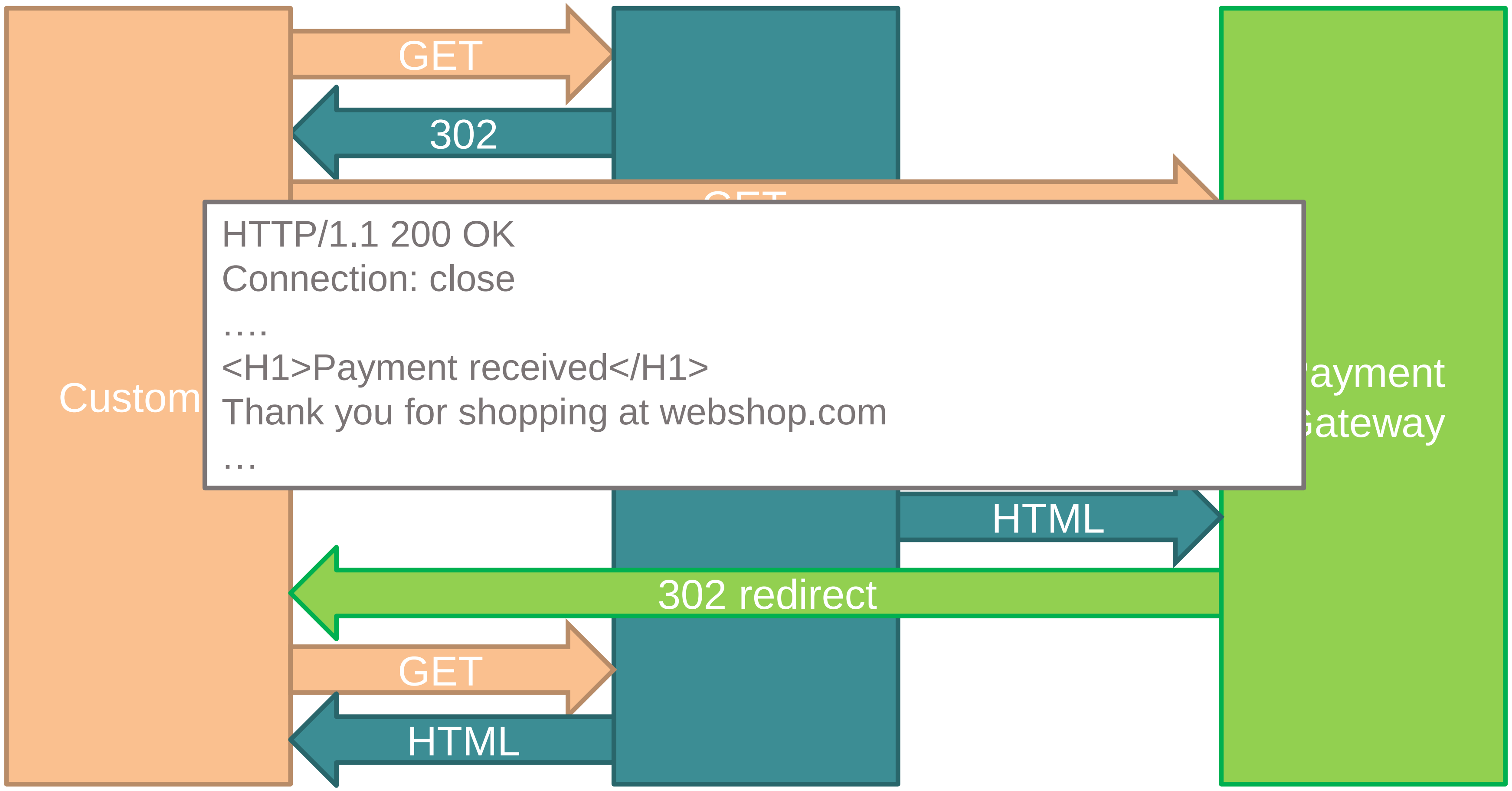
API – HOSTED RE-DIRECT ALTERNATIVE



```
GET /thanks.html HTTP/1.1
Host: www.webshop.com
…
```

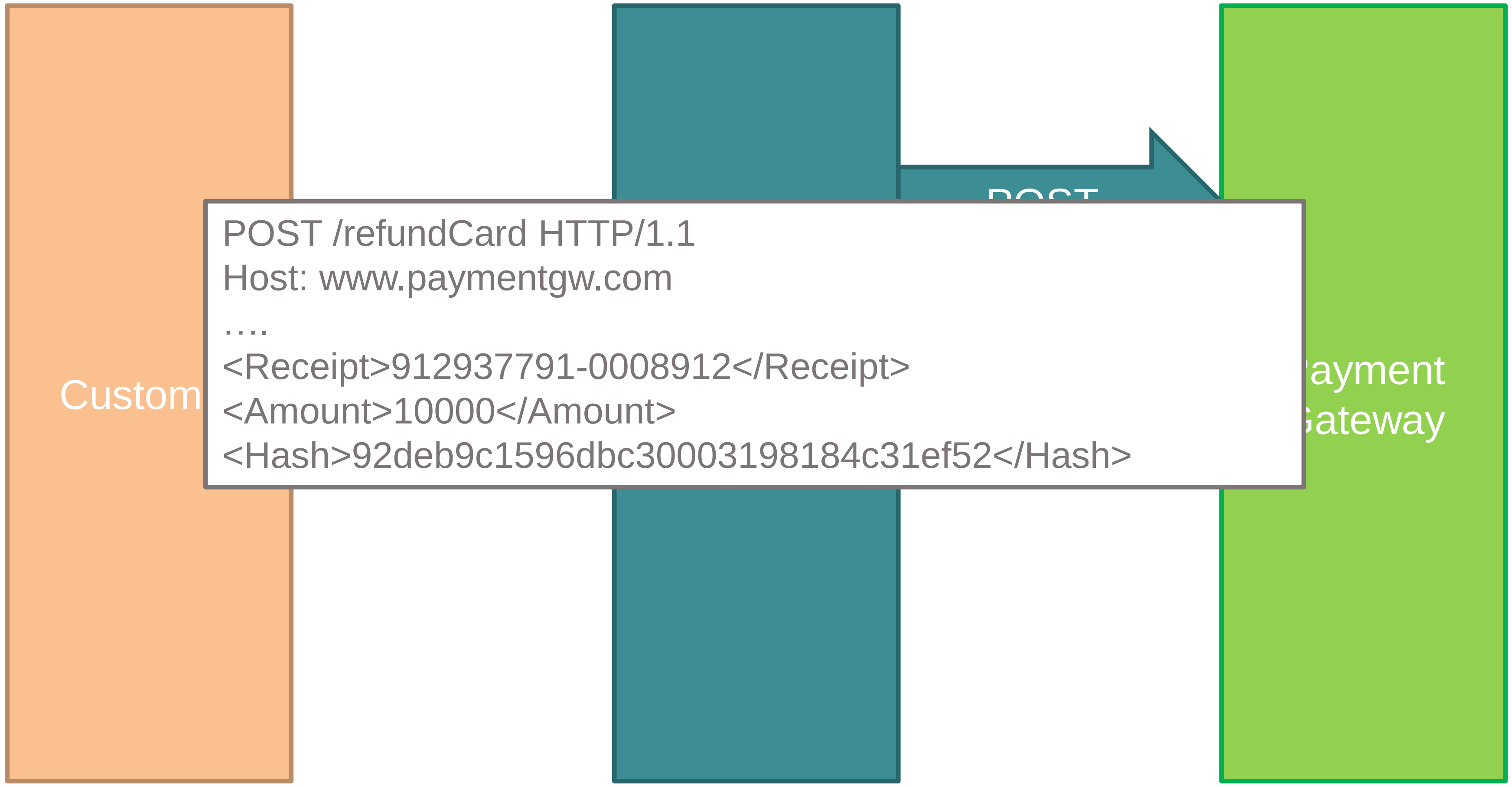
BAE SYSTEMS
INSPIRED WORK

API – HOSTED RE-DIRECT ALTERNATIVE



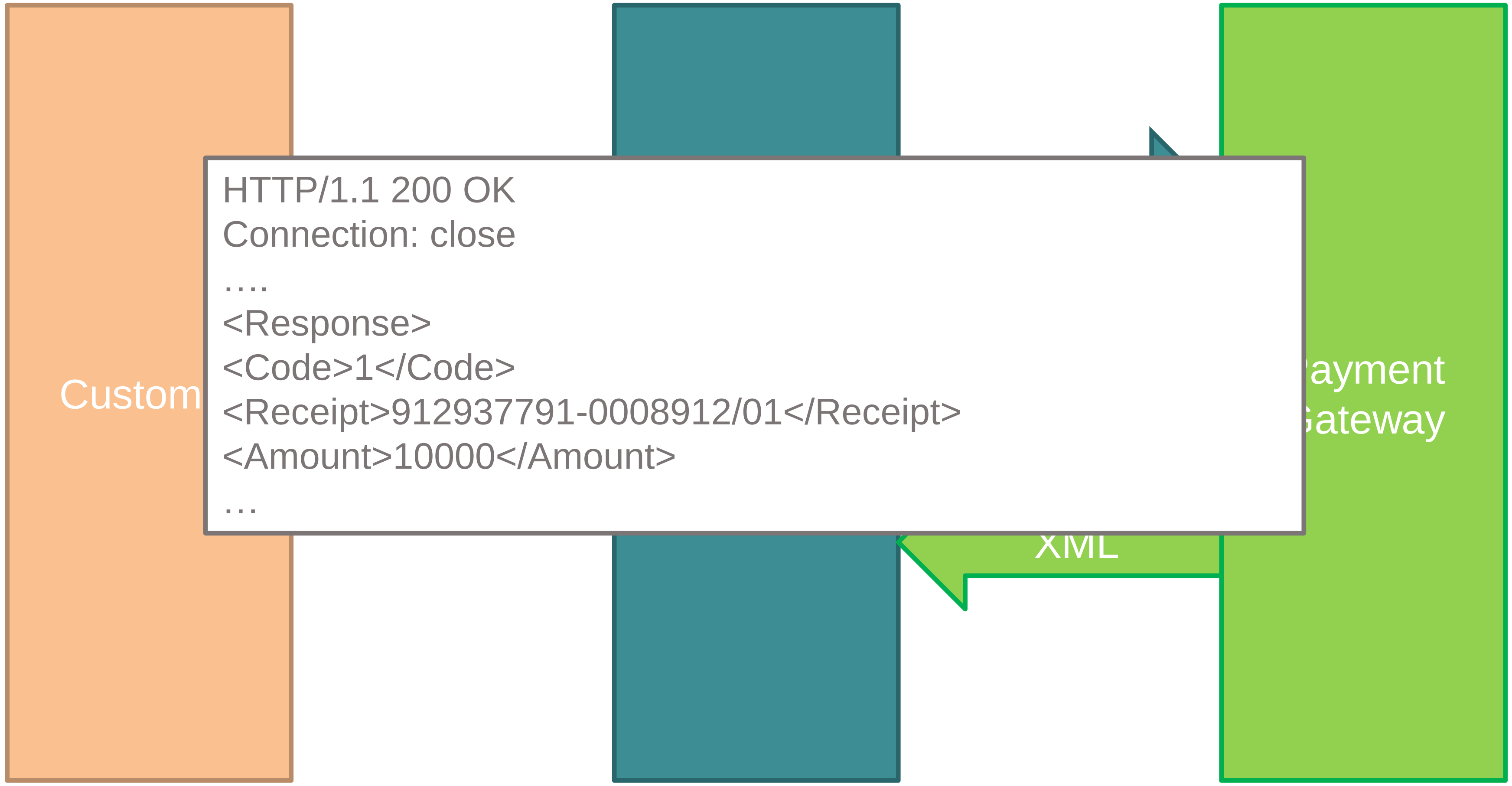
```
HTTP/1.1 200 OK
Connection: close
….
<H1>Payment received</H1>
Thank you for shopping at webshop.com
…
```

API - DIRECT



```
POST /refundCard HTTP/1.1
Host: www.paymentgw.com
….
<Receipt>912937791-0008912</Receipt>
<Amount>10000</Amount>
<Hash>92deb9c1596dbc30003198184c31ef52</Hash>
```

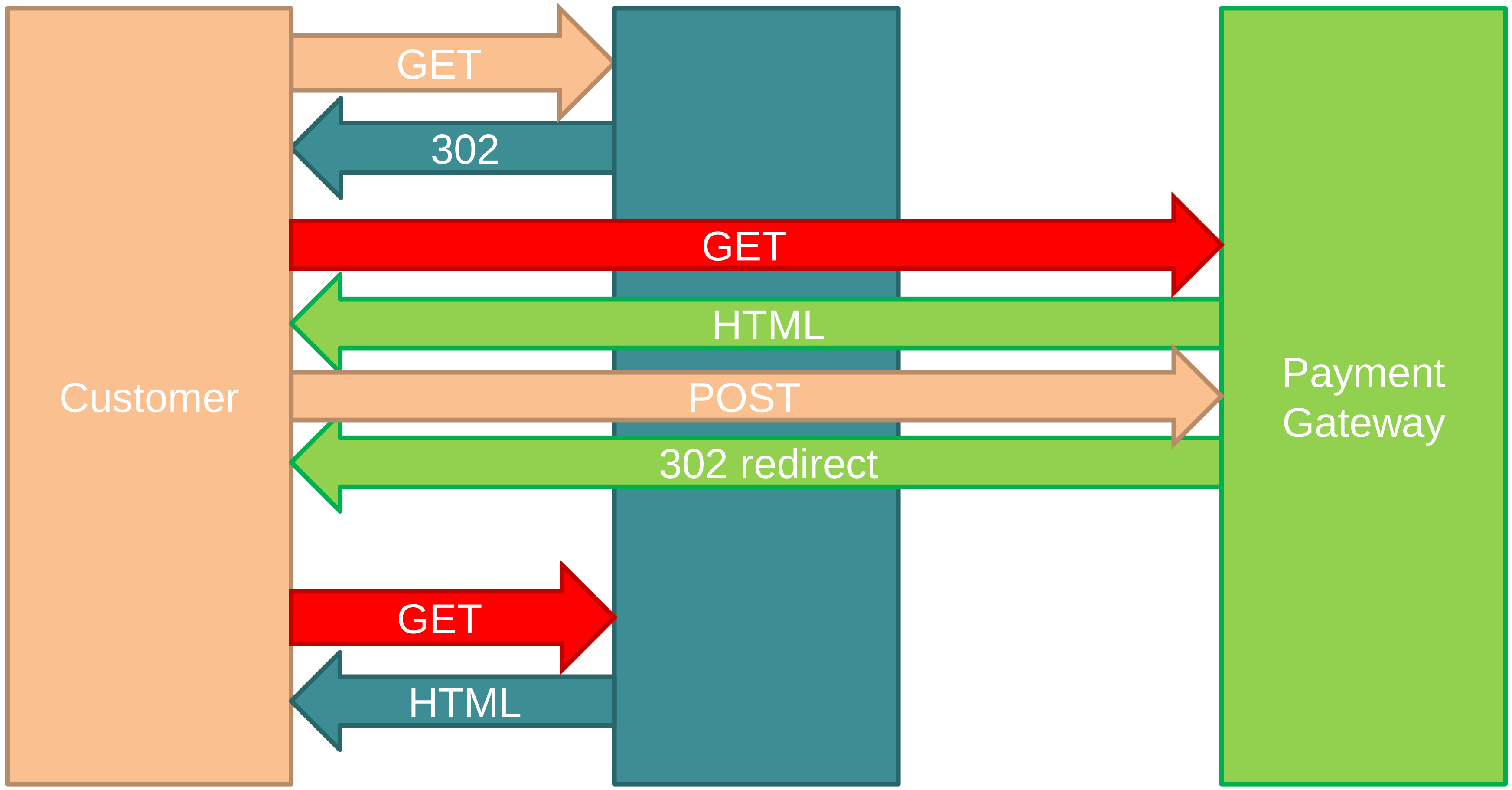
API - DIRECT



```
HTTP/1.1 200 OK
Connection: close
….
<Response>
<Code>1</Code>
<Receipt>912937791-0008912/01</Receipt>
<Amount>10000</Amount>
…
```

TRADITIONAL ATTACKS

TRADITIONAL ATTACKS



TRADITIONAL ATTACKS

- Change payment amount

  https://paymentgateway.com/pay? amount=0.01

  Solved with request validation!

- Spoof payment received message to return url

  https://merchant.com/return?Success=1&Amount=100.00&Message=Paid

  Solved with response validation!

REQUEST VALIDATION

- To validate the request of the payment page result, signed request is often used - which is the result of the hash function in which the parameters of an application confirmed by a «secret word», known only to the merchant and payment gateway.

REQUEST VALIDATION

- Protects the “**vital**” details of the transaction

Example:

- SHA1 of MERCHANTID, TXNTYPE, REFERENCEID, AMOUNT, CURRENCY, TIMESTAMP

BAE SYSTEMS
INSPIRED WORK

REQUEST VALIDATION EXAMPLE

```
sha1('ABC9999|password123|1|Invoice 986616|100.00|20140121222324‘)
4e65a02daacaf2f94f057fbc3d09c43883d10dc8
```

```
md5(‘password123abc9999100.00aud')
ce9b54a5bc2f08dd2a2bf5f3b2d2d8f0
```

```
md5(md5('20140121222324.ABC99999.Invoice
986616.100.AUD').'.Secrit123')
6a0a4eb970340d98fa33daf21400e5eb
```

RESPONSE VALIDATION

- Protects the “**vital**” details of the payment receipt

- Example:
- SHA1 of MERCHANTID, TRANSACTIONID, AMOUNT

BAE SYSTEMS
INSPIRED WORK

RESPONSE VALIDATION EXAMPLE

sha1('ABC9999|Secrit123|Invoice 986616|100.00')

c5af7bd81fec9eee6415fd1a4d77edc1e8ca9df6

md5('secrit123saltabc9999approved1-918490ae-9a1c-11de')

04beffd2eaf481e0d50ef2134188c6d0

md5(md5('20140121222324.ABC99999.Invoice 986616.00.Completed.auth.0000').'.Secrit123')

1f35ae73cf918f446cc45875948bd300

ABUSING REQUEST VALIDATION

- Bypass validation
- Abuse cryptographic properties
- Defeat secret key

BYPASSING REQUEST VALIDATION

- HTTP Parameter Pollution

  https://url/pay?amount=100.00&amount=0.01

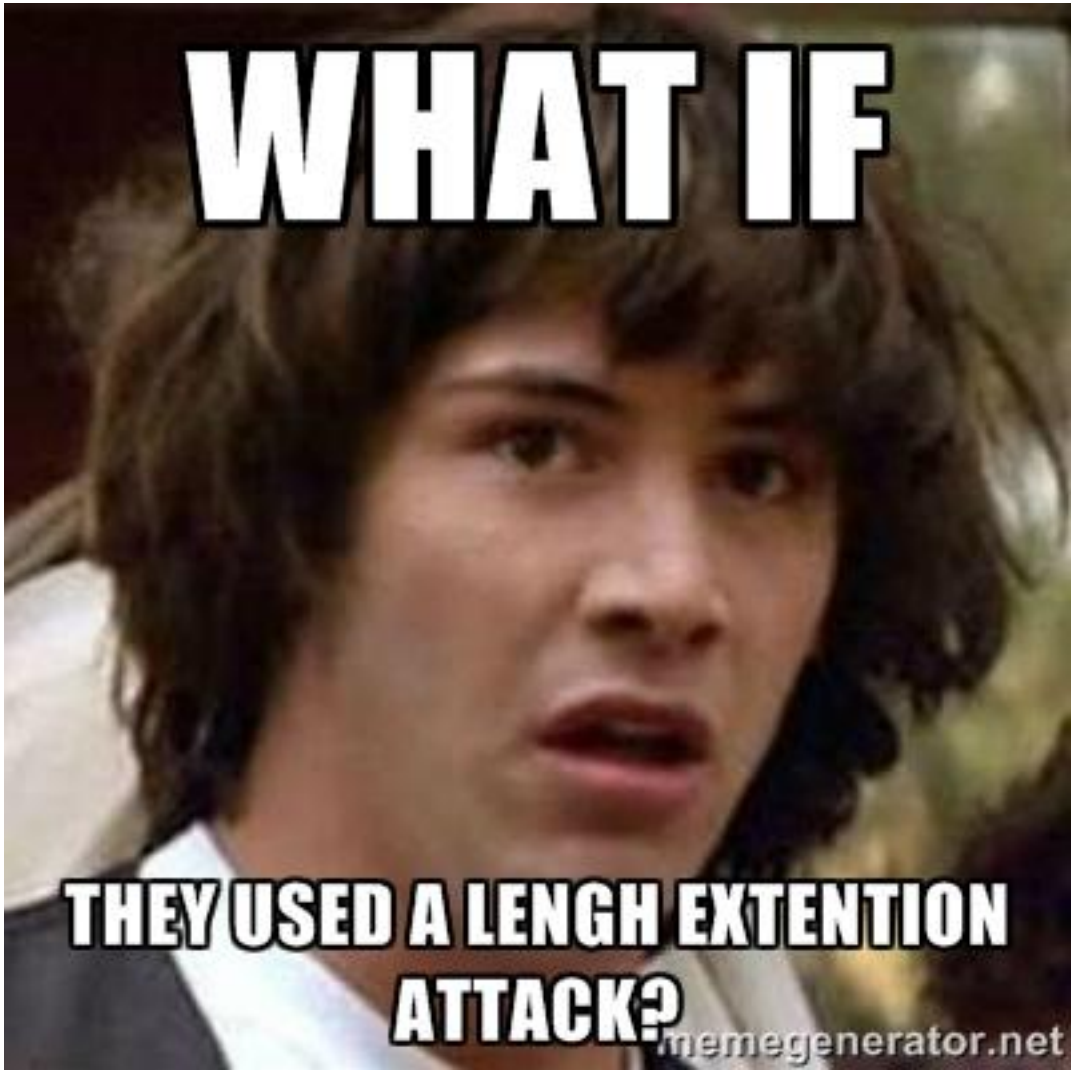
- Abusing unprotected parameters

  https://url/pay?expiry_date=31/12/2099

- Abusing application logic

  https://url/pay?pre_auth=1

ABUSE CRYPTOGRAPHIC PROPERTIES



LENGTH EXTENSION ATTACK

- The reason $H(k|m)$ is not the standard comes from the message extension attack
- Hashes operate on block of text
- Padding is used to fill out the blocks
- Attacker knows $H(k|m)$ and m
- Compute $H(k|m|p|m2)$
- p is the padding that would have applied to $k|m$
- $m2$ is an arbitrary message
- Attacker can now use $H(k|m|p|m2)$ and $m|p|m2$ to pass validation checks

LENGTH EXTENSION ATTACKS ARE COSTLY

DEFEATING REQUEST VALIDATION

Fingerprint	
EPS_MERCHANTID	
Password	
EPS_TXNTYPE	
EPS_REFERENCEID	
EPS_AMOUNT	
EPS_TIMESTAMP	

BAE SYSTEMS
INSPIRED WORK

DEFEATING REQUEST VALIDATION

```
<input hidden EPS_MERCHANT = “ABC999">
<input hidden EPS_TXNTYPE = "0">
<input hidden EPS_REFERENCEID ="Invoice 986616">
<input hidden EPS_AMOUNT ="100.00">
<input hidden EPS_TIMESTAMP ="20140121222324">
<input hidden EPS_FINGERPRINT
=“5f330cea9480efd63669b1b1464db1339c964bdf">
<input hidden EPS_RESULTURL = “https://www.merchantsite.com/”>
```

BAE SYSTEMS
INSPIRED WORK

DEFEATING REQUEST VALIDATION

Fingerprint	Web form
EPS_MERCHANTID	EPS_MERCHANT
Password	
EPS_TXNTYPE	EPS_TXNTYPE
EPS_REFERENCEID	EPS_REFERENCEID
EPS_AMOUNT	EPS_AMOUNT
EPS_TIMESTAMP	EPS_TIMESTAMP
	EPS_FINGERPRINT

DEFEATING REQUEST VALIDATION

Fingerprint	Web form
EPS_MERCHANTID	EPS_MERCHANT
Password	
EPS_TXNTYPE	EPS_TXNTYPE
EPS_REFERENCEID	EPS_REFERENCEID
EPS_AMOUNT	EPS_AMOUNT
EPS_TIMESTAMP	EPS_TIMESTAMP
	EPS_FINGERPRINT

DEFEATING REQUEST VALIDATION

Fingerprint	Web form
ABC0010	ABC9999
Secrit123	
0	0
Test reference	Invoice 986616
100.00	100.00
20120916221931	20140121222324
	5f330cea9480efd63669b1b1464db1339c964bdf

SHARED SECRET

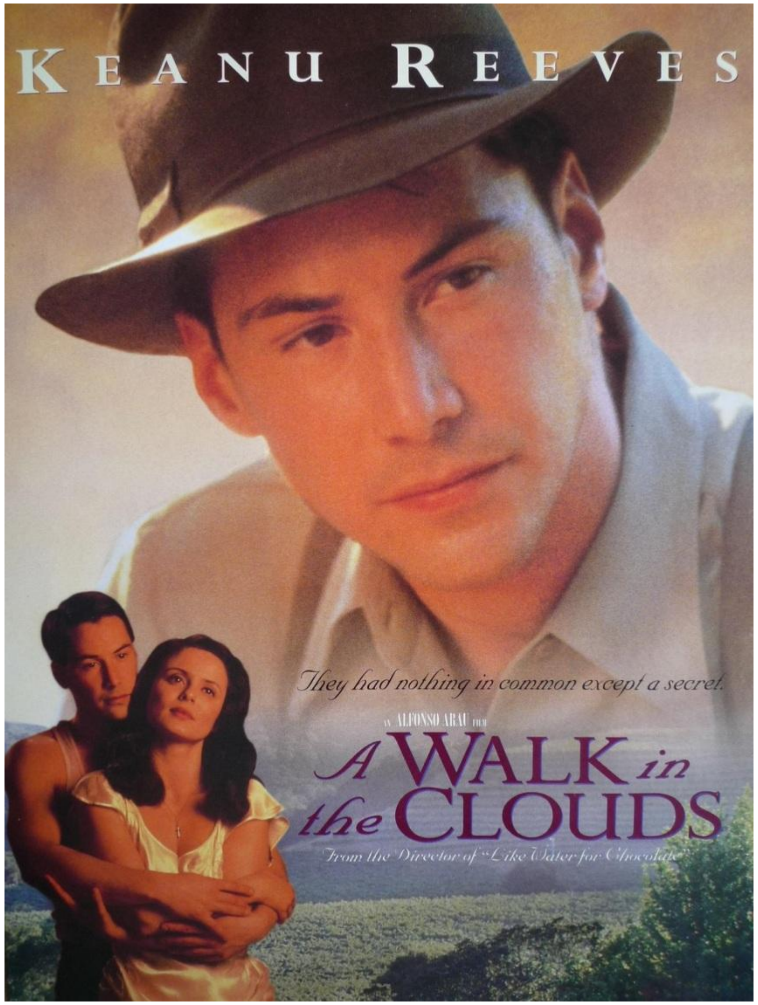
- Shared secret is usually:
- Vendor supplied
- Never changes
- Sometimes converted to upper/lower-case

BAE SYSTEMS
INSPIRED WORK

SHARED SECRET

Characters	Length
a-z0-9	8
a-zA-Z0-9	8
a-zA-Z0-9!@#$%^&*()[]-_=+;:'",./?	8
a-zA-Z0-9!@#$%^&*()[]-_=+;:'",./?	10
0-9a-f	32

TO THE CLOUD


KEANU REEVES
They had nothing in common except a secret.
AN ALFONSO ARAU FILM
A WALK in the CLOUDS
From the Director of "Like Water for Chocolate"

WHY USE CLOUD

- Easy alternative to having dedicated cracking hardware
- Low to no setup cost
- Readily available images for deployment
- Scales as required

CRACKING WITH JTR

- Jumbo distribution
- Define dynamic format
- Distributed cracking with MPI
- Increase performance with CUDA or OpenCL

JTR DYNAMIC FORMAT

```
[List.Generic:dynamic_1011]
Expression=md5($s.$p.$s2) (Payment gateway signature)
Flag=MGF_SALTED
Flag=MGF_SALTED2
Func=DynamicFunc__clean_input
Func=DynamicFunc__append_salt
Func=DynamicFunc__append_keys
Func=DynamicFunc__append_2nd_salt
Func=DynamicFunc__crypt_md5
Test=$dynamic_1011$c4a5babae57a7d58610ce33ca79622c9$ABC9999|$$2|Invoice 986616|100.00:xyz123
```

Validate:

./john --test --format=dynamic_1011

BAE SYSTEMS
INSPIRED WORK

HTML FORM TO DYNAMIC HASH

```
my $html =eval { local $/; open my $fh, "$ARGV[0]"; return <$fh>; close($fh); };
$html =~ m/(<h3>Credit Card Payment.*?<\/form>)/ms;
my $pwgform = $1;
my $form = HTML::Form->parse($pgwform, 'file:///');
my $merchantID = $form->find_input('MERCHANT_ID')->value;
my $amount = $form->find_input('AMOUNT')->value;
my $hash = $form->find_input('MD5HASH')->value;
my $account = $form->find_input('ACCOUNT')->value;
my $currency = $form->find_input('CURRENCY')->value;
my $notifyurl = $form->find_input('SHOP_DOMAIN')->value;
my $shopname = $form->find_input('SHOP_NAME')->value;
my $orderID = $form->find_input('ORDER_ID')->value;
my $floatAmt = $form->find_input('FLOAT_AMOUNT')->value;
my $timestamp = $form->find_input('TIMESTAMP')->value;
my $id_card = $form->find_input('ID_CARD')->value;
my $lang = $form->find_input('LANG')->value;
print "\$dynamic_1011\$".$hash."\$merchantID|\$\$2$orderID|$amount|$currency\n";
```

DISTRIBUTED CRACKING WITH MPI

Update makefile:

```
CC = mpicc -DHAVE_MPI -DJOHN_MPI_BARRIER -DJOHN_MPI_ABORT
MPIOBJ = john-mpi.o
```

Setup MPI over ssh using key based authentication

Create a MPI host file

```
192.168.1.2 slots=2
192.168.1.3
```

CRACKING WITH GPU

- GPU greatly outperforms CPU for hash calculation
- Scales with devices
- CUDA or OpenCL
- Available through some cloud providers

CRACKING WITH GPU

CUDA	OpenCL
Nvidia	Khronos group
Compiler builds kernel	Builds kernel at runtime
C language extensions	API only
Buffer offsets allowed	Buffer offsets not allowed
Pointer traversal allowed	Must use pointer arithmetic

BAE SYSTEMS
INSPIRED WORK

LOTS OF OPEN SOURCE OPTIONS

- Jtr
  http://www.openwall.com/john/
- Cryptohaze Multiforcer
  http://www.cryptohaze.com/multiforcer.php
- Wisecracker
  http://selectiveintellect.com/wisecracker.html
- Whitepixel
  http://whitepixel.zorinaq.com/
- Defuse gpu cracker
  https://defuse.ca/gpucrack.htm
- OCLcrack
  https://github.com/sghctoma/oclcrack

DEMO

BAE SYSTEMS
INSPIRED WORK

refundCard
merchantUUID
apiKey
transactionAmount
transactionCurrency
transactionID
refundAmount
hash

queryCard
merchantUUID
apiKey
transactionID
hash

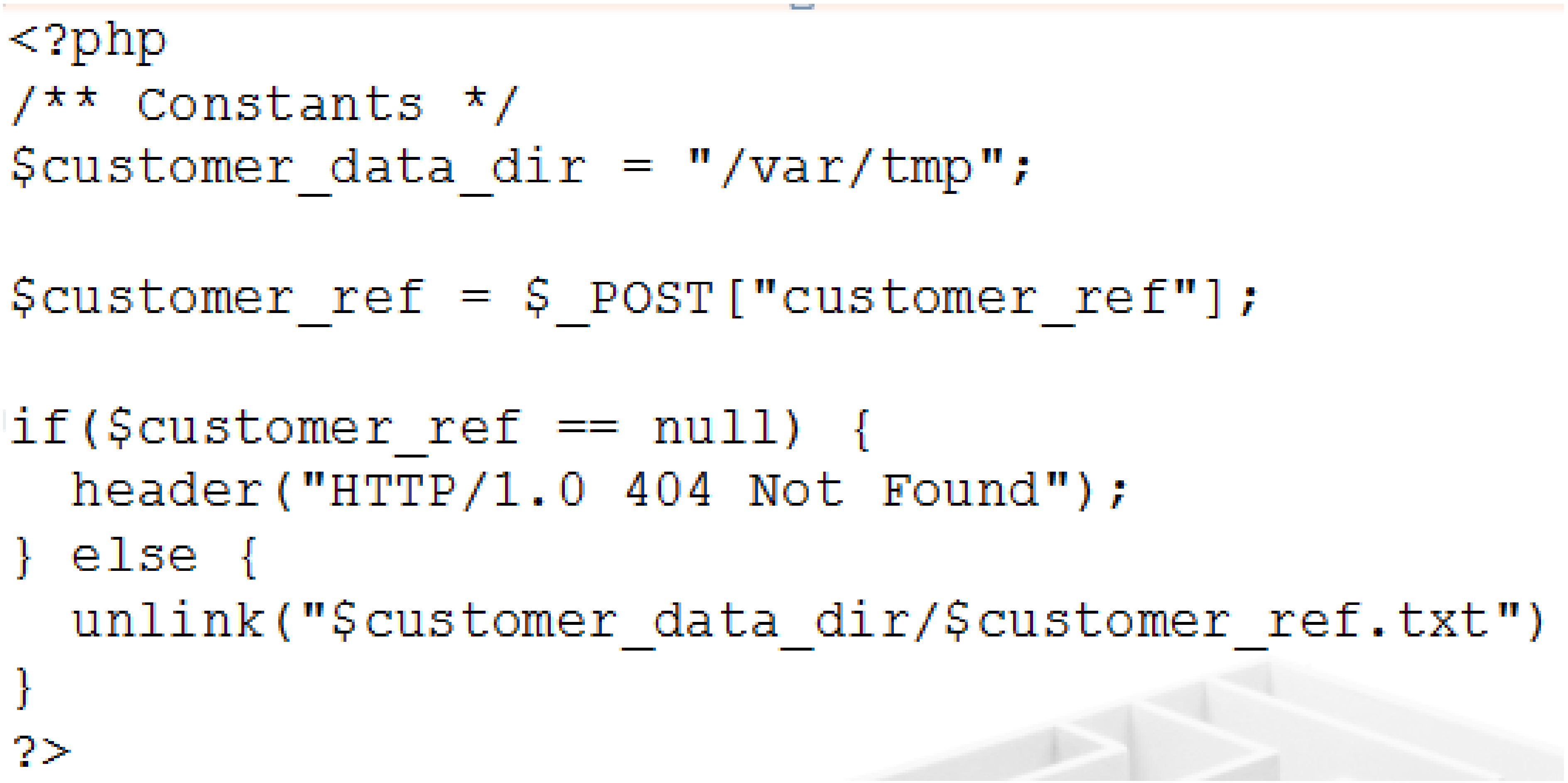
CONCLUSION

- Don't rely on the browser to drive traffic between the merchant website and the payment gateway
- Crypto is hard
- Use more than one unknown variable in request validation
- Use a long secret
- Use token based redirection
- Protect all parameters used in the request
- Use an established keyed-hash message authentication code (HMAC)
- Weak request validation does not equal an exploitable vulnerability

THERE WILL ALWAYS BE IMPLEMENTATION BUGS

BAE SYSTEMS
INSPIRED WORK

VULNERABLE VENDOR CODE



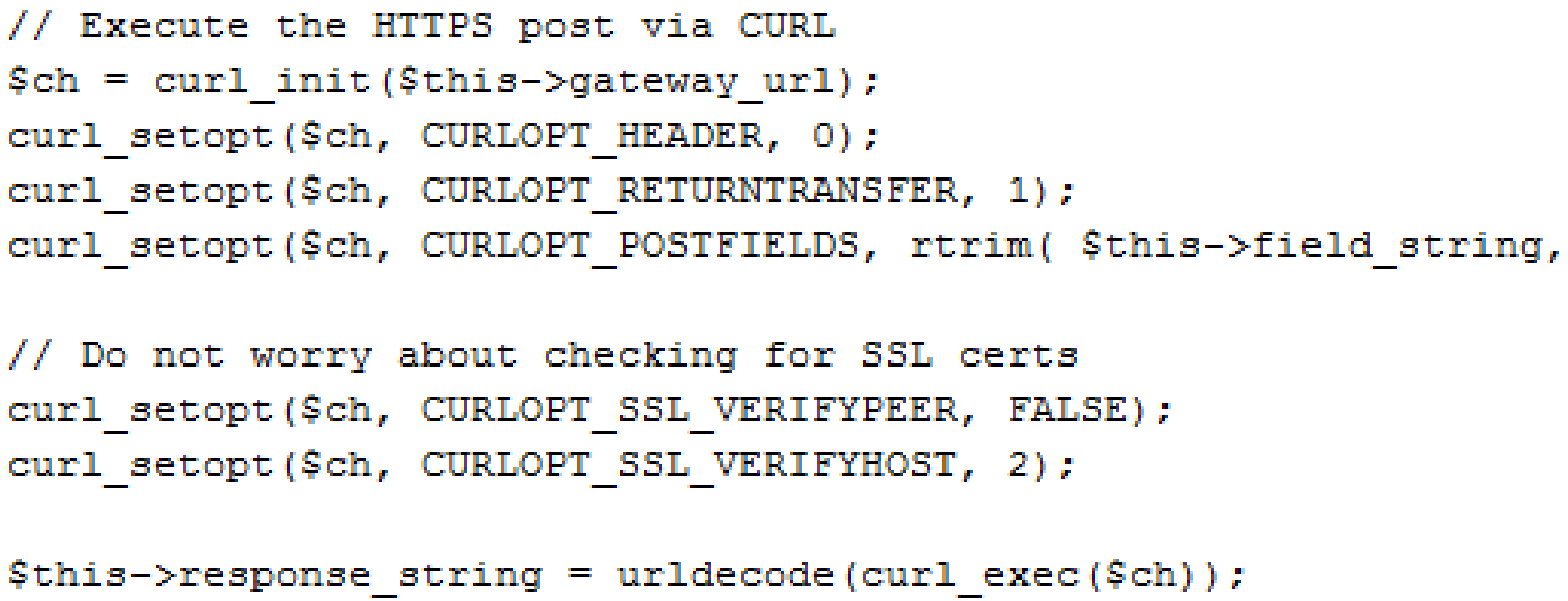
```php
<?php
/** Constants */
$customer_data_dir = "/var/tmp";

$customer_ref = $_POST["customer_ref"];

if($customer_ref == null) {
  header("HTTP/1.0 404 Not Found");
} else {
  unlink("$customer_data_dir/$customer_ref.txt")
}
?>
```

BAE SYSTEMS
INSPIRED WORK

BAD SSL PRACTISES

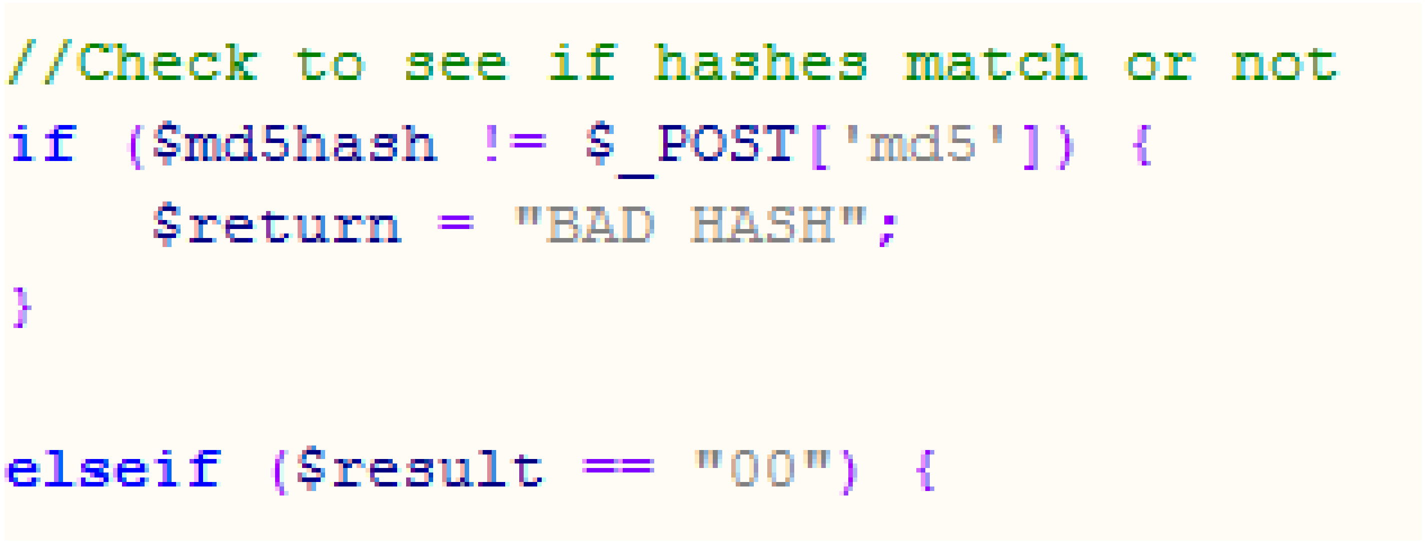


```
// Execute the HTTPS post via CURL
$ch = curl_init($this->gateway_url);
curl_setopt($ch, CURLOPT_HEADER, 0);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_POSTFIELDS, rtrim( $this->field_string,

// Do not worry about checking for SSL certs
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, FALSE);
curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, 2);

$this->response_string = urldecode(curl_exec($ch));
```

PHP'S TYPE JUGGLING

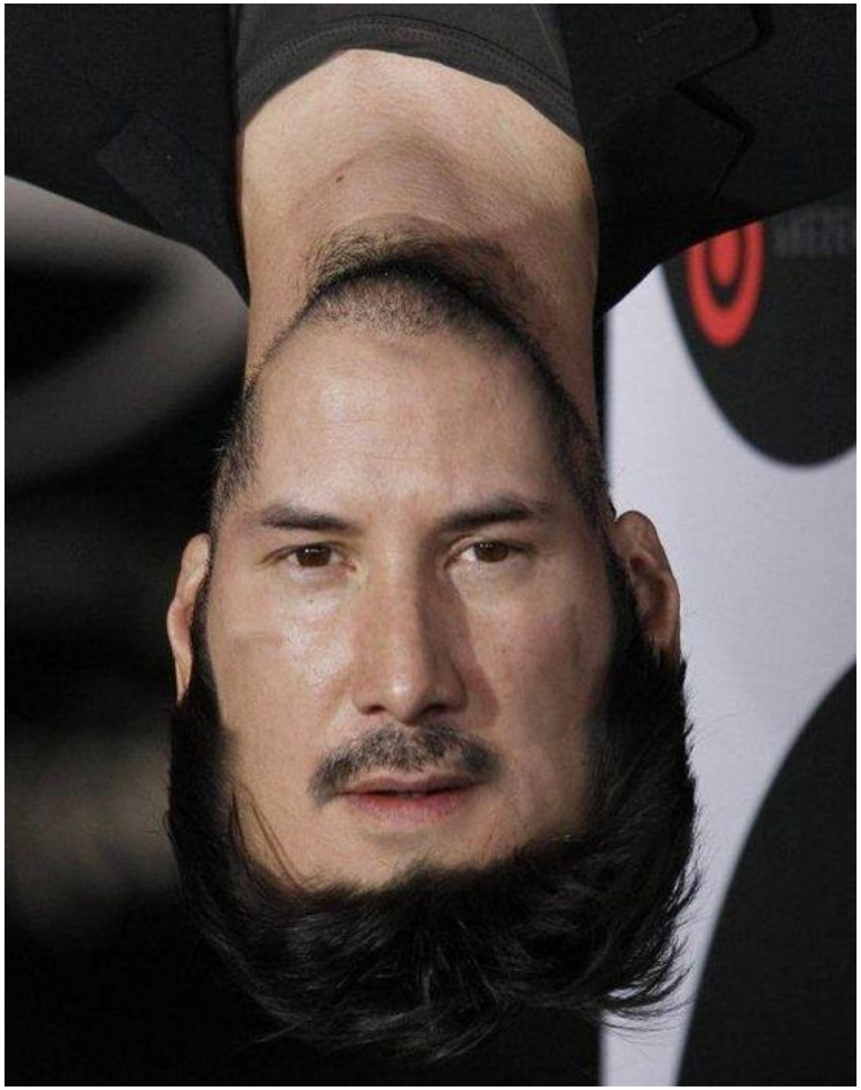


```
//Check to see if hashes match or not
if ($md5hash != $_POST['md5']) {
    $return = "BAD HASH";
}

elseif ($result == "00") {
```

BAE SYSTEMS
INSPIRED WORK

???QUESTIONS???



REPO

Slides and demo code can be found at:

https://github.com/wireghoul/presentations/BHAsia2014

Contact details

BAE Systems Applied Intelligence
Suite 1, 50 Geils Court
Deakin ACT 2600
Australia

Tel: +61 1300 027 001
Fax: +61 2 6260 8828

Email: australia@baesystemsdetica.com

Web: www.baesystemsdetica.com.au