ISO 9001 : 2008

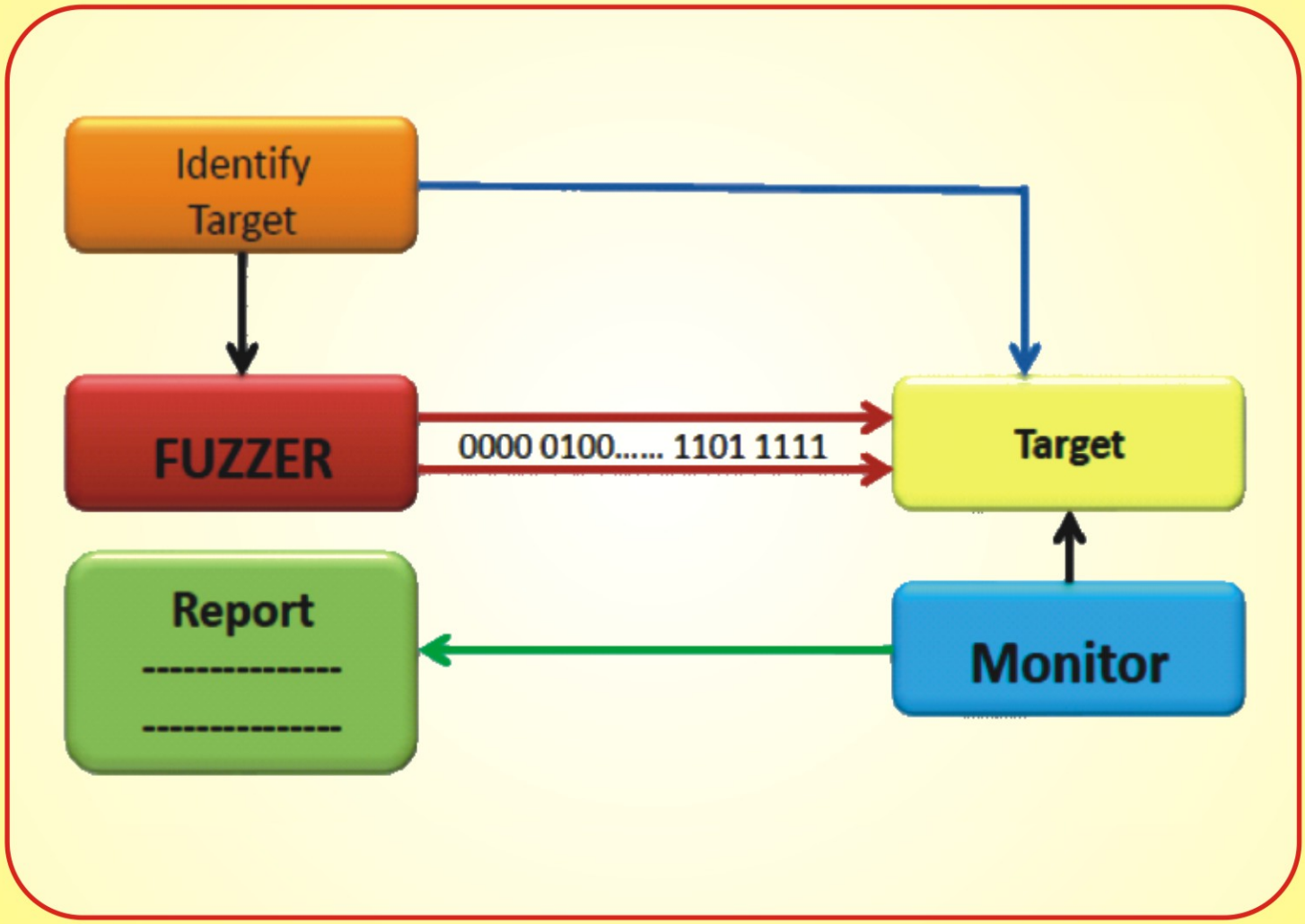
TEC

टी ई सी संचारिका

NEWSLETTER

VOL. 19 JULY 2015 ISSUE 2

FUZZ TESTING



ISO 9001:2008
TELECOMMUNICATION ENGINEERING CENTRE

IN THIS ISSUE

1.0 Introduction:

Fuzz testing or fuzzing is a software testing technique used to discover security vulnerabilities in network protocols, applications, file formats etc. Although the idea behind this technique is conceptually very simple, it is a well-known and widely established methodology employed in COTS software vulnerability discovery process. Fuzz testing was originally developed by Barton Miller at the University of Wisconsin in 1989. Since then the technique has evolved a lot and it is used internally by many companies to find bugs in their software. All the big companies such as Microsoft, Cisco, IBM, etc. have been using this technique to improve software quality. The real world is full of unexpected conditions and badly formed inputs. Softwares used in the telecom protocol stacks implementation and other associated implementations must be able to deal with the poorly formed inputs, unexpected actions and misuse of the software. Fuzzing is the process of sending intentionally malformed inputs to a piece of software to see whether it crashes or show any other unexpected behavior like CPU overload, memory leaks etc.

While using fuzzers to improve security, the end objective is not just finding vulnerabilities, but fixing them as well. Finding and fixing vulnerabilities in the products before their deployment in the network certainly saves money, protects customers and improves reputation.

2.0 Effectiveness :

"Fuzzers are a good way to find bugs, but not so good about making assurances that no bugs exist." - --- Michael Eddington ,author of the widely used open source fuzzer Peach Fuzz testing works best for vulnerabilities that can cause a program to crash, such as buffer overflow, cross-site scripting, denial of service attacks, format bugs and SQL injection. Fuzzing is less effective for dealing with security threats that do not cause program crashes, such as spyware, viruses, worms, Trojans and keyloggers. Fuzz testing can undoubtedly reveal defects that are overlooked when software is written and debugged. Nevertheless, it usually finds only the most serious

faults. This testing alone cannot provide a complete picture of the overall security of the target. Fuzzers are most effective when used in conjunction with other proven security testing methods.

3.0 Fuzzing Process

The basic phases involved in the fuzzing process (as shown in the figure placed at cover page of this newsletter) are as follows:

3.1 Identification of target

Before selecting the fuzzing tool it is required to identify the target, whether it is a protocol stack, application, file format or a web browser.

3.2 Identification of inputs

Most of the exploitations are caused due to the fact that target applications accept and process the user input without sanitizing and validating the input data. Hence, identification of inputs is a key to success of the fuzzing process.

3.3 Generation of fuzzed data

After identification of inputs, fuzzed data must be generated. This data may be some predetermined values, mutation of existing samples or some dynamically generated data depending upon the target and inputs identified.

3.4 Execution of fuzzed data

This step goes hand in hand with the previous one. It involves sending a fuzzed data packet towards the target after launching a target process.

3.5 Monitoring for exceptions

Exception or fault monitoring process is one of the most vital steps of fuzzing. Transmitting thousands of fuzzed data packets towards a target and ultimately causing that target to crash will be a futile exercise, if we are not able to pinpoint the packet responsible for crash.

3.6 Reporting

Finally, the findings of the fuzzing process are compiled in the form of a report, which later on helps in mitigation efforts.

4.0 Fuzz Testing approaches

One of the most important aspects of fuzzing is the generation of fuzzed inputs. These inputs or test cases are normally generated by using one of the following two approaches.

4.1 Generation based fuzzing

The starting point for this approach is a specification or RFC, which describes the file format or network protocol. Test cases are generated based on the input data format specified in these documents. Each generated test case should differ from the valid input to such an extent that a problem is caused in the target application, but should not be too invalid to be

discarded by the target application.Generation based fuzzing requires a significant amount 6 of upfront work to study the test specification and generate the test case from scratch. But, the extra knowledge gained by understanding the format results in higher quality test cases.

4.2 Mutation based fuzzing

In this approach valid data samples e.g. files, traffic captures etc. is collected and then modified. Examples of mutations include bit flipping, replacing small string with longer strings etc. The advantage of this approach is little or no knowledge of target is required. All that is needed is some good data samples. Since, the protocol stacks and applications contain large sections of code that may not execute with mutated inputs, mutation based fuzzers do not often perform an effective fuzzing.

Which approach is more suitable?

It depends on requirements, availability of time and target. For example,

i. Some clear text protocols such as FTP, SMTP, etc. lend themselves well to mutation fuzzing

ii. Some protocols such as SSH, IKE, etc will require generation fuzzing because of the complexity of the protocol.

Hence, it is always advisable to use both the approaches, if possible

5.0 Types of Fuzzers

Fuzzers are classified into the following categories

based on the targets.

5.1 Protocol Fuzzer

Flaws in the implementations of network protocols are some of the most serious security problems. Such flaws could allow a malicious user to attack vulnerable systems remotely over the Internet. A protocol fuzzer is used to discover implementation flaws in a protocol stack by sending the unusual inputs towards the target in hopes of producing unexpected behavior.

5.2 File format Fuzzer

A large number of applications deal with file input and output and are susceptible to vulnerabilities that occur while parsing maliciously crafted files.File format fuzzers discover these vulnerabilities. It dynamically creates different malformed files and sends them towards the target application and monitors the target for an unexpected behaviour.

5.3 Web application Fuzzer

Web application fuzzing can discover rvulnerabilities in the web application itself or any of the underlying components like database server with whom it might integrate with. For web 7 application fuzzing the delay caused in transport of inputs from fuzzer to the target is a challenge. Hence, if possible rather than running the target application on a remote server it should be hosted locally so that the packets don't have to traverse a network. Another option is to run the web application on a VM, while the fuzzer can be run locally. These fuzzers are used to discover vulnerabilities such as SQL injection, cross site scripting, command injection vulnerability etc.

5.4 Web browser Fuzzer

Vulnerabilities in web browsers have become a major area of concern as they are making their users victims of several attacks such as phishing attack, identity theft, creation of large botnets etc. Time and again, vulnerabilities have been discovered in almost all popular web browsers like internet explorer, Mozilla firefox, Google chrome etc. Web browser fuzzers are capable to discover these kinds of vulnerabilities, which occur while parsing of malformed HTML tags, javascript, image files like JPG, GIF and PNG, and ActiveX controls etc.

6.0 Open Source Fuzzing tools

Open source Fuzzing tools typically fall into one of three categories : Fuzzing frameworks, Special purpose fuzzers and General purpose fuzzers.Fuzzing frameworks are good if one is looking to write or develop a new fuzzer or need to fuzz a custom or proprietary protocol. They basically provide quick, flexible, reusable and homogeneous development environment for fuzzer developers.

Special purpose fuzzers are fuzzers that were written for a specific protocol or application. These fuzzers have a very limited scope and were usually developed to find few loopholes in a particular protocol or application and after that were left unmaintained.

General purpose fuzzers are generic fuzzers designed for minimizing setup time during fuzzing sessions and are especially useful for fast testing of proprietary and undocumented protocols.

6.1 Open Source Fuzzing frameworks

Some popular open source fuzzing frameworks are listed below.

i. **Antiparser ---** http://antiparser.sourceforge.net / Written in Python, antiparser is an API designed to assist in creation of random data for construction of fuzzers. This framework can be used to develop simple fuzzers that will run 8 across multiple platforms, but inadequate for handling complex tasks. This framework has not been updated since August 2005.

ii. **Autodafe ---** http://autodafe.sourceforge.net/ Autodafe supports fuzzing of both network protocols and file formats. It is designed to run on unix platforms and lacks Microsoft windows support. It takes an xml based packet dump as an input to generate fuzzing test cases and has discovered mainly buffer over flow vulnerabilities. This tool has not been updated since August 2006.

iii. **Peach Fuzzer -** http://peachfuzz.sourceforge.net/

Peach is a cross-platform fuzzing framework originally written in Python and released in 2004. It uses a combination of mutation and generation based fuzzing. Peach Fuzzer supports fuzz testing of any file format, network protocol, application protocol, Android device, or embedded hardware. It can be extended to enable fuzzing of proprietary systems and interfaces. Its commercial version is also available. It is under active development and the latest update was released in November 2014. Although, Peach is highly advanced in theory it is not so well documented.

iv. **Dfuz ---** http://genexx.org/dfuz/

Dfuz is a remote protocol fuzzer, which is capable of sending random data combined with the positive inputs. It also sends rule files to the target along with the inputs, which is parsed by the target to know how the input data is to be used. Dfuz can even be used by the nonprogrammers to fuzz protocols. But, there is a very limited scope for code reuse, as any duplication or modification of the code is not permitted without permission of the author. This framework has been used to uncover a variety of vulnerabilities affecting vendors such as Microsoft, Ipswitch and RealNetworks.

v. **SULLEY ---** http://fuzzing.org/sulley/

Sulley is a python fuzzing framework, which exceeds the capabilities of almost all previously published technologies in the public domain. Mostly, fuzzers are focused on data generation. Sulley not only has impressive data generation, but monitors the health of the target as well as is capable of reverting the target to a good state. It detects, tracks and categorizes detected faults. Sulley can pin point the unique sequence of test cases, which triggers faults. Sulley will be taken as a separate study paper during next year and may be used as an open source fuzzing framework in the upcoming security test lab.

6.2 Open Source Special-Purpose Fuzzing tools

These tools are generally written for a specific protocol or application. Some of the commonly available open source special-purpose fuzzing tools are:

i. **SPIKE Proxy ---** www.immunitysec.com/ resources-freesoftware.shtml for web applications.

ii. **Mangle ---** www.lcamtuf.coredump.cx/ for HTML file fuzzing

iii. **WebFuzzer ---** www.gunzip.altervista.org/ for web application fuzzing.

iv. **Ip6sic ---** www.ip6sic.sourcefourge.net/ for stress testing of an IPv6 stack implementation.

v. **Blue Tooth Stack Smasher ---** www.secuobs.com/ for fuzzing of Bluetooth devices.

vi. **Radius Fuzzer ---**
www.suse.de/projects/radiusfuzzer/ for fuzzing radius protocol.

6.3 Open Source General-Purpose Fuzzing tools

These tools are good beginning to fuzzing with minimal effort and to get ideas on how fuzzing should be done and how it works. Some of the most common open source general purpose fuzzing tools are mentioned below.

i. **SPIKE ---** www.immunitysec.com/ resources-freesoftware.shtml

It is the most preferred tool to analyze a new protocol for buffer overflows or similar weaknesses. It is available for Linux platform only and requires a strong knowledge of C to use. One of the first block

based open source fuzzing tool and is known to have found vulnerabilities in Microsoft Windows' RPC framework and other products. This tool has not been updated since August 2005.

ii. **RADAMSA -**

http://ouspg.googlecode.com/files/radamsa

It is designed to be a good general purpose tool for developers and vendors who want to test how well their products can withstand malicious inputs. It has been used to find previously unknown security vulnerabilities in handling of many kinds of file formats with very different structure, like bmp, png, gif, jpeg, svg, xml, ogg, avi, html, gz, bzip2, tiff, pdf and zip. This tool is available for platforms such as GNU/Linux, OpenBSD (and probably other BSDs). Mac OS X and Windows (experimental).

iii. **SDL MiniFuzz File Fuzzer --** http://www.microsoft.com/enus/download/details.aspx?id=21769 10

Microsoft SDL MiniFuzz File Fuzzer is designed to help detect code flaws that may expose security vulnerabilities in file-handling code. This tool creates multiple random variations of file content and feeds it to the application to exercise the code in an attempt to expose unexpected and potentially insecure application behaviors. This tool is available for windows platform only.

iv. **Flayer ---** https://code.google.com/p/flayer/

Google's security team has released this fuzz testing tool, which was used internally to find multiple vulnerabilities in Internet-critical software products. The fuzzer, called Flayer, is an analysis and flow alteration tool that has been used to find errors in real software. In the past years, results from Flayer has led to the discovery of security holes in several open - source products, including OpenSSH, OpenSSL, LibTIFF and libPNG.

7.0 Commercial Fuzzing Tools

Entrance of commercial players in a technology space strongly indicates that a certain level of maturity has already been achieved. This trend has already been witnessed with fuzzing technology. A large number of software developers such as Microsoft, Google etc. have adopted fuzzing as a means of identifying security vulnerabilities earlier in the SDLC. This has resulted into the emergence of companies to fill the need of robust, user friendly fuzzers.

Common Fuzz tool may run on Windows, UNIX and Linux platforms. It comes with a predefined set of protocol modules each containing a generation based fuzzer as per the full description of protocol in the relevant RFCs. Support for developing mutation based fuzzer for proprietary protocols is also available.

Testing suites are also available for file fuzzing. The negative test cases are sent towards the target in an attempt to see if the target survives the attacks and continue to provide the useful service. Hardware solution are available to perform a benchmark between security devices (firewalls, IDSs etc). Testing is done by creating complicated traffic scenarios or by capturing the network traffic and recreating it, altering it or amplifying it. The test suite include various interfaces including command line, GUI and web based interface.

8.0 Challenges and Future Trends in Fuzzing

Although there are many open source and commercial tools available for fuzz testing of targets such as network protocols, file formats, web browsers, web applications etc., many problems are expected to arise during the testing. For example, some mutation-based fuzzers may run indefinitely, endlessly modifying data and supplying it to the target. Assuming the target never crashes, how do we know when to turn off this type of fuzzer. Another problem, which is associated with generation-based fuzzer with finite set of fuzzed test case is even if the target does not crash after running all the test cases, what to do next? Should we conclude that the target is secure? Is there a measurement technique available for depth of fuzzing? Is there a way to improve the fuzzing test cases based on the results of previous test runs. The answer to all these questions can be found in a new evolving technology known as white box fuzzing. For white box fuzzing source code is required. It is basically code coverage along with fuzzing, which is used to measure the effectiveness of fuzzing. It can be used to identify the portions of the code, which was not covered during the fuzzing. Additional test cases can be generated to increase the effectiveness of fuzzing. Microsoft is already using an internally developed white box fuzzer "SAGE" for security testing of their software products.

9.0 Standardisation Activities

3GPP has designed a security evaluation approach under the name SECAM (Security assurance methodology), which has been documented in technical report TR33.805 release 12. Clause 5.2.4.4 of this document talks about basic vulnerability testing (BVT), which consist of requirements for running automated Free and Open Source Software (FOSS) and Commercial off-the-shelf (COTS) security testing tools against the external interfaces of a Network Product. This activity covers at least three aspects: Port Scanning, Vulnerability Scanning by the use of Vulnerability scanners and robustness/fuzz testing.

10.0 Telecom Network vulnerabilities discovered by Fuzz testing

In recent past, many software vulnerabilities in the telecom network entities have been discovered by fuzz testing. Some of them are mentioned below. i. In 2013, MME and HS-GW of a mobile network operator crashed, which affected more than 16M subscribers. Fuzz testing of MME and HS-GW software revealed bugs in the software, which were found vulnerable to DOS attacks. ii. In 2013 again another Asia Pacific MNO had a trouble with its HSS disclosing subscribers key material as a result of NAS injection attack from UE over the Air (OTA). Fuzz testing of mobile (LTE) packet core network revealed lack of sanitization on NAS stack on MME exposing HSS to OTA attacks iii. In 2010, HLR of a MNO crashed due to malformed SS7 traffic resulting into 12 hour downtime of the complete network. On fuzz testing of the HLR SS7 stack software, DOS bugs were found.

iv. Most recently in April 2014, Heartbleed bug (CVE - 2014-0160) in the Open SSL's implementation of the TLS/DTLS (transport layer security protocols) was discovered again with the help of fuzz testing.

11.0 Conclusion

Fuzz testing is simple and offers a high benefit-tocost ratio. Fuzz testing can often reveal defects that are overlooked when software is written and debugged. However, fuzz testing alone cannot provide a complete picture of the overall security of a target. It has to be used in conjunction with other security testing techniques such as static analysis, binary analysis etc. There are many open source and commercial tools available for fuzzing, each having its own strengths and weaknesses. No single approach is full-proof, which means by using multiple approaches for fuzzing more vulnerabilities can be discovered.

Activities at NTIPRIT (Jan to Jun 2015)

1. In-service training courses for DoT Officers were conducted at NTIPRIT on the following topics:
   i. Renewable Energy and Energy storage Technologies
   ii. IPv6 Applications Perspective
   iii. Energy conservation, Monitoring & Auditing
   iv. Workshop on Greening the Telecom for sustainable Growth
   v. Unified License-An overview
   vi. Workshop on cyber Security

   Total 57 officer trainees (OTs) participated in the above courses.
2. Induction Training courses for Officer Trainees of ITS-2012 batch were conducted on 'Lawful Interception &Monitoring' and 'Structure of Newtorks, Interconnection & Service Provisioning'.
3. Induction Training of the following Officer Trainees of ITS/BWSstarted on 08.06.2015:
   i. ITS-2012 Batch (2 officers)
   ii. ITS-2013 Batch (4 officers)
   iii. P&T BWS (Electrical)-2013 Batch (3 officers)
   iv. P&T BWS (Civil)-2013 Batch (3 officers)
   v. P&T BWS (Architect)-2010 Batch (2 officers)
4. Induction Training of the following JTO batch started from 01.06.2015:
   (i) JTO Trainees – 2011 Batch (4 officers)
   (ii) JTO Trainees – 2013 Batch (5 officers)
5. Apart from classroom training courses, the OTs went on a study tour to Major Telecom Installations, Telecom Industries and TERM Cells at Chennai, Banagalore & Hyderabad for a period of two weeks.
6. The OTs of ITS-2011 batch completed their On-Job Training in various units of DoT. Subsequently proceeded for Six-weeks customised training programme (4 weeks module on 'Management' and 2 weeks module on 'Government Acts and Laws for Administrative & Business functions') at Haryana Institute of Public Administration, HIPA, Gurgaon commencing from February 23 , 2015.
7. Smt. V. Sobhana, DDG (NGN & PR) and Shri C. S. Sharma, Director (NP) won Prizes at the Inter-Ministry Music, Dance & Short play Competition organised by Central Civil Services Cultural & Sports Board (DoPT) at CSOI, Vinay Marg, New Delhi on 17-20 Feb 2015.
8. Conducted the Professional Examination of 2010 & 11 batches of ITS officers successfully. (May 27 & 28, 2015)
9. Conducted the Hindi Test for 2010, 11& 12 batches of ITS officers successfully. (May 26, 2015)

Hon'ble MOC & IT has released Technical report on M2M

Hon'ble MOC & IT has released Five Technical Reports of TEC in a function in Vigyan Bhawan on 12th May 2015. These Technical Reports are on M2M enablement in Power, Automotive (Intelligent Transport Systems), Health (Remote Health Management), Safety & Surveillance Sectors and on M2M Gateway & Architecture

हिंदी कार्यशाला

दूरसंचार अभियांत्रिकी केंद्र में दिनांक 26/03/2015 और 16/06/2015 को हिंदी कार्यशाला का आयोजन किया गया।

दिनांक 26/03/2015 को आयोजित कार्यशाला के वक्ता श्री नरेंद्र चॉबे, सहायक महानिदेशक (एफ.एन.) द्वारा कार्यालय में फाईलों पर दैनिक प्रयोग में उपयोग होने वाले हिन्दी शब्दों पर चर्चा की गयी तथा फाईलों पर ज्यादा से ज्यादा टिप्पणियां हिन्दी में लिखने का सुझाव दिया।

दिनांक 16/06/2015 को आयोजित कार्यशाला के वक्ता श्री विक्रम सिंह, राजभाषा विभाग द्वारा यूनिकोड, हिंदी व्याकरण, फॉन्ट कन्वर्ट के संबंध में गूगल पर बोल कर सर्च करने तथा हिंदी के बारे में काफी रोचक एवं ज्ञानवर्धक जानकारियों उपलब्ध कराई गई।

(हिन्दी कार्यशाला में भाग लेते अधिकारी / कर्मचारी गण)

Approvals from JAN 2015 to JUN 2015

S.No.	Name of the Company /Name of Product & Modal No.
1.	M/s Huawei Telecommunications India Co Pvt Ltd
1.1	Transmission Equipment, OptiX OSN 550
1.2	Transmission Equipment, OptiX OSN 550 (other IR)
1.3	Transmission Equipment, OptiX OSN 500
1.4	Transmission Equipment, OptiX OSN 500 (other IR)
1.5	Switrching Node with N/W to N/W interface at STM-1, CSOFTX3000 with UMG Q900 (V1R8, V2R9)
2.	M/s Sunren Technical Solutions Pvt. Ltd.
2.1	Terminal for Connecting to PSTN,MAX EX
2.2	Terminal for Connecting to PSTN,CONVERGE PRO 880 TA
2.3	Terminal for Connecting to PSTN,CONVERGE PRO880 T
2.4	Terminal for Connecting to PSTN,CONVERGE PRO840 T
2.5	Terminal for Connecting to PSTN,CONVERGE PRO TH20
2.6	G3 FAX Machine C521 B
2.7	G3 FAX Machine C511 D
2.8	G3 FAX Machine C463 C
3	M/s Avaya India Pvt. Ltd
3.1	PABX, Avaya Aura Communication Manager
4	M/s ZTE Telecom India Pvt Ltd
4.1	Switrching Node with N/W to N/W interface at STM-1, ZXMSG9000 with ZXSS10SS1b
5	M/s NEC India Pvt. Ltd
5.1	PABX with SL1000
6	M/s Tejas Network Ltd
6.1	STM-16 TM/ADM & TJ1400
7	M/s Accord Comunications Ltd
7.1	PABX & ADX 600
8	M/s Clearone Inc
8.1	Terminal for connecting to PSTN & 880 TA
8.2	Terminal for connecting to PSTN & MAXEX
9	M/s ECL Telecom Ltd.
9.1	Digital Multiplexer (SDH) STM-16, BG64
9.2	Digital Multiplexer (SDH) STM-16, BG30
9.3	Digital Multiplexer (SDH) STM-1, BG20
9.4	Digital Multiplexer (SDH) STM-16, BG30 (other IR)
9.5	Digital Multiplexer (SDH) STM-64, BG64
10	M/s Intellicon Pvt Ltd
10.1	PABX for Network connectivity, KAREL DS 200
11	M/s Panasonic India Pvt Ltd
11.1	PABX for Network Connectivity, KX-NS300SX
11.2	PABX for Network Connectivity, KX-NS1000BX
12	M/s Aspect Contact Center Software India Pvt Ltd
12.1	System employing computer Telephony Integration DCP-00
13	M/s HP India Sales Pvt Ltd
13.1	G3 FAX Card BOISB-1102-00
14	M/s Brother International Pvt. Ltd.
14.1	G3 FAX Machine MFC-8910DW
14.2	G3 FAX Machine FAX-2840
14.3	G3 FAX Machine MFC-8510DN
14.4	G3 FAX Machine MFC-L2701 DW
14.5	G3 FAX Machine MFC-L2701D
15	M/s Polycom Unified Communication Solutions Pvt. Ltd
15.1	ISDN CPE
16	M/s Arvind Limited (Telecom Division)
16.1	PABX for Network Connectivity
17	M/s CDOT
17.1	Mini OLT based GPON

Important Activities of TEC during JAN -2015 to JUN-2015

New GRs/IRs issued on

- IR on WIFI Access Point
- GR on IP PABX
- GR on Power Meter
- GR on Hybrid M/W Radio equipment for 6GHz
- GR on Database Management system for telecom
- GR on 80 channel DWDM equipment with bit rate of 10G/40G/100G for Core/Metro Network applications
- GR on Aerial Drop Optical Fibre Cable (for last mile Application)
- GR on Radio Access system for Broadband Application in 3.3-3.4 GHz band.

Revised GRs/IRs issued on

- IR on Universal Subscriber Identity Module
- IR on Group3 Fax Machine/card
- IR on Electronic Telephone Instrument
- IR on ADSL 2+ system
- IR on Subscriber Identity Module
- IR on PABX for Network Connectivity
- IR on V.90 Modem
- IR on Terminal for Connectivity to PSTN
- GR on Universal Subscriber Identity Module
- GR on Short Message Service Cell Broadcast
- GL on Planning & Maintenance guideline for Solar Photovoltaic (SPV) Power Supply
- GL on Planning guideline for Switch Mode Power Supply (SMPS) Power Plant
- GR on Remote Fibre Monitoring System
- GR on L 4-7 load balancer switch
- GR on Ethernet Traffic Analyser for Ethernet transport service testing(Handheld)

Study/White Papers issued on

- OTN at Edge, IP PABX
- Lawful Interception in multi-access technology scenario
- Identity Management, Fuzz Testing
- PPDR Communication Networks
- Global Technological Trend in telecommunication
- Issues related to PMA, Indigenous development of telecom technologies & domestic manufacturing of equioment
- eMS NMS Architecture in current telecom network
- Lightning Protection requirement for telecom equipment

ISO : 9001-2008

Certifications issued by TEC

Type Approval (TA)

Interface Approval (IA)

Certificate of Approval (CoA)

Visit

www.tec.gov.in

Regional TEC Contact :

Eastern Region	:	**033-23570008**
Western Region	:	**022-26610900**
Northern Region	:	**011-23329464**
Southern Region	:	**080-26642900**

Other Activity

* Six national contributions were submitted in ITU-T/R.
* New work items on "QoS norms for interconnections of telecom networks" and "Security testing techniques for telecom networks" based upon proposals from TEC were accepted by ITU-T in study group 12 and 17 respectively.
* ITU-T Study Group 15 & 17 meeting held in the month of FEB 2015 in TEC.

Approvals issued by TEC during the period from JAN 2015 to JUN 2015

Interface Approvals 37
Type Approvals 0
Technology Approvals 1

DISCLAIMER : TEC Newsletter provides general technical information only and it does not reflect the views of DoT, TRAI or any other organisation. TEC/Editor shall not be responsible for any errors, omissions or incompleteness.

टी ई सी संचारिका : दूरसंचार अभियांत्रिकी केन्द्र
जुलाई 2015 : खुर्शीद लाल भवन
भाग 19 : जनपथ
अंक 2 : नई दिल्ली–110001

Editor : Sunil Purohit, DDG (NGS) Phone : 23329354 Fax : 23318724 E-mail : ddgs.tec@gov.in